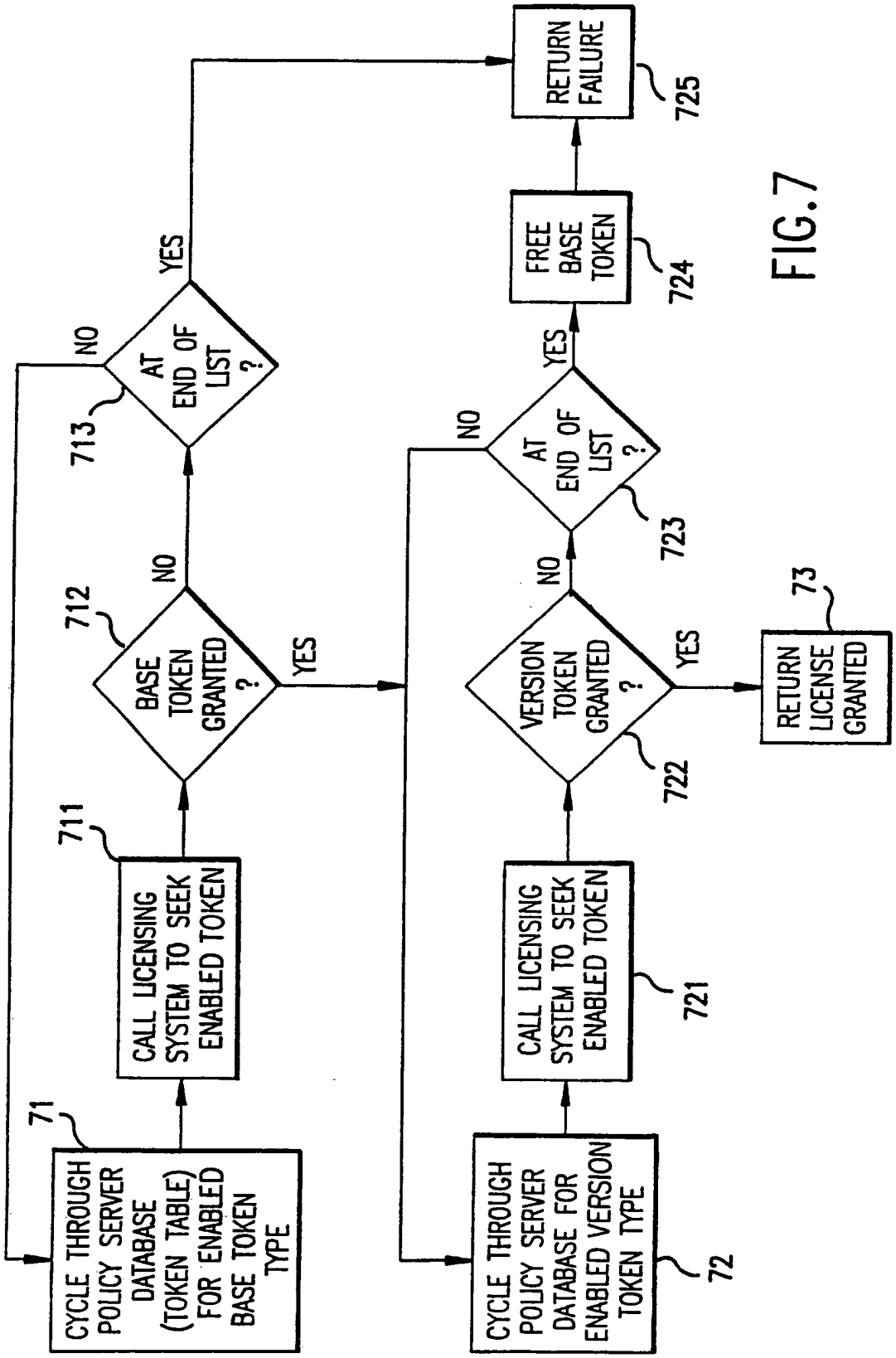


FIG.7

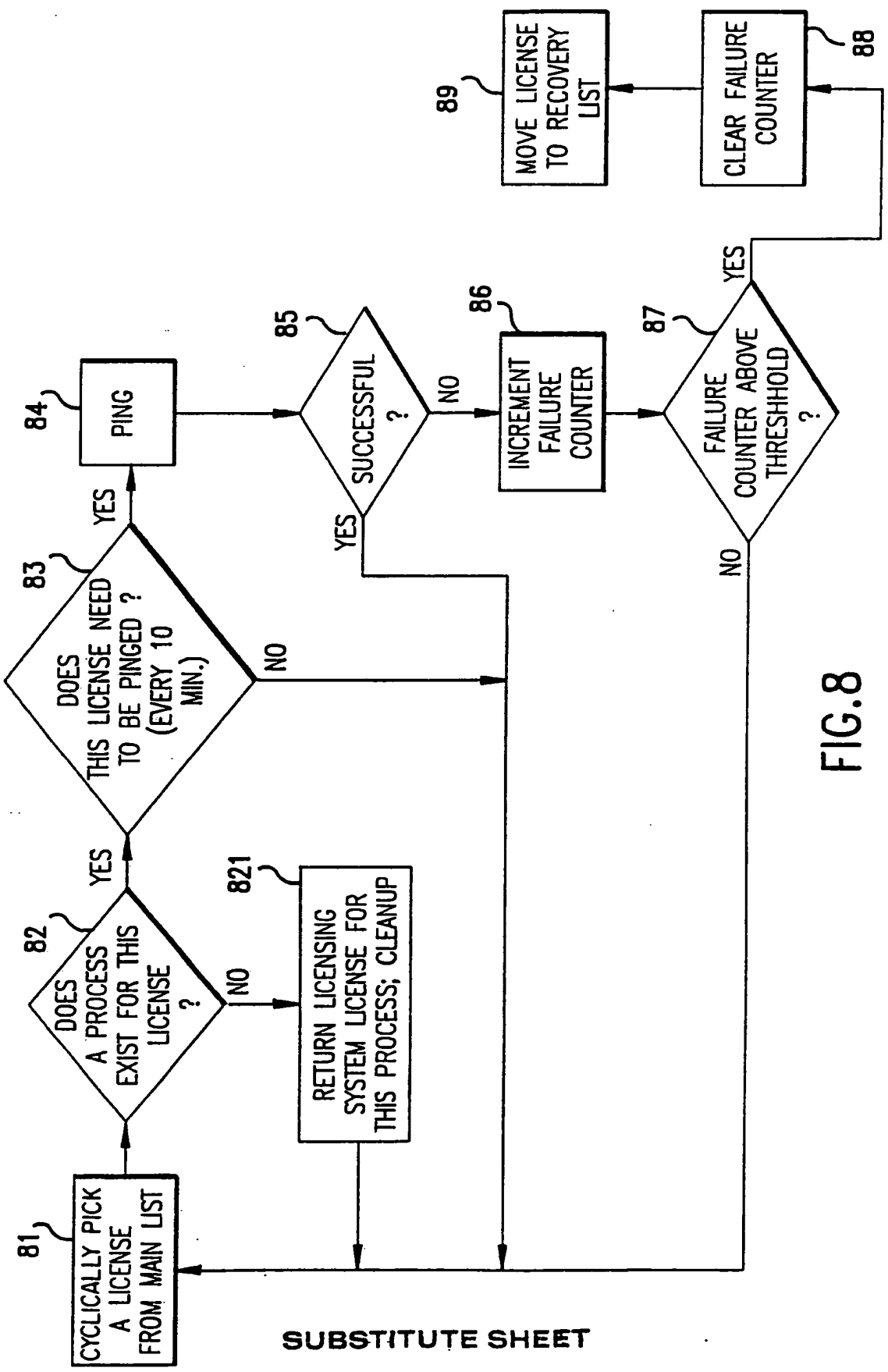


FIG.8

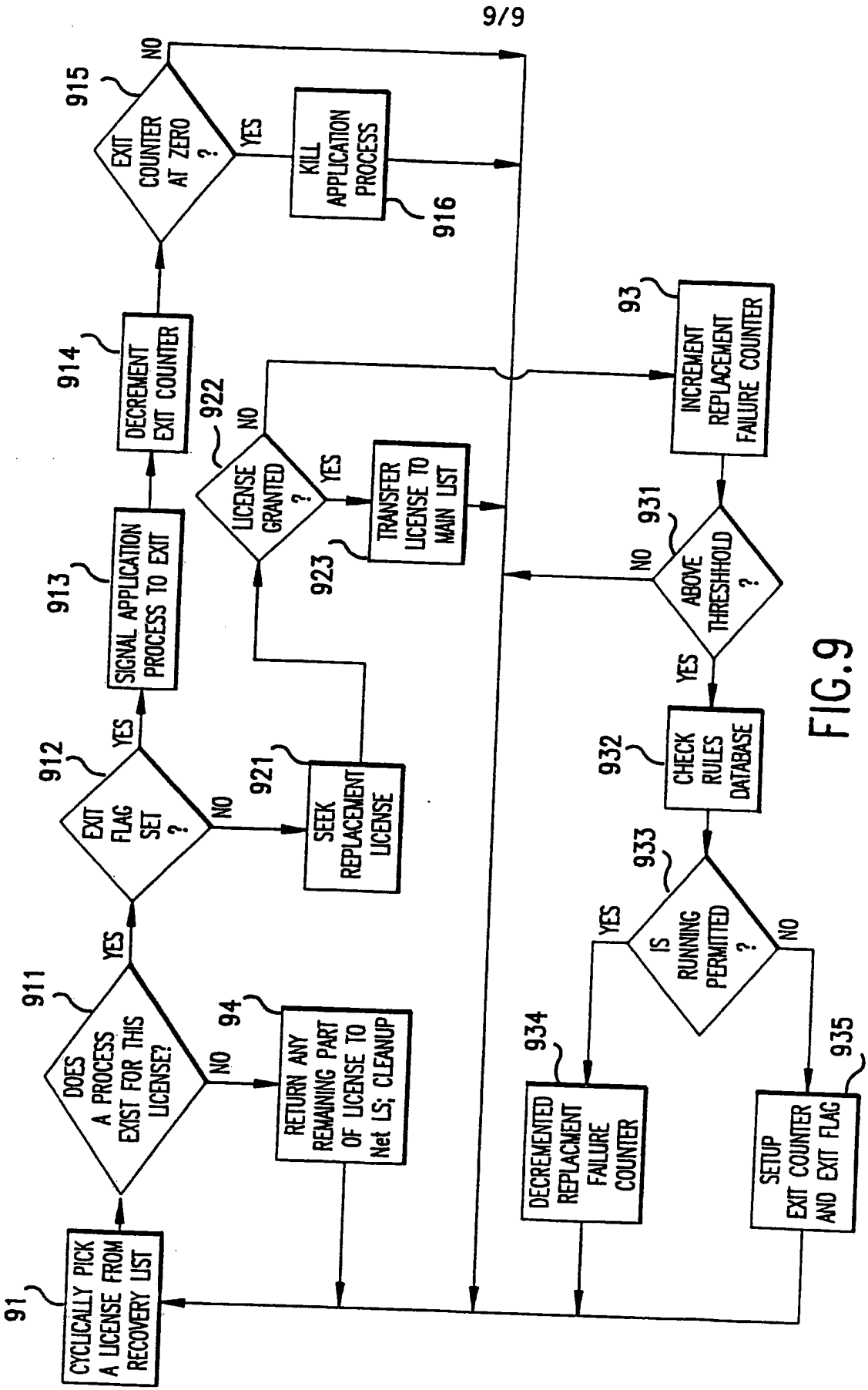


FIG.9

INTERNATIONAL SEARCH REPORT

PCT/US 92/10215

International Application No

I. CLASSIFICATION OF SUBJECT MATTER (if several classification symbols apply, indicate all)[6]

According to International Patent Classification (IPC) or to both National Classification and IPC

Int.Cl. 5 G06F1/00; G06F11/34

II. FIELDS SEARCHED

Minimum Documentation Searched[7]

Classification System	Classification Symbols
Int.Cl. 5	G06F

Documentation Searched other than Minimum Documentation to the Extent that such Documents are Included in the Fields Searched[8]

III. DOCUMENTS CONSIDERED TO BE RELEVANT[9]

Category °	Citation of Document,[11] with indication, where appropriate, of the relevant passages[12]	Relevant to Claim No.[13]
A	GB,A,2 236 604 (SUN MICROSYSTEMS, INC.) 10 April 1991 see page 9, line 11 - page 10, line 28 see page 12, line 16 - page 13, line 13 see page 14, line 20 - page 16, line 27 see figures 1-3	1-7, 13-15, 18-19
A	US,A,5 023 907 (APOLLO COMPUTER) 11 June 1991 see column 2, line 49 - column 5, line 42 see figures 1,2	1-5, 13-14, 18-19

-/--

° Special categories of cited documents :[10]

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

IV. CERTIFICATION

Date of the Actual Completion of the International Search	Date of Mailing of this International Search Report
26 FEBRUARY 1993	23. 03. 93
International Searching Authority	**Signature of Authorized Officer**
EUROPEAN PATENT OFFICE	JOHANSSON U.C.

Form PCT/ISA/210 (second sheet) (January 1985)

III. DOCUMENTS CONSIDERED TO BE RELEVANT (CONTINUED FROM THE SECOND SHEET)

Category °	Citation of Document, with indication, where appropriate, of the relevant passages	Relevant to Claim No.
A	EP,A,0 332 304 (DIGITAL EQUIPMENT CORP.) 13 September 1989 see column 3, line 31 - column 6, line 8 see column 6, line 42 - column 7, line 23 see column 8, line 33 - column 9, line 44 see figure 1 -----	1,2, 10-13,18

ANNEX TO THE INTERNATIONAL SEARCH REPORT ON INTERNATIONAL PATENT APPLICATION NO.

US 9210215
SA 67461

This annex lists the patent family members relating to the patent documents cited in the above-mentioned international search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information. 26/02/93

Patent document cited in search report	Publication date	Patent family member(s)		Publication date
GB-A-2236604	10-04-91	US-A-	5138712	11-08-92
		CA-A-	2025434	03-04-91
		JP-A-	4100148	02-04-92
US-A-5023907	11-06-91	None		
EP-A-0332304	13-09-89	US-A-	4937863	26-06-90
		JP-A-	2014321	18-01-90

EPO FORM P0479

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 5 : G06F 11/34, H04L 9/00

A1

(11) International Publication Number: WO 93/01550

(43) International Publication Date: 21 January 1993 (21.01.93)

(21) International Application Number: PCT/US92/05387

(22) International Filing Date: 30 June 1992 (30.06.92)

(30) Priority data:

724,180	1 July 1991 (01.07.91)	US
907,934	29 June 1992 (29.06.92)	US

(81) Designated States: AT, AU, BB, BG, BR, CA, CH, CS, DE, DK, ES, FI, GB, HU, JP, KP, KR, LK, LU, MG, MN, MW, NL, NO, PL, RO, RU, SD, SE, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IT, LU, MC, NL, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, SN, TD, TG).

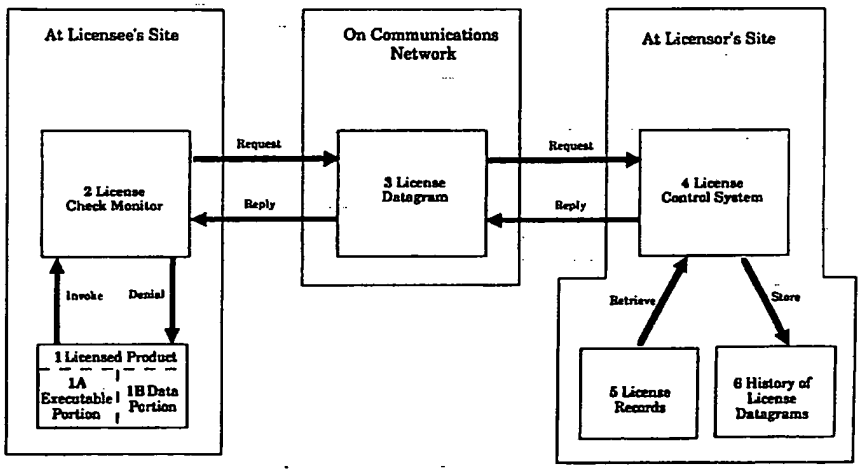
Published
With international search report.

(71) Applicant: INFOLOGIC SOFTWARE, INC. [US/US]; 1223 Peoples Avenue, Suite 5405, Troy, NY 12180 (US).

(72) Inventor: GRISWOLD, Gary, N. ; 1937 Regent Street, Schenectady, NY 12309 (US).

(74) Agents: LAZAR, Dale, S. et al.; Cushman, Darby & Cushman, Ninth Floor, 1100 New York Avenue, N.W., Washington, DC 20005-3918 (US).

(54) Title: LICENSE MANAGEMENT SYSTEM AND METHOD



(57) Abstract

A license management system and method for recording (6) the use of licensed product (1), and for controlling (4) its use. A licensed product invokes a license check monitor (2) at regular time intervals. The monitor generates request datagrams (3) which identifiy the licensee and the product and sends the request datagrams over a communications facility to a license control system (4). The license control system maintains a record (6) of the received datagrams, and compares the received datagrams to data stored in its licensee database (5). Consequently, the license control system (4) transmits reply datagrams with either a denial or an approval message. The monitor (2) generates its own denial message if its request datagrams are unanswered after a predetermined interval of time. The datagrams are counted at the control system to provide billing information.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	FI	Finland	ML	Mali
AU	Australia	FR	France	MN	Mongolia
BB	Barbados	GA	Gabon	MR	Mauritania
BE	Belgium	GB	United Kingdom	MW	Malawi
BF	Burkina Faso	GN	Guinea	NL	Netherlands
BG	Bulgaria	GR	Greece	NO	Norway
BJ	Benin	HU	Hungary	PL	Poland
BR	Brazil	IE	Ireland	RO	Romania
CA	Canada	IT	Italy	RU	Russian Federation
CF	Central African Republic	JP	Japan	SD	Sudan
CG	Congo	KP	Democratic People's Republic of Korea	SE	Sweden
CH	Switzerland			SN	Senegal
CI	Côte d'Ivoire	KR	Republic of Korea	SU	Soviet Union
CM	Cameroon	LI	Liechtenstein	TD	Chad
CS	Czechoslovakia	LK	Sri Lanka	TG	Togo
DE	Germany	LU	Luxembourg	US	United States of America
DK	Denmark	MC	Monaco		
ES	Spain	MG	Madagascar		

LICENSE MANAGEMENT SYSTEM AND METHOD

BACKGROUND

Field of the Invention

The present invention generally relates to systems for managing licenses of products such as computer software, video games, CD-ROM information, movies and other video products, music and other audio products, multimedia products, and other systems for up-to-date recording of actual usage of such a licensed product to enable efficient billing therefor.

Description of Related Art

Licenses for information products such as computer software, music, video products and the like usually provide licensees with limited rights. The licenses may restrict sites of use, duration of use, or number of concurrent uses of the products. The licenses also may limit the use of the products depending on currentness of licensee's payments. However, enforcing the conditions of the licenses is difficult, because, in general, the licensed products may be easily copied or "pirated" and used without the licensor's knowledge.

Compliance with limited license rights has been encouraged with copy protection. Known methods of computer software copy protection include putting a

physical hole or mark on the diskette containing a product, or placing data on the diskette in a location where no data is expected. A disk with an illegally copied software product usually would not contain the marks. At the beginning of its operation, a copy-protected, but illegally copied software product would search its own diskette for the marks. Upon failing to detect the marks, the software would abort from its normal procedures.

Most software products sold today do not have such copy protection, partly because copy protection renders legitimate duplication of copy protected software difficult, but not impossible. Copy protection frustrates the making of legitimate copies, while not eliminating unauthorized copying. Many software publishers have experienced higher sales by eliminating copy protection schemes.

Another method for enforcing limited licensing rights of computer software is described in U.S. patent No. 4,932,054 to Chou. Chou describes a "coded filter" hardware device which is plugged into a port of a computer. The "coded filter" outputs an authorization control code when a predetermined control code is sent to it. The licensed software functions properly only if the "coded filter" transmits the correct authorization control code to the software.

While devices such as described by Chou have existed for several years, they have not been well accepted by the market. Since the device is attached to the outside of a computer, it can easily be lost or stolen, preventing the use of licensed software. In addition, if a licensee purchased a number of software

products, each of which used Chou's protection scheme, the licensee would collect a stack of "coded filters."

Hershey, in U.S. patent No. 4,924,378, describes a method for limiting the number of concurrent uses of a licensed software product. Each workstation of a network has a license storage area in its local memory. License Management System (LMS) daemons are provided in the network in a number corresponding to the permissible number of concurrent uses of the software product. To use the software, a work station stores a daemon in its license storage area. If all daemons are in use, no further work stations may use the software.

Robert et al., in U.S. patent No. 4,937,863, describe a similar invention. This invention includes a license management facility which accesses a database of license information related to licensed computer software programs. When a user attempts to use a licensed program, the license management facility first checks the database. Access to the licensed product is prevented if licensing conditions related to the product are not satisfied (e.g., expiration of licensing dates, etc).

While the Robert et al. and Hershey patents show effective techniques for controlling licensed computer software, each also reveals components that cannot be easily managed by an average user. A system manager, or someone with special access privileges to the internals of a machine, must install the licensed software. This hinders the distribution of the software.

Licensable products other than computer software have not generally been copy-protected. For example,

video tapes can be easily copied by anyone with two VCR machines, and audio tapes and music CDs can be easily copied to tape. Computer CD-ROMs can be copied to magnetic disk; however, their large information storage capacity relative to that of magnetic disks makes this a very expensive proposition. The introduction of digital audio tape is being delayed, because some view its ability to easily produce very high quality copies as a threat to music royalties.

Hellman, in U.S. patent No. 4,658,093, describes means to bill by usage. This is accomplished via communication of an encrypted authorization code from a licensor to a base unit at the licensee's site. The encrypted authorization code contains information related to an identification of the base unit, a number of uses requested, and a random or non-repeating number; however, implementation of Hellman's scheme requires a "base unit", such as a computer, video game unit, record player, video recorder, or video disk player, with a unique identification number. The requirement is difficult to satisfy, because, at the present, only a fraction of such systems on the market have an internally readable serial number for identification. In addition, vendors of these systems provide no guarantees for the uniqueness of any given device's serial number. Furthermore, an internal serial number can change when hardware maintenance is performed on the device. Also, Hellman's approach requires that an identical copy of each software product be stored at the authorization site. These copies are used in the generation of unique keys. The unstated assumption that all copies of a specific version of a software

product are identical is unrealistic. Minor bug fixes to software are often made without generating a new version of the product. Also, some software products, such as those which run on Macintosh computers, are self-modifying.

While Hellman's invention counts each use of the software, it does not monitor the duration of use. Thus, Hellman's system would not be able to bill for extensive use of licensed software if the software were continuously operated. Finally, while Hellman suggests the inclusion of an automated communication system as part of his invention, he does not disclose how this communication system could be implemented. Instead, he mentions non-automated use of telephone and mail. In summary, Hellman's patent is an interesting discussion of cryptographic techniques, but it does not provide a practical, real-world implementation of those techniques.

Shear, in U.S. Patent No. 4,977,594, describes a system and method to meter usage of distributed databases such as CD-ROM systems. The method describes a hardware module which must be part of the computer used to access the distributed database. This module retains records of the information viewed. Once the module storage is filled, the module must be removed and delivered to someone who will charge for the usage recorded therein and set the module back to zero usage. Like Hellman's method, this method requires a hardware module which must be incorporated within the computer so the system can control user access. No database publisher will be able to use this method until there are a very large number of units containing such modules. Hardware manufacturers

will be hesitant to include the module in the design of their computers until there is sufficient demand from customers or publishers for this system. The method and apparatus according to the present invention can be implemented entirely in software and hence does not require special, dedicated computer subsystems.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a license management system and method which can ensure that a licensed product is used only on machines under which it is licensed.

It is another object of the present invention to provide a license management system and method which may terminate access to a licensed product once its license has expired.

It is yet another object of the present invention to provide a license management system and method which may terminate access to a licensed product when payment for a license is overdue.

It is a further object of the present invention to provide a license management system and method which can limit the number of concurrent uses of a licensed product.

It is yet another object of the present invention to provide a license management system and method which can bill licensees for the duration of actual usage of a licensed product.

The present invention provides an advantageous feature of quickly and effectively implementing license agreements between a licensor and licensee.

The present invention provides another advantageous feature of allowing logic used to control licenses to be easily changed.

The present invention provides yet another advantageous feature of detecting, at the licensor's site, many types of attempts to alter the license management system.

The present invention provides a further advantageous feature of permitting anyone without special access privileges to install a licensed product.

In the present invention, a licensed product generates request "datagrams," messages transmitted over a communications network. The request datagrams are sent to the licensor's site. At the licensor's site the datagram is compared to information stored in a license database. After the comparison, a reply datagram is sent to the licensee. Upon receiving the reply datagram, the licensed product reacts in accordance with the instructions therewithin. For example if a reply datagram contained a "denial," the licensed product would display an appropriate message to the user and then suspend further execution of its programs.

In the present invention, the licensed product is implemented on a network node attached to a communications network that includes the licensor. The network node may be a computer, a CD-ROM player, a tele-computer or other multimedia machine, or any other appropriate device. The node may also be an intelligent type of consumer electronic device used for presenting information, such as an intelligent television, VCR, videodisk player, music CD player,

audio tape player, telephone or other similar device. Further, the communications network may be any two-way network such as a computer network, telephone network, a cellular telephone network or other wireless network, a two-way cable TV network, or any other equivalent system.

Should the user detach the node from the network, the licensed product will fail to receive reply datagrams. Upon several failures to receive reply datagrams, the licensed product will generate its own denial.

After a request datagram has been sent out, a user may be permitted to use the licensed product for a limited duration. This feature may be necessary because of the delays in network communications. When networks are sufficiently fast, use of a licensed product can be postponed until the reply datagram is received.

In the preferred embodiment of the present invention, licensees' network addresses are used to identify the licensees. Other embodiments may use a licensed product serial number or hardware serial numbers for the identification.

A licensed product as in the present invention generates a request datagram after each period of product use. The number of request datagrams received by the licensor can be used to bill the licensee. For example, if datagrams are sent after every hour of product use, the licensee will be billed for the amount equal to the number of request datagrams received by the licensor multiplied by the hourly rate.

The embodiments of the present invention may incorporate a query system at a licensor's site for reporting on problem datagrams. This would allow the licensors to take appropriate actions in accordance with problems associated with each datagram.

BRIEF DESCRIPTION OF THE DRAWINGS

These and other objects and advantages of this invention will become more apparent and more readily appreciated from the following detailed description of the presently preferred exemplary embodiment of the invention, taken in conjunction with the accompanying drawings, of which:

FIGURE 1 is a general block diagram of the preferred exemplary embodiment of the present invention;

FIGURE 2 shows representative diagrams of the contents and formats of data at licensee's site, contained in datagrams, and at licensor's site;

FIGURE 3 illustrates a sequence of representative operations executed at the licensee's site and at the licensor's site, together with required inputs for the execution of the operations and with outputs produced therefrom;

FIGURE 4 illustrates a sequence of representative operations to send a request datagram, together with required inputs for the execution of the operations and with outputs produced therefrom;

FIGURE 5 illustrates a sequence of representative operations when a reply datagram is overdue, together with required inputs for the execution of the operations and with outputs produced therefrom;

FIGURE 6 shows a sequence of representative operations to process a reply datagram, together with required inputs for the execution of the operations and with outputs produced therefrom;

FIGURE 7 shows a sequence of representative operations to generate an authorization code, together with required inputs for the execution of the operations and with outputs produced therefrom; and

FIGURE 8 shows a sequence of representative operations to send a reply datagram, together with required inputs for the execution of the operations and with outputs produced therefrom.

DETAILED DESCRIPTION OF THE PRESENTLY PREFERRED EXEMPLARY EMBODIMENT

As shown in FIGURE 1, a licensed product 1 is located at a licensee's site. Product 1 may include a data portion 1B and a functional portion 1A such as computer software product or any other kind of information product used to control use of data portion 1B. If data portion 1B is CD-ROM database information, functional portion 1A should enable the licensee to search indexes and display text. If data portion 1B is video information, functional portion 1A should control the display of the video information. For audio information, functional portion 1A should play the audio information. If data portion 1B is an electronic book, functional portion 1A should display and turn pages. The above examples show some of the ways functional portion 1A can control data portion 1B; however, they are hardly exhaustive.

By including in product 1 both information and software which controls the information, product 1 is

an executable product. Non-software information in product 1 is preferably encrypted so that it cannot be easily extracted from the product.

License check monitor 2 sends license datagrams 3 to the licensor and also receives license datagrams 3 from the licensor. License check monitor 2 also prevents further use of product 1 when a datagram 3 containing a "denial" message is received.

License datagrams 3 are messages that describe information related to the use of licensed product 1. Datagrams 3 are sent over a communications network between the licensee and licensor. Initially, the licensee sends a request datagram 3 over the network to the licensor. The licensor then returns a reply datagram containing either an approval or denial. It is also possible to implement the present invention by having the licensor transmit a reply datagram only for approvals.

At the licensor's site, license control system 4 makes licensing decisions by comparing request datagram 3 with license records 5. After the comparison, control system 4 stores information related to request datagram 3 into history of license datagram record 6. It is noted that request datagrams 3 are periodically sent while product 1 is in use. Thus, the history of license datagrams in record 6 provides means for measuring the duration of use of product 1.

Representations of data and records stored at the licensee's site, contained in datagrams, and stored at the licensor's site are illustrated in FIGURE 2. At the licensee's site, network service 7, which handles delivery and transmission of datagrams 3, supplies

network address 8. It is by this address that license control system 4 identifies a location of use of product 1.

Licensed product record 9 is contained within monitor 2. Within the license product record 9 is an identification record 10, which contains the following two items: licensor's network address 11, and product model number 12 that identifies product 1. When a licensor has only one product, or uses different licensor network addresses 11 for each product, product model number 12 may not be needed.

Datagram sent record 13 stores information about the last sent datagram 3. It includes a datagram number 14, which uniquely identifies the last transmitted datagram 3, and the date and time 15 when the last datagram 3 was sent from the licensee's site.

Licensed product record 9 also contains control parameters record 16, which is used for controlling the timing of key events in the communication of license check monitor 2 with license control system 4. Send interval 17 specifies a time interval between each transmission of a new datagram 3 from the licensee to the licensor.

Wait interval 18 is the length of time that monitor 2 waits to receive a reply datagram 3 before resending the same request datagram 3. The duration of this interval depends on the speed of the communications network being used to deliver datagrams 3.

Disconnect allowed interval 19 is the duration of time that monitor 2 allows product 1 to be used without a reply datagram 3 from the licensor. The duration of this interval depends on the reliability

of the communications network. The interval must be long enough to take into consideration network downtime. For example, suppose a message was sent from the licensor and the network went down just afterwards. Disconnect allowed interval 19 should be long enough to allow the network to resume its normal operation and successfully deliver datagrams 3 from the licensor; otherwise, the licensee would be forced to stop using product 1 until the network was operational.

License datagram 3 contains header 20. Header 20 is used during execution of low level communication protocols within the network. Source network address 21 is the network address from where datagram 3 is sent. Destination network address 22 is the network address to where datagram 3 is sent. Additional data may be included in header 20 if required by low level protocols used in delivering datagrams 3.

Data 23, a part of datagram 3, conveys a message, and contains a number of fields. Product model number 24 and datagram number 25 identify product 1 and datagram 3, respectively. It is noted that retransmitted datagrams have an identical datagram number. Duplicate datagrams must be identified at a licensor's site so that they do not all contribute in billing a licensee.

Each datagram number 25 is unique for each request datagram 3 transmitted from the licensee, except for retransmitted datagrams. This allows a reply datagram 3 received by a licensee to be verified as an actual reply to a request datagram 3 from that licensee, as explained below.

Number of processes running 26 is the number of concurrent uses of product 1 at the time datagram 3 is sent. Authorization code 27 is used on reply datagrams 3 to indicate an approval or a denial. Message text 28 contains a message which will be displayed to the user upon a denial.

License database 29 at the licensor's site holds records of information about customers, licenses, and license usage. The types of information within license database 29 of the present embodiment are shown in FIGURE 2. However, a specific license management system may require its license database to hold types of information other than those in FIGURE 2. For example, licensee name and address may be incorporated as a part of a license database 29.

License record 5 contains information on licenses. Licensee network address 30 identifies a precise network node which is licensed to use product 1. If request datagrams are received which do not originate from known licensee network addresses 30, reply datagrams containing denial messages are transmitted. Product model number 31 is the model number of a licensed product. Termination date 32 is the expiration date of a license. When the license of a product is issued for an unlimited duration, termination date 32 should reflect a date very far into the future, relative to the licensing date.

The present embodiment allows licenses to be paid for in a lease-like or rental fashion. If a licensee were to rent or lease product 1, paid through date 33 would reflect the date through which the licensee has paid for using the product. Grace period 34 is the time interval for which the licensee is allowed to be

delinquent before services are disconnected. Grace period 34 would reflect a very large time interval if the license is not of a lease-like or rental type. When the license provides for a limit on the number of concurrent uses of a product 1, number of processes licensed 35 contains the limiting number. When the license does not provide for such a limit, number of processes 35 should be a very large number.

History of license datagrams 6 is an archive of datagrams 3 received from the licensee.

FIGURE 3 illustrates operations executed at the licensee's site and at the licensor's site. An overview of the processing at the licensee's site is described by steps 101.0 to 106.0, and an overview of the processing at the licensor's site is described by steps 107.0 to 110.0.

At the licensee's site, at step 101.0, product 1 invokes monitor 2. This is accomplished by first establishing monitor 2 as a handler for a timer expiration interrupt signal and for received datagrams 3. Next, a timer is set with a very short time to cause an initial call to monitor 2. At step 102.0, monitor 2 computes a time 36 since the last datagram was sent by determining the difference between the current date and sent time and date and time 15 that a datagram was last sent from the licensee's site. When product 1 commences execution, datagram sent date and time 15 is set to "null." Thus, time since send 36 is very large at the beginning of the monitor's execution. At step 103.0, time since send 36 is compared to send interval 17. If time since send 36 is greater than send interval 17, then a request datagram is transmitted, per the steps described in

FIGURE 4. Step 104.0 first checks if a reply to the last datagram has arrived and if wait interval 18 has expired. If a reply has not arrived and the wait interval has expired, steps 104.1-104.3 (FIGURE 5) are executed. Step 105.0 processes authorization code 27 in a reply when the reply is received, in accordance with steps 105.1 to 105.5 (FIGURE 6). At step 106.0, product 1 resumes normal execution of its programs until the next interrupt signal is generated.

At the licensor's site, license control system 4 receives and processes datagram 3, in accordance with steps 107.0 to 110.0. Step 107.0 receives request datagram 3. Step 108.0 generates authorization code 27, per steps 108.1 to 108.8 (FIGURE 7). Step 109.0 creates reply datagram 3 and transmits the datagram to the licensee via steps 109.1 to 109.5 (FIGURE 8).

FIGURE 4 shows the procedure which monitor 2 follows for sending request datagram 3 to the licensor. Step 103.1 sets source network address 21 in datagram 3 to the network address 8 of the licensee's location on the network. Step 103.2 sets destination network address 22 to licensor's network address 11. Step 103.3 encrypts product model number 12 for datagram 3. Step 103.4 assigns a unique number to datagram 3, encrypts the number, and stores it as datagram number 14. This number is altered when an entirely new datagram 3 is sent. Datagrams which are retransmitted have the same datagram number 25 as the original. As already explained, this allows license control system 4 to identify duplicate datagrams.

Step 103.5 counts the number of processes using product 1, currently running, encrypts the count, and stores the encryption as the number of processes

running 26. In the UNIX operating system, this procedure could be performed using the command "ps" to obtain a list of current processes, the command "grep" to extract the processes of product 1, and "wc" to count the number of processes. Step 103.6 sets authorization code 27 to number 255 and encrypts the number.

Number 255 indicates that datagram 3 is a request for authorization. Such an indication is needed to guard the present system against the following steps for circumventing the present invention: intercepting outgoing datagrams; and inputting the intercepted datagrams to monitor 2.

Step 103.7 stores the current date and time as sent date & time 15. This date is needed to compute when to send the next datagram 3. Step 103.8 assigns a value to send interval 17, which sets an alarm for invoking monitor 2 to send the next datagram 3. Step 103.9 sends datagram 3.

In the present embodiment a datagram is transmitted via a connectionless datagram service. Methods for transmission are well documented for some networking systems. For example, TCP/IP (Transport Control Protocol/Internet Protocol) includes a connectionless protocol called UDP (User Datagram Protocol). A method for sending a datagram using UDP protocol from a SUN Microsystem computer is documented in a SUN manual titled, Network Programming Guide, in section 9 titled "Transport Level Interface Programming."

Step 103.10 sets another alarm using wait interval 18 for retransmitting datagram 3, if no reply datagram has been received. The alarm causes monitor

2 to be invoked for checking whether a reply datagram 3 has been received. Monitor 2 will transmit a duplicate of the previously transmitted datagram, if no reply has been received. After the execution of step 103.10, "Send License Datagram" procedure returns system control to step 104.0 in FIGURE 3.

FIGURE 5 shows the operation of the "Reply Datagram is Overdue" procedure. Step 104.1 compares time since the last datagram was sent 36 to disconnect allowed interval 19, which, as described above, is the interval that product 1 is allowed to operate even if a reply is overdue. If time since send 36 is smaller than disconnect allowed interval 19, datagram 3 is retransmitted via executing step 103.9 in FIGURE 4. Step 104.2 "disconnects" product 1 from further service, if time since send 36 is greater than disconnect allowed interval 19.

Step 104.2 comprises a sequence of sub-steps 104.2.1-104.2.3. Step 104.2.1 assigns number 5 to authorization code 27 in the current datagram being processed. Value 5 is interpreted by monitor 2 as a denial. Step 104.2.2 sets message text 28 to the following: "A reply from licensor to numerous authorization requests was never received. This product must be connected to a communications network in order to function." Step 104.2.3 transfers system control to step 105.3 in FIGURE 6. Step 105.3 processes the current denial datagram 3 as if it were just received.

Through the execution of steps 104.1-104.3, the present system permits the use of product 1 for a prescribed period of time. After the prescribed

period of time has elapsed, the present system generates a denial.

FIGURE 6 illustrates the steps which monitor 2 follows in processing a reply datagram 3. Step 105.1 decrypts all encrypted data in the received datagram. Step 105.2 compares datagram number 25 with datagram number 14 associated with the last datagram. If datagram number 25 is not equal to datagram number 14, step 105.2 ignores the current datagram and transfers procedural control to step 103.9 (FIGURE 4) in order to resend the last transmitted datagram. After disconnect allowed interval 19 elapses, monitor 2 generates a denial.

In essence, step 105.2 guards against the circumvention of the present invention via: (1) intercepting a reply datagram 3 (from the licensor) containing an approval (2) storing the reply datagram 3; and (3) inputting the stored datagram to monitor 2.

If the execution of step 105.2 does not transfer its procedural control to step 105.3, and if authorization control 27 is not zero (indicating an unqualified authorization has not been received), step 105.3 processes authorization code 27 via steps 105.3.1 to 105.3.3. Step 105.3.1 retrieves message text 28 from datagram 3. If message text 28 is null, then the current datagram 3 is ignored, and monitor 2 resends the last transmitted datagram 3. Step 105.3.1 further protects the present system from attempts to generate fake datagrams and to feed the fake datagrams to monitor 2 by checking for a proper authorization code of zero.

If message text 28 is not null, step 105.3.2 presents the message 28 to the user on an output

device such as a CRT screen. Step 105.3.3 terminates the current use of product 1. This step may be implemented by subroutine or function call to a simple exit that saves any current user data to a file. Alternatively, product 1 may be designed so that, upon being directed to terminate further execution, it first gives the user an opportunity to save their data.

If authorization code 27 is zero, step 105.4 allows further use of product 1. Step 105.5 returns procedural control to 106.0 on FIGURE 3.

FIGURE 7 shows a sequence of operations within the "Generate Authorization Code" procedure. The procedure produces appropriate authorization code 27 when a request datagram 3 is received at the licensor's site.

Step 108.1 decrypts all encrypted data in the received datagram 3. Using source network address 21 and product model number 24 in the datagram 3, step 108.2 searches the license database 29 for matching licensee network address 30 and product model number 31. If license database 29 does not contain a record of product model number 24 of the product 1 being licensed to the licensee, step 108.3 sets authorization code 27 of its reply datagram 3 to 1 (i.e., the sending node is not a registered address) and authorization is denied.

Step 108.3 prevents copies of product 1 from being installed on multiple nodes independently of whether they are within or outside the licensee's organization. Step 108.3 also prevents the licensee from transporting product 1 from one node to another node without the licensor's approval. This is

important because the two nodes may have different processing capacities, and they may be billable at different rates.

If the date a request datagram is received is later than license termination date 32, step 108.4 sets authorization code 27 to number 2 (i. e., license has expired). Step 108.4 allows the licensor to fix licensing periods, or to determine free trial periods for the use of the product. The licensing period may be extended by resetting license termination date 32 at the licensor's site.

If the date when the datagram is received is later than the paid through date 33 as extended by the grace period 34, step 108.5 sets authorization code 27 to 3 (i.e., payment is past due).

If the number of processes running 26 exceeds a licensed number of concurrent uses of product 1 (at a particular node), then step 108.6 sets authorization code 27 to 4 (i.e. concurrent process usage limit is exceeded).

Step 108.7 sets authorization code 27 to 0 indicating processing can continue. It is noted that steps 108.3-108.7 are a part of a

```
IF (x1) then (y1)
ELSE if (x2) then (y2)
ELSE if (x3) then (y3) ...
```

statement of a procedure (e. g., FORTRAN, PASCAL, C, etc). Thus, only one of the steps 108.3-108.7 is executed. Step 108.7 sets authorization code 27 to 0 (indicating approval of further use) only if steps 108.3-108.6 do not execute the THEN portion of each step. Step 108.7 also stores the received datagram 3 in history of license datagrams 6.

Step 108.8 is the last of authorization processing rules 108.1-108.7. After the execution of steps 108.3-108.7, step 108.8 returns procedural control to step 109.0 in FIGURE 3.

FIGURE 8 illustrates the steps which license control system 4 follows to send reply datagram 3 to the licensee.

Step 109.1 encrypts authorization code 27 and writes the encrypted code into datagram 3. Next, step 109.2 writes message text 28 corresponding to authorization code 27 into datagram 3.

Step 109.2 may be replaced with the following method for relaying proper messages to a product user. Proper messages corresponding to each authorization code is stored in monitor 2 at each licensee's site. Upon reception of a reply datagram 3, monitor 2 would locate within itself the proper message corresponding to the authorization code, and use the message for various purposes. This method would reduce the size of reply datagrams 3. However, if the licensor wanted to implement new denial codes, each product would need to somehow incorporate the new message associated with the new denial code into itself. The list of messages, one of which may be written as message text 28, are as follows:

AUTHORIZATION CODE	TEXT MESSAGE
1	This product is not licensed to run at this location. Please contact the licensor to either license this product, or move an existing license of your organization to this location. Use of this product at this

	location is discontinued until this problem is resolved.
2	Your license on this product has expired. Please contact licensor in order to have your license extended. Use of this product is discontinued until this problem is resolved.
3	Payment on this licensed product is over due and past your grace period. Please have your accounting department send payment in order to continue your license. Use of this product is discontinued until this problem is resolved.
4	Your current use of this licensed product exceeds limits for the number of uses your organization has licensed. Please try again later.
5	A reply from licensor to numerous authorization requests was never received. This product must be connected to a communications network in order to function.
0	Authorization is OK. There is no message.

Step 109.3 swaps source network address 21 and destination network address 22. Step 109.4 transmits datagram 3 back to monitor 2.

At step 109.5, a communications network delivers datagram 3 to monitor 2. Subsequently, procedural control returns to step 107.0 in FIGURE 3 to process the next datagram 3.

Although only a few exemplary embodiments of this invention have been described in detail above, those skilled in the art will readily appreciate that many

modifications are possible in the preferred embodiments without materially departing from the novel teachings and advantages of this invention. For example, product 1 was described as sometimes consisting of information as well as software which controls the information. This approach provides the greatest flexibility, but it is also possible to include the software which controls the information in the networked machine at the licensee's site. In this case, product 1 is split, with part of it on media and part on the licensee's machine. By doing this, some space can be saved on the media containing product 1, but the capabilities of these products will be limited by the standard functions available on these machines.

Also, the presently described embodiment includes a product 1 which is at the licensee's site. This implies that product 1 is on some physical media such as diskette, tape, or CD. However, product 1 can be electronically delivered over communications lines to the licensee and therefore might exist in the memory of the licensee's machine, rather than any physical media. In the case of a product such as music, radio programs and the like, product 1 may even be broadcast to the licensee's site for playback; thus, the product 1 would not even be "resident" in the licensee's machine.

The presently described embodiment allows the licensee to access the licensed product concurrent with the sending and receiving of datagram 3. In this way, the present invention does not inconvenience the legitimate licensee; however, for sensitive licensed products such as confidential information, the license

check monitor 2 can prevent access to the product 1 until an authorization reply datagram 3 is received.

Further, monitor 2 could be realized as an integral part of product 1. Monitor 2 could also be implemented as: 1) a separate process which is the parent process of product 1 (Such a parent process would have the authority to cancel the use of product 1); 2) a single system level task which controls license checking of all products at the licensee's site; and 3) custom logic in a digital integrated circuit (the present invention could be implemented as hardware instead of software).

Also, though the above embodiment has been described as being implemented on a computer system network where operator messages are provided on a CRT monitor or the like, the invention may be practiced on other hardware platforms by incorporating appropriate changes known to those of ordinary skill in the art. For example, in an alternative hardware embodiment such as a music or video playback device, monitor 2 is invoked by the licensee's action of pushing the "play" or similar button, and in a broadcast music application or similar system, the monitor may be invoked simply by turning the device on. The processing of monitor 2 is as described in the presently described embodiment. However, when a denial message is received or generated, monitor 2 must be able to switch "play" to "off".

The presently described embodiment is designed to be used in conjunction with a connectionless UDP (User Datagram Protocol) in the TCP/IP protocol suite as an underlying protocol. However, the present invention could also be realized using a slower,

connectionless protocol such as electronic mail or a variety of connection protocols (e. g., File Transfer Protocols (FTP), Telnet).

It is noted that protocol suites quite different from TCP/IP could be used, such as ISO (International Standards Organization) protocol. In addition, datagrams 3 could be sent over telephone systems with communications protocols such as those specified by CCITT (Consultative Committee on International Telephony and Telegraphy). In this case, telephone numbers could serve as network addresses 21, 22. Communications protocols for wireless communications such as cellular telephone can also be used to send the datagram 3.

Accordingly, all such modifications are intended to be included within the scope of this invention as defined by the following claims.

WHAT IS CLAIMED IS:

1. A method for monitoring the use of a licensed product, comprising the steps of:

generating, at regular time intervals, datagrams including an address in a communications facility, said facility address identifying a licensee;

automatically sending said datagrams from at least one licensee's site over said facility to a licensor's site while said licensed product is in use;

receiving said datagrams at said licensor's site;

storing an indication of receipt of each of said datagrams; and

counting said datagrams from each licensee as an indication of the use by the licensee of said licensed product.

2. A method as in claim 1 further wherein:

said generating step includes the step of incorporating a model number of said product in said datagrams; and

said counting step includes the step of separately counting datagrams for each product model number for each licensee.

3. A method as in claim 1, wherein said generating step includes the step of automatically obtaining said facility address that identifies said licensee from said facility without any data being provided by said licensee.

4. A method for controlling use of a licensed product comprising the steps of:

generating a request datagram including an address in a communications facility, said facility address identifying a licensee;

automatically sending said request datagram from at least one licensee's site over said facility to a licensor's site while said licensed product is in use;

receiving said request datagram at said licensor's site;

comparing said received request datagram with rules and license data at said licensor's site to determine if use of said licensed product is authorized;

sending a reply authorizing datagram to said licensee's site if use of said licensed product is approved; and

receiving said reply authorizing datagram at said licensee's site and denying the use of said product when no reply authorizing datagram is received.

5. A method as in claim 4, wherein:

said generating step includes the step of incorporating a model number of said product in said datagram;

said comparing step includes the step of comparing said rules and license data for a particular model number; and

said sending step includes the step of transmitting said reply datagram for each product model number.

6. A method as in claim 4, wherein said generating step includes the step of automatically obtaining said facility address that identifies said licensee from said facility without any data being provided by said licensee.

7. A method as in claim 4 further comprising the step of sending a reply denial datagram if use of said licensed product is not approved as determined in said comparing step, said step of automatically sending said request datagram from a licensee's site including the step of resending said request datagram if neither a reply authorizing datagram nor a reply denial datagram is received from said licensor's site within a predetermined time from sending said request datagram from said licensee's site.

8. A method as in claim 4, wherein said step of automatically sending said request datagram from said licensee's site includes the step of sending a request datagram at regular time intervals.

9. A method as in claim 4, wherein:

said generating step includes the step of providing a datagram identification code within said datagram;

said reply datagram sending step includes the step of inserting the same datagram identification code in said reply datagram; and

said reply receiving step rejects said reply authorizing datagram if the datagram identification code included in said reply authorizing datagram does

not match the datagram identification code included in said request datagram.

10. A method as in claim 4, wherein:

said comparing step includes the step of comparing said facility address that identifies said licensee with a list of valid licensee addresses to determine if said facility address is a valid address; and

said reply authorizing datagram is not sent if said facility address that identifies said licensee is not valid.

11. A method as in claim 10 further comprising the step of sending a reply denial datagram if said facility address that identifies said licensee is not valid.

12. A method as in claim 4, wherein:

said comparing step includes the step of comparing a license expiration date with a date at which said datagram is received; and

said reply authorizing datagram is not sent if the license expiration date is later than the date at which said datagram is received.

13. A method as in claim 12, further comprising the step of sending a reply denial datagram if the license expiration date is later than the date at which said datagram is received.

14. A method as in claim 4, wherein:

said comparing step includes the step of checking currentness of payments from said license; and

said reply authorizing datagram is not sent if payment is overdue.

15. A method as in claim 14, further comprising the step of sending a reply denial datagram if payment is overdue.

16. A method as in claim 4, wherein:

said generating step includes the step of incorporating in said datagram data indicative of the number of processes currently using said product at said licensee's site;

said comparing step includes the step of comparing the number of processes using said product at the licensee's site to an authorized number; and

said reply authorizing datagram is not sent if said number of processes using said product exceeds said authorized number.

17. A method as in claim 16, further comprising the step of sending a reply denial datagram if said number of processes using said product exceeds said authorized number.

18. A method as in claim 4, wherein said sending step includes the steps of sending said reply authorizing datagram when use of said product is approved and sending a reply denial datagram when use of said product is not approved, said receiving step

denying use of said product when said reply denial datagram is received.

19. A method as in claim 18, wherein said receiving and denying step denies use of said product when neither a reply authorizing datagram nor a reply denial datagram is received within a predetermined time after said request datagram is sent.

20. A method as in claim 18, further comprising the step of indicating, at a licensee's site, a reason for denial when said reply denial datagram is received.

21. A method as in claim 4, wherein:

said licensed product comprises an executable portion and a data portion; and

said method further comprises a step of controlling use of said data portion with said executable portion.

22. A method as in claim 4 further comprising a step of allowing use of said licensed product before a reply datagram is received.

23. A system for controlling licensed product comprising:

a communications facility to which at least one licensee having a license for operating a licensed product from the licensor is connected;

monitoring means, connected to said facility at a site of each said licensee, for generating a request datagram including an address of said licensee

on said facility and transmitting said request datagram over said facility to a site of said licensor, and for receiving and processing a reply datagram; and

controlling means, connected to said facility at said licensor's site, for receiving said request datagram, comparing said request datagram with rules and license data to determine if use of said licensed product is authorized and sending a reply authorizing datagram to said licensee's site if use of said product is approved; and

said monitoring means including means for denying use of said licensed product when no reply authorizing datagram is received.

24. A system as in claim 23, wherein:

said monitoring means sends request datagrams at regular time intervals during use of said licensed product; and

said controlling means further comprises means for counting said request datagrams received at said controlling means and means for computing an amount to be billed to said licensee in response to said counting.

25. A system as in claim 23 wherein:

said monitoring means incorporates a model number for said product in said request datagram; and

said controlling means comprises means for counting datagrams for each product model number for each licensee, in order to compute an amount to be billed to each licensee.

26. A system as in claim 23, wherein said monitoring means automatically obtains said facility address of said licensee from said facility without any input from said licensee.

27. A system as in claim 23, wherein:

said controlling means sends a reply denial datagram to said licensee's site if use of said product is not approved; and

said monitoring means resends said request datagram if no reply authorizing datagram and no reply denial datagram is received within a predetermined period of time after said requesting datagram is sent.

28. A system as in claim 23, wherein said monitoring means transmits request datagrams at predetermined time intervals.

29. A system as in claim 23, wherein:

said monitoring means incorporates a unique identification code in said request datagram;

said controlling means incorporates the same request datagram identification code in said reply authorizing datagram; and

said monitoring means rejects any reply authorizing datagram which does not include the same identification code as included in said request datagram.

30. A system as in claim 23, wherein said controlling means compares said facility address of said licensee with a list of valid licensee facility addresses and does not generate a reply authorizing

datagram if said facility address of said licensee is not valid.

31. A system as in claim 30, wherein said controlling means sends a reply denial datagram when said facility address is not valid.

32. A system as in claim 23, wherein said controlling means compares an expiration date of a license of said product with a date at which said request datagram is received by said controlling means, and does not generate a reply authorizing datagram, thus denying use of said product, if the license expiration date is earlier than the date at which said request datagram is received.

33. A system as in claim 32, wherein said controlling means sends a reply denial datagram if the license expiration date is earlier than the date at which said request datagram is received.

34. A system as in claim 23, wherein said controlling means generates a reply authorizing datagram, thus denying use of said product, if a payment for the use of said product is overdue.

35. A system as in claim 34, wherein said controlling means sends a reply denial datagram if payment for the use of said product is overdue.

36. A system as in claim 23, wherein:

said monitoring means includes in said request datagram data indicative of the number of

processes, at a licensee's site, currently using said product; and

said controlling means does not generate a reply authorizing datagram, thus denying a use of said product, if more than a predetermined number of processes using said product are running at the licensee's site.

37. A system as in claim 36, wherein said controlling means sends a reply denial datagram if more than said predetermined number of processes using said product are running at the licensee's site.

38. A system as in claim 23, wherein said controlling means sends a reply denial datagram if use of said product is not approved.

39. A system as in claim 38, wherein said monitoring means denies use of said licensed product when no reply authorizing datagram and no reply denial datagram is received within a predetermined time from the sending of said request datagram.

40. A system as in claim 38, further comprising means for indicating, at a licensee's site, a reason for denial when said reply denial datagram is received.

41. A system as in claim 23, wherein:

said licensed product comprises an executable portion and a data portion; and

said system further comprises means for controlling use of said data portion with said executable portion.

42. A system as in claim 41, wherein said data portion controlling means is disposed within said executable portion.

43. A system as in claim 41, wherein said data portion controlling means comprises a first partial controlling means disposed within said executable portion and a second partial controlling means disposed within said monitoring means.

44. A system as in claim 23, wherein said monitoring means includes means for permitting use of said licensed product before a reply datagram is received.

45. A system for monitoring product comprising:

a communications facility to which at least one licensee having a license for operating a licensed product from a licensor is connected;

monitoring means, connected to said facility at a site of each said licensee, for generating datagrams including an address of said licensee on said facility and transmitting said datagrams at periodic intervals over said facility to a site of said licensor; and

control means, connected to said facility at said licensor's site, for receiving said request datagrams, storing an indication of receipt of each of said datagrams and counting said datagrams from each

licensee as an indication of the use by the licensee of said licensed product.

46. A system as in claim 45, wherein said monitoring means automatically obtains said facility address of said licensee from said facility without any input from said licensee.

47. A system as in claim 45, wherein:

said monitoring means incorporates a product model number in said request datagrams; and

said controlling means separately counts request datagrams for each product model number for each licensee.

48. A method for monitoring the use of a licensed product comprising the steps of:

generating, at regular time intervals, datagrams including an address in a communications facility, said facility address identifying a licensee; and

automatically sending said datagrams from at least one licensee's site over said communications facility to a licensor's site while said licensed product is in use.

49. A method as in claim 48 further wherein:

said generating step includes the step of incorporating a model number of said product in said datagrams.

50. A method as in claim 48, wherein said generating step includes the step of automatically

obtaining said facility address that identifies said licensee from said communications facility without any data being provided by said licensee.

51. A method for controlling use of a licensed product comprising the steps of:

generating a request datagram including a facility address that identifies a licensee in a communications facility;

automatically sending said request datagram from a licensee's site over said communications facility to a licensor's site while said licensed product is in use; and

receiving a reply authorizing datagram at said licensee's site and denying the use of said product when no reply authorizing datagram is received.

52. A method as in claim 51 wherein:

said generating step includes the step of incorporating a model number of said product in said datagram.

53. A method as in claim 51, wherein said generating step includes the step of automatically obtaining said facility address that identifies said licensee from said communications facility without any data being provided by said licensee.

54. A method as in claim 51, wherein:

said reply datagram is one of at least a reply authorization datagram and a reply denial datagram; and

said step of automatically sending said request datagram from a licensee's site includes a step of resending said request datagram if neither a reply authorizing datagram nor a reply denial datagram is received within a predetermined time from sending said request datagram from said licensee's site.

55. A method as in claim 51, wherein said step of automatically sending said request datagram from said licensee's site includes the step of sending a request datagram at regular time intervals.

56. A method as in claim 51, wherein:

said generating step includes the step of providing a datagram identification code within said datagram; and

said reply receiving step rejects said reply authorizing datagram if the datagram identification code included in said reply authorizing datagram does not match the datagram identification code included in said request datagram.

57. A method as in claim 51, wherein:

said generating step includes the step of incorporating in said datagram data indicative of the number of processes currently using said product at said licensee's site.

58. A method as in claim 51, further comprising the steps of:

receiving a reply denial datagram; and

displaying, at a licensee's site, a reason for denial when said reply denial datagram is received.

59. A method as in claim 51, wherein:

said licensed product comprises an executable portion and a data portion; and

said method further comprises a step of controlling use of said data portion with said executable portion.

60. A method as in claim 51 further comprising a step of allowing use of said licensed product before a reply datagram is received.

61. A system for controlling a licensed product comprising:

a communications facility to which at least one licensee is connected;

monitoring means, connected to said communications facility at a site of each said licensee, for generating a request datagram including an address of said licensee on said communications facility and transmitting said request datagram over said communications facility, and for receiving and processing a reply authorizing datagram; and

means for denying use of said product when no reply authorizing datagram is received.

62. A system as in claim 61, wherein:

said monitoring means sends request datagrams at regular time intervals during use of said licensed product.

63. A system as in claim 61 wherein:

said monitoring means incorporates a model number for said product in said request datagram.

64. A system as in claim 61, wherein said monitoring means automatically obtains said facility address of said licensee from said communications facility without any input from said licensee.

65. A system as in claim 61, wherein:

said monitoring means resends said request datagram if no reply authorizing datagram and no reply denial datagram is received within a predetermined period of time after said requesting datagram is sent.

66. A system as in claim 61, wherein said monitoring means transmits request datagrams at predetermined time intervals.

67. A system as in claim 61, wherein:

said monitoring means incorporates a unique identification code in said request datagram; and

said monitoring means rejects any reply authorizing datagram which does not include the same identification code as included in said request datagram.

68. A system as in claim 61, wherein:

said monitoring means includes in said request datagram data indicative of the number of processes, at a licensee's site, currently using said product.

69. A system as in claim 61, wherein:

said monitoring means denies use of said licensed product when no reply authorizing datagram and no reply denial datagram is received within a predetermined time from the sending of said request datagram.

70. A system as in claim 61, further comprising means for indicating, at a licensee's site, a reason for denial when a reply denial datagram is received.

71. A system as in claim 61, wherein:

said licensed product comprises an executable portion and a data portion; and

said system further comprises means for controlling use of said data portion with said executable portion.

72. A system as in claim 71, wherein said data portion controlling means is disposed within said executable portion.

73. A system as in claim 71, wherein said data portion controlling means comprises a first partial controlling means disposed within said executable portion and a second partial controlling means disposed within said monitoring means.

74. A system as in claim 61, wherein said monitoring means includes means for permitting use of said licensed product before a reply datagram is received.

75. A system for monitoring a licensed product comprising:

a communications facility to which at least one licensee is connected;

monitoring means, connected to said communications facility at a site of each said licensee, for generating datagrams including an address of said licensee on said communications facility and transmitting said datagrams at periodic intervals over said communications facility.

76. A system as in claim 75, wherein said monitoring means automatically obtains said communications facility address of said licensee from said communications facility without any input from said licensee.

77. A system as in claim 75, wherein:

said monitoring means incorporates a product model number in said request datagrams.

78. A method for monitoring the use of a licensed product comprising the steps of:

receiving datagrams at a licensor's site on a communications facility having at least one licensee's site thereon, said datagrams being generated at regular time intervals and including a facility address that identifies a licensee in said communications facility;

storing an indication of receipt of each of said datagrams; and

counting said datagrams as an indication of the use of said licensed product.

79. A method as in claim 78 further wherein:

said datagrams include a model number of each product; and

said counting step includes the step of separately counting datagrams for each product model number for each licensee.

80. A method for controlling use of a licensed product comprising the steps of:

receiving a request datagram at a licensor's site on a communications facility having at least one licensee's site thereon, said request datagram including a facility address identifying a licensee and being automatically sent over said communications facility to said licensor's site while said licensed product is in use;

comparing said received request datagram with rules and license data at said licensor's site to determine if use of said licensed product is authorized; and

sending a reply authorizing datagram if use of said licensed product is approved.

81. A method as in claim 80 wherein:

said datagrams include a model number of said product;

said comparing step includes the step of comparing said rules and license data for a particular model number; and

said sending step includes the step of transmitting said reply datagram for each product model number.

82. A method as in claim 80 further comprising the step of sending a reply denial datagram if use of said licensed product is not approved as determined in said comparing step.

83. A method as in claim 80, wherein:

said datagrams include a datagram identification code; and

said reply datagram sending step includes the step of inserting the same datagram identification code in said reply datagram.

84. A method as in claim 80, wherein:

said comparing step includes the step of comparing said facility address that identifies said licensee with a list of valid licensee addresses to determine if said facility address is a valid address; and

said reply authorizing datagram is not sent if said facility address that identifies said licensee is not valid.

85. A method as in claim 84 further comprising the step of sending a reply denial datagram if said facility address that identifies said licensee is not valid.

86. A method as in claim 80, wherein:

said comparing step includes the step of comparing a license expiration date with a date at which said datagram is received; and

said reply authorizing datagram is not sent if the license expiration date is later than the date at which said datagram is received.

87. A method as in claim 86, further comprising the step of sending a reply denial datagram if the license expiration date is later than the date at which said datagram is received.

88. A method as in claim 80, wherein:

said comparing step includes the step of checking currentness of payments from said license; and

said reply authorizing datagram is not sent if payment is overdue.

89. A method as in claim 88, further comprising the step of sending a reply denial datagram if payment is overdue.

90. A method as in claim 80, wherein:

said datagrams include data indicative of the number of processes currently using said product at said licensee's site;

said comparing step includes the step of comparing a number of processes using said product to an authorized number; and

said reply authorizing datagram is not sent if said number of processes using said product exceeds said authorized number.

91. A method as in claim 90, further comprising the step of sending a reply denial datagram if said

number of processes using said product exceeds said authorized number.

92. A method as in claim 80, wherein said sending step includes the steps of sending said reply authorizing datagram when use of said product is approved and sending a reply denial datagram when use of said product is not approved.

93. A system for controlling a licensed product comprising:

a communications facility to which at least one licensee and a licensor are connected at a licensee's site and at a licensor's site, respectively; and

controlling means, connected to said communications facility at said licensor's site, for: receiving a request datagram, said request datagram including an address of said licensee on said communications facility and being transmitted over said communications facility to a site of said licensor; comparing said request datagram with rules and license data to determine if use of said licensed product is authorized; and sending a reply authorizing datagram to said licensee's site if use of said product is approved.

94. A system as in claim 93, wherein:

said request datagrams are sent at regular time intervals during use of said licensed product; and

said controlling means comprises means for counting said request datagrams received at said

controlling means and means for computing an amount to be billed to said licensee in response to said counting.

95. A system as in claim 93 wherein:

said datagrams include a model number for said product; and

said controlling means comprises means for counting datagrams for each product model number for each licensee, in order to compute an amount to be billed to each licensee.

96. A system as in claim 93, wherein:

said controlling means sends a reply denial datagram to said licensee's site if use of said product is not approved.

97. A system as in claim 93, wherein:

said datagrams include a unique identification code; and

said controlling means incorporates the same request datagram identification code in said reply authorizing datagram.

98. A system as in claim 93, wherein said controlling means compares said facility address of said licensee with a list of valid licensee facility addresses and does not generate a reply authorizing datagram if said facility address of said licensee is not valid.

99. A system as in claim 98, wherein said controlling means sends a reply denial datagram when said facility address is not valid.

100. A system as in claim 93, wherein said controlling means compares an expiration date of a license of said product with a date at which said request datagram is received by said controlling means, and does not generate a reply authorizing datagram, thus denying use of said product, if the license expiration date is earlier than the date at which said request datagram is received.

101. A system as in claim 100, wherein said controlling means sends a reply denial datagram if the license expiration date is earlier than the date at which said request datagram is received.

102. A system as in claim 93, wherein said controlling means generate a reply authorizing datagram, thus denying use of said product, if a payment for the use of said product is overdue.

103. A system as in claim 102, wherein said controlling means sends a reply denial datagram if payment for the use of said product is overdue.

104. A system as in claim 93, wherein:

said datagrams include data indicative of the number of processes, at a licensee's site, currently using said product; and

said controlling means does not generate a reply authorizing datagram, thus denying a use of said

product, if more than a predetermined number of processes using said product are running at the licensee's site.

105. A system as in claim 104, wherein said controlling means sends a reply denial datagram if more than said predetermined number of processes using said product are running at the licensee's site.

106. A system as in claim 93, wherein said controlling means sends a reply denial datagram if use of said product is not approved.

107. A system as in claim 93, wherein:

said licensed product comprises an executable portion and a data portion; and

said system further comprises means for controlling use of said data portion with said executable portion.

108. A system as in claim 107, wherein said data portion controlling means is disposed within said executable portion.

109. A system for monitoring a licensed product comprising:

a communications facility to which at least one licensee and a licensor are connected at a licensee's site and at a licensor's site, respectively; and

control means, connected to said communications facility at a licensor's site, for: receiving request datagrams, said request datagrams

including an address of said licensee on said communications facility and being transmitted at periodic intervals over said communications facility to said licensor's site; storing an indication of receipt of each of said datagrams; and counting said datagrams from each licensee as an indication of the use by the licensee of said licensed product.

110. A system as in claim 110, wherein:

said request datagrams include a product model number; and

said controlling means separately counts request datagrams for each product model number for each licensee.

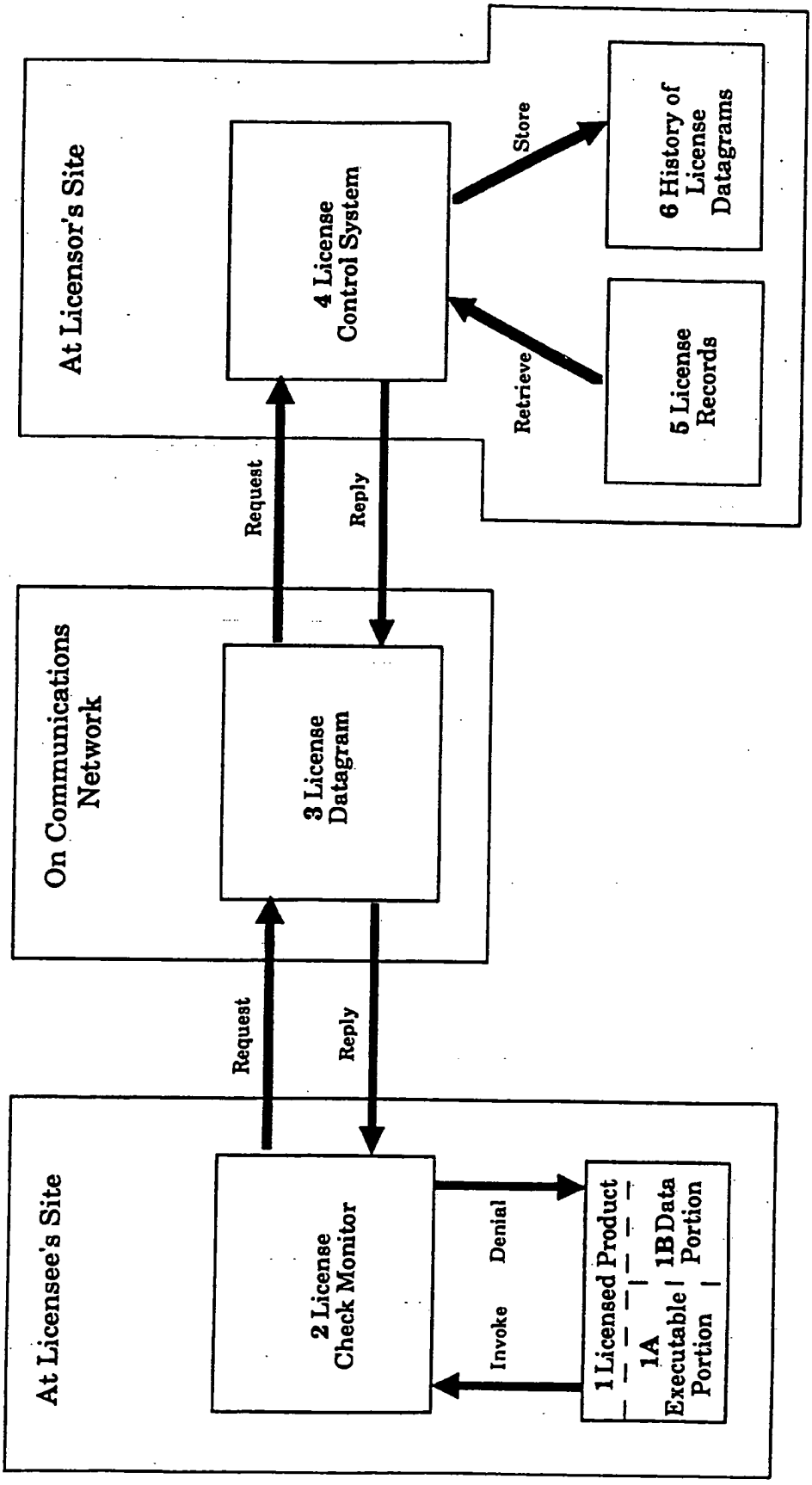

At Licensee's Site
2 License Check Monitor
1 Licensed Product
1A Executable Portion
1B Data Portion
Invoke
Denial
On Communications Network
3 License Datagram
Request
Reply
Request
Reply
At Licensor's Site
4 License Control System
5 License Records
6 History of License Datagrams
Retrieve
Store

FIG. 1

FIG. 2

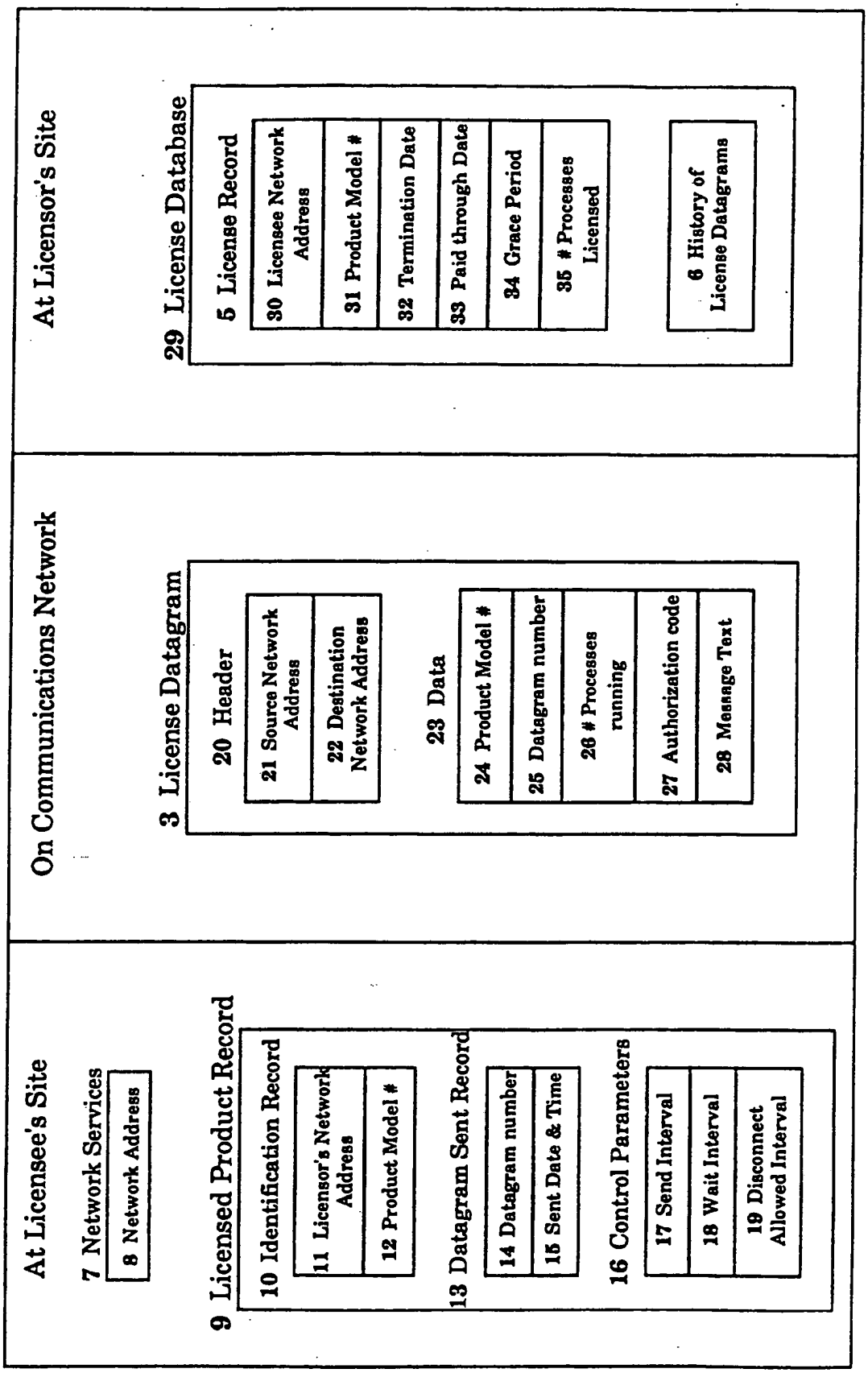


FIG. 3

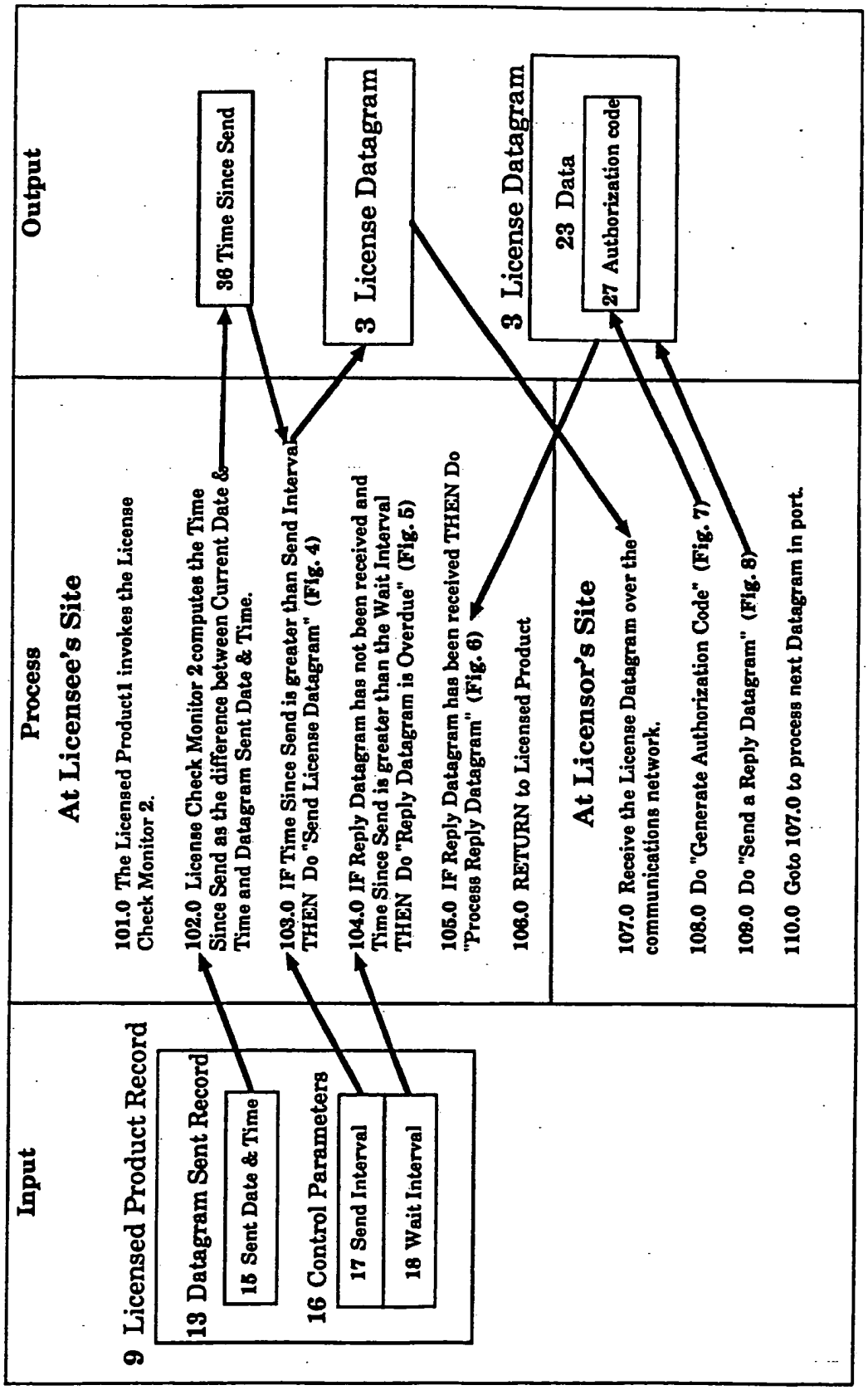


FIG. 4

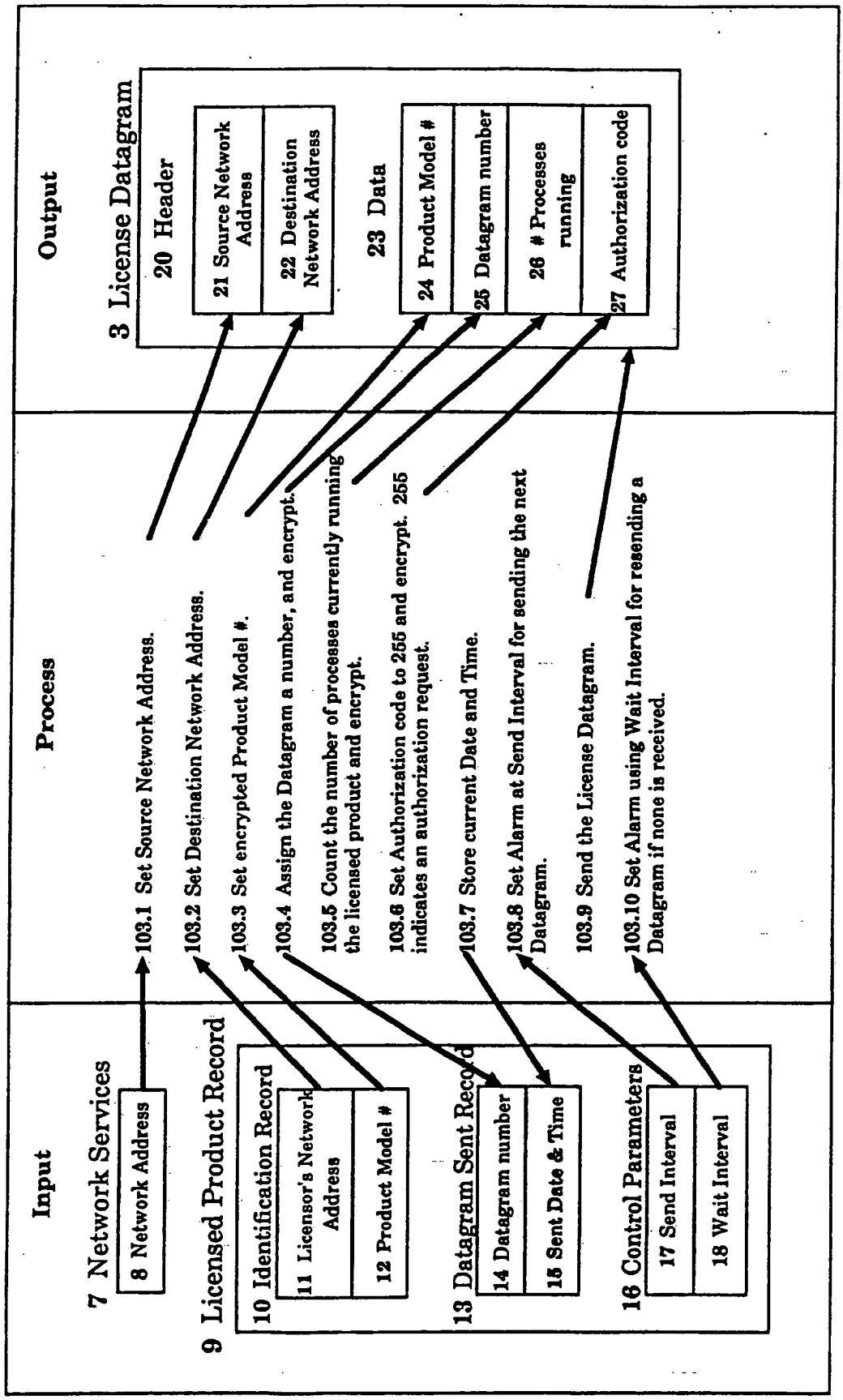


FIG. 5

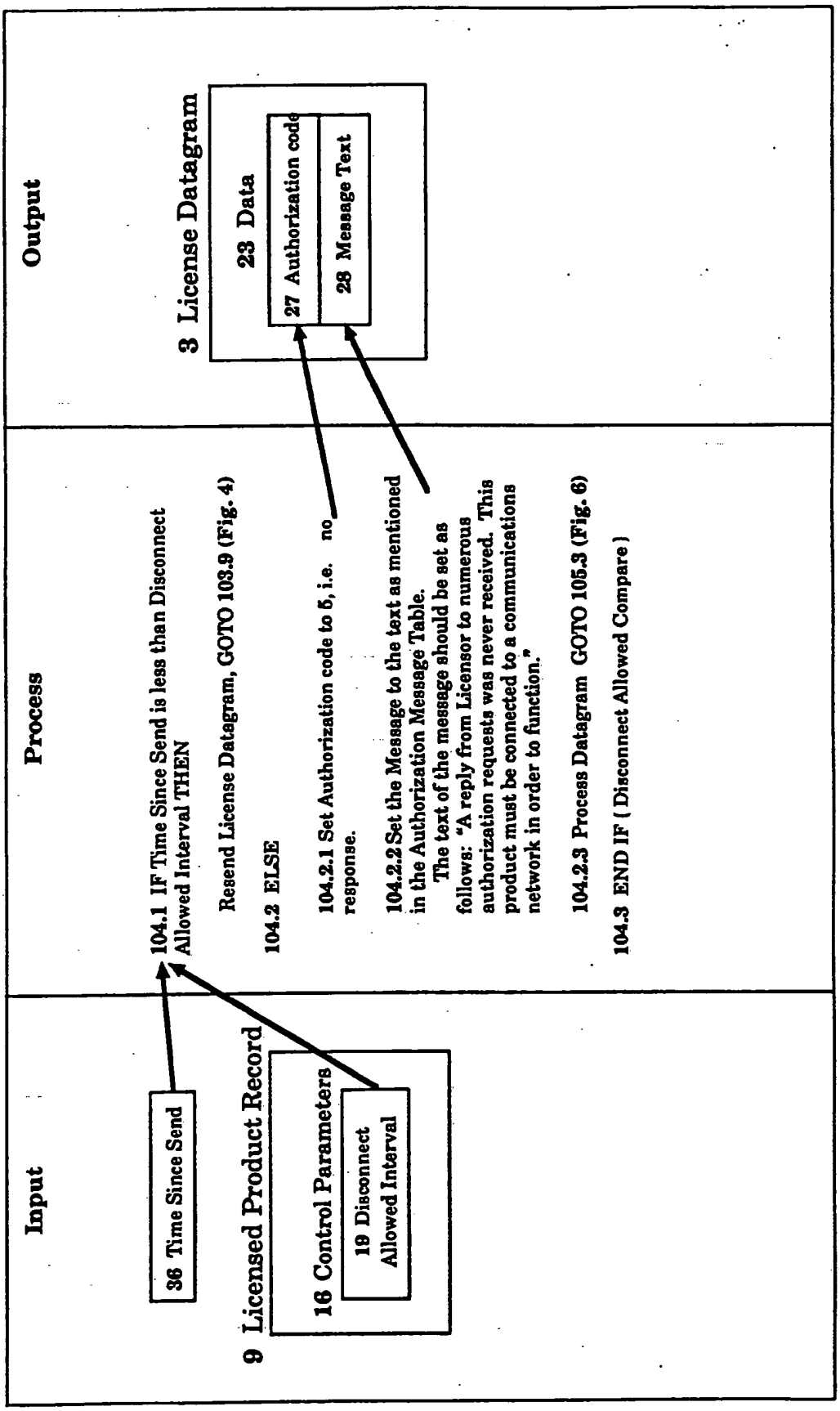


FIG. 6

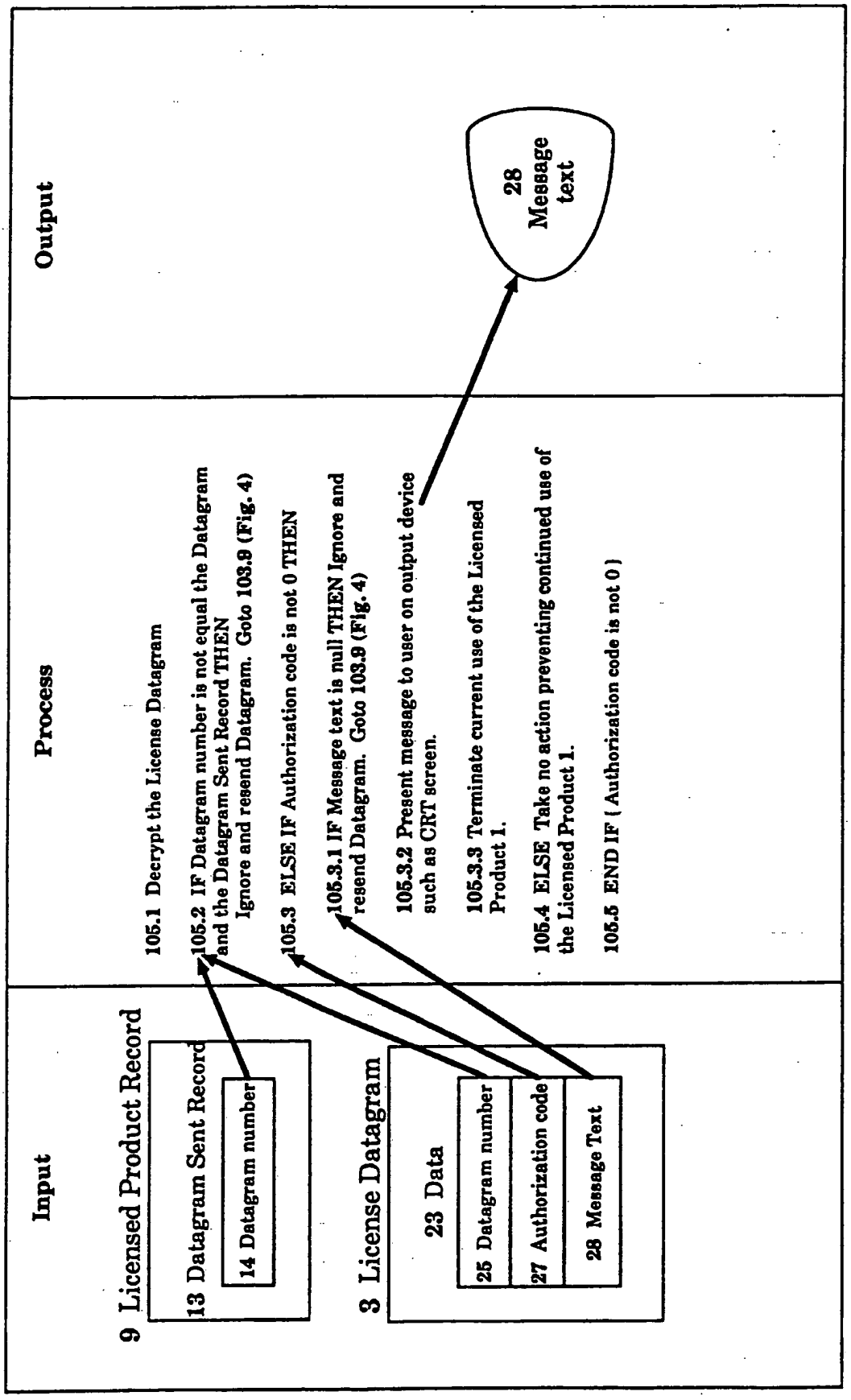


FIG. 7

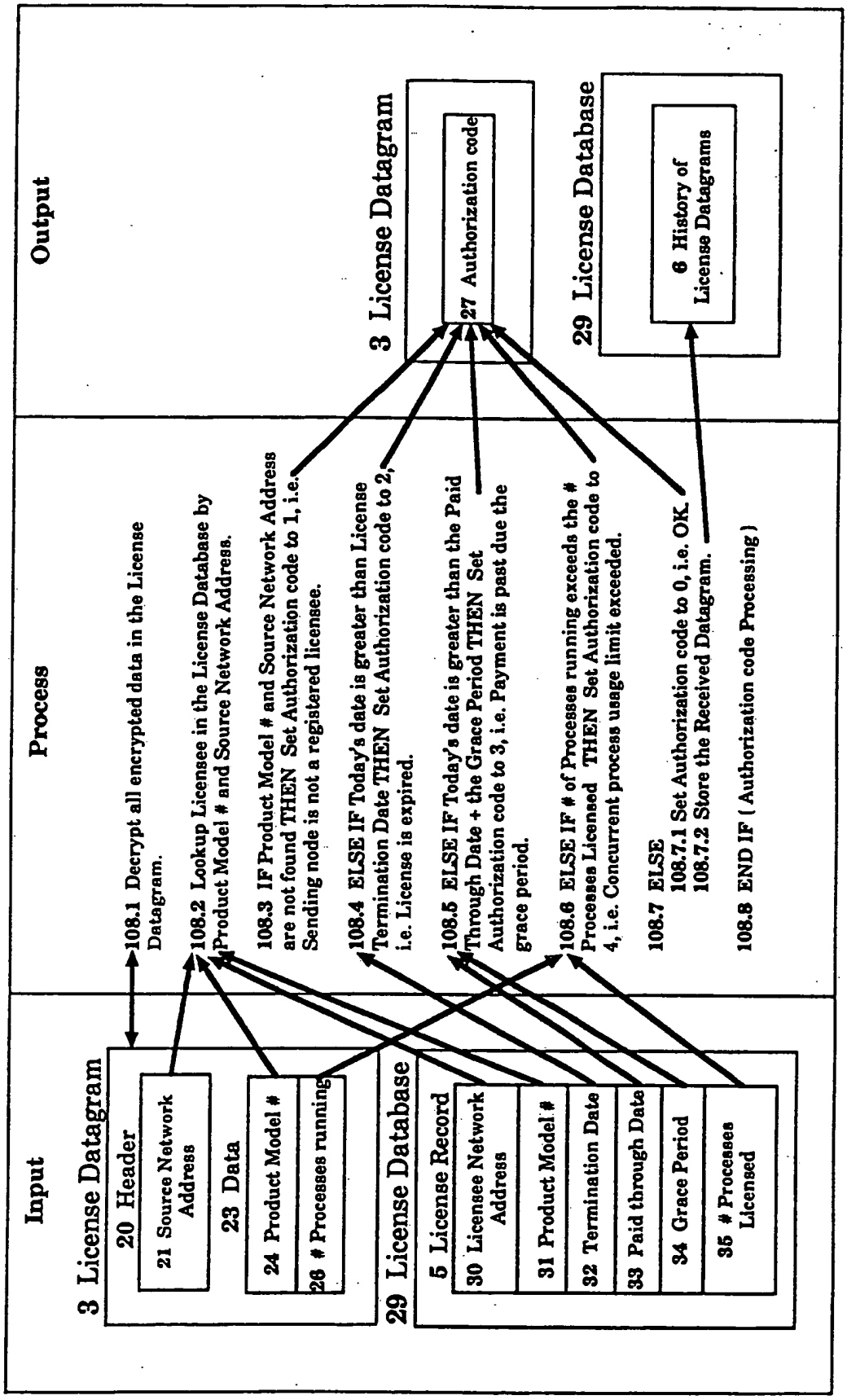
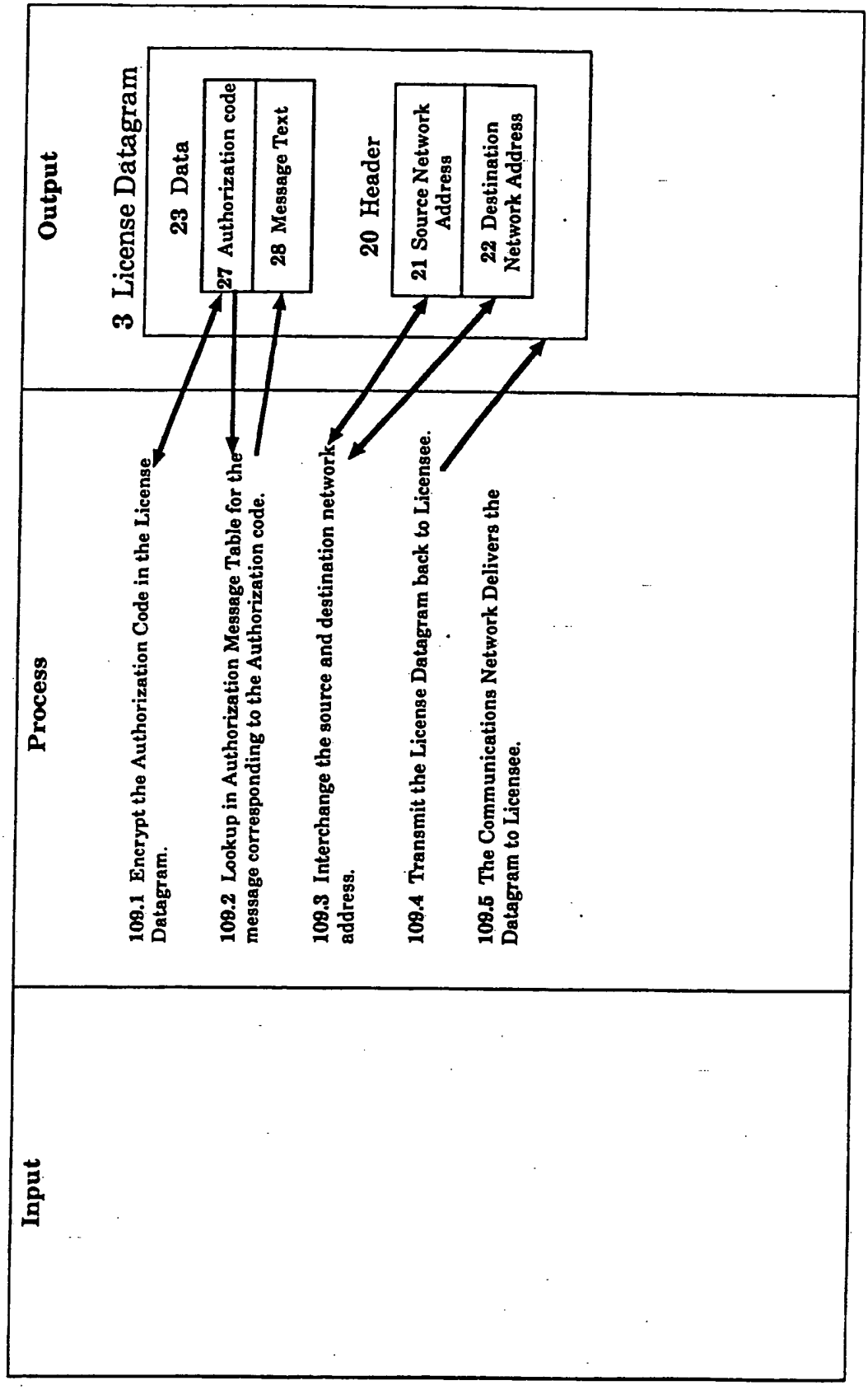


FIG. 8



INTERNATIONAL SEARCH REPORT

International application No.
PCT/US92/05387

A. CLASSIFICATION OF SUBJECT MATTER

IPC(5) :G06F 11/34; H04L 9/00
US CL :395/725; 380/4
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 364/406; 380/25

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

APS DATABASE: Software#, information, usage, monitor?, Licens?

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y,P	US,A, 5,103,476 (WAITE ET AL) 07 APRIL 1992 See entire text.	1-110
Y,P	US,A, 5,050,213 (SHEAR) 17 SEPTEMBER 1991 See column 6, lines 27-51.	1-6,9-21,23-26,29-43,45-53,56-59,61-64,67-73,75-110
Y,P	US,A, 5,047,928 (WIEDEMER) 10 SEPTEMBER 1991 See col. 6, lines 16-54.	1-6,9-21,23-36,29-43,45-53,56-59,61-64,67-73,75-110
Y	US,A, 5,023,907 (JOHNSON ET AL) 11 JUNE 1991 See entire document.	1-110

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be part of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 05 AUGUST 1992	Date of mailing of the international search report 04 NOV 1992
Name and mailing address of the ISA/ Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. NOT APPLICABLE	Authorized officer Debbie Williams KENNETH S. KIM Telephone No. (703) 308-1634

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US92/05387

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US,A, 5,014,234 (EDWARDS, JR.) 07 MAY 1991 See col. 3, lines 4-16.	1-110
Y	US,A, 5,010,571 (KATZNELSON) 23 APRIL 1991 See entire document.	1-6,9-21,23-26,29-43,45-53,56-59,61-64,67-73,75-110.
Y	MACMILLAN Publishing Company, 1985, WILLIAM STALINGS, Data and Computer Communications. p199-203.	1-110
Y,P	US,A, 5,113,519 (JOHNSON ET AL) 12 MAY 1992 See col. 6, lines 36-68.	1-110
Y	US,A, 4,937,863 (ROBERT ET AL) 26 JUNE 1990 See col. 3, lines 25-40.	1-6,9-21,23-26,29-43,45-53,56-59,61-64,67-73,75-110

Request for Comments: XXXX

David Raggett (Hewlett Packard)
12th July 1993

HTML+ (Hypertext markup language)

A proposed standard for a light weight presentation independent delivery format for browsing and querying information across the Internet

Status of this Memo

This document is a proposal for an Internet Draft, and specifies the HTML+ wide-area hypertext document format, with a view to requesting discussion[1] and suggestions for improvements. Distribution of this memo is unlimited.

Abstract

HTML+ is a simple SGML based format for wide-area hypertext documents, for use within the World Wide Web. Unlike desktop publishing formats, HTML+ captures the logical intent of authors. This simplifies the task of writing documents, and permits them to be effectively rendered on a wide range of display types as well as the printed page.

HTML+ represents a substantial improvement over the existing format: HTML, offering nested lists, figures, embedded data in foreign formats for equations etc, tables with support for titles and column headings, change bars, entry forms for querying and updating information sources and for use as questionaires for mailing. This document specifies the HTML+ format with guidelines on how it should be rendered by browsers.

Introduction

The World Wide Web is a wide area client-server architecture for retrieving hypermedia documents across the Internet. It also supports a means for searching remote information sources, for example bibliographies, phone directories and instruction manuals. There are three main ingredients:

a) Universal naming scheme for documents. The universal resource location syntax specifies documents in terms of the protocol to be used to retrieve them, their Internet host and path name. A format for location independent lifetime identifiers is currently being defined by working groups of the IETF. A network protocol will allow universal resource numbers (URNs) to be resolved to the URL for the nearest available copy.

b) Use of available protocols for retrieving documents over the network, including FTP, NNTP, WAIS, Gopher, and HTTP. The latter is designed specifically for use with the World Wide Web, and combines efficiency with an ability to flexibly exchange information between clients and servers.

c) A document format supporting hypertext links based on URLs and URNs which can specify documents anywhere in the Internet. HTML+ is designed for rendering on a wide variety of different display types and platforms.

Information browsers can display information in a wide variety of formats, e.g. plain text, rich text in the HTML+ format, images in the GIF and JPEG formats, MPEG movies, and MIME documents. The hypertext format has a special significance as it allows users to navigate from one document to the next at the click of a button. It provides the basis for menus, cross references, either within a document or to other documents,

---

[1]Please mail comments to the author `dsr@hplb.hpl.hp.com`, or to the WWW discussion group: `www-talk@nxoc01.cern.ch`

perhaps on the other side of the world. It also provides a means of building larger scale collections of documents that act as journals, books or encyclopedias. The format is also intended to act as a building block for creating wide area groupware applications.

HTML+ follows on from an earlier standard - HTML, see [Berners-Lee 93a], which has been widely used as the basis for hypertext documents in the World Wide Web. The new format grew out of experience with HTML, culminating in the desire to add new features, e.g. inline images, tables, and form fields for greater flexibility in querying remote information sources. This document specifies the HTML+ format and suggests ways in which browsers can choose to render it on a variety of different display types.

2. HTML+ and SGML

HTML+ itself is based on the Standardised General Markup Language (SGML), an international standard for document markup that is becoming increasingly important. The term markup derives from the way proof-readers have traditionally pencilled in marks that indicate how the document should be revised.

SGML grew out of a decade of work addressing the need for capturing the logical elements of documents as opposed to the processing functions to be performed on those elements. SGML is essentially an extensible document description language, based on a notation for embedding tags into the body of a document's text. It is defined by the international standard ISO 8879. The markup structure permitted for each class of documents is defined by an SGML Data Type Definition, usually abbreviated to DTD.

Working groups in ISO have recently produced a range of SGML DTDs for documents, e.g. ISO 12083 defines DTDs for books and ISO 10744, which defines the HyTime standard for hypermedia/time-based documents. These standards are large and complex, and perhaps best suited as interchange standards that facilitate conversion between proprietary document formats. By contrast, HTML+ provides a lightweight delivery format that can be rendered by relatively simple browsers, and which has grown out of two years practical experience with wide-area hypertext information systems in the Internet community.

HTML+ and HyTime

The HyTime standard provides a rich range of architectural forms, but is not aimed at run-time efficiency. Suggestions have been made as to how the HTML DTD could be adapted to comply with HyTime's clink architectural form [Kimber 93]. This would necessitate documents declaring links as external entities and the use of local names in link definitions, but in the absence of any immediate benefit, there has been little enthusiasm for this within the World Wide Web community. Instead, it is believed that a straightforward filter program should be used to map HTML and HTML+ documents into a format which is strictly compliant with HyTime, when this becomes appropriate.

A simple example of HTML+

The following is a simple example of an HTML+ document,which illustrates the basic ideas involved in SGML.

```
<title>A Simple HTML+ Document</title>

<h1 id="a1">This is a level one header</h1>

<p> This is some normal text which will wrap at the window margin. You
can emphasise <em>parts of the text</em> if you wish.

<p> This is a new paragraph. Notice that unlike title and header tags,
there is no matching end tag.
```

The text of the document includes tags which are enclosed in <angle brackets>. Many tags have matching end tags for which the tag name is preceded by the "/" character. The tags are used to markup the document's logical elements, for example, the title, headers and paragraphs. Tags may also be accompanied by parameters, e.g. the "id" attribute in the header tag, which is used to define potential destinations for hypertext jumps.

Unlike most document formats, SGML leaves out the processing instructions that determine the precise appearence of the document, for example the font name and point size, the margins, tab settings and how much white space to leave before and after different elements. The rendering software makes these choices for itself (perhaps guided by user preferences), and so can avoid problems with different page sizes or missing fonts.

Logical markup also preserves essential distinctions that are often lost by lower level procedural formats, making it easier to carry out operations like indexing, and conversion into other document formats.

Practical experience has shown that people often make mistakes when they have to type in the markup for themselves. As a result, most browsers are tolerant of bad markup. This problem is being minimised by keeping the format as simple as possible and encouraging the development of WYSIWYG editors.

The HTML+ Document Format

The following sections go through the various features of the format with suggestions as to how browsers should render them. The DTD for HTML+ is given in Appendix I.

Parsing HTML+ Documents

By default, HTML+ documents are made up of 8-bit characters in the ISO 8859 Latin-1 character set. In future, 16 bit character sets may be used to cover a wider range of languages. The HTTP network protocol uses the MIME standard (RFC 1341) to specify the document type and the character set. It is assumed that the chosen character set includes the printable 7 bit US ASCII characters as a subset.

The DTD specifies the syntax of the document structure, in particular, which tags are permitted in any given context. Certain tags are only permitted at the start of the document. Tags and attribute names are case insensitive, thus <TITLE> is equivalent to <title>. End tags may be minimised to </> instead of say </title>.

In general, SGML entity definitions are used to represent characters which would otherwise be confused with markup elements:

&	is represented by	`&`
<	is represented by	`<`
>	is represented by	`>`

Such entity definitions should be used in all places except within attribute values for tags (tag names and attribute names cannot contain these particular characters). Entity definitions can also be used for special characters, e.g. "é" for a small e with an accute accent. The full list is given in Appendix II. Additional entities may be defined within documents using the SGML entity declaration tag !ENTITY, e.g.

```
<!ENTITY % sgml "Standardised General Markup Language">
```

The browser will then insert the full form whenever it comes across "&sgml;".

Repeated white space characters such as space, tab, carriage return, line feed and form feed are ignored except within preformatted text, i.e. it doesn't matter which white space characters you use or how many of them you put between words, or before or after markup elements, the effect is the same as a single space character.

It is recommended that HTML+ documents start with the following external identifier, indicating that the document conforms to the HTML+ DTD. This will ensure that other SGML parsers can process HTML+ documents, without needing to include the DTD with each document.

```
<!DOCTYPE htmlplus PUBLIC "-//Internet/RFC xxxx//EN">
```

HTML+ departs slightly from pure presentation independence by allowing authors to specify rendering hints, e.g. to use a bold font for a given type of emphasis. This step was taken to give authors greater control over the final appearence, and is based upon practical experience with the earlier HTML format. In addition, attribute values are used to distinguish different subcategories of markup, rather than adding extra tags. New logical categories of emphasis etc. can be added in future without needing to change existing browsers. These decisions have made it practical to restrict HTML+ to a very small set of tags.

Backwards Compatibility with HTML

The format is designed to be largely compatible with the earlier format HTML, and HTML+ browsers will be able to display documents in the HTML format with little extra cost. Suggestions on how to map HTML elements to HTML+ are given in Appendix III.

Notes for Implementors

Please ensure that browsers can tolerate bad markup. In practice, this is quite straightforward to achieve, provided a naive top-down SGML parser is avoided. A forgiving parser should be able to cope with tags in unexpected positions, e.g. the <A> tag bracketing a header[2]. Unknown tags should be simply ignored.

Implementors should endevour to make sure that documents can be scrolled efficiently regardless of their length. Always parsing from the start of the document leads to jerky performance. Two strategies for efficiently scrolling through documents are:

a) Establish regular landmarks throughout the document for which the state of the parse is known. The browser can then work forward from the nearest landmark, when it needs to refresh the screen after a scroll operation. The landmarks need updating when users make changes, while using a WYSIWYG editor.

b) When scrolling up, parse backwards to work out the state at earlier points in the document. This can be done via a combination of skipping back, looking for markup which causes a line break etc. and then parsing forward until the current position, to find the change of state. This can be repeated until the parser reaches a point prior to the new top of the window.

Practical experience has shown the importance of providing cues to users on progress in retrieving documents over the network. These will depend on the protocol, but should show how much data has been received at any point. The network connections shouldn't block, and an abort button is essential[3]. It is generally better to avoid displaying the retrieved document in a new window, unless explicitly requested by the user, e.g. by holding down the shift key when clicking the hypertext link.

Normal Text

This is generally shown with a serif font and wraps on the right window margin. It can include:

- Entity references, e.g. "`>`" and "`&eacute;`"
- Significant Line breaks (the BR tag)
- Non-breaking spaces - the SP tag
- Hypertext links - the A tag
- Inlined graphics or icons - the ICON tag
- Various styles of logical emphasis - the EM tag
- Embedded data in an external format, e.g. TeX equations - the EMBED tag
- Input fields for forms - the INPUT tag.

Line breaks and

This tag causes the renderer to start a new line at the current left margin setting. There is no corresponding end tag. The BR tag is *empty*, that is to say, it doesn't act as a container around other text or markup.

---

[2]Headers typically cause a line break and leave a vertical gap. If the hypertext link definition is parsed prior to the beginning of the header, the starting position for the button will be in the wrong place - browsers should therefore adjust this position to the beginning of the text.

[3]For X11 on Unix systems, the *select* system call can be used with non-blocking I/O to poll the event queue at regular intervals. The *XtAddInput* call acts as a wrapper around select for this very purpose. Users can then continue to view the current document as well as being able to click an abort button (which sets a global variable, polled by the comms software). Be careful to disable unsafe actions, e.g. trying to get a second document while still waiting to get the first (a race hazard).

Non-breaking spaces and <SP>

This allows authors to be certain that browsers won't break a line at an inappropriate place. Authors may also use SP at the start of a line to indent text. This is deprecated.

Some authors like to have a slightly longer space after punctuation at the end of sentances. While this is a stylistic issue, browsers need to be able to distinguish periods which denote the end of sentances from those used in abbreviations. One way of doing this in HTML+ is to use the EM tag to delimit abbreviations.

Hypertext Links

When the user clicks on a hypertext link in the document, the current document is replaced by the one referenced by the link. Links can be made to a wide range of document types, based on the URL[4] and URN[5] notations. Some document types permit links to be made to specific sections within a document[6]. The syntax for links within the same document or to documents in the same directory is particularly simple:

```
Links are defined with the <a href="#z1">A tag</a>. HTML+ supports a
number of <a href="links.html">different link types</a>.
```

In a browser this might look like:

Links are defined with the *A tag*. HTML+ supports a number of *different link types*.

The first link is to an anchor named "z1" in the current document. The second is to a file named "links.html" in the same directory as the current document. The caption for the link is the text between the start and end tags. The value for the HREF attribute defines the destination point, and can be abbreviated in certain cases. If practical, word the caption in such a way that continues to make sense when the document is printed out. The link should be shown in a clearly recognisable way, e.g. as a raised button, or with underlined text in a particular color. For displays without pointing devices, it is suggested that a reference number is given in square brackets, which can then be typed by the user.

A more general discussion of hypertext links and their treatment in HTML+ is presented in a later section.

Inlined Graphics or Icons

These are treated like characters and inserted as part of the text, e.g.

```
This line has a egyptian hieroglyph at the end of the
line. <img src="ankh.tiff">
```

The URL notation is used to name the source of the graphics data. The *align* attribute can be used to control the vertical position of the image relative to the current text line in which the IMG element is placed. Use a value of "top", "middle" or "bottom" to align the top, middle or bottom of the image with the current text line. The *seethru* attribute allows authors to include a chromakey, i.e. a colour that designates portions of the image to be left unpainted so that the background shows through. The format for this attribute's value is dependent on the type of graphics data, and has yet to be defined.

Note that you can create simple iconic buttons, e.g.

```
<a href="bigpic.gif"><img src="littlepic.tiff"></a>
```

If the user clicks anywhere on the image, this will cause the browser to retrieve its bigger version. This approach allows users to preview images which may take significant time to download. Note that there is little additional penalty for displaying the same image at multiple points in the document. The *ismap* attribute is provided for backwards compatibility with HTML. When present the browser will send all mouse clicks and drags on the image, to the server.This mechanism is explained in more detail for the FIG tag.

---

[4]The notation for universal resource locators is defined in [Berners-Lee 93b].

[5]The notation for universal resource numbers and the protocol for resolving them to the nearest available copy is currently under study by the IETF URN working group.

[6]At the time this document was written, such links were restricted to named anchors within HTML and HTML+ documents

Sophistocated HTML+ editors should allow authors to modify images using an external editor. Larger images should be specified with the FIG tag.which provides support for flowing text around figures, along with captions, overlays and active areas.

Various Styles of Emphasis[7]

This allows you to emphasise a portion of the text. The simplest approach is:

```
<em>default emphasis, usually shown in an italic font</em>
```

The logical role of emphasis denotes the semantic significance, e.g. a citation, or text to be input by a user for a computer program.The physical style of emphasis controls its appearence. Note that EM elements can include inlined graphics.

Logical Role of Emphasis

It is strongly recommended that the logical role of the emphasis is given with the *role* attribute, e.g.

```
<em role="cite">a citation</em>
```

Providing a logical role allows browsers to apply differing rendering styles according to the role, but more importantly, it allows indexes to be constructed automatically, e.g. the list of bibliographic references in a technical report. These can be used for searching through collections of documents according to semantic keys giving better focussed searches compared with full text indexes .

The list of recommended roles are as follows:

For references to other works:

CITE	a reference to a related work
PUB	a publication containing a referenced work
AUTHOR	an author of a referenced work
EDITOR	an editor of a referenced work
CREDITS	e.g. the rights owner of a photograph
COPYRIGHT	the holder of the copyright
ISBN	for ISBN numbers
ACRONYM	for acronyms like "NATO" and "US"
ABBREV	for abbreviations

For annotations:

FOOTNOTE	shown as footnote or pop-up
MARGIN	shown as margin note or pop-up

For computer instruction manuals:

DFN	defining instance of a term
KBD	something a user would have to type
CMD	command name, e.g. "chmod"
ARG	command arguments, e.g. "-I"
VAR	named place holder, e.g. "filename"
INS	an instanced of a named printer, directory or file etc.
OPT	an option of some kind
CODE	an example of code (shown with a fixed pitch font)
SAMP	a sequence of literal characters

On dumb terminals annotations should be shown in round brackets. Margin notes should be right aligned, and may include graphics via the IMG tag. The set of recommended roles will be kept by the HTML+ registration authority.

---

[7]The name EM was chosen in preference to EMPH because it allows existing HTML browsers to show all HTML+ emphasis in italics. It also allows HTML+ browsers to correctly process the common case for emphasis in HTML documents.

Physical Styles

The appearence can be modified by adding optional rendering hints from the list:

```
<em b>              bold text
<em i>              italic text
<em u>              underlined text
<em sup>            superscript text
<em sub>            subscript text
<em tt>             type writer font (courier)
<em hv>             sans serif font (helvetica)
<em tr>             serif font (times roman)
```

These hints can be combined, e.g.

```
<em b i> for bold italic text </em>
```

Note that these are only hints and may be ignored by browsers. Indeed, arbitrary combinations will present difficulties for most browsers. If the display is limited to a single font, colour or underlining can be used, but should be clearly differentiated from hypertext links and headers. Dumb terminals can use email conventions, e.g. switching to all capitals, or delimiting with the * or _ characters. Subscript and superscript text should be shown in a smaller point size, vertically offset as appropriate.

Browsers may choose to simplify or ignore hints, but should aim to do so in a consistent manner. At the simplest level, browsers can ignore the attributes and render all emphasis in the same style.

Nested Emphasis

Emphasis can be nested as in:

```
<em b>bold text, and <em i>bold italic text</em></em>
```

Nested emphasis is better suited for grouping logical roles together, for instance, you could use the EM to separately tag author, title, and publication, and then wrap these up as a citation. Without this, indexing programs will have difficulty in grouping markup into the correct references.

Embedded data in an external format

The EMBED tag provides a simple form of object level embedding. This is very convenient for mathematical equations and simple drawings. It allows authors to continue to use familiar standards, such as *TeX* and *eqn*. Images and complex drawings are better specified using the FIG or IMG elements. The *type* attribute specifies a MIME content type and is used by the browser to identify the appropriate shared library or external filter to use to render the embedded data, e.g. by returning a pixmap. It should be possible to add support for new formats without having to change the browser's code, e.g. through using a common calling mechanism and name binding scheme. Sophistocated browsers can link to external editors for creating or revising embedded data. Arbitrary 8-bit data is allowed, but &, < and > must be replaced by their SGML entity definitions.

Input Fields for Forms

Input fields can be arranged with considerable freedom, as part of normal paragraphs, preformatted text, lists or tables. Examples of how to do this are given later on in the section describing the FORM tag. The INPUT tag has the following attributes:

name	Used to name this input field, e.g. `name="phone number"` (required attribute).
type	Defines the type of data the field accepts (the type name is insensitive to upper/lower case). If missing, the field is assumed to a a free text field.
size	Specifies the size/precision of the input field according to its type (optional).
value	The initial value for the field, or the value when checked for checkboxes and radio buttons (optional, except for radio buttons).
checked	When present, this attribute indicates that a checkbox or radio button is selected.
disabled	When present, this attribute indicates that this field is temporarily disabled. Browsers should show this by greying out or via a similar visual clue. Users are unable to set the focus to disabled fields, or change their values.

error When present, this attribute indicates that the current value for this field is in error in some way, e.g. because it violates some consistency constraints. Browsers should indicate this by a change to the shape and colour (red) of the field's border. This should be accompanied by an error message and a beep.

The following types of field should be supported:

TEXT	Single or multi-line text entry fields. Use the *size* attribute to specify the width and height in characters, e.g. size="24" or size="32x4".
URL/URN	For fields which expect document references.
INT	For entering integer numbers, the maximum number of digits may be given with the *size* attribute, e.g. size=3 for a 3 digit number[8].
FLOAT	For fields restricted to floating point numbers.
DATE	Restricted to a recognised date format.
CHECKBOX	Use these for simple boolean attributes, or for attributes which can take multiple values at the same time from some set of alternatives.
RADIO	Use these for attributes which can take a single value from a set of alternatives (groups input fields with the same *name*).

For the purposes of sending the contents of a form to a server, as part of a query, the input fields are mapped to a list of properties. In most cases the *name* and current *value* are used to define a property/value pair for each field. Radio buttons and check boxes are left out if they are unselected. This ensures that only the selected radio button yields a property/value pair. By missing out the *value* attribute for check boxes, these fields will map to a simple (value-less) property. The representation of property lists is defined as part of the HTTP protocol.

Browsers can choose to notify the server whenever a field is changed (i.e. when a field looses the focus and its contents have changed) or wait until the form is completed. This choice will depend on network latency.

Headers and Titles

The title tag is generally used to define the window banner when viewing a particular document, e.g.

```
<title>Reference Guide to HTML+</title>
```

This element should appear at the start of the document. There are six levels of headers, H1 to H6, with H1 the most important, and H6 the least. A common convention is to begin the body of the document with a level one header. e.g.

```
<h1>Introduction to HTML+</h1>
```

Header names should be appropriate to the following section of the document, while the document title should cover the document as a whole. There are no restrictions on the sequence of headers, e.g. you could use a level three header following a level one header. Browsers should render headers with a line break before and after the header text. A common convention for headers is to use a sans serif font, e.g. Helvetica, with a smaller point sizes for less significant headers, and a serif font, e.g. Times Roman, for normal text.

Headers can include an identifier, unique to the current document, for use as destinations of hypertext links, e.g.

```
<h1 id="intro">Introduction to HTML+</h1>
```

This allows authors to make links to particular sections of documents. It is a good idea to use something obvious when creating an identifier, to help jog your memory at a later date. WYSIWYG editors may automatically generate the identifiers. In this case, they should also provide a point and click mechanism for defining links, so that authors don't need to deal explicitly with the identifiers.

---

[8]Perhaps the syntax should permit integer ranges, e.g. size="1 to 6", in which case a more appropriate name for the attribute than *size* would be desirable.

The attribute "margin" when present acts as a hint to the browser to insert the header into the margin and causes the following text to be vertically aligned with the start of the margin header. By convention, margin headers are left justified, e.g.

```
<h4 margin> Deleting the Curve </h4>

The Delete command allows you to delete any selected symbol or text
block.
```

Note that headers don't act as containers for the subsequent text. You can group the header and text with the GROUP tag, see later for details.

Indexing

A good index plays an important role in helping users find their way to the material they need. It allows users to type in one or more keywords to see a meaningful list of matching topics. Alternatively they can browse through the index and take advantage of serendipity, and gain a feeling for the limits of what is covered in the associated document. The two approaches can be combined, when the characters typed act dynamically to control the viewing position within the index. Typically each keyword entry in the index is associated with one or more topics. This notion of guiding the user is absent from full text indexes like WAIS, where users are given very little help in choosing the keywords to search on.

Generating a conventional index for a document is a skilled task, and HTML+ allows authors to include annotations for creating an index. These directives can be included with document titles, headers and emphasis etc. using the *index* attribute. This allows each such element to be included in one or more entries in the index, under primary or secondary keys, e.g.

```
<h3 id="z23" index="Radiation damage/shielding from as difficult">Radiation shielding</h3>
```

This resulting index looks like:[9]

```
Radiation damage
    classical target theory
    dominance of
    in molecular mills.
    shielding from as difficult
    simple lifetime model
    track-structure lifetime model
Radicals
  and so on.
```

Where each entry is a hypertext link to the associated anchor. The *index* attribute can specify multiple entries, each separated with the ";" character. The optional secondary key (*shielding from as difficult*) is introduced by the "/" character. Secondary keys are useful when the primary key occurs more than once. To allow for future extension, primary keys should not start with the "#" character. This prefix is being reserved to designate indirect index entries. Use "\/", "\;", "\#" and "\\" to escape "/", ";", "#" and "\" respectively.

Paragraphs and Preformatted Text

HTML+ includes support for paragraphs and preformattted or verbatim text.

Defining Paragraphs with <P>

The <P> tag splits normal text into paragraphs. Unlike headers, there is no corresponding end tag, so don't use </P>. The following optional attributes can be used:

id An identifier, unique to this document, which can be used as a destination in a hypertext link. Note that the paragraph tag acts as a container for the paragraph.

---

[9]Taken from K. Eric Drexlers's "Nanosystems, Molecular Macherinery, Manufacturing and Computation".

role The role of the paragraph, see the following list for supported types.

align A rendering hint to the browser to justify lines. The supported values should be: `align="left"`, `align="center"` and `align="right"`. This is useful for single line paragraphs or when the lines are made explicit with the <BR> tag.

indent When present, this hint suggests that the left and right margins are indented by an amount dependent on the browser, e.g. about 4 character widths.

The *role* attribute is used to indicate the logical role of the paragraph, e.g. a stanza in a poem or a cautionary note in a computer manual. Browsers may apply particular rendering styles to certain roles. The role name is case insensitive. The following roles are recommended:

quote A paragraph quoted directly from some other work. Browsers could indent the paragraph and maybe use a different font.

byline Information about the author of the document, e.g contact details. This could be displayed in a different font, and perhaps right aligned.

note Advisory note in an instruction manual. The browser could display a hand icon in the margin.

caution Cautionary note. The browser could display an warning road sign in the margin.

error A note describing error conditions. The browser could indicate the importance of the note by displaying a stop sign in the margin.

An example of a paragraph element:

```
<p role="note"> If you accidentally delete a symbol other than the red
circle, immediately press ALT+BKSP to choose the undo command, and
then select the red circle and delete it again.
```

Paragraphs can be rendered by indenting the first line, or by leaving a vertical gap equal to half the current line spacing. When using the latter style, browsers should take care to avoid this vertical gap when the paragraph element immediately follows a header. This rule ensures that authors can tag paragraphs directly following a header without causing unwanted extra space before the start of the text.

Ordered, Unordered and Definition Lists

There are three kinds of lists: ordered or numbered lists, unordered lists and definition lists. Ordered and unordered lists can be nested arbitrarily, and browsers should progressively inset the left margin for each level of nesting.

Ordered Lists with <OL>

The list items are automatically numbered, e.g.

```
<OL>
  <LI>Wake up
  <LI>Get dressed
  <LI>Have breakfast
  <LI>Drive to work
</OL>
```

Is displayed as:

1) Wake up
2) Get dressed
3) Have breakfast
4) Drive to work

The *compact* attribute when present has the effect of reducing interitem spacing, e.g. <ol compact>. Authors can also make both the OL tag and the LI tag potential destinations for hypertext links with the *id* attribute. List item text can include normal text and paragraph elements, but not headers.

Unordered Lists with <UL>

These are bulleted lists, e.g.

```
<UL>
  <LI>Wake up
  <LI>Get dressed
  <LI>Have breakfast
  <LI>Drive to work
</UL>
```

Is displayed as a bulleted list:

- Wakeup
- Get Dressed
- Have breakfast
- Drive to work

The *compact* attribute when present has the effect of suppressing bullets and reducing interitem spacing, e.g. <ol compact>. Multicolumn lists can be requested with the *narrow* attribute, e.g. <ul narrow>. This causes the browser to try to lay out the list as a number of columns, depending on the window width. This attribute should only be used when all the items are less than 20 characters long. Authors can also make both the UL tag and the LI tag potential destinations for hypertext links with the *id* attribute. List item text can include normal text and paragraph elements, but not headers. For nested unordered lists, browsers may use different bullet symbols for different levels, in addition to progressively insetting the left margin. The *src* attribute on the LI tag can be used to specify an icon for use in place of the standard bullet symbols.

Definition Lists with <DL>

These consists of pairs of terms <DT> and definitions <DD>. The following example is part of a french dictionary:

```
<DL>
  <DT>endetter
  <DD>Engager dans des dettes

  <DT>endeuiller
  <DD>Plonger dans le deuil, remplir de tristesse

  <DT>endiablé, ée
  <DD>D'une vivacité extrême
</DL>
```

Is commonly displayed as:

endetter	Engager dans des dettes
endeuiller	Plonger dans le deuil, remplir de tristesse
endiablé, ée	D'une vivacité extrême

With the *compact* attribute, e.g. <dl compact>, this is altered to:

endetter Engager dans des dettes

endeuiller Plonger dans le deuil, remplir de tristesse

endiablé, ée D'une vivacité extrême

In this style, the term and definition appear in the same paragraph, with the term text emphasised in a bold font. The definition text follows on, and wraps to a left margin a little further inset than the term text. This style is common place in dictionaries.

Term text following the <DT> is restricted to normal text. The definition text after the <DD> tag can additionally include paragraph elements and ordered/unordered lists. Headers are not allowed in either case. Authors can make the DL, DT and DD tags potential destinations for hypertext links with the *id* attribute.

Authors are reminded to check that DT and DD are paired up. Common misunderstandings lead to people repeating DD tags to separate paragraphs (use <P> instead), or leaving out the DT tag altogether to indent text (use <p indent> or <group indent>). The ability of browsers to cope with bad markup seems to encourage such problems, which will hopefull fade away as wysiwyg editors become commonplace

Figures

Figures provide great flexibility:

- linked or embedded graphics
- control of picture alignment and text flow
- Figure description for when the image can't be shown
- caption placement
- scaled or pixel-based coordinates
- hypertext links with active areas
- text and image overlays

The following simple example will set the scene for the description of the various features:

```
<fig align="right" src="map.gif"> How to get to my house </fig>
```

Here, the image is defined by a link to an external document. The caption "How to get to my house" will appear at the bottom of the image. The *align* attribute directs the browser to display the figure at the right of the widow, and to flow subsequent text around the left of the image.

Using embedded graphics data

Instead of the *src* attribute, you can include an EMBED element immediately following the <fig> tag. This is useful for graphs etc. defined in an external format.

Figure Description

The FIGD tag allows you to give a textual description which can be shown when the figure itself can't be shown, e.g. for browsers working on dumb terminals, e.g.

```
<FIGD> This is an aerial photograph of central London, showing
Buckingham Palace and the Houses of Parliament. On the left you can see
Hyde Park and in front the Albert Hall and the Natural History
Museum.</FIGD>
```

Alignment and Text Flow

The *align* attribute controls the horizontal position of the figure: "left", "right", or "center". The default is "left". Browsers may flow text when there is sufficient room, unless the figure is center aligned or the *noflow* attribute is present.

Caption Placement

The *cap* attribute allows you to ask the browser to position the caption text to the "left", "right", "top" or "bottom". The default is to place the caption at the bottom of the figure. Text flow will occur around the figure and caption, leaving a suitable gully. The browser will ignore this attribute if there is insufficient room for the requested placement.

Pixel-base or Scaled Coordinates

The upper left of the figure is designated as x,y = (0, 0), with x increasing across the page, and y down the page. If points are given in real numbers, the lower right is taken as being (1.0, 1.0), otherwise with integer values, the coordinates are assumed to be in pixels[10]. Note that using scaled coordinates is much safer, especially for graphics! The extent of the image in pixels may change, e.g. as a result of format negotiation with the server, and by retrieving images with lower resolution when network performance is poor.

Active areas

The *ismap* attribute causes the browser to send mouse clicks on the figure, back to the server using the selected coordinate scheme. The mouse button-up event is sent with the URL formed by adding "?x,y" as a suffix to the URL for the current document. You can also designate rectangular regions of interest in the picture by holding the mouse button down while dragging the mouse. The browser should show a rubber band outline for the rectangle defined by the current location of the mouse pointer and the point at which the mouse button was pressed. The region is named by taking the current URL and adding the suffix: "?x1,y1;x2,y2", where (x1, y1) and (x2, y2) define the points at which the mouse button went down and came up, respectively. The *ismap* mechanism is relatively slow, but makes sense when the active regions change their boundaries over time, e.g.

```
<fig ismap src="weather.gif">Click on your area for todays weather</fig>
```

You can also designate arbitrary areas of the figure as hypertext links. Mouse clicks are handled locally, and the browser can provide visual clues that the pointer is over an active area, for example, by changing the pointer from an arrow to a hand symbol, or highlighting the area in some way.

Active areas are defined with the FIGA tag. This has two attributes:

href	A URL specifying the link to traverse when clicked (required)
area	Defines a polygonal[11] area as a list of points: "x1, y1; x2, y2; ..." (optional)

The *area* attribute lists a sequence of points defining a polygon. Closure is ensured by joining the last point in the list to the first (i.e. a triangular area is defined with a list of 3 points).When the *area* attribute is missing, the whole of the picture is assumed. Polygons may be non-convex or even intersect themselves, thereby complicating the definition of what is enclosed by the polygon. Holes should be excluded. Note that active areas defined with FIGA take precedence over the *map* mechanism.

Overlays

The FIGT tag allows you to position text and image overlays on top of the figure, e.g.

```
<fig src="map.giff">
  <figt at="0.2, 0.3" framed>A text overlay</figt>
  The figure caption
</fig>
```

The overlay can contain a wide variety of elements including text, images (IMG), lists and tables. Figures shouldn't be nested. Any hypertext links in the overlay text will take precedence over the *href* attribute in FIGT. The following attributes are permitted:

at	The upper left of the overlay, relative to the figure.
width	As a fraction of the figure, e.g. width="0.3". This allows you to limit the lengths of wrapped text lines. The vertical extent is then determined automatically.
framed	Directs the browser to draw a frame around the overlay and to colour in the background in some way.
href	Allows you to make the overlay into a hypertext button.

---

[10]This mechanism was designed to be backwards compatible with the *ismap* feature as used with IMG in HTML, and as a consequence forces the choice of y increasing down rather than up the page. A simple test to distinguish the two schemes is to check if the "." character occurs anywhere in the list of points.

[11]The code for hit testing polygons is tricky, but quite fast. A public domain version of the code would be helpful.

Tables

Tables are defined with the TBL tag. Cells are designated as being headers or data. You can join adjacent cells, e.g. to define a header spanning two columns.

An Example of a Table

	average		other
	height	weight	category
males	1.9	0.003	yyy
females	1.7	0.002	xxx

This is defined by the markup:

```
<tbl border>
   <tt top> An Example of a Table
   <th rowspan=2> <th colspan="2"> average <th> other <tr>
   <th> height <th> weight <th> category <tr>
   <th align=left> males <td> 1.9 <td> .003 <td> yyy <tr>
   <th align=left> females <td> 1.7 <td> .002 <td> xxx
</tbl>
```

The *border* attribute for TBL directs the browser to draw borders. The *compact* attribute is used when you want the table to appear in a smaller size.

The optional <tt> tag defines a title. By default (i.e. when *top* is missing) this should be positioned below the table. The <th> and <td> tags define header or data cells respectively. The <tr> tag acts as a separator between rows. In the example, you can see that the first header in each of the first two rows is void.

TH, and TD all have the same permitted attributes:

colspan	Columns spanned by this cell, see example
rowpan[12]	Rows spanned by this cell, see example
align=left	Left justify the cell's content
align=center	Center justify the cell's content
align=right	Right justify the cell's content

By default, headers are centered, while other cells are left justified. If practical, browsers should be smarter than this, e.g. if all the cells in a column are shorter than the column header, then indent the cells to make them appear under the middle of the header.

Browsers need to carry out a pre-parse (e.g. when sizing the vertical scroll bar) in order to determine the number of columns and their widths. The following guidelines may be useful:

- There is no need to declare empty cells at the end of a row, so the number of columns for the table is given by the row with the most columns.
- Restricting text to a fixed pitch font may simplify matters.
- If a column only contains numbers or empty cells then align on units and set width to maxium precision needed (before and after decimal point, allowing for an exponent). This rule also applies when currency symbols are used.
- Otherwise set column width to the minimum of a threshold width and the maximum text length for all cells in the column. Text is left aligned and wrapped if it exceeds the chosen column width.

---

[12]This is tricky to handle. The parser should carry a spanned cell over to the next row, the definition of which should miss out the spanned cell, i.e. the next row will have one fewer explicit cell definitions.

The threshold column width can be set according to the number of columns and the width of the display window. It is also necessary to take the column headers into account in this process. Header text wraps to the next line if the column is too narrow. Browsers will by default center the header in the column.

A complication occurs when a header or data cell spans more than one column, as specified by the *s* attribute. This can be used to give complex headers which share a header between columns followed by individual headers on the next line.

Vertical gaps can be introduced with the <tb> element - this inserts 1/2 line space into the next row. Header and Data rows can be intermixed. Authors can use alternate header and data rows when the rows alternate between text and numbers. The vertical alignment of numbers only applies to data fields.

Tables which don't fit into this model should be defined as figures using an external format, e.g. Postscript, Tex or Computer Graphics Metafile.

Forms

A document can include one or more forms. Each form is defined by a FORM element, which contains a number of input fields laid out with normal and preformatted text, lists and tables. The browser should manage the input focus, e.g. with the tab key and mouse clicks. The Return key can be used to mean that the user has filled in the form and wants the appropriate action to be taken. Browsers may also display "Accept" and "Cancel" buttons as part of the document (or perhaps on another part of the browser). Note that forms shouldn't be nested.

The action to be taken is specified by the *action* attribute of the FORM tag. If missing the URL for the current document is assumed. This attribute uses a URL to specify a server to query, or an email address to send the form to. When sending the form to a server as a query, the form's contents are encoded as a property list (see definition of the INPUT tag). The precise encoding is dependent on the HTTP protocol and defined in [Berners-Lee 93c][13]. When the form is to be mailed, it is first converted into plain text, closely resembling the appearence on the screen. You can include multiple RFC 822 mail headers with the MH tag. The *hidden* attribute may be used to hide the headers when browsing the document. The following is an example of a simple questionaire:

```
<form action="mailto:www_admin@info.cern.ch">

<mh hidden>
  Subject: WWW questionaire
</mh>

Please help us to improve the World Wide Web by filling in the
following questionaire:

<p>
Your organisation? <input name="org" size="48">

<p> commercial? <input name="commerce" type="checkbox">
How many users? <input name="users" type="int">

<p> Which browsers do you use?

<ol compact>

<li> X Mosaic <input name="browsers" type="checkbox" value="xmosaic">

<li> Cello <input name="browsers" type="checkbox" value="cello">

<li> Viola <input name="browsers" type="checkbox" value="viola">

<li> Others? <input name="other browsers" size="48x4">

</ol>

A contact point for your site: <input name="contact" size="48">

<p>Many thanks on behalf of the WWW central support team.

</form>
```

[13]This and the *ismap* feature rely on the forthcoming definition of HTTP as an official Internet standard.

Floating Panels

The PANEL tag can be used to define panels or boxes which are free to float with respect to the standard flow of text. These are often used in magazine articles for asides on background material. The panel is typically shown with a distinctive background colour and border. The lay out software positions the panel to coincide with the page boundaries in printed media. For on-line use, panels can be rendered as pop-up windows. The body of the panel can be defined by a link to a separate document or included in the current document.

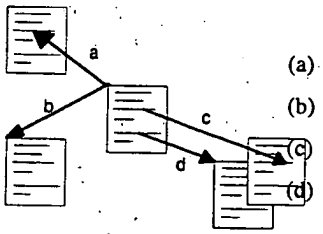
The following optional attributes are permitted with the <panel> tag:

id	An identifier, unique to this document, which can be used as a destination in a hypertext link.
at	An identifier elsewhere in this document. The panel mustn't be placed before this point. (Defaults to the current position if the *at* attribute is missing).
href	This attribute allows authors to fill the panel from a separate document, as specified by a URL. Note that the matching end tag: </panel> is always needed.

The text contained by the panel element can include any of the markup elements and looks like a separate document (panels themselves can't be nested). If the *href* attribute is used the text delimited by <panel> ... </panel> may be used as the caption for a pop-up. The *at* attribute allows you to include the panel definition at a convenient point in the HTML+ document, rather than interrupting the main flow of the document.

More on Links

Before describing the details of how links are represented in HTML+ it is worth looking more generally at the nature of hypertext links. First a terminological point: a *node* is the atomic unit for information retrieval, while *documents* may consist of one or more nodes, perhaps arranged as a hierarchy. A node may even be shared between several documents. Hypertext links start and end on nodes or anchor points within nodes.



The diagram illustrates the basic possibilities:

(a) Link from a node to an anchor

(b) Link from a node to a node

Link from an anchor to another anchor

Link from an anchor to a node

Links in HTML+ are represented with the LINK and A tags. The LINK tag is used for cases (a) and (b), while the A tag is used for cases (c) and (d). These links are held in the source node only, so there is a risk that the destination may have disappeared. Organisations can manage this risk by continued support for a few well published nodes (servers can use redirection to hide name changes). Links to other subsidiary nodes are at higher risk. This structured approach allows people to become familiar with the major routes through the web, without needing to worry about the minor routes.

In most cases URLs and URNs explicitly specify a node/anchor. The nodes may be explicit files or generated as the result of some process invoked by the server, e.g. a hypertext listing of a directory or a list of matches for a given search string. The search string can be explicitly encoded as part of a link, or dynamically defined by the user (see the ISINDEX tag, as described later on).

Links may be held separately from the source and destination nodes[14]. This is particularly appropriate for annotations and discussion groups. For example, consider making an annotation on a document held by a server located far away in another organisation. You could take a local copy and directly annotate it, but this is only appropriate for private use. The remote server might even support a protocol to add your annotations in place. More likely though, you will have to use an annotation server. This mechanism can be used to obtain a

[14]These correspond to HyTime's *ilink* architectural form.

copy of the document with the annotations inserted as hypertext links and shown as pop-ups or separate documents.

Context Dependent Links

For discussion groups, responses are made asynchronously, and include one or more references to other articles. In this situation, context dependent links are appropriate. The resolution to an explicit node can be carried out by either the client or server. The former approach is often appropriate, but requires special support, e.g. for network news and nntp.

Context dependent links are also useful for links to the table of contents for documents consisting of multiple nodes, when some of the nodes also appear in other documents. The appropriate table of contents for a given node will depend on which document is currently being viewed. In this case, the context will depend on how the current node was reached. This is quite simple to track if the links from the table of contents are differentiated from cross reference links.[15]

Hypertext paths are recommended routes through a set of nodes, and generally shown by next and previous buttons on a toolbar. Paths can be defined using explicit links in a node, or held separately in another node. The latter case once again, depends on the context. Paths and tables of contents all fall under the general category of navigating around a hierarchy of nodes forming a document too large or unwieldy to be held in a single node.

Types of Links

There are several motivations for differentiating between types of links:

how it is viewed	The potential to show different cues depending on the type and size of the node to be retrieved. If this information is explicitly stated as part of the link, there is the risk that it will become out of step with the linked node.
what happens	Whether the linked document replaces the current one, or appears in a new window, or as a pop-up overlay on top of the current one.
printed appearance	Whether links are treated as references, footnotes or as separate sections
effect on context	After traversing the link, will there be implicit values for the table of contents, and hypertext path etc?

Link Attributes

The A tag has the following attributes:

id	An identifier unique to this document which can act as a hypertext anchor
name	The same as *id* and included for backwards compatibility with HTML. New documents should use the *id* attribute for consistency with the other tags.
href	The URL or URN identifying the destination of the link.
role	A string giving the role of the link, e.g. `role="partof" or "annotation"`
effect	A string defining how the linked node is shown: "replace", "new", "overlay", with the default effect of replacing the current document.
print	How should the link be printed: "reference", "footnote" and "section", defaulting to "reference" (i.e. a footnote stating the link's URL).
title	The title to show when otherwise undefined for the node.
type	The MIME content type for the linked node for use with presentation cues.
size	The size in bytes for the linked node. This allows the browser to show a gauge indicating progress in retrieving long documents or images etc.

The LINK tag has only the *href* and *role* attributes.

---

[15]This is more general than deriving the role of the link from that of the node alone.

The *role* attribute is appropriate when context dependent properties such as table of contents (toc) are implied for the linked node, e.g. if the current node is a toc (as defined by the html or group tags) and the link has the role "partof", then the current node should act as the *toc* for the linked node. This property propagates down "partof" links, but not normal links. The *next* and *prev* properties are given by the sequence of "partof" links in the parent node. The *parent* property is only defined if the current node was reached via a "partof" link.

The LINK tag is used to express these properties in an explicit form, e.g.

```
<LINK href="toc.html" role="toc">
```

The recommended property names are:

toc	Table of contents for current node.
next	The next node in a hypertext path.
prev	The previous node in a hypertext path.
parent	The next level up in the hierarchy.
style	The style sheet appropriate to this node.

Style sheets provide a way for authors to express their detailed preferences for fonts, and layout, whether for the screen or when the node is printed out. A possible format is given in [Raisch 93].

The *effect* attribute is a hint and may be disregarded by browsers. It allows you to click on an image and to see a linked movie as an overlay at the same position. The browser tries to position the overlay at the same origin as the link. In some cases, the linked node is a description of the current node. By including `effect="new"`, the linked node will appear in a new window so that users can see both nodes at the same time. This hint should be used sparingly!

The *print* attribute makes it practical to print nodes along with relevant linked nodes. By default each link appears as a footnote stating the link's URL. Short nodes can be included in their entirety as footnotes, and longer ones as sections in their own right. This approach could be extended in future, to reorder the sequence of nodes from that defined by the position of the links in the source node, and to control the level that nodes appear as, e.g. chapter, section or subsection.

The *title* attribute is useful for nodes without titles of their own, e.g. Gopher menus. The *type* attribute can be used to show cues for the node type, e.g. iconic decorations[16]. The size attribute allows browsers to show a gauge on how much of a document has been retrieved at a any time. These attributes are liable to get out of step with the target node, and should be treated as hints only.

Groups

The GROUP tag allows you to define arbitrary groups, e.g. books, chapters, and sections. The *role* attribute is used to name the logical role of the group. You can use most markup elements inside a group element, including group itself. The *inset* attribute is a rendering hint to inset the left margin. Using the A tag with role="partof" allows you to designate a node as being included within the group, allowing hierarchies of groups which cross multiple nodes. See previous discussion of how properties are propagated.

Groups offer opportunities for presenting and searching documents at different levels of abstraction. For example, you might first describe a book by its title, author, publisher and ISBN number. The next level down could add a cover illustration together with a summary of the book's contents, some comments by reviewers and a short biography of the author. A number of books to be presented in an iconic form using a miniature version of the "cover page". Publishers could include copyright and other details in a standard place.

---

[16]The appropriate cue might also depend on the role of the link, e.g. for annotations browsers could show an icon of a drawing pin (as in attaching a note to a pin board). The colour of the pin could then vary according to the media type of the annotation.

Change Bars

Authors can indicate a part of a document has been changed using the CHANGED tag. This may appear anywhere that normal text is allowed (as designated by the entity reference `%text;` in the DTD). This tag signals the beginning or end of changes, which should be rendered by a vertical bar in the left margin. The tag can have one (but not both) of the following attributes:

id An identifier unique to the current document, which can also be used as a a destination for hypertext links. This signals the beginning of changes.

idref This must be an identifier matching the preceding changed element. It signals the end of changes. Note that you mustn't have both *id* and *idref* together.

Miscellaneous Tags

The remaining tags must appear at the start of the node like TITLE and LINK. They describe properties which apply to the node as a whole.

The HTML tag.

This is intended to provide short informal classifications for use in cataloging documents held by HTTP servers. The *role* attribute identifies the purpose of the node, for example `<html role="home page">`. Another common role is "toc" for table of contents. See previous discussion of link attributes.

The ISINDEX tag

This specifies that the URL designated with the *href* attribute is searchable (defaults to this document's URL). Browsers should allow users to enter a search string of one or more keywords. When the Return key is pressed the search string is appended to the designated URL, after a "?" character and sent to the server specified by the URL. Certain characters should be escaped as specified by the standard URL syntax, for example, the space character is mapped to "+". The newer HTTP protocol offers an alternative means for specifying that documents are searchable. In this case, the search string is sent as part of an RFC 822 style header. See [Berners-Lee 93c] for details.

The NEXTID tag

This is used by browsers that automatically generate identifiers for anchor points. It specifies the next identifier to use, to avoid confusionwith old (deleted) values, e.g. `<nextid n="id56">`. The identifier should take the form of zero or more letters followed by one or more digits. The numeric suffix should be incremented to generate successive identifiers.

The BASE tag

The *href* attribute gives the full URL of the document, and is added by the browser when the user makes a local copy. Keeping the original URL in a local copy is essential when subsequently viewing the copy as it allows relative URLs in the document to be resolved to their original references.

Note that one motivation for using relative URLs is to allow a group of documents to be copied without the need to alter any links between them. In this case, the BASE tag is inappropriate, since it would cause links to be intepreted as being to the original documents rather than their copies.

The HEAD and BODY tags

The HEAD tag can be used to delimit properties which apply to the document as a whole, and if used, must be present at the start of the document, followed by the BODY tag which then delimits the rest of the document.

Acknowledgements

I would like to thank the many people on the *www-talk* mailing list who have contributed to the design of HTML+ and to the management of HP Labs for their support during this work.

David Raggett, Hewlett Packard Laboratories, July 1993.

Email: `dsr@hplb.hpl.hp.com`, *Phone*: `+44 272 228046`

Appendix I - The HTML+ DTD

```
<!DOCTYPE HTMLPLUS [

<!-- DTD for HTML+ It assumes the default <!SGML> declaration

Markup minimisation should be avoided with the exception of </>
for the endtag. Browsers should be forgiving of markup errors.

Common Attributes:

   id       the id attribute allows authors to name elements such as
            headers and paragraphs as potential destinations for links.
            Note that links don't specify points, but rather extended
            objects.

            index allows authors to specify how given headers etc should
            be indexed as primary or secondary keys, where "/" separates
            primary from secondary keys, ";" separates multiple entries
-->

<!-- ENTITY DECLARATIONS

  <!ENTITY % foo "X | Y | Z"> is a macro definition for parameters and in
  subsequent statements, the string "%foo;" is expanded to "X | Y | Z"

  Various classes of SGML text types:

    #CDATA   text which doesn't include markup or entity references

    #RCDATA  text with entity references but no markup

    #PCDATA  text occurring in a context in which markup and entity
             references may occur.
-->

<!ENTITY % URL "CDATA" -- a URL or URN designating a hypertext node -->

<!ENTITY % text "#PCDATA|A|IMG|EM|EMBED|INPUT|SP|BR|CHANGED">

<!ENTITY % paras "P|PRE|FIG">

<!ENTITY % lists "UL|OL|DL">

<!ENTITY % misc "TBL|FORM|PANEL|GROUP">

<!ENTITY % heading "H1|H2|H3|H4|H5|H6">

<!ENTITY % table "%text;|P|%heading;|%lists;">

<!ENTITY % main "%heading;|%misc;|%lists;|%paras;|%text;">

<!ENTITY % setup "(TITLE? & HTML? &ISINDEX? & NEXTID? & LINK* & BASE?)">

<!--

<!ELEMENT tagname - - CONTENT> elements needing closing tags

<!ELEMENT tagname - O CONTENT> elements without closing tags

<!ELEMENT tagname - O EMPTY> elements without content or closing tags

   The content definition is:

      a)    an entity definition as defined above

      b)    a tagname

      c)    (brackets enclosing the above)

   These may be combined with the operators:

      A*    A occurs zero or more times

      A+    A occurs one or more times
```

```
      A|B   implies either A or B
      A?    A occurs zero or one times
      A,B   implies first A then B
-->

<!ELEMENT HTMLPLUS O O ((HEAD, BODY) | ((%setup;), (%main;)*))>

<!ELEMENT HEAD - - (%setup;)>

<!ELEMENT BODY - - (%main;)*>

<!-- Document title -->

<!ELEMENT TITLE - - (#PCDATA | EM)+>

<!ATTLIST TITLE
      id       ID      #IMPLIED -- link destination --
      index    CDATA   #IMPLIED -- entries for index compilation -->

<!-- Document role for cataloging documents held by servers -->

<!ELEMENT HTML - O (EMPTY)>

<!ATTLIST HTML role CDATA #IMPLIED -- home page, index, ... -->

<!-- Floating panel which can be moved around relative to the normal text
flow. Often rendered with a different background and possibly framed. The
panel can be anchored to a named point in the document as specified by
the AT attribute. The panel may be placed at that point or after, but not
before.
-->

<!ELEMENT PANEL - - (TITLE?, (%main;)*)>

<!ATTLIST PANEL
      id       ID      #IMPLIED -- defines link destination --
      at       IDREF   #IMPLIED -- anchor point --
      index    CDATA   #IMPLIED -- entries for index compilation -->

<!-- Document headers -->

<!ELEMENT (%heading;) - - (#PCDATA | EM)+>

<!ATTLIST (%heading;)
      id       ID      #IMPLIED -- defines link destination --
      index    CDATA   #IMPLIED -- entries for index compilation -->

<!-- logical emphasis with optional style hints -->

<!ELEMENT EM - - (%text;)*>

<!ATTLIST EM
   role   CDATA  #IMPLIED -- semantic category e.g. CITE --
   b      (b)    #IMPLIED -- render in bold font --
   i      (i)    #IMPLIED -- render in italic font --
   u      (u)    #IMPLIED -- underline text --
   tt     (tt)   #IMPLIED -- render in typewriter font --
   tr     (tr)   #IMPLIED -- render in serif (Times Roman) font --
   hv     (hv)   #IMPLIED -- render in sans serif (Helvetica) font --
   sup    (sup)  #IMPLIED -- superscript --
   sub    (sub)  #IMPLIED -- subscript --
   index  CDATA  #IMPLIED -- entries for index compilation -->

<!-- Paragraphs with different roles and optional style hints -->

<!ELEMENT P - O (%text;)+>

<!ATTLIST P
   id       ID         #IMPLIED -- link destination --
   role     CDATA      #IMPLIED -- semantic role --
   align    CDATA      #IMPLIED -- left, center or right --
   indent   (indent)   #IMPLIED -- indented margins --
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!ELEMENT BR - O EMPTY -- line break -->
```

```
<!ELEMENT SP - O EMPTY -- unbreakable space -->

<!-- Preformatted text with fixed pitch font, respecting original spacing
and newlines. Authors can also request proportional fonts. Further
control is possible with EM. -->

<!ELEMENT PRE - - (%text;)+>

<!ATTLIST PRE
   id       ID       #IMPLIED -- link destination --
   style    CDATA    #IMPLIED -- various styles --
   tr       (tr)     #IMPLIED -- serif (Times Roman) font --
   hv       (hv)     #IMPLIED -- sans serif (Helvetica) font --
   width    NUMBER   #IMPLIED -- e.g. 40, 80, 132 --
   index    CDATA    #IMPLIED -- entries for index compilation -->

<!-- Lists which can be nested -->

<!ELEMENT OL - - (LI | UL | OL)+ -- ordered list -->

<!ATTLIST OL
   id       ID        #IMPLIED
   compact  (compact) #IMPLIED
   index    CDATA     #IMPLIED -- entries for index compilation -->

<!ELEMENT UL - - (LI | UL | OL)+ -- unordered list -->

<!ATTLIST UL
   id       ID        #IMPLIED      -- link destination --
   compact  (compact) #IMPLIED      -- reduced interitem spacing --
   narrow   (narrow)  #IMPLIED      -- narrow perhaps multi columns --
   index    CDATA     #IMPLIED -- entries for index compilation -->

<!-- List items for UL and OL lists -->

<!ELEMENT LI - O (P|%text;)+>

<!ATTLIST LI
   id       ID        #IMPLIED
   src      %URL;     #IMPLIED -- icon for use in place of bullet --
   index    CDATA     #IMPLIED -- entries for index compilation -->

<!-- Definition Lists (terms + definitions) -->

<!ELEMENT DL - - (DT,DD)+ -- DT and DD *MUST* be paired -- > <!ATTLIST DL
   id       ID        #IMPLIED
   compact  (compact) #IMPLIED
   index    CDATA     #IMPLIED -- entries for index compilation -->

<!ELEMENT DT - O (%text;)+ -- term text -- >

<!ELEMENT DD - O (P|QUOTE|UL|OL|%text;)+ -- definition text -- >

<!ATTLIST (DT|DD)
   id       ID        #IMPLIED
   index    CDATA     #IMPLIED -- entries for index compilation -->

<!-- Tables with titles and column headers, e.g.

   <tbl border>
      <tt> An Example of a Table
      <th> <th s="2"> average <th> other <tr>
      <th> <th> height <th> weight <th> category <tr>
      <td> males <td> 1.9 <td> .003 <td> yyy <tr>
      <td> females <td> 1.7 <td> .002 <td> xxx
   </tbl>

-->

<!ELEMENT TBL - - (TT?, (TH|TD|TR|TB)*) -- mixed headers and data -->

<!ATTLIST TBL
   id       ID        #IMPLIED
   compact  (compact) #IMPLIED      -- if present use compact style --
   border   (border)  #IMPLIED      -- if present draw borders --
   index    CDATA     #IMPLIED       -- entries for index compilation -->

<!ELEMENT TT - O (%text;)+ -- table title -->
```

```
<!ATTLIST TT top (top) #IMPLIED -- place title above table -->

<!ELEMENT TH - O (%table;)* -- a header cell -->

<!ATTLIST TH
   colspan  NUMBER     1          -- columns spanned --
   rowspan  NUMBER     1          --. rows spanned --
   align    CDATA    #IMPLIED     -- left, center or right -->

<!ELEMENT TD - O (%table;)* -- a data cell -->

<!ATTLIST TD
   colspan  NUMBER     1          -- columns spanned --
   rowspan  NUMBER     1          --. rows spanned --
   align    CDATA    #IMPLIED     -- left, center or right -->

<!ELEMENT TR - O EMPTY -- row separator -->

<!ELEMENT TB - O EMPTY -- vertical break of 1/2 line spacing -->


<!-- Forms composed from input fields and selection menus

These elements define fields which users can type into or select with
mouse clicks. The browser should manage the input focus e.g. with the
tab/shift tab keys and mouse clicks.

The enter/return key is then taken to mean the use has filled in the form
and wants the appppropriate action taken:
   -  send as query/update to WWW server
   -  email/fax to designated person

The action is specified as a URL, e.g. "mailto:dsr@hplb.hpl.hp.com You
can specify additional mail headers with the MH tag:

<MH>Subject: Please add me to tennis tournament</MH>

Each FORM should include one or more INPUT elements which can be layed
out with normal and preformatted text, lists and tables.
 -->

<!ELEMENT FORM - - (MH, (%main;)*)>

<!ATTLIST FORM
   id       ID         #IMPLIED
   action   %URL;      #IMPLIED
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!ELEMENT MH - - CDATA -- one or more RFC 822 header fields -->

<!ATTLIST MH hidden (hidden) #IMPLIED -- hide the mail headers from view -->

<!-- INPUT elements should be defined within a FORM element.

Users can alter the value of the INPUT element by typing or clicking with
the mouse. Use radio buttons for selecting one attribute value from a set
of alternatives. In this case there will be several INPUT elements with
the same name. Attributes which can take multiple values at the same time
should be defined with checkboxes: define each allowed value in a
separate INPUT element but with the same attribute name. For checkboxes
and radio buttons, the value doesn't change, instead the state of the
button shown by the presence or absence of the checked attribute in each
element.

The size attribute specifies the size of the input field as appropriate
to each type. For text this gives the width in characters and height in
lines (separated by an "x"). For numbers this gives the maximum
precision.

-->
```

```
<!ELEMENT INPUT - O EMPTY>

<!ATTLIST INPUT
   name     CDATA     #IMPLIED -- attribute name (may not be unique) --
   type     CDATA     #IMPLIED --TEXT,URL,INT,FLOAT,DATE,CHECKBOX,RADIO--
   size     CDATA     #IMPLIED -- e.g."32x4" for multiline text --
   value    CDATA     #IMPLIED -- attribute value (altered by user) --
   checked  (checked) #IMPLIED -- for check boxes and radio buttons --
   disabled(disabled)   #IMPLIED    -- if grayed out --
   error    (error)   #IMPLIED -- if in error -->

<!-- Embedded Data

You can embed information in a foreign format into the HTML+ document.
This is very convenient for mathematical equations and simple drawings.
Images and complex drawings are better specified as linked documents
using the FIG or IMG elements.

Arbitrary 8 bit data is allowed but any occurrences of the following
chars must be escaped as shown:

      "&"      by      "&amp;"
      "<"      by      "&lt;"
      ">"      by      "&gt;"

The browser can pipe such data thru filters to generate the corresponding
pixmap The data format is specified as a MIME content type, e.g.
"text/eqn"
-->

<!ELEMENT EMBED - - (RCDATA)>

<!ATTLIST EMBED
   id       ID        #IMPLIED
   type     CDATA     #IMPLIED  -- mime content type --
   index    CDATA     #IMPLIED -- entries for index compilation -->

<!-- Figures

The image/drawing is specified by a URL or as embedded data for simple
drawings. The element's text serves as the caption. Use the emphasis with
style = "credits" to record photo credits etc.

-->

<!ELEMENT FIG - - (EMBED?, FIGD?, (FIGA|FIGT)*, (%text;)*)>

<!ATTLIST FIG
   id       ID        #IMPLIED
   align    CDATA     #IMPLIED -- position: left, right or center --
   cap      CDATA     #IMPLIED -- caption at left, right, top, bottom --
   noflow   (noflow)  #IMPLIED -- disables text flow --
   ismap    (ismap)   #IMPLIED -- server can handle mouse clicks/drags --
   src      %URL;     #IMPLIED -- link to image data --
   index    CDATA     #IMPLIED -- entries for index compilation -->

<!ELEMENT FIGD - - (%table;) -- figure description -->

<!-- Figure anchors designate polygonal areas on the figure which can be
clicked with the mouse. The default area is the whole of the figure. This
mechanism interprets mouse clicks locally, and browsers can choose to
highlight the designated area (or change the mouse sprite) when the mouse
is moved over the area.

Note that polygons may be non-convex or even intersect themselves,
thereby complicating the definition of what is enclosed by the polygon.
Holes are excluded.
-->

<!ELEMENT FIGA - O EMPTY>
```

```
<!ATTLIST FIGA
   href    %URL;     #REQUIRED -- link to traverse when clicked --
   area    NUMBERS   #IMPLIED -- x1,y1,x2,y2,x3,y3,... -->

<!-- FIGT Text on top of an figure background, or in a colored background
box which sits arbitrarily on top of an figure background. The text can
include headers, lists and tables etc. The width attribute allows you to
limit the width of the text box. The height is then determined
automatically by the browser.

FIGT can also be used to position a graphic on top of a picture using an
IMG element within FIGT. In this case the chromakey attribute may allow
parts of the underlying image to show through.

You can make the whole of the box into a hypertext link. This will act as
if it is underneath any hypertext links specified by the overlay markup
itself.
-->

<!ELEMENT FIGT - - (%main;)>

<!ATTLIST FIGT
   at      NUMBERS #IMPLIED -- upper left origin for text --
   width   NUMBER  #IMPLIED -- given as fraction of picture --
   framed  (framed)#IMPLIED -- framed with coloured background --
   href    %URL;   #IMPLIED -- link to traverse when clicked -->

<!-- inline icons/small graphics
The align attribute defines whether the top middle or bottom of the
graphic and current text line should be aligned vertically

The SEETHRU attribute is intended as a chromakey to allow a given colour
to be designated as "transparent". Pixels with this value should not be
painted. The exact format of this attribute's value has yet to be
defined.


Use the FIG tag for captioned figures with active areas etc.
-->

<!ELEMENT IMG - O EMPTY>

<!ATTLIST IMG
   src     %URL;   #REQUIRED -- where to get image data --
   align   CDATA   #IMPLIED  -- top, middle or bottom --
   seethru CDATA   #IMPLIED  -- for transparency --
   ismap   (ismap) #IMPLIED  -- send mouse clicks/drags to server -->

<!-- Hierarchical groups for books, chapters, sections etc. -->

<!ELEMENT GROUP - - ((TITLE|LINK*), (%main;)*)>

<!ATTLIST GROUP
   id     ID       #IMPLIED
   role   CDATA    #IMPLIED -- book, chapter, section etc. --
   inset  (inset)  #IMPLIED -- rendering hint: indent margins -->

<!-- change bars defined by a matched pair of CHANGED elements:

      <changed id=z34> changed text <changed idref=z34>

This tag can't act as a container, since changes don't respect
the nesting implied by paragraphs, headers, lists etc.
-->

<!ELEMENT CHANGED - O EMPTY>

<!ATTLIST CHANGED -- one of id and idref is always required --
   id       ID     #IMPLIED -- signals start of changes --
   idref    IDREF  #IMPLIED -- signals end of changes -->
```

```
<!-- Hypertext Links from points within document nodes -->

<!ELEMENT A - - (#PCDATA | IMG | EM | EMBED)*>

<!ATTLIST A
   id       ID       #IMPLIED -- as target of link --
   name     ID       #IMPLIED -- backwards compatibility --
   href     %URL;    #IMPLIED -- destination node --
   role     CDATA    #IMPLIED -- role of link, e.g. "partof" --
   effect   CDATA    #IMPLIED -- replace/new/overlay --
   print    CDATA    #IMPLIED -- reference/footnote/section --
   title    CDATA    #IMPLIED -- when otherwise unavailable --
   type     CDATA    #IMPLIED -- for presentation cues --
   size     NAMES    #IMPLIED -- for progress cues -->

<!-- Other kinds of relationships between documents -->

<!ELEMENT LINK - O EMPTY>

<!ATTLIST LINK
   href     %URL;    #IMPLIED -- destination node --
   role     CDATA    #IMPLIED -- role played, e.g. "toc" -->

<!-- Original document URL for resolving relative URLs  -->

<!ELEMENT BASE - O EMPTY>

<!ATTLIST BASE HREF %URL; #IMPLIED>

<!-- Signifies the document's URL accepts queries -->

<!ELEMENT ISINDEX - O (EMPTY)>

<!ATTLIST ISINDEX href %URL; #IMPLIED -- defaults to document's URL -->

<!-- For use with autonumbering editors - don't reuse ids, allocate next
one starting from this one -->

<!ELEMENT NEXTID - O (EMPTY)>

<!ATTLIST NEXTID N NAME #REQUIRED>

<!-- Mnemonic character entities. -->

<!ENTITY AElig "&#198;" -- capital AE diphthong (ligature) -->
<!ENTITY Aacute "&#193;" -- capital A, acute accent -->
<!ENTITY Acirc "&#194;" -- capital A, circumflex accent -->
<!ENTITY Agrave "&#192;" -- capital A, grave accent -->
<!ENTITY Aring "&#197;" -- capital A, ring -->
<!ENTITY Atilde "&#195;" -- capital A, tilde -->
<!ENTITY Auml "&#196;" -- capital A, dieresis or umlaut mark -->
<!ENTITY Ccedil "&#199;" -- capital C, cedilla -->
<!ENTITY ETH "&#208;" -- capital Eth, Icelandic -->
<!ENTITY Eacute "&#201;" -- capital E, acute accent -->
<!ENTITY Ecirc "&#202;" -- capital E, circumflex accent -->
<!ENTITY Egrave "&#200;" -- capital E, grave accent -->
<!ENTITY Euml "&#203;" -- capital E, dieresis or umlaut mark -->
<!ENTITY Iacute "&#205;" -- capital I, acute accent -->
<!ENTITY Icirc "&#206;" -- capital I, circumflex accent -->
<!ENTITY Igrave "&#204;" -- capital I, grave accent -->
<!ENTITY Iuml "&#207;" -- capital I, dieresis or umlaut mark -->
<!ENTITY Ntilde "&#209;" -- capital N, tilde -->
<!ENTITY Oacute "&#211;" -- capital O, acute accent -->
<!ENTITY Ocirc "&#212;" -- capital O, circumflex accent -->
<!ENTITY Ograve "&#210;" -- capital O, grave accent -->
<!ENTITY Oslash "&#216;" -- capital O, slash -->
<!ENTITY Otilde "&#213;" -- capital O, tilde -->
<!ENTITY Ouml "&#214;" -- capital O, dieresis or umlaut mark -->
<!ENTITY THORN "&#222;" -- capital THORN, Icelandic -->
<!ENTITY Uacute "&#218;" -- capital U, acute accent -->
```

```
<!ENTITY Ucirc "&#219;" -- capital U, circumflex accent -->
<!ENTITY Ugrave "&#217;" -- capital U, grave accent -->
<!ENTITY Uuml "&#220;" -- capital U, dieresis or umlaut mark -->
<!ENTITY Yacute "&#221;" -- capital Y, acute accent -->
<!ENTITY aacute "&#225;" -- small a, acute accent -->
<!ENTITY acirc "&#226;" -- small a, circumflex accent -->
<!ENTITY aelig "&#230;" -- small ae diphthong (ligature) -->
<!ENTITY agrave "&#224;" -- small a, grave accent -->
<!ENTITY amp "&amp;" -- ampersand -->
<!ENTITY aring "&#229;" -- small a, ring -->
<!ENTITY atilde "&#227;" -- small a, tilde -->
<!ENTITY auml "&#228;" -- small a, dieresis or umlaut mark -->
<!ENTITY ccedil "&#231;" -- small c, cedilla -->
<!ENTITY eacute "&#233;" -- small e, acute accent -->
<!ENTITY ecirc "&#234;" -- small e, circumflex accent -->
<!ENTITY egrave "&#232;" -- small e, grave accent -->
<!ENTITY eth "&#240;" -- small eth, Icelandic -->
<!ENTITY euml "&#235;" -- small e, dieresis or umlaut mark -->
<!ENTITY gt "&#62;" -- greater than -->
<!ENTITY iacute "&#237;" -- small i, acute accent -->
<!ENTITY icirc "&#238;" -- small i, circumflex accent -->
<!ENTITY igrave "&#236;" -- small i, grave accent -->
<!ENTITY iuml "&#239;" -- small i, dieresis or umlaut mark -->
<!ENTITY lt "&lt;" -- less than -->
<!ENTITY ntilde "&#241;" -- small n, tilde -->
<!ENTITY oacute "&#243;" -- small o, acute accent -->
<!ENTITY ocirc "&#244;" -- small o, circumflex accent -->
<!ENTITY ograve "&#242;" -- small o, grave accent -->
<!ENTITY oslash "&#248;" -- small o, slash -->
<!ENTITY otilde "&#245;" -- small o, tilde -->
<!ENTITY ouml "&#246;" -- small o, dieresis or umlaut mark -->
<!ENTITY szlig "&#223;" -- small sharp s, German (sz ligature) -->
<!ENTITY thorn "&#254;" -- small thorn, Icelandic -->
<!ENTITY uacute "&#250;" -- small u, acute accent -->
<!ENTITY ucirc "&#251;" -- small u, circumflex accent -->
<!ENTITY ugrave "&#249;" -- small u, grave accent -->
<!ENTITY uuml "&#252;" -- small u, dieresis or umlaut mark -->
<!ENTITY yacute "&#253;" -- small y, acute accent -->
<!ENTITY yuml "&#255;" -- small y, dieresis or umlaut mark -->

<!-- dash entities -->

<!ENTITY endash "--"  -- En dash -->
<!ENTITY emdash "---" -- Em dash -->

<!-- The END -->

]>
```

Appendix II - Entity Definitions

ISO Latin 1 character entities in HTML+ derived from "ISO 8879:1986//ENTITIES Added Latin 1//EN". The corresponding 8-bit character codes are given in the DTD.

Æ	capital AE diphthong (ligature)
Á	capital A, acute accent
Â	capital A, circumflex accent
À	capital A, grave accent
Å	capital A, ring
Ã	capital A, tilde
Ä	capital A, dieresis or umlaut mark
Ç	capital C, cedilla
Ð	capital Eth, Icelandic
É	capital E, acute accent
Ê	capital E, circumflex accent
È	capital E, grave accent
Ë	capital E, dieresis or umlaut mark
Í	capital I, acute accent
Î	capital I, circumflex accent
Ì	capital I, grave accent
Ï	capital I, dieresis or umlaut mark
Ñ	capital N, tilde
Ó	capital O, acute accent
Ô	capital O, circumflex accent
Ò	capital O, grave accent
Ø	capital O, slash
Õ	capital O, tilde
Ö	capital O, dieresis or umlaut mark
Þ	capital THORN, Icelandic
Ú	capital U, acute accent
Û	capital U, circumflex accent
Ù	capital U, grave accent
Ü	capital U, dieresis or umlaut mark
Ý	capital Y, acute accent
á	small a, acute accent
â	small a, circumflex accent
æ	small ae diphthong (ligature)
à	small a, grave accent
å	small a, ring
ã	small a, tilde
ä	small a, dieresis or umlaut mark
ç	small c, cedilla
é	small e, acute accent
ê	small e, circumflex accent
è	small e, grave accent
ð	small eth, Icelandic
ë	small e, dieresis or umlaut mark

í	small i, acute accent
î	small i, circumflex accent
ì	small i, grave accent
ï	small i, dieresis or umlaut mark
ñ	small n, tilde
ó	small o, acute accent
ô	small o, circumflex accent
ò	small o, grave accent
ø	small o, slash
õ	small o, tilde
ö	small o, dieresis or umlaut mark
ß	small sharp s, German (sz ligature)
þ	small thorn, Icelandic
ú	small u, acute accent
û	small u, circumflex accent
ù	small u, grave accent
ü	small u, dieresis or umlaut mark
ý	small y, acute accent
ÿ	small y, dieresis or umlaut mark

In addition, there are two entity definitions for horizontal dashes longer than the "-" character.

&endash;	En sized horizontal dash (--)
&emdash;	Em sized horizontal dash (---)

Appendex III - Compatibility with HTML

HTML documents can be easily converted into the HTML+ format, and only a few changes are needed. Most documents won't need any changes at all. HTML+ browsers should be able to view HTML documents with very little effort. Older browsers will be able to view HTML+ documents which don't contain, tables or forms.

Lists

<menu>	becomes	<ul compact>
<dir>	becomes	<ul narrow>

Emphasis

HTML+ replaces the various tags used by HTML with a single tag. It may be worth changing the name for the emphasis tag in HTML+ from EM to EM, to gain compatibility with this common form. However, using EM might be confused with the typographical term *em* as in em dash (you also get en dash). EM has the merit of being unambigous. **I would like to get peoples views on this.**

	becomes	
<tt>	becomes	<em tt>
	becomes	<em b>
<b>	becomes	<em b>
<i>	becomes	<em i>
<u>	becomes	<em u>
<code>	becomes	
<samp>	becomes	
<kbd>	becomes	
<var>	becomes	
<dfn>	becomes	
<cite>	becomes	

Miscellaneous

Some tags which are deprecated in HTML are now obsolete, and should be mapped to preformatted text:

<plaintext>	becomes	<pre>
<xmp>	becomes	<pre>
<listing>	becomes	<pre>

The following two tags have been absorbed into the standard mechanism for paragraphs:

<address>	becomes	<p role="byline" align="right">
<blockquote>	becomes	<p role="quote">

References

This is missing the appropriate references to work on the syntax and name service for URNs. The HTTP definition needs updating to cover the encoding of form data (and *ismap* ?).

[Berners-Lee 93a] "*Hypertext Markup Language (HTML)*", Tim Berners-Lee, March 1993.
URL=ftp://info.cern.ch/pub/www/doc/http-spec.ps

[Berners-Lee 93b] "*Uniform Resource Locators*", Tim Berners-Lee, January 1992.
URL=ftp://info.cern.ch/pub/ietf/url4.ps

[Berners-Lee 93c] "*Protocol for the Retrieval and Manipulation of Textual and Hypermedia Information*", Tim Berners-Lee, 1993.
URL=ftp://info.cern.ch/pub/www/doc/html-spec.ps

[Raisch 93] "*Style sheets for HTML*", Robert Raisch, June 1993, O'Reilly & Associates
email: raisch.ora.com

[Kimber 93] Article in comp.text.sgml newsgroup, 24th May 1993 by Elliot Kimber (drmacro@vnet.almaden.ibm.com),
URL=news:19930524.152345.29@almaden.ibm.com

A4

Intellectual Property Rights For Digital Library And Hypertext Publishing Systems: An Analysis of Xanadu

Pamela Samuelson

University of Pittsburgh School of Law

Robert J. Glushko

Hypertext Engineering
Pittsburgh, PA

ABSTRACT

Copyright law is being applied to works in digital form. The special character of digital media will inevitably require some adjustments in the copyright model if digital libraries and hypertext publishing environments are to become as commercially viable as the print industries have been. An intellectual property system works only when it embodies a reasonably accurate model of how people are likely to behave, but it is hard to predict author and reader behavior in an environment that has yet to be built. By far the most ambitious proposal for a digital library and hypertext publishing environment is Ted Nelson's Xanadu system. This paper reviews the intellectual property scheme in Xanadu and contrasts it with current copyright law. Xanadu's predictions about reader and author behavior are examined in light of how people currently behave in computer conferencing, electronic mail, and similar existing systems. These analyses identify some respects in which intellectual property systems might have to be changed to make digital libraries and hypertext publishing systems viable.

INTRODUCTION

An intellectual property system works only when it embodies a reasonably accurate model of how people are likely to behave. Copyright law is based on a relatively simple and straightforward model of author and reader behavior. Authors are assumed to be motivated to produce interesting and valuable texts, and to make these works available to others by copyright's reassurance that authors can control the sale of copies of their works. Readers are motivated to purchase the texts, or to urge institutions, such as libraries, to purchase the texts, so that they can have access to the work. Authors have generally had little control over what uses readers make of the copies after the first sale of the work to the public, and U.S. copyright law has sometimes regarded this lack of control over uses as a virtue. But while it can be said that the absence of use control promotes the dissemination of knowledge, the truth may be that in the print world it is infeasible to maintain meaningful control over uses anyway.

Copyright should be accounted a great success at modeling author and reader behavior, for the basic framework of this law has lasted nearly three hundred years. During this period, copyright industries have flourished and copyright law has broadened to include a wide variety of intellectual products besides those manufactured by printing presses.

Computers and the concomitant capability they have provided for making copyrighted texts available in digital form have created many new and exciting opportunities,

including the potential to create digital libraries and hypertext publishing systems. Active development of such systems is now underway ([Arms90]; [Enge90]; [Kahn88]; [Neuw90]). While there are many difficult technical problems that must be solved to build these systems, they are thought to be surmountable. Less clear, however, is what kind of intellectual property scheme is needed to make digital library or hypertext publishing systems commercially viable. While the copyright model is still being utilized for all manner of texts in digital form, the behavior of authors and readers is being changed by the new digital technologies. It is becoming increasingly likely that some adjustments will have to be made in the copyright model to make digital libraries and hypertext publishing environments as commercially viable as the print industries have been. But few new models have yet been constructed, and work in this direction has only just begun ([Kahn89]; [Zahr89]).

DIGITAL MEDIA AND INTELLECTUAL PROPERTY LAW

Elsewhere the first author has identified six characteristics of works in digital form that seem likely to change significantly the contours of copyright law [Samu90b]. The first and second of these, namely, the ease of replication of works in digital form, and the ease with which such works can be transmitted and accessed by multiple users, will create strong incentives for copyright industries to move away from their traditional focus on the sale of copies, and toward greater control over uses of protected works. That it is now feasible to control uses through controlling access to computer systems containing works in digital form will also affect this trend.

A third characteristic of digital works is the ease with which they can be manipulated and modified. While this plasticity offers users some important advantages over the print medium (printed works are sometimes *too* fixed to be maximally usable), copyright law is more experienced dealing with works that are permanently fixed. The law may need to be adjusted to cope with the new benefits and new problems that this plasticity will entail.

A fourth is that the traditional copyright distinctions among different kinds of works tend to break down when the works are in digital form. Federal copyright law recognizes seven categories of copyrighted works and provides each with different degrees of protection [USC88a]. Is a hypertext version of Mozart's "Magic Flute" that contains the music, the libretto, textual commentary, pictures of Mozart, and other media a "literary work", a "musical work", a "sound recording", a "pictorial work", or an "audiovisual work"? The answer to this question under copyright law cannot be all of the above--even if it is. Copyright's classification scheme, oriented as it is toward the appearance of works, seems in need of adjustment if the statutory differences are absent from the digital representation.

A fifth is that digital works are so compact as to be virtually invisible to user/readers. Consequently, user/readers are more dependent on user interfaces and navigation aids of a sort that the print world has not needed to provide. Intellectual property protection for interfaces and navigation aids are already a source of controversy, both on copyright and patent fronts, and seem likely to be more so in the future. Despite repeated Supreme Court rulings that algorithms are unpatentable [Samu90a] and evidence that practitioners believe strong protection by copyright and patent is bad for the software industry [Samu89], the U.S. Patent Office has been issuing many software patents in recent years. Many of these claim rights to certain functions and user interfaces for hypertext systems (see, for example, [Garb90]).

A sixth characteristic of digital media is the potential they provide for new search and linking activities, which may give rise to new classes of protected intellectual property products.

THE INTELLECTUAL PROPERTY SYSTEM IN XANADU

The most complete proposal for making digital library or hypertext publishing systems commercially viable has come from Ted Nelson, who coined the word "hypertext" and is often--and rightly--perceived as a hypertext visionary. For over two decades Nelson has been writing and talking about a proposed system called Xanadu, a vast digital library containing all of the world's literature [Nels87].[1] Because Xanadu will allow users to create new and derivative documents via links, Xanadu is also a hypertext publishing system. Xanadu can usefully be understood as an attempt to create an institution that will be writing environment, publishing environment, library, and bookstore in one.

Despite his visionary reputation, Nelson is practical enough to realize that the commercial success of the Xanadu proposal critically depends on the way it deals with intellectual property issues. The intellectual property system in Xanadu has sometimes been summarized in writings about the Xanadu system in popular magazines [Fraa87], but has been subject to little serious analysis.

After an introduction to the intellectual property system in Xanadu, this paper will discuss some respects in which the Xanadu proposal differs from the existing copyright system. While Xanadu contains some interesting ideas about how to solve certain problems with digital library and hypertext publishing systems, some aspects of the Xanadu model of author and user behavior may be unworkable. This analysis suggests some respects in which intellectual property systems might have to be changed to make digital libraries and hypertext publishing systems viable.

The Xanadu system builds on the foundation of copyright law, but goes beyond it to include some features that differ significantly from the standard copyright model. Nelson proposes to contract with all authors whose works are stored in the Xanadu system about derivative uses that can be made of documents in the system. Varying the "default setting" of copyright by contract is not, in itself, a novel thing. The motion picture industry is an example of a copyright industry that has historically depended for commercial success on contract-based distributions of copies, rather than on the outright sale of copies which has typified most copyright industries. Nelson's scheme is novel in proposing to use a contract-based scheme for commercial distribution of written texts, the prototypical subject matter of copyright.

Revenue and Royalty Incentives and Mechanisms

Revenues are generated in Xanadu from two sources: one, as author fees for renting space for their documents in the Xanadu system, and two, as user fees for their usage of the system. A portion of the usage fee (estimated at 10-20%) is to go to authors whose documents are accessed by users; the rest will go to the system to recoup costs and make profits. (If public domain documents, such as Shakespeare's plays, are accessed, the author portion of the fee will go into an "author's fund" for scholarships and the like.)

Nelson expects that authors will want to put their documents into the Xanadu system because once the documents are in the system, authors will be able to earn royalties whenever users make use of their documents. For the sake of administrative convenience,

---

[1] Nelson has described Xanadu in numerous publications, presentations, and interviews. Many of the publications have appeared in multiple editions, so it is hard to identify any one work as the definitive specification for Xanadu. Furthermore, Xanadu is being commercialized by Autodesk, a highly profitable firm with a track record of successful products. A commercial version is likely to differ from Nelson's vision, but it is instructive to consider Nelson's proposal in its "pure" form to understand some of the changes and compromises Autodesk is likely to make.

Nelson intends for the usage fees, and consequently the royalties as well, to be set on a per byte delivery basis. Nelson expects that people will pay to use the Xanadu system, because not only will it contain as much of the world's literature as Nelson can get into it, but there will also be legion opportunities in the Xanadu system for users/browsers/readers to make money by adding value to the system through their creative uses of the system.

There are two main ways Nelson intends to let users make money in the Xanadu system. One is by making derivative works of documents already in the system, such as new versions of other authors' documents, compound documents consisting of portions of a number of different documents, or commentaries on other documents in the system. By creating derivative documents, users would become system authors themselves, and thereby become able to earn royalties when other users access their derivative documents. No special permission would be needed to make derivative documents from other authors' documents, for Nelson will make it a condition of storing documents in Xanadu that authors agree to allow others to make whatever derivative uses they want of published documents in the system.

Nelson relies on two factors to motivate authors to agree to allowing derivatives to be made of their documents. One is that they will then be able to do to others' documents what others can do to theirs. But more importantly, when a third party accesses the derivative document on Xanadu, the author of the underlying document, as well as the author of the derivative document, will earn a royalty because the derivative document will be connected to the original document; bytes from both will be called up when third parties access the derivative document. Hence, both authors will receive royalties.

A second way for users to generate revenues when using the Xanadu system will be by creating links between (or among) documents in the system. Nelson expects some links to be very elaborate, such as a specialized index to certain classes of documents in the system; others may be modest, such as a connector between two documents. User links between documents, in effect, become new documents in the system. Each time other users traverse a set of links, the link author will receive a royalty, as will the authors of the documents on either end of the link. Although Vannevar Bush was the first to perceive that information trailblazers would be needed for computerized information systems [Bush45], Nelson deserves credit for recognizing the need to give incentives to information pioneers to cut paths through the invisible contents of a digital library.

Nelson's scheme would also provide authors with the opportunity to store private as well as published documents in the Xanadu system. Authors will be able to define who can have access to the private documents and under what conditions. Private documents can be withdrawn without difficulty from Xanadu by their authors. The same will not be true for published documents because of the effect withdrawal would have on the interests of authors who have linked to or otherwise built upon the foundation of the published document. Nelson attempts to create a strong incentive for authors to publish their documents in the Xanadu system by making system royalties unavailable to authors for private documents, even those with unrestricted distribution (i.e., from which derivatives can be made, and to which links can be constructed). Because publication imposes obligations on the Xanadu operator and the author, publication of a document in the Xanadu system is a formal event, requiring a signature of the author on a form affirming the intent to publish the work.

HOW THE XANADU INTELLECTUAL PROPERTY SYSTEM DIFFERS FROM THE COPYRIGHT SYSTEM

Nelson refers to copyright in a positive way in a number of passages in his book, and takes great care to establish a plausible case that nothing in Xanadu violates existing copyright law. Xanadu gives authors new ways to generate revenues from their works--even some that copyright might not provide--and so aims to create incentives to authorship, revealing a predisposition in keeping with traditional copyright incentives.

But the Xanadu system is more different from copyright than might be apparent from a cursory examination.

Accounting by Uses, not by Copies

One difference between the Xanadu intellectual property system and traditional copyright is that Xanadu aims to derive revenues for authors by charging for each and every *use* of their documents, rather than, as has traditionally been done in copyright industries, on the sale or other commercial distribution of *copies* of copyrighted works. Numerous other commercial computer data bases do much the same thing. Such arrangements seem likely to become increasingly common for works in digital form.

Blurring the "Idea" and "Expression" Distinction and Eliminating the "Fair Use" Provision

More novel are the set of differences from copyright that flow from Xanadu's treatment of links. Fundamental to the copyright regime is a distinction between "ideas" (which are unprotected by copyright) and "expression" (which is what copyright protects). Under the copyright regime, authors generally do not expect remuneration whenever other authors comment on, quote from, use ideas from, or make reference to their work. The "fair use" provision allows even literal copying of copyrighted text if the amount is small and for research, educational, or critical purposes. Only if other authors take a fairly hefty chunk of "expression" from the protected work do copyright holders expect compensation. Even for printed works, however, there is no exact boundary between "small" and "hefty" copying under fair use provisions, and some authors and publishers avoid the issue by obtaining rights to use even a handful of words.

In Xanadu, because information can be included in a document by linking, the definition of what information to count as a single work becomes unclear. This would complicate the determination of what constitutes fair use in any case. Nevertheless, Xanadu allows no fair use copying, and authors in the Xanadu system are expected to get varying royalties based on how many bytes were linked to, merely for being linked to. While Nelson presents arguments for this scheme, an intellectual property system that compensates authors without regard to whether chunks of expression have been appropriated may tend to undermine the "idea/expression" distinction that has been a staple part of the copyright system.

Treating Linking as Authorship

Nelson's decision to treat linking as a kind of authorship--an intellectual activity that should be encouraged, that should serve as the basis for earning royalties when users traverse the links, but that should not be controllable by authors of the documents being linked to--diverges somewhat from the traditional copyright model [Samu90b]. While an extensive set of links, such as an index, might readily be protectable by traditional copyright law as a compilation, many of the kinds of links that Nelson would treat as works of authorship might be unprotectable under traditional copyright law. A link between a passage in document A and a passage in document B might, for example, be considered a "discovery" that the statute says copyright cannot protect [USC88b]. Additionally, traditional copyright law would not regard it as a compensable use of a copyrighted work for readers to traverse the links among documents referred to in a printed article [Samu90b]. Yet link authors in Nelson's scheme would be compensated for link traversal.

Defining "Rights to Do" Rather than "Rights to Exclude"

It seems natural for people to think of intellectual property rights in terms of what authors should be able to get compensation for, what users should be able to do with documents in the system, and the like. This intuitive "rights to do" framework is used by

Xanadu. The law tends to define intellectual property rights in a somewhat different way. The law focuses on what rights owners have to *exclude* other people from doing certain kinds of things with the protected work. (The law, in general, tends to identify certain conduct as prohibited, leaving all else as legal conduct.)

Copyright law defines the ownership rights of authors by saying what kinds of activities they can stop unauthorized people from doing, which chiefly are: making copies of the work, making derivative works, and selling unauthorized copies or derivative works. The only exclusive right Xanadu seems to contemplate is whether or not to put a document into the Xanadu system in the first place. Xanadu is more like a compulsory license system than an exclusive rights system. While U.S. copyright law does contain some compulsory license provisions, compulsory licenses are generally an anathema to owners of intellectual property rights because the license fee generally bears little or no relation to what the market would bear if the issue were left to the market.

Extending the Duration of Rights Indefinitely

The Xanadu system seems to contemplate no end to the duration of author rights. As long as authors (or their heirs) continue to pay for storage on his system, Xanadu will to continue to pay royalties for uses of the documents. Copyright must, under the U.S. Constitution, only grant authors exclusive rights for limited times. Upon expiration of the copyright, the work is in the public domain. While Nelson may intend to include this aspect of copyright in the Xanadu system, he makes no mention of it. Certainly, he does not intend to reduce the usage fee for accessing public domain materials; royalties from them go into the "author's fund" over which he undoubtedly will exercise some control.

Making Publication a Formal Event (again)

Publication is an important formal event in the Xanadu system. Under "old" copyright law, an author only "copyrighted" his or her work when the work was published. Since 1976, federal copyright law has protected works of authorship from the moment of their first fixation in a tangible medium. Between 1976 and 1989, publication was mainly important because authors had to attach a copyright notice to published copies of the work. In 1989, this notice requirement was dropped, which made publication into a nonevent in copyright law. By making publication into a significant event, Nelson's scheme resembles "old" copyright more than "new" copyright.

Making publication a formal event in Xanadu is necessary because it creates a contract between the Xanadu operator and the author to guarantee the existence of the published document for a period of time. This provides an integrity to links and citations generally absent in the print world, where only law reviews, with their armies of student citation-checkers, assure the reader that the cited document exists and supports the proposition for which it was cited.

In short, the Xanadu intellectual property system is more different from copyright than one might think from reading Nelson's books. Nelson's insights about linking--the need to create incentives to do it, a willingness to treat linking as authorship and to treat the traversing of links by users as deserving of compensation to link authors, and the inability of authors to control who can link to their documents--are his most important and original contributions to current thought about how intellectual property issues should be handled for digital library and hypertext publishing systems. But some aspects of the Xanadu intellectual property system depend on assumptions about how authors and readers will behave that may be incorrect.

MODELS OF AUTHOR AND READER BEHAVIOR IN EXISTING COMPUTER INFORMATION AND MAIL SYSTEMS

How can one question a model of a system that has yet to be built? Some clues about how authors and readers might behave in digital libraries and hypertext publishing systems come from how people use computer bulletin boards, information services, and electronic mail systems. Instead of viewing these systems as technical precedents, it is instructive to consider them as experiments to develop appropriate models for intellectual property and human behavior to be applied to more ambitious applications like Xanadu.

Prodigy and CompuServe

Prodigy and CompuServe are commercial services that provide a variety of information services, bulletin boards, electronic mail, and entertainment. They embody significantly different intellectual property models and the behavior of their users is markedly different. Prodigy is targeted to the consumer and home market, and treats its users as relatively passive information consumers who do not interact much with each other. Prodigy is marketed in part for its entertainment value, and Prodigy's services are made available for a fixed monthly fee; usage-insensitive pricing is made possible by the paid advertising that Prodigy presents along with nearly every screen of information displayed by users. When Prodigy imposed a usage-pricing scheme for sending electronic mail, many users felt that their contract with Prodigy had been violated.

In contrast, CompuServe is oriented toward business and professional users and has always had usage-based pricing based on connect time. CompuServe information services are specifically focused, organized into a complex hierarchy of bulletin boards and databases, many of which are moderated by an expert, who in some circumstances is compensated by CompuServe. This finer-grained categorization enables CompuServe to impose surcharges for supposedly more timely or valuable information, but its user population is presumably used to paying for information according to its value. Users engage in heated electronic dialogues with each other on bulletin boards, commenting on and criticizing each other's postings.

The Internet

The Internet is a vast network of networks that interconnect thousands of computing sites in government, industry, and academia. The Internet has evolved from primarily providing electronic mail services to become the infrastructure for significantly broader services of information exchange and collaborative work. Like CompuServe, the heart of the Internet is a vast collection of newsgroups in which participants from around the world post and comment on messages. Some people take on the role of newsgroup moderators, but the overwhelming majority of newsgroups are unmoderated.

Author and reader behavior on the Internet are governed by norms or "netiquette" that have evolved over time and that are enforced both by system administrators and by the informal but effective sanctions of "flames" (critical messages) directed at violators. Included in these norms are rules about selecting newsgroups in which to post messages, choosing titles, sensitivity to authors of cited messages, and other topics that improve the lot of both authors and readers.

Users of the Internet vary greatly in their perception of intellectual property laws as they apply to this new kind of publishing system. Some users (especially new users who are college students) act as if the Internet services and the information it contains are completely free, and copyrighted material from newspapers or books often is posted

without permission.[2] At the same time, other authors explicitly assert copyrights on the messages that they post.

Authors are not paid to publish and receive no royalties, and readers do not have to pay to read, but it is fair to state that these activities are often being paid for (or at least subsidized) by their employers. Hence, it can be argued that anything posted on the Internet that is work-related is the intellectual property of the employer who provides access to the Internet by paying for the computers and telecommunications infrastructure. Employers may feel that the value of the information their employees glean from the Internet outweighs the costs of the time to obtain it, but it is unlikely that few employers explicitly make this analysis.

THE XANADU MODEL OF AUTHOR BEHAVIOR

One fundamental question raised by the Xanadu system is whether authors, particularly good ones, will be willing to pay to publish their works in Xanadu. Some authors may publish documents in Xanadu out of misplaced confidence in the value of their work, just as authors now post messages of dubious information content to CompuServe or Internet newsgroups. They, of course, will get feedback at the end of the first rental period when no royalties are credited to their account. Some authors also seem likely to decide not to renew their document rental space in Xanadu if no one linked to them during the first rental period, even though if they had stayed in the system, their documents would have eventually have been discovered and made them a fortune. Still other authors may lack confidence in their work or may be too poor to afford the rental fee, which may cause them to withhold from the Xanadu system documents that would have been widely utilized if published there. Xanadu might benefit from a scheme by which authors can solicit sponsors willing to subsidize the inclusion of their works in Xanadu in exchange for some portion of the royalties.

Authors may not, in other words, behave in the way Xanadu's designers might expect them to behave. Authors may prefer the print world's system which does not require authors to pay directly for the privilege of being published. Authors may feel it is quite enough to have had to work hard to write the text in the first place. Some of the trick of authoring is writing something that publishers are willing to risk their capital to publish. A system that would make authors pay to get published may end up either deterring authorship or sending authors in search of another digital library/hypertext publishing system in which to place their work.

The Xanadu model may also have underestimated how reluctant many authors may be about giving other people unlimited rights to make derivatives of their work. Although authors seem likely to have no objection to letting Xanadu users link to their documents, they are likely to feel quite differently about allowing any Tom, Dick, or Susan make their own versions of the authors' works, or to combine portions of their documents with portions of others' documents. It will be little consolation to such authors that they too might get royalties when the revised version or compound document was accessed by Xanadu users. Authors often regard their writings as expressions of their personalities. They tend to regard any tampering with their text as a "mutilation" of the work, as objectionable as if someone had the effrontery to walk up to you and cut your hair without your permission. In many countries, authors are expressly granted "moral rights" in their intellectual products, one of which protects the integrity of the work. In the U.S., the derivative work right of copyright owners protects authors' economic interests in controlling adaptations of their works. Nelson, like many members of the computer community, may have a much more positive attitude about taking someone else's work

---

[2] These same college students would likely be more sensitive to issues of plagiarism and infringement applicable to printed works when they write term papers.

and building on it to create a better modified version. Nelson seems to have assumed this attitude is more widespread in the authorial community than may, in fact, be true.

THE XANADU MODEL OF USER BEHAVIOR

The Xanadu intellectual property system is also based on a model of user behavior. Nelson has proposed for Xanadu a set of incentives for people to make use of the system for a wide variety of purposes, from research to entertainment to hobby to full-time occupation. Probably his most creative idea is that by which he contemplates transforming the digital library part of Xanadu into a hypertext publishing system, incenting users to become system authors through linking and other derivative uses of documents in the system.

But the royalty mechanism in Xanadu may create some unfortunate, unintended incentives. The system would seem to give an especial premium to those who are first to mention a particular topic in the Xanadu system, even if the first treatment of the topic was shallow or wrong. This may create incentives to rush documents into the system rather than to craft them to be deeper and more accurate.[3] An example will illustrate one such problem.

Suppose a journalist attended the first conference of scientists concerning the just-formed Human Genome Initiative, that he was an avid Xanadu user, and that at the first break in the conference schedule, the journalist authored a document for Xanadu describing in a shallow but intelligible way what HGI was about. By virtue of being the first to mention HGI in Xanadu, this journalist's entry might be, for a time at least, the most frequently linked to source on HGI in Xanadu, which would make him the most compensated author on the topic.

A naive user of Xanadu, when faced with a decision to access the journalist's HGI description or a later much deeper one by a scientist who was a founder of the HGI, might see that the first had been linked to a thousand times, whereas the scientist's document had been linked to only five times in the time it was on the system. This might cause the user to choose the more frequently cited source over the better but less frequently cited source, again causing more royalties to flow into the journalist's account, and incenting rushed documents over considered documents in the system. In the print world, the shallow first treatment on a topic will tend to be ignored by later authors, but in Xanadu, the first document to mention a subject might always be called up on a user search, and not until the user reads the shallow document (and hence pays the author royalties on it) will the user know to ignore it. Even creating a derivative document advising users to ignore the underlying document will result in royalties to the author of the underlying document.

Suppose that the journalist's Xanadu document on HGI contained some errors. Other Xanadu users might well notice the errors, and make derivative documents containing the needed corrections. Although this would correct the error, an inadvertent result of the scenario would be that the journalist might make a lot of money from putting out an erroneous document, for every time someone people linked to his document or created a revised version of it, the journalist would share in the revenues. The more, and more noticeable, were the errors in the document, the larger the number of Xanadu users likely to notice the errors, to link to his document, and/or revise it, which once again would

---

[3] This phenomenon is well-known in conventional publication media, of course. A visit to a bookstore or grocery store uncovers scores of slipshod books that report on the latest fad, war, movie, or entertainment personality. But Xanadu seems likely to increase the odds that first-in authors are rewarded because it doesn't allow readers to scan the work while waiting in line at the cash register to discover how shallow it really is.

generate more revenues for the journalist. This would seem to over-reward the journalist for rushing to get his document on HGI into the Xanadu system and not deter entry of erroneous information.

Usage-based systems, such as Xanadu, may also have the disadvantage, at least for price-sensitive users, of making those with the most curiosity and tenacity in research to pay the highest cost. They are the ones who will presumably use Xanadu for longer periods of time. Now, one might argue that this is fair because those who use the system the most are those who pay most. But some may conceive the issue differently, and think it one of the great virtues of the library systems of the print world that scholars do not have to pay more than casual users for access to the library. We want to encourage deep scholarship; by not making scholars pay more for their use of the library, the print world encourages scholarship. Digital library and hypertext publishing systems may also need to find ways to encourage good scholarship and curiosity without making it prohibitively expensive.

But a more serious problem perhaps than this may be figuring out how to motivate users to be persistent and creative in their use of the Xanadu system. It is difficult enough for ordinary folk to use libraries with print materials in it which they can walk around and browse through until they find something to interest them. In Xanadu, the clock will be ticking and the price will be rising as one browses. Digital libraries, because of their invisibility to the user, may be, for ordinary folk, too abstract to be enjoyably browsable. Once again, Nelson may have mistakenly modeled the Xanadu user in terms of his own persistence and creativity which others may not share.

QUESTIONS ABOUT PRICING INCENTIVES

But perhaps the single most questionable element of the Xanadu intellectual property scheme, from the standpoint of economic incentives, may be limitations of its pricing scheme. If one looks at the universe of copyrighted works in the print-dominated world, one will immediately observe that copies are priced according, more or less, to what the publisher/distributor and author/creator think the market will bear for the number of copies of it that it is reasonable to think can be sold or licensed. Xanadu posits a flat fee for Xanadu connect time and a fixed royalty for authors based on per byte delivery for certain kinds of usage of the document. This is like mandating that all books must be priced according to the number of pages they contain and all pages must be priced at the same amount. The CompuServe example seems to suggest that differential pricing of information is necessary to encourage the development of specialized markets. Unless Xanadu were the world's only digital library and hypertext publishing system, which remains Nelson's vision but which is unlikely, Xanadu will lack the negotiating power to compel authors to accept fixed pricing per byte of their information.[4]

People who own copyrights in very valuable intellectual properties simply won't use a system that won't let them make market-based pricing decisions. The only options authors of very valuable intellectual properties would have in the world Nelson envisions is to put the work in Xanadu as an encrypted private document and contract with users for access to the document, or to withhold the document from Xanadu altogether. While encryption might allow market pricing to occur, Xanadu does not facilitate these transactions; they are to be dealt with between the parties, but if Xanadu does not facilitate the transactions, it is difficult to see how they can occur. The transaction costs of individual negotiations which must occur outside Xanadu in order to access the

[4] If Xanadu were the only means for authors to publish their works, it would enjoy what economists call a monopsony, a situation with only one buyer for many sellers, which generally leads to exploitation.

encrypted document in Xanadu would seem inordinately high.[5] Nelson contemplates that authors or publishers of some valuable copyrighted works will choose not to put their documents in Xanadu. Nelson has an answer to this problem that may end up getting him in trouble if it works. Nelson says that it will still be possible for Xanadu users to link to and create derivative documents of works not stored in the Xanadu system. It would not be surprising if a copyright lawsuit was brought to stop such derivative activity.

CONCLUSION

Whether digital library or hypertext publishing systems can be made commercially viable will depend on how they deal with intellectual property rights issues. The traditional copyright model will require adjustments in order to facilitate these new kinds of institutions. Ted Nelson offers one model of how such adjustments might be made. While Nelson's intellectual property scheme for the Xanadu system is bold and innovative, there are a number of respects in which his system can be questioned. Most uncertain are the accuracy of the Xanadu model of author and user behavior, and the adequacy of financial incentives for authors to put their most valuable copyrighted works in the Xanadu system.

A generation of exposure to tape recorders and VCRs, and a raft of new digital technologies for scanning, frame grabbing, and sampling are making it harder to predict how people understand and relate to intellectual property. What is legal, and what is merely technically possible to copy? What constitutes "fair use" of digitally-encoded copyrighted works? Laws that were suited for traditional kinds of copyrighted works no longer seem to fit.

More work is needed to develop new models of author and user behavior and the economics that will yield the right level of incentives for creation of digital library and hypertext publishing systems. The law can be made to conform to these new models, but only after we figure out what the right ones are.

ACKNOWLEDGMENTS

We thank Mark Bernstein, Joe Farrell, Anna Belle Lieserson, James Moore, and several anonymous reviewers for their helpful criticism.

REFERENCES

[Arms90] Arms, C. (Ed.). *Campus strategies for libraries and electronic information.* Bedford, MA: Digital Press.

[Benn91] Benn, N. Copyright Collectives and Reproduction Rights in Electronic Media. *New Media News*, 5(1), 21-23, Winter 1991.

[Bush45] Bush, V. As we may think. *Atlantic Monthly*, July 1945. (Reprinted in Lambert & Ropiequet (Eds.), 1986, *CD-ROM - The New Papyrus*, 101-108. Microsoft Press.)

---

[5] Similar complaints about transaction costs for licensing of rights for digital media are motivating the development of new copyright collectives for electronic works modeled after ASCAP and BMI for the music industry [Benn91].

[Enge90] Engelbart, D. Knowledge-domain interoperability and an open hyperdocument system. *CSCW'90: Proceedings of the Conference on Computer-Supported Cooperative Work.* New York: ACM, 1990, 143-156.

[Fraa87] Fraase, M. Hyper-Mania, *The MACazine*, 64 (Nov. 1987).

[Garb90] Garber, S., Kozak, D., Kruse, J., & Clare, M. *Intelligent optical navigator dynamic information presentation and navigation system.* U.S. Patent 4,905,163, issued February 27, 1990.

[Kahn88] Kahn, R., & Cerf, R. *The Digital Library Project.* Corporation for National Research Initiatives, 1988.

[Kahn89] Kahn, R., & Cerf, V. Knowbots in the real world. *Workshop on the protection of intellectual property rights in the digital library system.* Corporation for National Research Initiatives, 1989.

[Nels87] Nelson, T. *Literary Machines* (Ed. 87.1).

[Neuw90] Neuwirth, C., Kaufer, D., Chandhok, R., & Morris, J. Issues in the design of computer support for co-authoring and commenting. *CSCW'90: Proceedings of the Conference on Computer-Supported Cooperative Work.* New York: ACM, 1990, 183-195.

[Samu89] Samuelson, P., & Glushko, R. Comparing the views of lawyers and user interface designers on the software copyright "look and feel" lawsuits. *Jurimetrics*, 30(1), Fall 1989.

[Samu90a] Samuelson, P. *Benson* Revisited: The Case Against Patent Protection for Algorithms and Other Computer Program-Related Inventions. *Emory Law Journal*, 39(4), Fall 1990.

[Samu90b] Samuelson, P. Digital Media and the Changing Face of Intellectual Property Law, *Rutgers Computer & Technology Journal*, 16, 323, 1990.

[USC88a] 17 U.S.C. sec 102(a), 1988.

[USC88b] 17 U.S.C. sec 102(b), 1988.

[Zahr89] Zahry, P., and Sirbu, M. The Provision of Scholarly Journals by Libraries via Electronic Technologies: An Economic Analysis. Engineering and Public Policy Department Technical Report, Carnegie-Mellon University, 1/16/89.

SoftLock Services Introduces

V

```
*************************************************************************
*      This message was requested from SoftLock Services' Email-Robot.     *
* * * * * * * * * *  We do not send "junk" mail. * * * * * * * * * * * * *
* If you receive any unsolicited or unwanted Email from SoftLock Services, *
*     please inform Jonathan Schull, Schull@SoftLock.Com, 215-993-9900     *
*************************************************************************
```

SoftLock Services introduces...SoftLock Services

Jonathan Schull, President

Schull@SoftLock.com

716-242-0348

SoftLock Services is a new kind of business for a new kind of product. While we offer an automated credit card processing service that will be a boon to Shareware authors (and many other people), the big news is our innovative suite of patent-pending Services and Tools for SoftLocking and selling freely-copyable computer programs and data.

In a nutshell, a SoftLocked computer program or document is a full-featured, freely copyable computer program or digital document, with a "lock" on certain "advanced features" (such as the ability to print, or to run at full speed). When a User tries to access one of those advanced features, the product displays a message like the following.

> In order to access the advanced features of this program in their full capacity, just...
>
> 1. Contact SoftLock Services any time, day or night, by telephone, modem, electronic mail, or fax.
>
> 2. Tell them you want a password for Product Number 87654321
>
> 3. Tell them the unique ID for that product on your computer is 12345678
>
> 4. Give them your credit card number (or SoftLock Voucher Number)
>
> And within 30 seconds, they'll give you a password that will unlock the advanced features, on this hard drive, forever!
>
> (And by the way, please give copies of this product to your friends. It will be of value to them, and if they want to unlock the advanced features they can just call us for the unique password they will need for THEIR hard drives).

MORE

All of this can be customized to suit the requirements of a particular product, but basically, that's all there is to it. In a typical SoftLocked application program, the developer might elect to SoftLock an advanced feature such as the ability to print professional-quality hard copy at full speed. In a SoftLocked electronic journal, some articles, and abstracts of all encrypted articles would be freely readable in any text browser or mail reader. Viewed through a SoftLock-aware text browser, an instant password could unlock one or more specific articles (and render this material readable, or printable, or modifiable, etc. at the author's discretion). Thus, SoftLocking can be used to strike the appropriate balance between "free sample" and "purchased product" all the while preserving the user's ability to backup, copy and pass on to others the entire SoftLocked application or document.

The business of SoftLock Services rests on three foundations:

1. SoftLock's (patent pending) technology: product- and machine- specific IDs can create a "lock" that requires a unique and unpredictable password available from SoftLock's central password "dispensaries".

2. A suite of commercial services, to make life easier for consumers and producers of SoftLocked products.

3. A philosophy: digital products should be virtually free until their value to the user is clear. Then, the creator of the product should be compensated fairly.

In keeping with this philosophy, our software tools are available to potential developers at cost, in order to create the market for our reasonably priced Services. By the same token, our client's products can be made available at nominal cost to Users who will (hopefully) find them useful when locked, and invaluable when unlocked. Users can "try before they buy" and then purchase passwords to instantly unlock any and all advanced features.

SoftLock's essential Service (and the one for which we take a modest commission) is the quick and convenient sale of Passwords and other products, round the clock, and around the world. Payment can be tendered on-line via credit card, or via SoftLock Vouchers through a variety of convenient channels. We provide information producers with sales, distribution, physical fulfillment and customer registration services and a variety of other support services, as well as prompt and accurate accounting and payment. We can provide these services to our Clients for far less than it would cost them to do it themselves.

SoftLock Services thus creates a whole new venue for the sale and distribution of freely copyable information. We do not pretend to know how our tools and services will best be exploited, but we are confident that we are in a position to help our clients and their customers find out.

With SoftLock Services...

Programmers and authors can now implement a business operation the same way one implements a computer operation -- by adding a few lines of code to their products. The resulting SoftLocked product can be cast upon the electronic waters, and (if Users find them valuable) checks for 80% of the retail cost will be deposited in the developers account.

Users can obtain software at little or no cost, "try before they buy", and unleash the full power of the software within minutes of deciding to make their purchase.

Software Producers can release complete programs and documentation, without copy protection, and still be assured that serious users will purchase the programs they use.

Hardware manufacturers can put computers and storage media into the hands of the people at cost (or below) by loading their goods with SoftLocked software and sharing in the profits from Password sales (by arrangement with SoftLock's clients). Thus, the computer becomes a "software vending machine", and digital technology becomes an inexpensive but invaluable commodity.

To make all of this possible we provide the following:

SoftLock Programmers' ToolKits

SoftLock Toolkits for C and Pascal programmers on PC and Macintosh computers, make it easy for programmers to create SoftLocked application programs that are unlocked by the Passwords we sell. Other ToolKits are under development, and SoftLock is eager to support those who would like to help produce them.

SoftLock Writers' Tools

Anyone who can use a word processor or spreadsheet can create a SoftLocked document. SoftLock Services has sponsored the development of freely-copyable SoftLock-aware text browsers and document-encrypting programs for DOS and Macintosh computers Using these and other tools we provide at nominal cost, SoftLocking a document can be done as easily as writing one.

Thus, the ability to SoftLock text files puts a virtual printing press, distribution channel, and sales operation into the hands of anyone with knowledge to sell.

SoftLock Fulfillment Services

We expect that many of our Clients' customers will want printed documentation, backup disks, etc. to follow up on their instant password purchases. SoftLock provides inexpensive on-demand fulfillment services for Clients who want to concentrate on what they do best -- create valuable digital products (not stuff envelopes).

SoftLock Customer's Tools

Privacy Enhanced Email and Digital Certification

There is now a standard for Privacy Enhanced electronic Mail which is so secure that it is subject to export restrictions by the US government. SoftLock Services is among the first mass-market information providers to empower users with the same kind of encryption technology used by banks and major corporations to protect their own business transactions. We will be making available to our customers two version of the RIPEM public-key encryption package: the original written by Mark Riordan, and a Macintosh version written by Ray Lau, author of the popular data compression program, Stuffit. SoftLock will honor and respond in kind to confidential messages, including credit-card based password purchases, through unsecured public Email channels. By endorsing and promoting this public standard, and by honoring Privacy-Enhanced electronic orders, SoftLock Services Inc., offers secure means for all of us to mind our own electronic business dealings. As usual (for us), all of this is free of charge. SoftLock Services will also provide "digital certificates" for attaching highly-secure "digital signatures" to PEM communications. Electronically "signed and sealed envelopes" are now a reality for everyone.

SoftLock communication tools

Our customers would be well served by a variety of communication tools to facilitate and automate their interactions with SoftLock Services. So we hereby invite developers to create some SoftLocked products for the purpose, which we will happily promote on their behalf, and of course sell under the auspices of our "standard deal".

Among the extensions we have planned are
--simple communications robots for the PC and Mac that will log in to the SoftLock Central bulletin board system, "squirt out" a pre-formatted order form, receive a password, and give the password back to the calling program (which will install it), and then hang up.
--a communications robot for bulletin board SYSOPs which will accomplish the same thing, and allow them to serve as on-line SoftLock resellers for their callers.
--an analogous internet-daemon to interact with SoftLock.com
--and so on.
Developers, get in touch!

SoftLock Vouchers

SoftLock Vouchers are analogous to "gift certificates" for SoftLocked Products. Each Voucher has a unique serial number and a specific monetary value, and can be used in lieu of a credit card number to purchase products from SoftLock Central. Each Voucher number is "retired" as soon as it is used. SoftLock Vouchers are useful in a variety of situations.

Software manufacturers can include a SoftLock Voucher in their shrink-wrapped packages, and thereby entitle their pre-paid customers to one (or more) pre-paid passwords.

Authorized SoftLock Resellers can sell SoftLock Vouchers to customers for whom credit card purchases are inconvenient or impractical. We offer quantity discounts on Vouchers to Authorized SoftLock Resellers in order to insure that customers anywhere in the world will be able to purchase passwords with local currency. Vouchers can be purchased and redeemed electronically, so we expect Authorized SoftLock Resellers to be able to help convey passwords to customers without convenient electronic access to SoftLock Central.

SoftLock Central

SoftLock Central is our virtual Customer Service headquarters, established to provide real-time processing of purchases from anywhere in the world at any time, through a variety of communication channels.

Touch-Tone Telephone. (1-800-SOFTLOCK)

Anyone with a touch-tone telephone can use our voice-response system to punch in a few numbers (ProductNumber, SoftLockID, Credit Card or SoftLock Voucher Number) and receive a password in approximately 60 seconds. Dongles and other consumer products can be ordered in a similar fashion.

Modem

Users with modems and communications software can log in to the SoftLock Central Bulletin Board System and purchase passwords on-line, order Dongles or ToolKits, etc. We are eager to encourage the development of automated communication tools that would log our system, submit SoftLock IDs and receive passwords

Email, Fax, and paper mail

SoftLock will respond within minutes to Email messages (including credit card and voucher-based purchases) addressed to SoftLock.com and to faxes sent to our Fax number. (We will also respond in kind to paper mail.) Many SoftLocked products will have the ability to generate order forms suitable for mailing or faxing to us. And as noted above, we are ready and able to respond to PEM-encrypted Email messages.

Voice

We don't want anyone left out. Customer Assistants at 215-993-9900 await your calls during business hours.

SoftLock Client Services

SoftLock's success depends upon the success of its clients -- the programmers and authors who produce SoftLocked products. We offer them a variety of services at minimal cost.

Developers can order ToolKits, from SoftLock Central, any time. They will soon be available at ftp sites and on Bulletin Board Systems worldwide.

SoftLock Certifications

SoftLock-aware text browsers must follow certain guidelines designed to protect the wishes of text-authors. SoftLock will only license SLX decryption technology to developers and products that are demonstrated to adhere to those guidelines.

SoftLock Resellers are enrolled in our certification program so that our customers can be confident they are buying valid SoftLock Vouchers from those able to provide appropriate customer support.

Certified Digital Signatures for Privacy Enhanced Mail are currently available at nominal cost to employees of many high-tech organizations. We will make this key to electronic security available to everyone else.

MORE.

SoftLock Central BBS Developer's Forum

This branch of the SoftLock Central BBS exists to register and serve SoftLock clients, and to provide them with the unique Product Numbers and Feature IDs they need to produce their own products. Developers without modem access to our bulletin board can receive the same services from our Developer's Assistants during business hours.

The BBS will allow Developers to:
- --set the prices of their products
- --see how their products are selling
- --access account information
- --collect user registration information
- --provide our fulfillment operation with documentation files, etc.
- --trade tips and referrals with each other and SoftLock Services.

The SoftLock Central BBS also hopes to be able to offer developers of large-volume products their own sub-forums on the BBS which they can use to offer support to their own customers.

We are eager to facilitate any other collaborative interactions on the internet or on the BBS that will put tools, inspiration and support into the hands of our clients, customers and collaborators.

In conclusion

We think we have created a new niche in cyberspace.

We do not pretend to know what will evolve in that niche.

We hope you'll join us.

[Submitted by: ANDREW WILLIAMSON (cijs26@vaxb.strathclyde.ac.uk)
Fri, 28 Jan 94 11:04 GMT]

Appendix III - Compatibility with HTML

HTML documents can be easily converted into the HTML+ format, and only a few changes are needed. Most documents won't need any changes at all. HTML+ browsers should be able to view HTML documents with very little effort. Older browsers will be able to view HTML+ documents which don't contain, tables or forms.

Lists

<menu>	becomes	<ul compact>
<dir>	becomes	<ul narrow>

Emphasis

HTML+ replaces the various tags used by HTML with a single tag. It may be worth changing the name for the emphasis tag in HTML+ from EM to EM, to gain compatibility with this common form. However, using EM might be confused with the typographical term *em* as in em dash (you also get en dash). EM has the merit of being unambigous. **I would like to get peoples views on this.**

	becomes	
<tt>	becomes	<em tt>
	becomes	<em b>
<b>	becomes	<em b>
<i>	becomes	<em i>
<u>	becomes	<em u>
<code>	becomes	
<samp>	becomes	
<kbd>	becomes	
<var>	becomes	
<dfn>	becomes	
<cite>	becomes	

Miscellaneous

Some tags which are deprecated in HTML are now obsolete, and should be mapped to preformatted text:

<plaintext>	becomes	<pre>
<xmp>	becomes	<pre>
<listing>	becomes	<pre>

The following two tags have been absorbed into the standard mechanism for paragraphs:

<address>	becomes	<p role="byline" align="right">
<blockquote>	becomes	<p role="quote">

References

This is missing the appropriate references to work on the syntax and name service for URNs. The HTTP definition needs updating to cover the encoding of form data (and *ismap* ?).

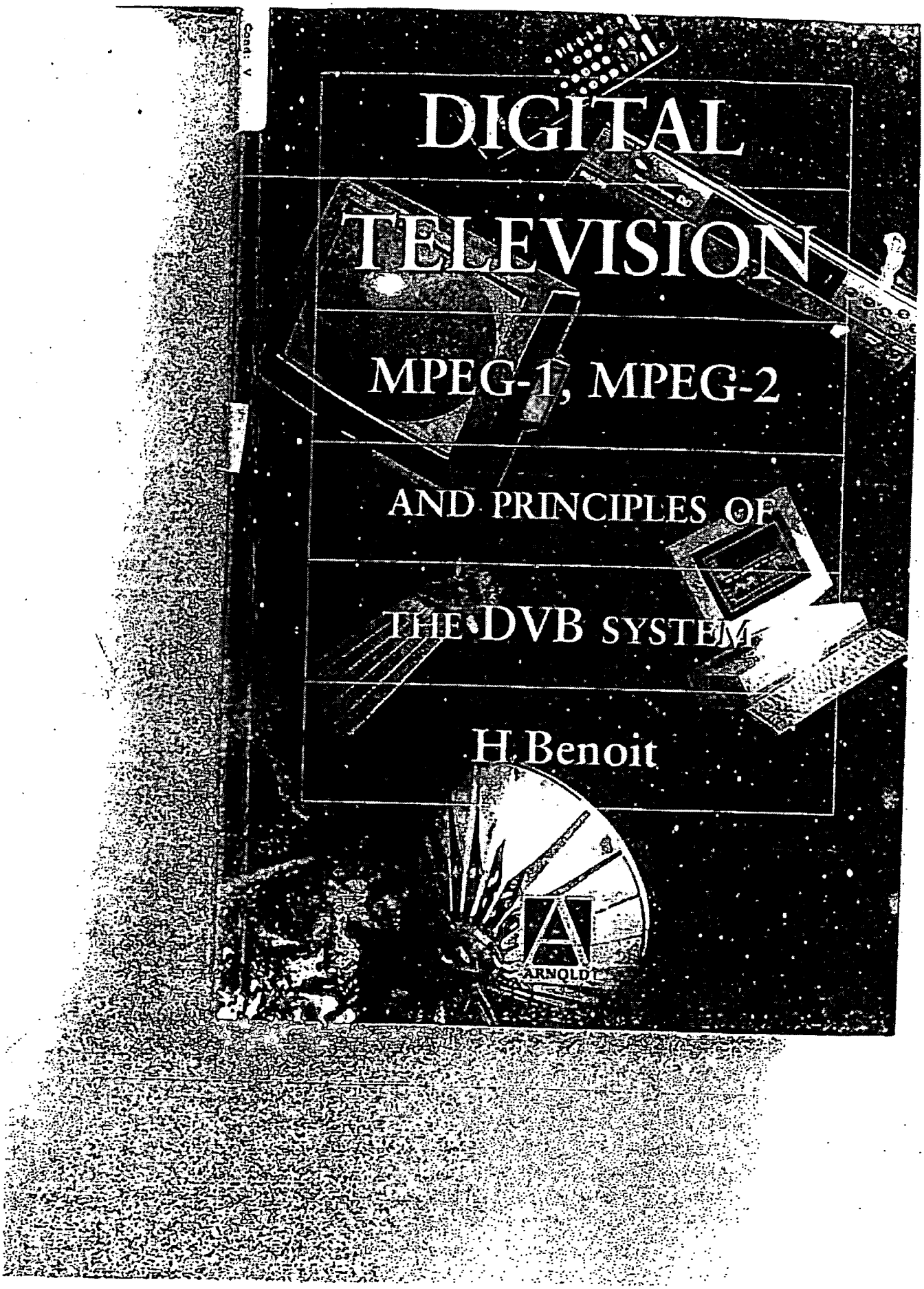
[Berners-Lee 93a] "*Hypertext Markup Language (HTML)*", Tim Berners-Lee, March 1993.
URL=ftp://info.cern.ch/pub/www/doc/http-spec.ps

[Berners-Lee 93b] "*Uniform Resource Locators*", Tim Berners-Lee, January 1992.
URL=ftp://info.cern.ch/pub/ietf/url4.ps

[Berners-Lee 93c] "*Protocol for the Retrieval and Manipulation of Textual and Hypermedia Information*", Tim Berners-Lee, 1993.
URL=ftp://info.cern.ch/pub/www/doc/html-spec.ps

[Raisch 93] "*Style sheets for HTML*", Robert Raisch, June 1993, O'Reilly & Associates
email: raisch.ora.com

[Kimber 93] Article in comp.text.sgml newsgroup, 24th May 1993 by Elliot Kimber (drmacro@vnet.almaden.ibm.com).
URL=news:19930524.152345.29@almaden.ibm.com


DIGITAL
TELEVISION
MPEG-1, MPEG-2
AND PRINCIPLES OF
THE DVB SYSTEM
H Benoit
ARNOLD

ısily the large number of

s not as quick as with
:hronization process can
the complex operations
top box, i.e.:

ıg part (only when chan-

the system clock of the

:eal decoding (this alone

5 Scrambling and conditional access

The proportion of free access programmes among analogue TV transmissions by cable or satellite is decreasing continuously, at the same time as their number increases; hence, it is almost certain that the vast majority of digital TV programmes will be pay-TV services, in order to recover as quickly as possible the high investments required to launch these services. Billing forms will be much more diversified (conventional subscription, pay per view, near video on demand) than what we know today, made easier by the high available bit-rate of the system and a 'return channel' (to the broadcaster or a bank) provided by a modem.

The DVB standard, as explained in the previous chapter, envisages the transmission of access control data carried by the conditional access table (CAT) and other private data packets indicated by the program map table (PMT). The standard also defines a common scrambling algorithm (CSA) for which the trade-off between cost and complexity has been chosen in order that piracy can be resisted for an appropriate length of time (of the same order as the expected lifetime of the system).

The conditional access (CA) itself is not defined by the standard, as most operators did not want a common system, everyone guarding jealously their own system for both commercial (management of the subscribers' data base) and security reasons (the more open the system, the more likely it is to be 'cracked' quickly). However, in order to avoid the problem of the subscriber who wishes to access networks using different conditional access systems having a stack of boxes (one set-top box per network), the DVB standard envisages the following two options:

1. **Simulcrypt**. This technique, which requires an agreement between networks using different conditional access systems but the same scrambling algorithm (for instance the CSA of the DVB), allows access to a given service or programme by any of the conditional access systems which are part of the agreement. In this case, the transport multiplex will have to carry the conditional access packets for each of the systems that can be used to access this programme.
2. **Multicrypt**. In this case, all the functions required for conditional access and descrambling are contained in a *detachable module* in a **PCMCIA** form factor which is inserted into the transport stream data path. This is done by means of a standardized interface (common interface, **DVB-CI**) which also includes the processor bus for information exchange between the module and the set-top box. The set-top box can have more than one DVB-CI slot, to allow connection of many conditional access modules. For each different conditional access and/or scrambling system required, the user can connect a module generally containing a smart card interface and a suitable descrambler.

The multicrypt approach has the advantage that it does not require agreements between networks, but it is more expensive to implement (cost of the connectors, housing of the modules, etc.). The DVB-CI connector may also be used for other purposes (data transfers for instance).

Only the future will tell us which of these options will be used in practice, and how it will be used.

5.1 Principles of the scrambling system in the DVB standard

Given the very delicate nature of this part of the standard, it is understandable that only its very general principles are available, implementation details only being accessible to network operators and equipment manufacturers under non-disclosure agreements.

The scrambling algorithm envisaged to resist attacks from hackers for as long as possible consists of a cipher with two layers, each palliating the weaknesses of the other:

- a *block layer* using blocks of 8 bytes (reverse cipher block chaining mode);
- a *stream layer* (pseudo-random byte generator).

The scrambling algorith
odd) alternated with a fre
make the pirate's task mo
control words is transmitt
(ECM) during the period
control words have to be s
descrambling device. Ther
could be used for free acc
little interest.

The DVB standard fores
different levels (transport
used simultaneously.

Scrambling at the transp

We have seen in the prece
packet header includes a 2
flags'. These bits are used
is scrambled and with whic
below.

Table 5.1 Meaning of transport

Transport_scrambling_flags
00
01
10
11

Scrambling at transport
the whole payload of the t
the multiplexer being 'in
only contain data coming f
scramble at transport level
part of a programme of th

Scrambling at the PES le

In this case, scrambling ge
multiplexing, and its prese
the 2 bit PES_scrambling_
format of which is indicat
possible options.

The scrambling algorithm uses two control words (even and odd) alternated with a frequency of the order of 2 s in order to make the pirate's task more difficult. One of the two encrypted control words is transmitted in the entitlement control messages (ECM) during the period that the other one is in use, so that the control words have to be stored temporarily in the registers of the descrambling device. There is also a *default* control word (which could be used for free access scrambled transmission) but it is of little interest.

The DVB standard foresees the possibility of scrambling at two different levels (transport level and PES level) which cannot be used simultaneously.

Scrambling at the transport level

We have seen in the preceding chapter (Fig. 4.6) that the transport packet header includes a 2 bit field called 'transport_scrambling_flags'. These bits are used to indicate whether the transport packet is scrambled and with which control word, according to Table 5.1 below.

Table 5.1 Meaning of transport_scrambling_flag bits

Transport_scrambling_flags	Meaning
00	No scrambling
01	Scrambling with the DEFAULT control word
10	Scrabling with the EVEN control word
11	Scrambling with the ODD control word

Scrambling at transport level is performed after multiplexing the whole payload of the transport packet, the PES at the input of the multiplexer being 'in the clear'. As a transport packet may only contain data coming from one PES, it is therefore possible to scramble at transport level all or only a part of the PES forming part of a programme of the multiplex.

Scrambling at the PES level

In this case, scrambling generally takes place at the source, before multiplexing, and its presence and control word are indicated by the 2 bit PES_scrambling_control in the PES packet header, the format of which is indicated in Fig. 4.4. Table 5.2 indicates the possible options.

Table 5.2 Meaning of PES_scrambling_control bits

PES_scrambling_control	Meaning
00	No scrambling
01	No scrambling
10	Scrambling with the EVEN control word
11	Scrambling with the ODD control word

The following limitations apply to scrambling at the PES level:

- the header itself is of course, not scrambled; the descrambling device knows where to start descrambling due to information contained in the PES_header length field, and where to stop due to the packet_length field;
- scrambling should be applied to 184 byte portions, and only the last transport packet may include an adaptation field;
- the PES packet header should not exceed 184 bytes, so that it will fit into one transport packet;
- the default scrambling word is not allowed in scrambling at the PES level.

5.2 Conditional access mechanisms

The information required for descrambling is transmitted in specific conditional access messages (CAM), which are of two types: entitlement control messages (ECM) and entitlement management messages (EMM). These messages are generated from three different types of input data:

- a *control_word*, which is used to initialize the descrambling sequence;
- a *service_key*, used to scramble the control word for a group of one or more users;
- a *user_key*, used for scrambling the service key.

ECM are a function of the control_word and the service_key, and are transmitted approximately every 2 s. EMM are a function of the service_key and the user_key, and are transmitted approximately every 10 s. The process for generating ECM and EMM is illustrated in Fig. 5.1

In the set-top box, the principle of decryption consists of recovering the service_key from the EMM and the user_key, contained

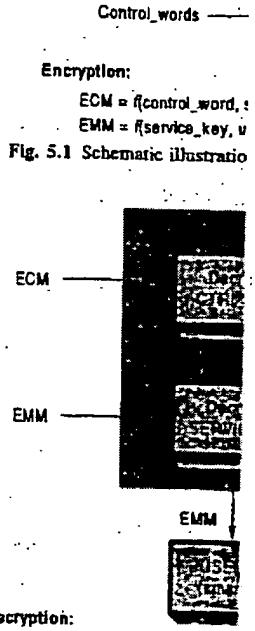


Fig. 5.1 Schematic illustratio

Fig. 5.2 Principle of decryption o

for instance in a smart c
decrypt the ECM in order
initialization of the descran
matically the process for re
and the EMM.

Fig. 5.3 illustrates the p
EMM required to descrar
gramme no. 3):

1. the program allocation
packets with PID = 0 >
packets carrying the prc

control bits

th the EVEN control word
th the ODD control word

mbling at the PES level:

nbled; the descrambling
ling due to information
d, and where to stop due

te portions, and only the
laptation field;

ed 184 bytes, so that it

ved in scrambling at the

nisms

bling is transmitted in
AM), which are of two
M) and entitlement man-
ages are generated from

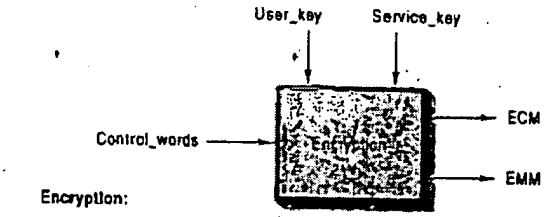
tialize the descrambling

ıtrol word for a group of

rvice key.

ord and the service_key,
2 s. EMM are a function
are transmitted approxi-
ating ECM and EMM is

yption consists of recov-
l the user_key, contained

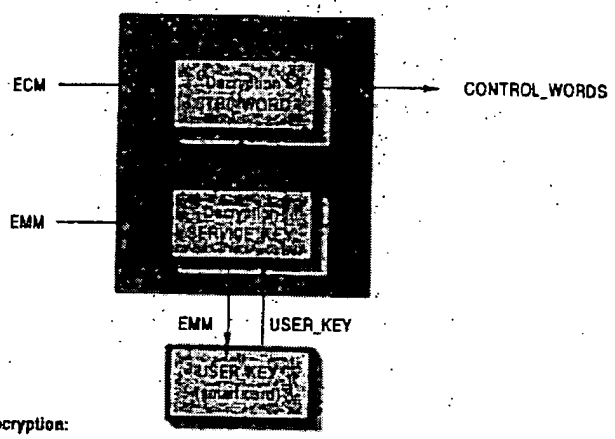


Encryption:

ECM = f(control_word, service_key)

EMM = f(service_key, user_key)

Fig. 5.1 Schematic illustration of the ECM and EMM generation process



Decryption:

control_word = f(ECM, service_key)

service_key = f(EMM, user_key)

Fig. 5.2 Principle of decryption of the control words from the ECM and the EMM

for instance in a smart card. The service_key is then used to decrypt the ECM in order to recover the control_word allowing initialization of the descrambling device. Fig. 5.2 illustrates schematically the process for recovering control_words from the ECM and the EMM.

Fig. 5.3 illustrates the process followed to find the ECM and EMM required to descramble a given programme (here programme no. 3):

1. the program allocation table (PAT), rebuilt from sections in packets with PID = 0 × 0000, indicates the PID (M) of the packets carrying the program map table (PMT) sections;

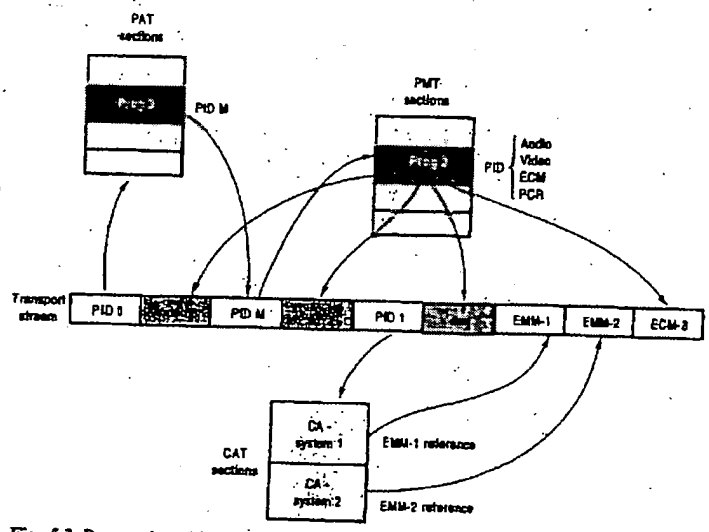


Fig. 5.3 Process by which the EMM and the ECM are found in the transport stream

2. the PMT indicates, in addition to the PID of the packets carrying the video and audio PESs and the PCR, the PID of packets carrying the ECM;
3. the conditional access table (CAT), rebuilt from sections in packets with PID = 0 × 0001, indicates which packets carry the EMM for one (or more) access control system(s);
4. from this information and the user_key contained in the smart card, the descrambling system can calculate the control_word required to descramble the next series of packets (PES or transport depending on the scrambling mode).

The above-described process is indeed very schematic; the support containing the user_key and the real implementation of the system can vary from an operator to another. The details of these systems are, of course, not in the public domain, but their principles are similar.

6 Channe (forwar correct

Once the source coding ope multiplexing and eventuall of 188 byte packets is availa a radiofrequency link (satel

We previously indicated unfortunately, not error-fre disturbances which can c interference, echoes). Hov when almost all its redunda low bit error rate (BER) fo of 10^{-10}–10^{-12}, correspon for a bit-rate of 30 Mb/s) called *quasi-error-free* (QI

It is therefore necessar modulation in order to all correction in the receiver c cal transmission channel. consist of reintroducing a (which obviously reduces t grouped under the terms 'channel coding' (this ter cess). Such measures will physical transmission med

The *virtual channel* thus encoder on the transmiss decoder on the receiver si referred to as a *super cha*

9 Future prospects

The fully digital era in consumer video applications is only just starting, and rapid and numerous changes in the services offered to the public are to be expected in the coming months. Due to the high investments required, it is understandable that these transmissions will start with pay TV services in nearly all countries.

However, even if the DVB standard has a powerful unifying role (27 countries had adopted the standard by mid-1996), in a very similar manner to that pioneered by the GSM standard for the mobile telephone some years ago, it could not impose a common conditional access control in the way GSM did at that time. As a result, a 'war of boxes' has already started between the various European and extra-European players, everyone trying to impose their technology in the field of conditional access and user interface (electronic program guide), and surprising alliances can sometimes be observed.

We will not, therefore, attempt to make any predictions in this field, as the risk of being contradicted by events taking place between the time of writing and the time of reading is far from negligible, and the person who could predict the winner would be very clever indeed. Nevertheless, we will try to list the main foreseeable technical changes in this field up to the turn of the century.

9.1 Terrestrial digital TV

Within Europe, the United Kingdom seems to be the first country willing to start a terrestrial service (as early as 1998, with the 2K

variant of DVB-T). This
the current PAL analog
coexistence between bot

One of the proposals f
'simulcasting' on the sam
of each of the four or five
one possibility would th
versions, one channel at

This would allow the
bandwidth, which could
multiplexes to increase
new bandwidth-hungry
mainly for cost reasons
OFDM modulation sche
the digital European ra
DAB). Its detractors wo
scheme similar to the
forward error correction

A similar approach is
HDTV 'Grand Alliance
replacement of analogu
However, the choice of
will certainly differ fron
states of AM modulatio
of ASK modulation wit
sidebands is almost co
halve the required banc

As long as the techn
ment of analogue tran
chance that the digital
TV set. As a first step,
ably be hybrid (capabl
missions), unless the 1(
beginning.

9.2 Evolution of

9.2.1 Functional i

The diagram in Fig. 8.
European IRD of the fi
market in the first half
corresponds to a majo

variant of DVB-T). This new system would eventually replace the current PAL analogue transmissions after 10–15 years of coexistence between both systems.

One of the proposals for the transition period would consist of 'simulcasting' on the same RF channel of 8 MHz a digital version of each of the four or five existing analogue channels; in this case, one possibility would then be to stop transmitting the analogue versions, one channel at a time, over a period of 15 years or so.

This would allow the progressive freeing up of an important bandwidth, which could be reallocated either to new digital TV multiplexes to increase the number of programmes, or even to new bandwidth-hungry communication services. However, mainly for cost reasons, there has been some criticism of the OFDM modulation scheme, already adopted some years ago for the digital European radio system (Digital Audio Broadcasting, DAB). Its detractors would prefer a less robust but cheaper QAM scheme similar to the one used for DVB-C with a reinforced forward error correction.

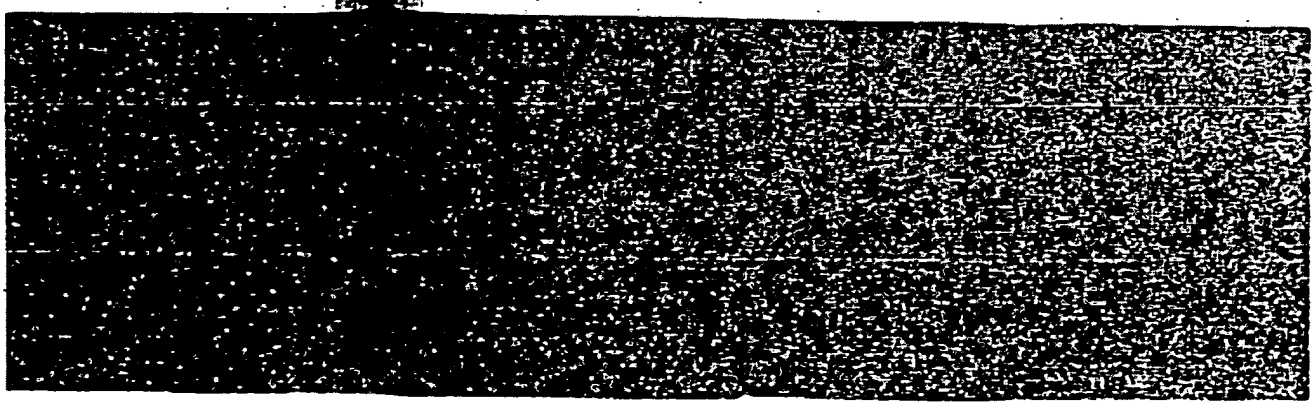
A similar approach is envisaged in the USA, where the digital HDTV 'Grand Alliance' project had as one of its objectives the replacement of analogue NTSC transmissions in the year 2008. However, the choice of the modulation for terrestrial and cable will certainly differ from DVB, since it will probably be 8-VSB (8 states of AM modulation with a vestigial sideband). This is a kind of ASK modulation with more than two states, where one of the sidebands is almost completely removed by filtering in order to halve the required bandwidth for transmission.

As long as the technical and political choices for the replacement of analogue transmissions remain unclear, there is little chance that the digital TV decoders will be integrated into the TV set. As a first step, the sets integrating digital TV will probably be hybrid (capable of receiving analogue and digital transmissions), unless the 100% simulcast approach is chosen from the beginning.

9.2 Evolution of the set-top box

9.2.1 Functional integration

The diagram in Fig. 8.2 represents the functional partitioning of a European IRD of the first generation (available on the commercial market in the first half of 1996), and each of its functional blocks corresponds to a major integrated circuit.



Most of these ICs are fabricated with a C-MOS technology, with geometries of 0.6 or 0.5 µ. As a result of the very rapid progress of the IC technology, we will soon be able to group these functions in circuits with 0.35 µ geometries according to the possible following scheme:

- reduction of the (S)DRAM packages from four of 256 K × 16 to one of 1 M × 16;
- use of a RISC processor integrating a part of the external interfaces (IEEE1284 etc.);
- grouping of the transport stream processing (demultiplexer and descrambler functions) with the RISC processor or perhaps the MPEG-2 audio/video decoder;
- use of a monochip for channel decoding (for satellite as well as for cable), grouping the demodulator and the error correction, and possibly the input ADC(s).

By 1988–1989, in addition to the power supply, the tuner and the necessary memory, such a decoder (a block diagram of which is shown in Fig. 9.1) could consist of three main ICs:

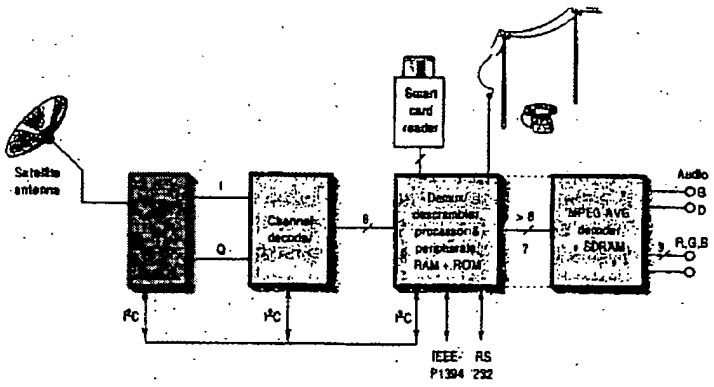


Fig. 9.1 Possible architecture of a DVB satellite receiver in 1997/1998 (dotted lines represent possible further integration options)

- channel decoder
- RISC processor with transport stream processing
- source decoder.

Integration will continue ever smaller geometries (0. the end of the century, to th PAL/RGB encoder with t processing. In addition, pr memory management will size required for decodin processor, leaving more r EPG or other application unthinkable that all the ac nel and source decoding) IC'. Only the power suppl of software would then be

9.2.2 Functional evo

In addition to the integrat mainly to reduce the cost added which will increa with the models at the e

- Interfaces to the exter that the analogue PA recording will be enha digital I/O, for instanc der (the **IEEE1394** int some important manu perhaps replace the cu is used for data interc
- For IRDs connected to preferred means of ac high speed *return cha* simple telephone line opening the door to co net access could the functions that could modification to the se TV set-top box are, am the international Digi
- Among the list of ne set-top box, the newl digital versatile disk) the existing MPEG-2

th a C-MOS technology,
result of the very rapid
on be able to group these
netries according to the

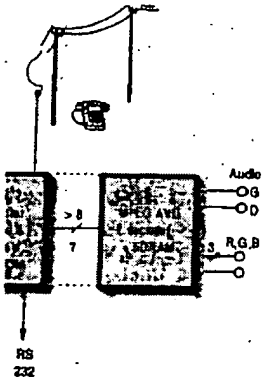
rom four of 256 K × 16

a part of the external

ssing (demultiplexer and
processor or perhaps the

g (for satellite as well as
and the error correction,

upply, the tuner and the
ck diagram of which is
main ICs:



er in 1997/1998 (dotted lines
options)

ocessing

Integration will continue inexorably after this step, by the use of ever smaller geometries (0.25 µ or less), which will lead, well before the end of the century, to the integration of the source decoder and the PAL/RGB encoder with the microprocessor and transport stream processing. In addition, progress in the decoding algorithms and in memory management will make possible a reduction in the memory size required for decoding, which could then be shared with the processor, leaving more room for high resolution graphics for the EPG or other applications. In much less than 10 years, it is not unthinkable that all the active functions (processor, memory, channel and source decoding) will be integrated into one single 'super IC'. Only the power supply, a tuner, this super IC and a fair amount of software would then be required to build a set-top box.

9.2.2 Functional evolution of the decoder

In addition to the integration of the existing functions, which aims mainly to reduce the cost of the basic IRD, new functions will be added which will increase the appeal to the consumer, starting with the models at the expensive end of the market:

- Interfaces to the external world will evolve, and it is probable that the analogue PAL or SECAM output used mainly for recording will be enhanced (and later replaced) by a high speed digital I/O, for instance, for connection to a digital video recorder (the **IEEE1394** interface seems to be the preferred choice of some important manufacturers, and when introduced it could perhaps replace the current parallel interface, IEEE1284, which is used for data interchange with a PC).
- For IRDs connected to a cable network, which could become the preferred means of access to the 'information superhighway', a high speed *return channel* using the cable network instead of a simple telephone line will provide a much better interactivity, opening the door to completely new services. High speed Internet access could then be one of the possible value-added functions that could be integrated without major hardware modification to the set-top box. These new features of a digital TV set-top box are, among others, being defined and specified by the international Digital Audio Visual Council (**DAVC**).
- Among the list of new functions which could be added to the set-top box, the newly standardized DVD (digital video disk or digital versatile disk) is a high priority, as it could make use of the existing MPEG-2 decoder to realize, at relatively low cost, a

combined IRD/DVD player (or even recorder) within the body of an existing VCR.

- Last, but not least, digital video broadcasting will ease the integration of TV and PC functions as techniques converge, and the new added functions will be more attractive due to the high data throughput offered by the transport channels. This will perhaps bring about the true home multimedia and multipurpose machine announced some years ago.

9.3 Other changes

9.3.1 The future MPEG-4 standard

Work continues within the MPEG group in order to elaborate new digital audio/video standards. The defunct MPEG-3, initially intended as the basis for a digital HDTV standard, was eventually included in the upper levels of MPEG-2. A new MPEG-4 committee has now been created, with the objective of defining a standard for audio/video coding at a very low bit-rate (from 10 kb/s to 1 Mb/s for moving pictures, and from 2 to 64 kb/s for the associated sound!).

This work is not, in principle, aimed at digital TV broadcasting, but at interactive multimedia applications at the interface between audio-visual and computer (radio)communications, such as videotelephony based on LAN, ISDN and standard telephone or radiotelephone networks. For these applications, MPEG-4 is going to define new coding principles, including being object oriented to ease editability and interactivity, and various scalability possibilities to allow easy adaption to a variety of transmission channel capacities.

The objective is to obtain an international standard by the end of 1998, which means that commercial applications based on MPEG-4 will not be available much before the year 2000.

9.3.2 New trends for signal processing

Despite the complexity of the algorithms used, the constant increase in the processing power of the new processors already allows purely software-based full-screen MPEG decoding in real time. Although this approach is definitely not economically attractive at present for a stand-alone set-top box, it is already of interest in some microcomputer based applications.

In recent years and months, new specialized processors have been developed which integrate a certain number of functional

blocks dedicated to mu
VLIW processor (audi
motion estimators, var
processors are generall
sors in a microcomputer
most probably appear f
ing, for instance, all the

The advantage of thi
that a well designed pi
reprogramming, in ver
same board could be
MPEG decoder or a n
pay being the amount o
application).

In conclusion, we wi
book if readers have a
view of the new techn
which will perhaps give
subject which is still in

In order to satisfy the
books much more de
aspects of this vast sub
paration) will most pro
The short bibliography
existing references.

/en recorder) within the body

broadcasting will ease the
ions as techniques converge,
ill be more attractive due to
d by the transport channels.
ie true home multimedia and
l some years ago.

ındard

group in order to elaborate new
ie defunct MPEG-3, initially
HDTV standard, was eventually
G-2. A new MPEG-4 committee
ective of defining a standard for
-rate (from 10 kb/s to 1 Mb/s for
kb/s for the associated sound!).
ed at digital TV broadcasting, but
ns at the interface between audio-
ıications, such as videotelephony
telephone or radiotelephone net-
G-4 is going to define new coding
oriented to ease editability and
/ possibilities to allow easy adap-
annel capacities.
ıternational standard by the end
ımercial applications based on
uch before the year 2000.

processing

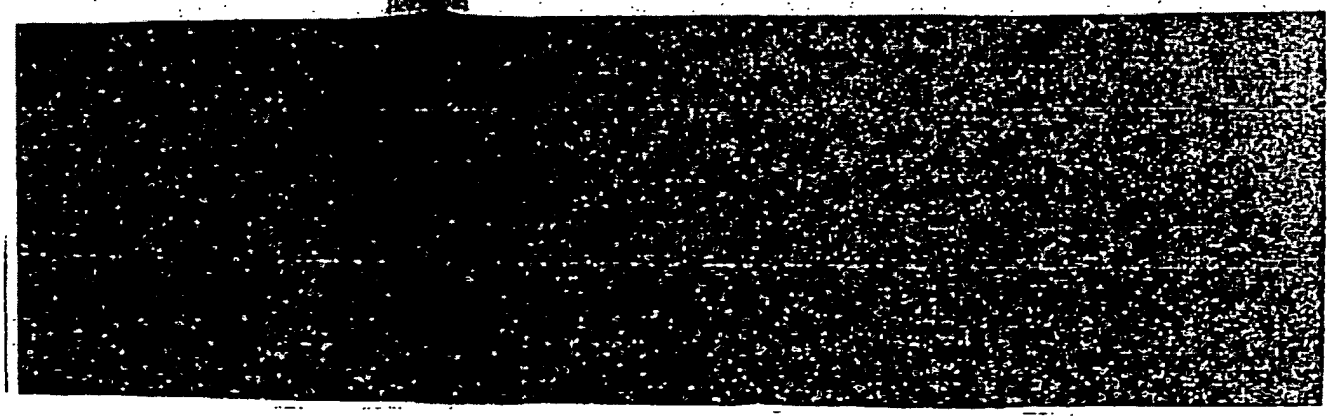
algorithms used, the constant
r of the new processors already
l-screen MPEG decoding in real
is definitely not economically
-alone set-top box, it is already
er based applications.
iew specialized processors have
a certain number of functional

blocks dedicated to multimedia processing on top of a RISC or VLIW processor (audio, video and communication interfaces, motion estimators, variable length coder/decoder, etc.). These processors are generally used as dedicated multimedia coprocessors in a microcomputer environment, but specific derivatives will most probably appear for use as stand-alone processors, combining, for instance, all the control and decoding functions of an IRD.

The advantage of this approach is its very high flexibility, so that a well designed piece of hardware will be usable, by simple reprogramming, in very different applications (for instance, the same board could be used as a videoconferencing codec, an MPEG decoder or a music editing sytem, the only penalty to pay being the amount of memory required for the most demanding application).

In conclusion, we will have reached our goal at the end of this book if readers have acquired a (hopefully clear enough) global view of the new techniques used in digital television systems, which will perhaps give them the desire to investigate further this subject which is still in its infancy.

In order to satisfy the thirst for knowledge of interested readers, books much more dedicated to and specialized in particular aspects of this vast subject (including the new standards in preparation) will most probably be published in the coming months. The short bibliography given at the end provides some of the existing references.



AZ

Deposit, Registration and Recordation in an Electronic Copyright Mangement System

by Robert E. Kahn

ABSTRACT

This document proposes the development of a testbed for deposit, registration and recordation of copyright material in a computer network environment. The testbed will involve the Library of Congress and provide for electronic deposit of information in any of several standard formats, automated submission of claims to copyright, notification of registration and support for on-line clearance of rights in an interactive network. "Digital signatures" and "privacy enhanced mail" will be used for registration and transfer of exclusive rights and other copyright related documents. Electronic mail will be used for licensing of non-exclusive rights with or without recordation. Verification and authentication of deposits can be carried out within the testbed using the original digital signatures. A system of distributed redundant "Repositories" is assumed to hold user deposits of electronic information. The testbed provides an experimental platform for concept development and evaluation, a working prototype for system implementation and a basis for subsequent deployment, if desired.

INTRODUCTION AND BACKGROUND

Deposit, registration and recordation of copyright material and its associated claims to rights have generally been handled manually. Over the past two decades, the economics of information technology has enabled an electronic foundation for such material and claims. The key elements of this foundation are the personal computers, workstations, computer networks and peripheral devices such as scanners, printers and digital storage systems which have now become sufficiently powerful and cost effective to be put into widespread use. It is now essential that the underlying systems used to manage copyright be conformed to be compatible with the promise of this new computer networking environment. This paper addresses several essential steps that should now be taken to facilitate that process.

In the current manual system, claims to copyright are registered with the Copyright Office, Library of Congress. Deposits are accepted and stored in physical form including tapes and diskettes as well as paper and other substances. Notification of registration is also made in physical form. In addition, documents transferring copyright ownership and other documents pertaining to copyright may be submitted to the Copyright Office for recordation. While an on-line record of recent registrations and recordations may be accessed at the Copyright Office, there is only limited external dissemination of this information in electronic form for access at remote sites.

This approach requires considerable physical storage at the Library of Congress for deposited materials which can only increase over time. Materials stored in physical form will slowly degrade unless deposited in digital media in which case the contents may be reproduced subsequently without loss of information but at some cost for duplication. Even if it is available digitally, much, if not most, of this material will not generally be accessible on-line from any source. Rights to use the information in a computer network environment cannot usually be acquired easily or quickly, even if the identity of the rightsholder is accurately known. Fortunately, these limitations can also be overcome with the use of information technology and only minor modification to the current manual system.

COMPONENTS OF THE PROPOSED SYSTEM

This document proposes building a testbed to develop and evaluate key elements of an electronic copyright management system. These elements include:

a. Automated copyright registration and recordation

b. Automated transactional framework for on-line clearance of rights

c. Privacy enhanced mail and digital signatures to facilitate on-line transactions

d. Methodology for deposit, registration, recordation and clearance

Current registration and recordation activities of the Library of Congress would be maintained and enhanced in the proposed testbed. It provides for repositories and recordation systems both within and without the Library of Congress, which would serve as agents for authors and other copyright owners which seek to register works with the library. In addition, the testbed provides for automated rights clearance, outside of but linked to the library, which would accelerate permissions and royalty transfers between users and rightsholders.

Electronic Copyright Management Testbed

A testbed is proposed to develop and evaluate these concepts and to obtain experience in the implementation and operation of an experimental system (see Figure 1). The proposed testbed consists of a Registration and Recording System (RRS), a Digital Library System (DLS) and a Rights Management System (RMS). The RRS will be operated by the Library of Congress and will permit automated registration of claims to copyright and recordation of transfer of ownership and other copyright related documents. The RRS would also provide evidence of "chain of title." The DLS will be a distributed system involving authors, publishers, database providers, users, and numerous organizations both public and private. It will be a repository of network accessible digital information and contain a powerful network based method of deposit, search and retrieval. The RMS will be an interactive distributed system that grants certain rights on-line and permits the selective use of copyright material on the network.

Information may be stored in the DLS, located within the DLS and retrieved from the DLS using any of several mechanisms such as file transfer, electronic mail or agents such as Knowbot programs. Material may be imported into the DLS from other independent systems, from paper and other sources or exported from the DLS to other independent systems, to paper or to other materials such as CD-ROM, DAT, and microcassettes. The electronic copyright management system described in this document would be directly linked to the DLS.

The testbed would contain a digital storage system connected to an applications gateway (which is, in turn, connected to multiple communication systems including the Internet) to which documents would be submitted. The storage system would constitute an experimental repository for information. The applications gateway would be designed to support multiple access methods including direct login. The RRS and RMS would be servers connected to the Internet. Initially, they would be on a common machine, but they could later be easily separated. After development, the RRS would be relocated to the Library of Congress or its designated agent prior to being placed in operation. After initial implementation, the repository and the RMS would be replicable at other sites.

Electronic Bibliographic Records

An electronic bibliographic record (EBR) is created by the user for each digital document submission and supplied with the document for registration. The EBR is also suitable for use in cataloging and retrieval. The EBR may be supplied to other systems without the actual document but with a pointer to it. The EBR must contain a unique name for the document per author. If a name is provided that has already been used by the same author, it will be rejected with an explanation. An acknowledgment of deposit will be returned to the user along with a unique numerical identifier and a retrieval pointer to the document, and, in the event of a claim to copyright, a certificate of registration from the RRS.

Claims Registration

When the EBR indicates a claim to copyright, the RRS will be supplied a copy of the EBR by the repository along with a digital signature (to be described shortly) that can be used to verify the accuracy of a deposit at a later time. The actual work would remain in the repository. The digital signature consists of a few hundred bytes of data and is approximately the size of the EBR. It should allow the authenticity of the retrieved document to be formally established at any time for legal and other purposes.

Repositories

The RRS need not be collocated with a repository. It is expected that an operational RRS would be operated by the Library of Congress. The repositories would be operated by the Library of Congress as well as other organizations or individuals. Deposits in certain

qualified repositories will constitute deposit for public record purposes. The Library of Congress will maintain its own repository of selected deposits.

Although a set of distributed repositories is envisioned for a widely deployed system, the proposed testbed will only have a single repository for experimentation. The repositories would be established in such a way as to insure the survival of the deposited information with perhaps different degrees of confidence (much like the treasury, banks and brokerage houses, for example). Certain information would probably not be deposited for purposes of registration and might be stored at the users local site or in a commercial repository. Highly valued information could be stored in rated repositories (5-star down to 1-star) with varying degrees of backup and corresponding costs. The most critical information, as determined by Copyright Office regulations, might be stored at the Library of Congress or the National Archives as a safeguard. The structure of such a system of repositories should be developed as part of the project.

The advantages of a distributed repository system are:

1. Large amounts of physical storage is not required to be made available at the Library of Congress.
2. Access to the original documentation is guaranteed by the DLS to the confidence level selected by the user's choice of repository (again like the banks).
3. Repositories serve as interfaces to the users, thus offloading and insulating any central servers and systems such as the RRS from potentially large user loadings and specialized customer service requests.
4. Access to the RRS in transaction mode is available only to authorized repositories and RMSs that are qualified to use the RRS in that mode. An individual author, a collective licensing organization, a government or corporate entity or others may run an RMS. Authors and other copyright owners, as well as users may also connect directly to the RRS through a separate interactive user interface.

The Computer Network Environment

There are three specific actions of concern in a network environment. One is the movement of information already contained in a computer network environment thereby greatly facilitating the creation of multiple copies in multiple machines in fractions of a second. The second is the importation of external information, such as print material or isolated CD-ROM based material, which must first be scanned or read into the system before it can be used. The third is export of internal network based information to paper using digital printers or facsimile machines or copied to separable media such as tape or DAT for external transport to others. Some of these actions, such as local use on paper in very small quantities, may or may not be covered by fair use provisions. However, non fair use actions would require approval of rightsholders.

In addition to the above three actions, there is a fourth action that is facilitated by the computer network environment. Information in digital form has the property of being easily manipulated on a computer to produce derivative works. Such derivative works can also

be easily moved about in a computer network environment and be subject to further manipulation by other parties. The technology makes it possible for parallel and concurrent manipulation of such information to result in an exponential proliferation of such derivative works.

Rights Management System

The four actions described above form a basis for a rights management system. In general, there will be many such systems operated by rightsholders or their agents for required permissions on either an exclusive or non-exclusive basis for a given type of action. To locate an RMS, a user requires the existence of a domain server that knows about the network names and addresses of all RMS servers. Transactions involving rights may be handled by direct exchange on-line between the user system and the corresponding RMS. Typically, this exchange would occur rapidly on-line, and we refer to this as the interactive clearance of rights. Privacy enhanced electronic mail would be available for exclusive licenses and other transfers of rights. Non-exclusive licenses might be handled by regular electronic mail.

Transfer of copyright ownership would usually involve recordation in the RRS and could conceivably be handled automatically by the RMS on behalf of the rightsholder and the user to facilitate matters. The confirmation from the RRS would also be passed back to the rightsholder and user directly or via the RMS using privacy enhanced mail. Various enabling mechanisms in the normal screen-based computer interface could be developed and invoked by a user to achieve rapid clearance. If included in the user interface, this capability would have the effect of creating an instant electronic marketplace for such information.

Recordation is defined to mean the official keeping of records of transfers of copyright ownership and other documents pertaining to copyright by the Copyright Office, Library of Congress. For legal purposes, proof of official registration of claims and recordations will be provided by the Copyright Office (via the RRS). Other registrations (at repositories) and non-exclusive licenses (via RMSs) will be certified by privacy enhanced mail. It will be up to the parties to such registrations and recordations to maintain electronic records of their transactions. These could also be stored within the DLS.

Identification Systems

The electronic copyright management system actually requires several types of domain servers. First, documents can be easily retrieved via the DLS if the citation is accurately known or through one or more search and browsing processes otherwise. However, the mapping of a bibliographic pointer (to the designated repository) into its network name and address may require a separate server. Second, the above mentioned domain server for RMSs is needed. Third, the date and time that transactions have been requested and taken may need to be formally validated. An electronic notary and time server would provide such a capability as part of the privacy enhanced mail system.

Retrieval, Appearance and Submission of Documents

public part of a pair of keys could use it to prepare a message which would remain confidential until the person knowing the private key used it to decrypt the message. The public keys could be listed in public directories without any special protection since knowing them did not help anyone decrypt messages encrypted using the public key. This feature makes it far simpler to manage key distribution since the public part need not be protected.

Three researchers at MIT, Rivest, Shamir and Adelman developed a pair of functions meeting the requirements specified by Diffie and Hellman. These functions are now known as the RSA algorithms (from the last names of the inventors).

Digital Signatures

Since either key of a public key cryptography pair can be used to perform the initial encryption, an interesting effect can be achieved by using the secret key of the pair to encrypt messages to be sent. Anyone with access to the public key can decrypt the message and on doing so successfully, knows that the message must have been sent by the person holding the corresponding secret key. The use of the secret key acts like a "signature" since the decryption only works with the matching public key.

Buyers could send digitally signed messages to sellers and the sellers could verify the identity of the sender by looking up the public key of the sender in a public directory and using it to verify the source of the message by successfully decrypting it.

Privacy- Enhanced Mail (PEM)

Public key cryptography can be combined with electronic mail to provide a flexible way to send confidential messages or digitally signed messages or both. In actual practice, a combination of public key, conventional secret key and another special function called cryptographic hashing is used to implement the features of privacy- enhanced mail. The public key algorithms require a substantial amount of computing power compared to conventional secret key algorithms. The older secret key algorithms, such as the Data Encryption Standard (DES) developed by the National Institutes of Standards and Technology (NIST), are much more efficient. Consequently, confidential messages are typically encrypted using a conventional secret key which, itself, is sent, encrypted in the public key of the recipient. Thus, only the recipient can decrypt the conventional secret key and, eventually, decrypt the message.

To send digitally- signed messages, each message is run through a "hashing" algorithm which produces a compressed residue which is then encrypted in the private key of the sender. The message itself is left in plain text form. The recipient can apply the same hashing algorithm and compare the compressed residue against the one that was sent (after decrypting it with the sender's public key).

One of the basic problems with this application of public key cryptography is knowing whether the public key found in the directory for a given correspondent is really that correspondent's key or a bogus one inserted by a malicious person. The way this is dealt with in the Privacy- Enhanced Mail system is to create certificates containing the name of

the owner of the public key and the public key itself, all of which are digitally signed by a well- known issuing authority. The public key of the issuing authority is widely publicized so it is possible to determine whether a given certificate is valid. The actual system is more complex because it has a hierarchy of certificate issuers, but the principles remain the same.

Notarization

Using digital signatures, it is possible to establish an on- line notarization service which accepts messages, time- stamps them and digitally signs them, then returns them in that form. If the person desiring notarization digitally signs the message at the time it is sent to the notarizing service, then it will be possible, later, to establish that the person requesting the notarizing had the document/message in question at the time it was notarized. One can imagine that the originator of a message might have it notarized for the record and the recipient might independently do so. By this means, for instance, evidence of a contract's existence in the hands of each party at particular times might be established.

VERIFICATION, AUTHENTICATION AND CERTIFICATION

The verification process uses stored digital signatures to ascertain whether a given copy is identical to the version which was originally deposited. If any portion of the copy differs from the original, the verification process will fail. Authentication or formal certification of deposits may be provided to a requesting party in traditional ways or via electronic mail. Privacy enhanced mail would be used to certify the authenticity of a deposit, as well as to certify registration and recordation records, for legal purposes.

The deployment of an electronic deposit, registration and recordation capability for use in a computer network environment would greatly facilitate and accelerate the move to a network base for information creation and dissemination. The system would be compatible with the current manual system and would support the ability of the Library of Congress to provide automated registration and recordation services. It would provide a foundation for straightforward and easy expansion and evolution and provide a direct linkage for the Library of Congress to the DLS. It would provide a prime working example for all other kinds of activities where claims registration and rights management come into play. Verification and authentication of copies of deposits may be performed electronically using digital signatures. Formal certification of deposits, as well as registration and recordation records, using privacy enhanced mail may be provided for legal purposes. A testbed which demonstrates the relevant concepts and ideas can be implemented within a two to three year period with initial limited use within a year.

Robert E. Kahn, Ph.D.
President
Corporation for National Research Initiatives
Suite 100
1895 Preston White Drive
Reston, VA 22091-5434

#13

THE DIGITAL LIBRARY PROJECT
VOLUME 1: The World of Knowbots
(DRAFT)

AN OPEN ARCHITECTURE FOR A DIGITAL LIBRARY SYSTEM

AND

A PLAN FOR ITS DEVELOPMENT

Robert E. Kahn and Vinton G. Cerf
Corporation for National Research Initiatives
March 1988

Table of Contents

Summary

This volume describes an open architecture for an important new kind of national information infrastructure which we call the Digital Library System (DLS). The architectural framework includes the DLS functional components, the methodology by which the participating systems communicate with each other, and active, mobile software components, called Knowbots, which perform services for the users. Subsequent volumes will address detailed technical aspects of the architecture such as the design of Knowbots and the protocols required to bind the DLS components together. This research was carried out by the Corporation for National Research Initiatives to specify the overall structure and function of a DLS and to provide a basis for subsequent creation of an experimental system to evaluate the concept with real users.

The term "library" conjures a variety of different images. For some, a library is a dim and dusty place filled with out-of-date texts of limited historical interest. For others, it is a rich collection of archival quality information which may include video and audio tapes, disks, printed books, magazines, periodicals, reports and newspapers. As used in this report, a library is intended to be an extension of this latter concept to include material of current and possibly only transient interest. Seen from this new perspective, the digital library is a seamless blend of the conventional archive of current or historically important information and knowledge, along with ephemeral material such as drafts, notes, memoranda and files of ongoing activity.

In its broadest sense, a DLS is made up of many Digital Libraries sharing common standards and methodologies. It involves many geographically distributed users and organizations, each of which has a digital library which contains information of both local and/or widespread interest.

Each user in the DLS manages his information with a Personal Library System (PLS) uniquely tailored to his needs. A PLS has the ability to act as a stand-alone system for its user, but under normal conditions it will be connected into a rich network of public, personal, commercial, organizational, specialized and national digital libraries.

The DLS provides each user with the capability to use other cooperating digital libraries and provides the necessary search, retrieval and accounting capabilities to support ready access to local and network-based information. The various digital libraries and the associated access to network based capabilities are integral parts of the Digital Library System. Convenient access to local and remote information, without regard for its location, is an essential goal of the system design.

The initial application of the DLS will be retrieval of specific documents for which a user may be able to supply only an imprecise description. For this purpose, we assume each retrieval request has a known target document which the user cannot describe or locate with precision, but can recognize when retrieved. Natural language and visual aids are used to assist the user in this process. However, the possible uses of a DLS are extensive and several innovative applications, discussed in the document, will be explored during the course of the project.

The potential utility for the Digital Library System technology is extremely high if agreement can be reached on appropriate standards and the relevant parties participate on a national scale. The efforts of multiple organizations such as computing equipment suppliers, publishers and other information providers and communications companies are needed to achieve the goals of the project. If successful, the results of this research offer the distinct possibility of enhancing productivity and should stimulate others to develop a vast array of new information products and services. The societal implications of success are very significant.

This document presents one practical path to the creation of such a capability. The benefits of this work depend on the outcome of the scientific research proposed herein. A plan is outlined for the development of an experimental digital library system which depends upon the active involvement of both the research community and the suppliers of equipment and services. The Corporation for National Research Initiatives looks forward to playing a leadership role in exploring the feasibility of this concept.

1. Introduction

This volume describes an open architecture for the development of a Digital Library System. The many users of such a system, even those with only limited or even no knowledge of information technology, can benefit enormously from quick and easy access to the information it contains. Its initial users will be drawn from the research community. However, the system is designed to accommodate a broad class of users (researchers and all others) in productive use of the digital library.

The Digital Library System design allows individual organizations to include their own material in the Digital Library System or to take advantage of network based information and services offered by others. It includes data that may be internal to a given organization and that which crosses organizational boundaries. This document presents a plan to develop such a system on an experimental basis with the cooperation of the research community. Finally, it addresses the application of a Digital Library System to meet a wide variety of user needs.

The productivity gains from having access to a Digital Library System are easily as large as those derived from internal combustion engines and electric motors in the early part of this century. Just as a car on an interstate highway is vastly more effective than one on a rutted dirt road, computer-based information "vehicles" can be made dramatically more effective given the proper operating environment. Computer and communications technology has made it possible for old fashioned, slow retrieval methods to be replaced by virtually instantaneous electronic retrieval. Each user of this technology can anticipate enormous potential benefit, but we lack the natural infrastructure to support this capability on a widespread basis today. This absence of infrastructure represents both a barrier and an opportunity of dramatic proportions.

1.1 A Perspective

Let us assume it is now 2003 and a little over a decade and a half has passed since the work which led to the development of a new national Information Infrastructure was started. The reality has surpassed early technological visions in unexpected ways although some of the more esoteric research ideas are still the subject of active investigation. To understand the profound nature of the revolution which has resulted from the establishment of this new infrastructure, we must reach back into history to trace the roots of human information processing.

In the early history of man, we had only an oral tradition with which to maintain our increasing fund of knowledge and recollection of history. Fathers handed down to sons the oral tradition of the tribe. The capture and rendition of this knowledge was a time-consuming process requiring frequent repetition to avoid its loss. With the invention of writing, we were able to speed up the process of information capture and simplify its reproduction. The price we paid to obtain maximum benefits for this improvement was a need to teach a larger community to read and write.

With the invention of paper and, especially with Gutenberg's invention of the printing press, the time and cost required to reproduce information was reduced dramatically. The invention of

the typewriter brought personal printing within reach of a mass market, but the modification of printed documents to reflect changes and new ideas was still a laborious process which often required the re-typing and/or reprinting of the entire document.

Then dry-process reproduction methods were discovered and subsequently fashioned into copier products. This brought rapid reproduction of printed material to a mass market at an affordable cost. Once again, the time from the creation of a document to having multiple copies available for distribution dropped dramatically.

Computers brought yet another increment in flexibility and speed to the task of recording and sharing human knowledge. In the mid-to-late 1960's, time-sharing applications and networking, along with CRT displays, made it far less costly to prepare and alter documents before committing them to paper or other permanent media. Computer-supported text editing grew even more accessible and affordable with the emergence of microprocessors in word processors and then personal computers. By the early 1980's, most documents were prepared using word processing software running on personal computers.

An adjunct development, the computer-controlled laser printer, brought an additional level of convenience and flexibility to the recording of human knowledge. At very little cost, it was possible to produce fully formatted, multi-font documents with the same degree of revisability as one had with earlier, single-font systems. Products were developed which permitted the integration of imagery and graphics in digital form along with the textual components of documents. These products were called, collectively, "desktop publishing systems."

In the late 1960's and throughout the 1970's, the sharing of mainframe resources and information through networking was a popular method for distributing the cost of gathering and maintaining special information bases. Among the many such information services developed during that period, the bibliographic retrieval systems were among the most popular. Services such as the National Library of Medicine's MEDLINE, Lockheed's DIALOG and the Bibliographic Retrieval Service (BRS) became important reference tools for a variety of users. In the legal community, Mead Data's Lexis and Nexis databases became important resources supporting the preparation and evaluation of legal cases. These tools provide full text in addition to citations to their researchers.

The practitioners of Library Science, like many others dealing with increasing amounts of information, turned to computer-based methods for assistance. The Library of Congress, the Research Libraries Group at Stanford, the National Library of Medicine and the Online Computer Library Center, Inc., joined together in a project to exchange information between their databases. This was called the Linked Systems Project.

Other information services, focused around the concept of remotely-accessible, on-line databases, emerged in this period. These included the Dow Jones/News Retrieval Service, the Data Resources, Inc., economic databases, the Thomas Register of companies and products, and hundreds of special databases reachable through Compuserve and The Source. These services, together, probably did not generate more than $300M/year by 1985; but there was a growing

interest in access to information in this computer-processible form. Most of the information service providers were desired technology which would make more uniform the varied environment in which they had to work.

Networking, along with time-sharing and personal computing combined to form the technical support base for another important technology: electronic messaging. This technology emerged in the computer science research community in the 1970's and in the public domain in the 1980's. Electronic messaging further reduced the potential time delay for propagation of documents by allowing them to be sent electronically to the appropriate recipients. By the early 1990's, standards had been established and business relationships forged which permitted the interoperation of public and private electronic messaging services. The development and deployment of Integrated Services Digital Networking facilities, which emerged slowly in the marketplace, reached an average penetration of 30-35% by 1994. The bulk of this penetration was in the commercial sector where over 70% of businesses had some form of electronic messaging service installed, while residential use had reached only about 15% of the market by that time. By the year 2000, this class of telecommunications had reached about 95% of the business market and about 35% of the residential market. Much of the early usage in the business market was attributable to "electronic data interchange" or "EDI" applications. In these applications, business documents (purchase orders, confirmation, shipping manifests and the like) were exchanged electronically along with electronic funds transfers.

The availability of a prototype Digital Library System (DLS) in 1992, with its innovative approach to intellectual property tracking, opened up a new publishing medium for information providers. In addition to the traditional book, magazine and newspaper markets and existing interactive database markets, the new Digital Library publications allowed the user to selectively view, reorganize and even update the contents for his or her personal use. Certain literary objects even had the ability to automatically update themselves with fresh information or to provide references to recent arrivals in the DLS.

Certain fundamental effects of the Digital Library System were the consequence of four distinct but strongly interacting developments during the 1990's. First, the conventions adopted for the representation of documentary material in the Digital Library were widely implemented by all vendors of combined document processing, database and spread sheet software, by vendors of electronic desktop publishing systems and by public and private libraries around the country. The existence of such common conventions made it feasible for virtually any personal computer or workstation to access and use information produced by any other similarly equipped workstation.

Second, the continuing trend in reduced cost, increased power and memory in portable computers finally reached the point where real-time speech recognition applications became cost-effective. This meant that the transcription of voice to text became affordable to the business community in 1997 and to individual users by 2001. This same computing capacity, along with the development of high resolution, flat screen, touch sensitive displays, provided a basis for the recognition and transcription of hand written or pre-printed material as well.

Direct interaction with a tablet/display and/or processing of scanned material became an affordable alternative to manual (i.e., keyboard) transcription.

The ready capture of imagery through high resolution scanning and telemetry added a third leg to the convenient creation, capture and processing of compound documents. Real-time manipulation and storage of material was also achieved.

Fourth, the incorporation of digitized sound, recorded or synthesized voice and high definition video sequences into documents stored in the Digital Library made it possible to combine most traditional forms of information publication into a common digital format. Conversion into and out of the digital forms and into the more traditional media provided bridges to older existing technologies. The structure and elementary content of printed material were determined during the scanning process.

The ability to register, store, catalog, search, retrieve and manipulate digital information in the library, combined with the variety of affordable media conversion capabilities available by the early 21st Century have led to a revolution in our social, economic and intellectual frameworks. Aided by computer-based Knowbots, easily reproduced and distributed computing cycles augmented human brainpower in the collection, use and creation of information in virtually every aspect of our lives.

Spurred, in part, by the focus of scientific attention on biology and biochemistry during the 1980's, and by the application of computer-intensive processing to non-invasive medical evaluations, the technology of the Digital Library System was applied to the capture and storage of high resolution magnetic resonance imagery (MRI), sonograms, X-ray and other similar diagnostic information. Increasingly detailed genetic analysis capabilities in combination with atomic level biochemical simulations have made it possible to carry out patient-specific bio- and chemo-therapy unthinkable in the past.

Massive amounts of detailed patient history, including the various kinds of digitized imagery, were stored in Digital Libraries around the country. This information provided a basis for epidemiological studies, simulation of experimental therapies, analysis of the population for various health trends, tissue matching and statistical analyses for predictive or retrospective purposes. Coupled with the increasing use of computers for the fabrication of prosthetics, the conduct of surgery and the evaluation of drugs for therapeutic effects, Digital Library Systems are now playing a central role in health care in the 21st century.

Virtually all economic and social transactions are now recorded in Digital Libraries: property exchanges and documentation of ownership, the creation and dissolution of businesses and other legal entities, regulations, the judgments of courts and the acts of legislatures, births, deaths, marriages and divorces, the filing of intellectual property claims and the publication of intellectual works of all kinds are registered within the framework. Entertainment and advertising, product information and actual products, if representable in digital form, are lodged in and made available through these systems. Blueprints and designs for buildings, and other kinds of physical components are required to be deposited in Digital Libraries.

The exploration of information accumulated in Digital Libraries is now an essential part of our educational and research infrastructure. Computer-based tools for search and retrieval of information (including documents) are readily available to students at all levels. The results of manned and unmanned space exploration are indelibly recorded and made accessible as part of the system. Similar aggregations of information are accumulated daily from national and international high energy physics research activities. Economic information, generated and captured in the natural course of daily transactions, is sorted, analyzed and mined by tireless Knowbots making their endless journeys through information space. Malthusian concerns about data overpopulation are easily solved by a combination of advances in high density storage systems and techniques which allow data to die a natural death.

The users of these systems draw upon natural language and visual capabilities embedded within them to find the information they need and to put it into a form suitable for further use. This information-rich, computer-aided environment has significantly changed our ability to organize into groups to achieve specific objectives. Our business organizations have taken on a much more fluid and "horizontal" character, now that the assembling and sharing of information has been made a natural side-effect of everyday interaction. New information-based products are introduced daily and are often discovered and used by other programs that serve our needs, without the need for our personal intervention at all.

Digital Libraries have now become such a pervasive part of everyday living that it's hard to remember what life was like without them. Like other infrastructure, one never really thinks about how it works, how it evolved or, how it is maintained, any more than one thinks about water, electricity, telephones and highways when they are readily available.

1.2 Technology and Infrastructure

Infrastructure plays a key role in the economic vitality of every nation. Viewed from an evolutionary perspective, infrastructure develops in response to the creation of new technologies. For example, the invention of the steam engine and its application in locomotives led to the development of railroads which, in the U.S., were instrumental in opening up the American West. Similarly, the harnessing of electricity and the invention of the light bulb preceded and motivated the development of a national power generation and distribution system. The invention of the automobile and the capability for its mass production ultimately led to the national interstate highway system which drove the evolution of suburban America. The telephone and the underlying communications technology led to a national telecommunications infrastructure.

Few inventions lead to the creation of infrastructure, but every so often, technology appears which drives this kind of development. Nearly every application which emerges at the heart of an infrastructure has an aspect of geographic dispersion and connectivity (e.g., telephone, television, roads, railroads, power generation). However, some technologies can form a kind of infrastructure without connectivity. Videocassette recorders are a prime example. Their

penetration into the residential market is the basis for the cassette rental business which could not exist otherwise.

An important characteristic of infrastructure is simplicity of use. As with electricity, the user's view of the telephone, television, and automobile is essentially simple, although each of the underlying systems is quite complex.

Simple standards governing the use and application of an infrastructure also contribute to its utility in the social and economic structure. For instance, while power generation itself can be complex, ordinary 60 cycle, 120 VAC service is easily described and used to support an unending array of devices.

Computer technology, especially the personal computer or workstation, has all the characteristics consistent with infrastructure. PCs and workstations are widely dispersed in the geographic sense. Like many new technologies, their initial applications displace older methods of achieving similar objectives (word processing versus typewriters, for example) just as cars were thought of initially as simply horseless carriages. Once these displacement applications achieve sufficient penetration, though, it is possible to introduce quite new applications which have no previous counterparts. Moreover, the relatively recent development and spread of packet communication and internetting technology adds an important ingredient, namely connectivity.

These observations suggest that the ingredients are present for the creation of a new information infrastructure based on the wide and increasing penetration of computing and communication technology into the American social and economic fabric. Although personal computers and workstations still suffer from user interface complexity, techniques have emerged (e.g., icons, mice, windows, natural language) which have the potential to simplify the use of computers considerably.

Infrastructure does not happen by accident. It is planned (either well or poorly) and deliberately created often with the direct involvement of the government. It is also preceded by a great deal of experimentation and research. The development of an information infrastructure will be no different in this regard. Unlike the industrial revolution which focused on the augmentation of human manual skills and abilities, computers offer the opportunity to enhance human cognitive capability and capacity. Over the last 40 years, the evolution of digital electronics, communication networks and computer-based applications has amply demonstrated the fertile potential of this technologies.

What is different at this juncture is that computers and digital storage technology are now readily accessible at reasonable cost to be applied to personal information tasks. At the home or office, and even on travel, the availability of computation is becoming pervasive. In the near future, shirt-pocket-size CD-ROMs (perhaps even writeable versions) will be commonplace. Use of networks to access remote databases will also continue to grow.

It is now likely that a substantial portion of the written information we encounter in the U.S. was, at one time, in computer manipulable form. Much of it has never been in that form, but the rate of production of information is so high that the more recent material significantly dominates that which has been produced in the past. Of course, the bulk of this information arrives as "marks on paper" in part because our information distribution methods are still dominated by the low cost and convenience of the printed medium.

Nowhere is the effect of this enormous influx of printed information more painfully recognized than in the research world where rapid access to relevant current and historically important results may make the critical difference between impasse and breakthrough. Finding relevant material, and even learning of its existence, is often a massive challenge. This problem is not unique to the research domain. It plagues virtually every information-dependent human endeavor.

Even if much of this information exists, however fleetingly, in computer processible form, it may not be saved or made accessible in that form. It is usually impossible for others to obtain access to it, even if they know about its existence. The need exists to establish an electronic information infrastructure to capture and allow this information to be made available (under the control of its proprietor). Computer and communication technology can be applied to augment our ability to search for, correlate, analyze and synthesize the available information. We describe such a strategy in this document. Our initial results will make it possible to find and access copies of relevant documentation rapidly and in digital form, which will be a major improvement over current practice. Moreover, it will demonstrate an important example of information infrastructure which can provide the seed for a quantum leap in the way we handle all forms of information.

As information becomes increasingly accessible and fungible (in the sense that once in digital form, it can be readily processed by computer), the entire framework for compensating the creators of intellectual property may have to shift. At present, the basis for intellectual property protection in the U.S. is Patent and Copyright law. The large scale aggregations of information found on CD-ROMs and the selective access to information found in on-line databases may require substantial re-thinking of the ways in which the creators and owners of such information are compensated for its use.

There are many issues at stake in this area, not the least of which relate to the ease with which information can be replicated once in digital form and the rapidity with which large quantities of information can be processed (accessed, transferred, analyzed, integrated, etc.). Concepts of value and pricing and royalty for use of information could require considerable revision if the cost of such use is to remain within reason. One does not now pay an author a royalty each time a book is read. However, a royalty may be earned each time a song is played in public, though not in private. If a thousand books are combined on a single CD-ROM and the acquirer of the CD-ROM only intends to read one of them, what sort of royalty arrangement is appropriate to compensate the copyright owners? How would compensation be extended for cases in which electronic copies are provided to users? In fact, the concept of copying or

duplicating a work may no longer be the essential factor in calculating royalties since far more complex actions may now be taken on digital information.

These questions are not trivial in nature nor have many workable solutions been proposed thus far. It is critical that the interplay of various user and provider interests in information be considered and reflected in the design of the new information infrastructure.

1.3 The Digital Library Project

The digital library project is a broadly based effort to achieve coherent development of our national information resources. The existence of an open architecture for Digital Library Systems will provide the necessary structure for developing rapid access to existing information resources and for creating new information resources; some will be public, some commercial, some organizational and some personal. These will all be pieces of a larger composite library system if they adhere to the open architecture. Just as the highway system required judicious choices within each region and coordination at the boundaries, so will the Digital Library System. It can and should evolve to provide a seamless structure of access to information to encompass, in as far as practicable, the needs of all members of society.

By making it easier to use existing information resources, more people will utilize them naturally and hence the size of the user base will grow. The approach outlined here is to allow the user to stipulate simply what he or she wants to have happen and to let the system take the necessary actions. For example, to retrieve and print a specific document, the user would simply cite it by name. The library system would provide the necessary means for locating the information, retrieving it, and subsequently billing the user (the user could identify that he wants to know the cost before printing).

An overall architecture is needed to guide our use of such information in the future. The Digital Library System represents one practical path to the development of a coherent information base for the management and retrieval of data. The embodiment of this architecture and its assorted functions, protocols and standards in tangible experimental system will be a major contribution to the information infrastructure of the nation.

With the development of Digital Library Systems, enormous opportunities can be foreseen for creating and selling new products and services and for stimulating very significant increases in the demand for existing products such as workstations and print servers. One potential new product is a Personal Library System (PLS) which can manage all of a user's information needs. Personal and organizational data systems are logical extensions of today's myriad software packages and numerous services based on them can easily be envisioned. As with word processing and spread sheets, the use of a PLS within the business community has the potential to streamline operations and improve productivity. For the research community, the ability to achieve quick and easy dissemination of results through electronic publishing will allow source knowledge to be propagated rapidly. For educational use, convenient and rapid access to reference material will quicken the educational pace and stimulate individual initiatives in teaching and learning.

To obtain the potential benefits of an information infrastructure, it is essential to promote the digitization of information and to insure that it becomes computer-accessible. Just as the widespread proliferation of the video cassette recorder has formed a technology base supporting an entirely new alternative to broadcasting, cablecasting and motion pictures, the provision of easy and affordable access to computer processible information leads to interesting new notions such as 1) "digital-back" publications as counterparts to hardcover and paperback books, 2) multi-media documents, whose elements may range over a substantial portion of computer-based publications, and 3) semi-automated retrieval services which can scan very large quantities of published and unpublished material for relevance to research and analysis.

Satisfying all the demands for access to on-line digital information is an overwhelming task for any one organization to undertake. Some of this information will be provided by existing suppliers and some will be created in computer-based form for the first time by new suppliers to the Digital Library. A significant portion of the material in the user's personal library will be created by the user or collected from a variety of informal sources such as personal electronic mail, clippings and intra-organizational memoranda.

For the library to be a repository for personal and organizational information, it must have the participation of many different individuals and groups within each organization. By collectively engaging the creative energies of the many individuals and organizations in the country, a critical mass effort can be realized on a national scale. This broadly based participatory theme is an important aspect of the evolution of the Digital Library System.

The introduction of an information infrastructure is strongly affected by the environment from which it must emerge. There already exists an array of mass media types (newspapers, television, magazines, books) and some fragmentary electronic facilities such as electronic mail, and computer-based teleconferencing services, on-line financial, bibliographic, technical and business databases. Alternate technologies for mass publication of digital information are beginning to proliferate. For example, Compact Disk Read Only Memories (CD-ROMs) appear to be very attractive for many applications. These include the storage of large quantities of geographic, topographic and medical imagery (e.g. Defense Mapping Agency databases, NASA LANDSAT imagery, medical magnetic resonance imagery, etc.) and for large amounts of text and imagery as might be found in an encyclopedia, Patent and Trademark files, design documents (architectural, aircraft, ships, integrated circuits, automobiles, etc.) or other reference volumes.

The development of an advanced information infrastructure must take into account a variety of existing and likely future interests and capabilities if it is to succeed. Publishers and authors must have reasonable incentives to make use of the new infrastructure. Existing libraries and their users must be able to make use of new technologies. Likewise, the educational system must be able to acquire and apply the products and services arising from a new information infrastructure if it is to serve their needs.

As viewed in this volume, the new electronic information infrastructure has a heavy computer-based aspect to it. Moreover, because the information is likely to be kept in digital form, the

telecommunications industry (including telephone, television, local and wide-area networks, cable, fiber and satellite elements) will have an important role to play in support of access, retrieval and dissemination of digital information. For example, the planned development of Integrated Services Digital Networks and the longer term Broadband-ISDN could have a profound impact on the evolution of information infrastructure by providing easily used, variable rate, switched digital communication facilities. However, the role of the carriers could change in unforseen ways due to uncertainty in the regulatory arena.

1.4 Spectrum of the Digital Library System

A large amount of information is already available in computer-based form but is not easily accessible; therefore, relatively little use is made of it. Unless one already knows how to access such information, it may not be obvious even how to get started. Exploring databases for new information is at best a highly speculative process that is often expensive and unproductive. To the providers of database services, and the suppliers of user equipment, this situation translates directly into unrealized potential. Moreover, the vast majority of information that a user may ultimately wish to retrieve surely exceeds the currently available supply by a considerable amount. Without a system for convenient and widespread access to such information by unsophisticated as well as experienced users, it may never be economical to provide it. Until it is provided, however, widespread use may be stifled. Here we see a classic chicken-egg dilemma and hence progress on both fronts moves at a glacial pace.

The spectrum of possibilities for use of a Digital Library System system ranges from the tangible to the intangible, from the very specific to the vague and from the visual to the invisible. We depict one such range of possibilities by the series of six overlapping circles in Figure 1.

At the right-hand side of the spectrum, we denote fixed format documents intended to be read by people. These are generally assumed to be prepared for publication and have definite presentation formats. These documents are stared and retrieved in their presentation form. They are guaranteed to be reproduced as they were originally created, subject only to scale and resolution limitations of the print server.

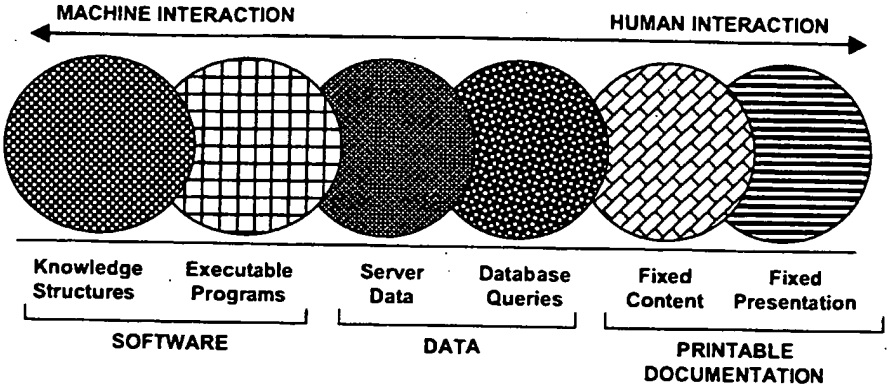


Figure 1 The Spectrum Of Library Contents

Fixed Content, flexible format documents, shown just to the left of the fixed presentation documents in Figure 1, require the user or his system to specify how to present the information, assuming its content remains unchanged. For this class of documentation, the user might wish the text to be single spaced, double spaced, margins adjusted, page boundaries adjusted, fonts changed and so forth. This is in marked contrast with the fixed format documents, where no substantial visual changes of any kind are permitted.

In the middle of the figure are shown database queries and data of the kind collected from sensors. The system treats sensor data along with database entries as if the were new types of objects in the library; this treatment requires understanding the semantics of objects in the library for the purpose of analysis and question/answering. When prestored answers are available without the need for searching documents, retrieval requests can be satisfied more quickly. Obviously, it will not be possible to anticipate all such questions in advance.

To the extreme left in Figure 1 are the two most speculative aspects of the spectrum of the Digital Library content. Although many attempts have been made to achieve reusable software, the infrastructure to reach this goal is still largely unexplored. Further, the preparation and reuse of knowledge structures in the development of intelligent systems is also virgin territory. This latter subject will be the focus of the second volume in this series.

The initial version of the Digital Library System will be tailored for the domain of printable documents (the two right hand circles in Figure 1). However, the underlying technology will be designed to allow evolution to cover the remaining portions of the spectrum. Ultimately, we see the library system encompassing the entire range of possibilities shown.

Even with this initial restriction on content, the span of possibilities for inclusion in the library is enormous. In the implementation plan (see Section 4), we discuss how the library system will be developed aid how the supply of documentation can begin and expand.

Most users subscribe to a given information service to retrieve highly selective pieces of information. Rarely do they learn to use the full complement of capabilities available on that or

any other system. Almost all existing on-line informational services support users that connect via simple alphanumeric terminals or PCs in terminal emulation mode. Most users are able to do little more than print a received text string or view it on a screen. The power of personal computers is rarely used to exploit further processing of received on-line information. With the exception of spread sheet programs that accept certain financial data obtained electronically, and mail systems that allow for forwarding, little or no user processing of received information typically occurs.

The underlying technology of the Digital Library System allows a user to access any available document within the entire Digital Library System. Using the PLS, he can modify a document in any way he chooses, incorporate it in another document, print it, search it or supply it as input to another program for further processing or display. Parts of the document can be extracted and manipulated.

Unlimited access to specific documents raises fundamental issues of intellectual property protection. A technological approach to this problem is outlined briefly in Section 3.1 of this report.

We plan to explore how to support vague and imprecise retrieval requests for specific printable documents while insuring that other well-defined requests are effectively handled as well. Requests for a specific manual, report, or equipment specification might be precise enough for the system to retrieve straightforwardly. The same might not be true if there was any uncertainty in the request. For example, if the author or title of a report were unknown, and only a general description of its subject was available, an intermediate process would be required to resolve the query further, to ask more questions of the user, or to produce a list of possible documents for selection.

1.5 A Guide to the System

A schematic description of the Digital Library System is shown in Figure 2. Its components are Personal Library Systems for the users, Organizational Library Systems for serving groups of individuals or activities, new as well as existing Databases stored locally and across the country, Database Servers to handle remote requests, and a variety of system functions to coordinate and manage the entry and retrieval of data. The system components are assumed to be linked by means of one or more interconnected computer networks.

Local requests for information, if not satisfiable by the local Personal Library, are dispatched to other, larger or possibly more specialized sources of information available through the network. A single inquiry may spawn tens to thousands of exchanges among various parts of the full Digital Library System. This could easily happen if the system must first query several databases before responding to a particular inquiry.

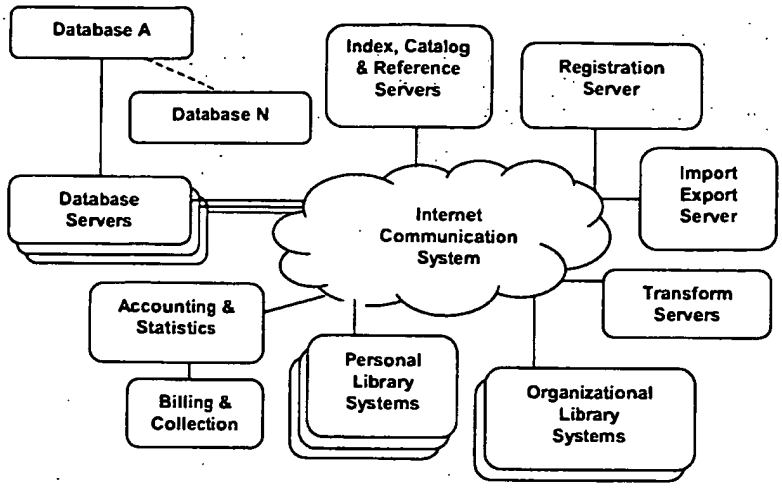


Figure 2 Structure of the Digital Library System

These exchanges are, for the most part, mediated by Knowbots, which are active intelligent programs capable of exchanging messages with each other and moving from one system to another in carrying out the wishes of the user. They may carry intermediate results, search plans and criteria, output format and organization requirements and other information relevant to the satisfaction of a user's query.

A Knowbot is typically constructed on behalf of a user at his Personal Library System and dispatched to a Database Server which interfaces the network to a particular database or set of databases. To accommodate existing database systems which are not capable of direct interaction with Knowbots, these servers can assist Knowbots in translating their information requests into terms which are compatible with the existing database's access methods. In the future, we expect to witness the development of databases systems with built-in mechanisms for housing and catering to resident or transient Knowbots. It is possible, and even likely, that more than one Knowbot may be dispatched either directly from a Personal Library System or indirectly as a result of actions taken at a particular Database Server. These various Knowbots may rendezvous at a common server or all return to the originating workstation for assembly of the results.

Two important components of the DLS shown in Figure 2, are the Import/Export Servers and the Representation Transformation Servers. The former components are responsible for accepting new documents into the Digital Library System and for dispatching documents out of the system. The latter components convert document from one internal representation to another. Depending on the nature of the output required, the obtained results may be passed through a Representation Transformation Server for conversion before being delivered. The results may be destined for either an originating PLS, a target PLS (or other workstation) designated in the original query or to an Import/Export Server if the destination is outside the particular Digital Library System "universe". For example, if the results are to be produced on

CD-ROM delivered physically to the user, this process will involve passage of the results out of the Export Server.

When Knowbots and originating workstations or other intermediate information servers need assistance in finding information, they invoke Indexing, Cataloging and Referencing Servers by causing one or more Knowbots or messages from them to be dispatched there. The Indexing, Cataloging and Referencing Servers collectively contain information about the content and organization of the Digital Library System and help to identify which Database Servers should be accessed to respond to particular types of queries.

The Indexing, Cataloging and Referencing Servers are, in turn, kept up-to-date by the Registration Server which accepts new information into the Digital Library System. The Registration Server makes use of Indexing, Cataloging and Referencing Server(s) to determine where to store new information. The Registration Server also updates the Accounting Server so that providers of information can be identified and compensation for the use of information in the Library can be properly accounted for. Users of the Digital Library are also registered with the Registration Server and information about them passed to the Accounting Server, so that access to information and billing for its use can be supported.

Records of accesses and results are collected, by means of additional Knowbots, and reported to an accounting and statistics collection system for subsequent rating or analysis. The results of accounting collection are passed to a billing and collection system for further action.

1.6 Applying the Digital Library System

The Digital Library System will only be as effective as the various uses to which it is put. A few of these will be developed during the DLS project; the remainder will occur over time through the determination of motivated individuals and organizations. By way of comparison, it is noted that most of the applications for electricity came well after its introduction. However, a few key needs drove its early development such as its use for urban lighting. Applications using electrical motors came later on.

We can only begin to speculate on the many uses of a Digital Library System. However, several needs seem clearer than others; four of them are outlined in Section 3. One need is to examine and prioritize the contents of various publications which have been identified in advance and are known to be relevant to a given worker's field. A second need is to support computer-based design activities in which access to prior designs and their context or rationale is essential. A third need is to support research activities which involved searching for documents which contain relevant information and extracting critical portions for further and possibly detailed analysis. A fourth need is to link images and text for diagnosis. Many additional needs will undoubtedly occur to others.

2. The Architecture of a Digital Library System

Before describing specific features of the Digital Library System, it will be helpful to review some of the fundamental assumptions which strongly affect its design. Perhaps the most dominant of these assumptions are that the system is distributed, heterarchical, hierarchical, networked and strongly display-oriented. In addition, it must have an ability to interact with other autonomous Digital Library Systems that do not adhere to its internal standards and procedures.

The rationale behind the first assumption (distribution), in part, is that existing digital information sources are not physically collocated and that, as a practical matter, the Digital Library System design has to accommodate many geographically distributed components. The distributed system design does not rule out the centralization or at least concentration of resources where this meets pragmatic needs for minimizing operating costs, aggregating communications facilities, and so on. The important point is that the design forces neither centralization nor pure decentralization but accommodates both styles.

We assume that users will access the services of the Digital Library from powerful, geographically distributed and often locally networked workstations. This assumption places networking at the center of the distributed architecture. Even if all the data content of the Digital Library were centralized, its users cannot be.

Distinctions between entirely different (autonomous) library systems leads to at least one level of hierarchical structure in the architecture. Components which can interact among themselves using an internal set of conventions are distinguishable from the set of components which use an external conventions. Distributed, decentralized but hierarchically structured computer services seem to be a natural consequence of the organization of the present and foreseeable marketplace for the use of systems like the Digital Library. Computer services which cross the jurisdictional boundaries between organizations, or even between divisions or departments of one organization, require management structures for access control and accounting. Services which span multiple organizations typically exhibit two or more levels of hierarchical structure stemming from the necessity to draw boundaries around component operating and management responsibilities.

Another rationale behind the hierarchical structure of the system is to constrain the scope of the data management problems so that system growth does not lead to exponential amounts of database updating and consistency checking activity. Similar motivations often impose structure on otherwise unstructured telecommunication networks, for example.

The importance of scaling in all dimensions cannot be over-emphasized. The architecture must scale in sizes and numbers of databases, numbers of users, numbers of components, bandwidth of underlying data communication, varieties of archived content and variation in presentation media and access methods.

By deliberately treating parts of the Digital Library as distinct, networked components, it becomes possible to simplify implementations and to identify explicit protocol, management and control interfaces required to carry out the functions of the system. Such structuring also has the benefit of accommodating potential competition among multiple sources for the provision of products, services and functions, which in the long run, improves user choices and enhances the opportunities for growth of the Digital Library System.

The assumption that users will access and use the services of the Digital Library through powerful, display-oriented workstations is rooted partly in the observation that personal computing and graphics-based workstation technologies are rapidly converging. As costs drop, personal computer users tend to buy increasing capability at the same cost, rather than spending less to obtain previously available capabilities. Economics aside, another reason for assuming the use of high power workstations is the need to support multi-font text, graphics, imagery and, possibly, other modalities (sound and video for example), if the full range of potential Digital Library services be supported.

Such reasoning does not rule out catering to "disadvantaged workstations," but these are treated explicitly with the realization that there is a potential loss of fidelity, functionality, or quality of service when accessing Digital Library services through these less capable devices.

The heterarchical assumption is motivated by the likelihood that more than one such Digital Library System will emerge as the national and global information infrastructure evolves. In the past, architectural designs for distributed systems often have been based on the assumption that there is only a single, monolithic, integrated architecture. Such assumptions usually lead to serious limitations on interactions between autonomous distributed systems and thus inhibit any ability for them to coordinate, cooperate and interoperate. Examples of such lack of vision may be found in many of the private and public electronic mail systems which make no provision for addressing messages outside the domain of the specific mail system in question. The Digital Library design specifically contemplates the existence of multiple instances of autonomously operating Digital Library Systems requiring compatible external interfaces. Each Personal Library System will also comprise multiple internal components which need to interact closely.

The second assumption motivating the heterarchical design stems from a belief that useful, self-contained, workstation-based, personal digital libraries are needed which can interoperate seamlessly with other internal or external library components of an organizational, regional, state or national character. The system design supports crosslinks among components at various levels in the structure and, in fact, makes heavy use of such linkages to achieve efficient interactions.

2.1 Overview of Major Library System Components

In the sections that follow, we will examine each of the major Library System components in turn, describing their functionality and relationships with other components. Figure 2 illustrates a top-level view of the Digital Library System. The rationale for the order in which these

components are described is based on following a document [or, more generally, an object as it makes its way into the Digital Library and is then accessed and used.

The principal components of the system are:

i) Import/Export Server
ii) Registration Server
iii) Indexing, Cataloging and Reference Servers
iv) Database Servers
v) Accounting and Statistics Servers
vi) Billing System
vii) Representation Transformation Servers
viii) Personal Library System

In addition to these eight basic components, there are two fundamental concepts which are intrinsic to the interaction of these various subsystems. These concepts are Knowbots and Shared Icon Geography which are discussed in more detail in Section 3. The initial information in the Digital Library system is assumed to be material which was originally intended to be printed (including multi-font text, graphics, bitmapped imagery) or otherwise displayed in static form. In addition to books, reports and periodicals, the system can include other material such as electronic mail, VLSI designs and organization charts. However, the underlying concepts will be easily extendable to allow more ambitious kinds of information such as holographs and digital films. The initial formulation of the system is organized around printable information to give the project focus and a concrete development target.

2.2 Import/Export Servers

An Import/Export Server acts as a primary interface between the Digital Library System and the outside world. Contributions to and acquisitions for the Digital Library are presented through an Import Server. The method of interaction with an Import Server forms one of the most important interfaces in the system. An Import Server will be capable of accepting contributions to the Library in many forms. Contributions and submissions might arrive as part of an electronic mail message, as a CD-ROM, as a magnetic tape, as a PC diskette or even as a facsimile scan. The common denominator is that the information has been converted to some definable digital form. One of the most important steps in the Digital Library design will be the determination of how many and which arrival formats will be acceptable. Conversion from analog to digital form, while an important consideration, is outside the scope of the library project.

The arriving objects (e.g., documents) must come with additional information if they are to be successfully entered into the DLS. Among other things, the Digital Library needs to know the origin of the object (bona fides); the owner of it (especially if any intellectual property rights are to be accounted for); terms and conditions for use, reproduction and access (including access control lists on an individual or organizational basis, for instance); descriptive information

which might aid in retrieval; relation to existing information in the Library (e.g., part of a periodical series, book series, revision, etc.); and format definition.

Information which is not in a form which can be directly accepted at an Import Server will have to be prepared by services outside the Digital Library (an opportunity for any number of public agencies or private businesses). Similarly, Import Servers for particular classes of information might be implemented and operated or sold competitively.

An Import Server extracts the information relevant to registration from the arriving submission, packages it for processing by a Registration Server, and then forms and launches a Knowbot to deliver it there. At this point, the simple model is to send all of the information, including the actual submission, along with the registration Knowbot. This could prove impractical for significant contributions such as books. An alternative is for the registration Knowbot to carry only the information needed by a Registration Server and to carry references to the storage facilities at an Import Server for use when the information is to be transferred and incorporated into a database or catalogued by an Indexing, Cataloging and Reference Server.

An Import/Export Server also provides a basic mechanism for the equivalent of interlibrary exchange services. It should be possible for several, otherwise distinct, Digital Library Systems to exchange information, queries, responses and library contents. Analogous to conventional inter-library loans, this capability is essential if the Digital Library System technology is to be independently proliferated to support a variety of products and services. Every effort must be made to assure that the architecture is free of the assumption that a single system is unique in the information universe. This does not rule out the need to tightly integrate some Digital Library components into a particular coherent system, but emphasizes the need to tolerate and accommodate diversity.

It is not yet clear whether the inter-library exchange facility can be implemented merely as an electronic message exchange or whether the interaction should also permit more immediate and direct forms of Knowbot exchange. The latter may require too much context sharing or accounting/billing and authentication mechanism to be implemented for essentially distinct Digital Library Systems. Additional research will be required on these matters. For the present, it is assumed that an electronic message exchange convention will be the basis for interactions among distinct Digital Library Systems.

All such systems, if they are to interact at all, must share a common name and address space to support message exchange. This could be provided by relying on international electronic messaging conventions which include provision of such a common name and address space for electronic mailboxes.

In addition to its import functions, an Import/Export server has the responsibility for exporting information (objects) from the local library environment to other environments. The latter may be other libraries or other presentation media (paper, CD-ROM, facsimile, etc.). An object may be exported either as the result of an action taken by a user (or a Knowbot acting on behalf of a user) or as a consequence of a request for service imported from another library system.

Although the inter-library exchange mechanism is assumed to be based on electronic mail, other less general but perhaps more efficient choices are possible. Other media conversions (e.g., to print) may have to be handled in idiosyncratic ways.

2.3 Registration Servers

Registration Server(s) are responsible for 1) receiving messages from or hosting arriving Knowbots carrying new information (or references to new information) to be added to the Digital Library, and 2) registering new users, sources of information (databases) or other components newly added to the system.

One of the most important tasks of a Registration Server is to associate a unique identifier with any new object in the system. Ideally, it should never be necessary to re-use any identifier; thus the identifiers need to be allowed to increase in length. If identifiers are to be assigned by more than one Registration Server, methods must be invoked to assure uniqueness (e.g., by prefixing the object identifiers with Registration Server identifiers).

A Registration Server reports the existence of a new object to the relevant Digital Library component. If the object is a new user, this is reported to the Accounting System and to the Indexing, Cataloging and Reference Server(s) so that queries regarding that particular user can be properly answered. New information to be added to the Library is likewise reported to the Accounting system in the event that charges are to be associated with its access and use. A Registration Server may also supply a description of the charging algorithm to be used for this information. This might be as simple as a reference to a standard algorithm or as complex as a program for computing use charges for the particular item.

If it is readily apparent which database server(s) should house the arriving object, a Registration Server will so inform the Indexing, Cataloging and Referencing Server(s) and direct the Registration Knowbot to ferry the data to the appropriate Database Server. Alternatively, if the information did not come along with the registration Knowbot, a Registration Server can form a new Knowbot to pick up the information from the Import Server and deliver it to the appropriate Database Servers.

Registration Servers interact directly with Indexing, Cataloging and Referencing Servers by providing them with an instance of the object being registered. An Indexing, Cataloging and Referencing Server determines which database can house the object (there may be more than one) and reports this information to the Registration Server. Other items, in addition to documents, which require registration in the reference database include, inter alia, all intra-library servers, users and other known Digital Library Systems.

2.4 Indexing, Cataloging and Referencing Servers

The principal function of the Indexing, Cataloging and Referencing Server(s) in the Digital Library System is to provide global cataloging and indexing services for the retrieval of Library content. The system is organized to support multiple, cooperating servers. It is also planned to

accommodate alternative, specialized Indexing, Cataloging and Referencing Servers within this architecture to take advantage of new ideas and implementations without requiring the removal or replacement of existing services.

An important design issue will be the control of potentially open-ended interactions between Registration Servers and multiple Indexing, Cataloging and Referencing Servers to avoid network congestion and deal with the resulting multiple copy database update problem. Criteria for selecting among alternative Indexing, Cataloging and Referencing Servers must be worked out, if several deal with the same or inter-related information. It is easier to deal with the case that knowledge about the content of the Digital Library is partitioned non-redundantly among several multiple servers. For instance, one server might specialize in cataloging and indexing electronic mail messages, another in books and a third in journals or other periodicals. Alternatively, if redundancy is to be supported, it might be based on multiple, complete, copies of identical indexing and cataloging information, rather than overlapping or partitioned components. Maintaining a consistent set of registration database copies is an interesting challenge in its own right.

Indexing, Cataloging and Referencing Servers are also used to locate services and users as well as information in the Digital Library. This function has an analog in the electronic mail domain in which name servers make it possible to find mailboxes associated with users. Search criteria for the name servers may be as simple as first and last personal names or complex conditional expressions, involving job title and/or function, company name, special interests (if known), locale and other identifying characteristics.

There are two distinct questions which can be answered by the Indexing, Cataloging and Referencing Server when it is dealing with Library content:

- "Here is the data, where should it be stored?"
- "Here is the kind of data I want, where is it?"

These two functions are, in fact, very similar and require the same, base level input information. Thus, any tools developed for one function can potentially carry over to the other.

Each Indexing, Cataloging and Referencing Server is capable of carrying out a repertoire of functions which can be invoked by Knowbots arriving at the Server. Knowbots arriving at a Indexing, Cataloging and Referencing Server will usually be performing one of several specific tasks:

- Cataloging/indexing of a new Library acquisition.
- Searching for a cataloged or indexed item.
- Collecting statistics about the content or usage of the Library.

When a new item is registered, a Knowbot is dispatched to a Indexing, Cataloging and Referencing Server for guidance in cataloging and indexing. The arriving Knowbot carries with it any key word or other cataloging and index terms that may have been assigned on

publication (e.g., by the Library of Congress, the journal publisher, the author, etc.). It may also carry the actual item content so as to support cataloging and indexing algorithms which operate on the full "text" of the new item. Of course, the Knowbot also carries information such as the source (author), copyright owner (if any), International Standard Book Number (or other identification of this type), publisher, date, place (and time?) of publication. Both published and unpublished works could be included.

The indexing or cataloging information may vary depending on the nature of the new item. For example, arriving electronic mail would typically be indexed by origin, To: and CC: recipients, date and time of origin, unique message identifier, originating mail system, subject matter, and depending on the Indexing, Cataloging and Referencing Server, by key words or user-provided search terms.

2.5 Database Servers

The design of the Digital Library System is intended to accommodate existing databases and database services and to provide a framework for new databases organized around the concept of Knowbotic information storage and retrieval. Database Servers bridge the gap between already existing, database services and the Digital Library System by providing support for resident and arriving Knowbots and exchange of inter-Knowbot messages. The principal tasks of the Database Servers are:

- To accept and store new information, and
- To house arriving Knowbots bearing queries

Some Database Servers may only provide the second of these functions as is likely to be the case if the actual database is managed and updated essentially outside the Digital Library System context. For database systems which are designed to operate within the Knowbotic paradigm of the Digital Library System, the functions of the Database Server may actually be combined with the database system itself. It is possible, of course, that these functions might still be supported by a separate Database Server for efficiency reasons.

Another motivation for including the Database Server in the architecture is to utilize new parallel processing technologies to speed the search and retrieval Process for both new and existing database systems. Full text databases could be searched in their entirety at very high speed. Coupled with the Knowbot concept, such special purpose servers could revolutionize the utility of existing databases. To achieve this goal, it would probably be necessary to collocate the Database Server and the database system it serves so as to provide an economical but very high speed interconnection between the two. For existing databases, such a specialized Database Server would absorb the entire database so as to permit ultra-high speed and novel searching algorithms to be applied independent of its pre-existing computational base.

Such an intimate link between the Database Server and the database will doubtless require both technical and business arrangements, particularly in cases where the database is considered to be proprietary. Where such an arrangement proves infeasible, the alternative is to configure the

Database Server so that it looks to the database as an ordinary user but provides all the of required framework for interfacing to the Knowbots of the Digital Library System.

2.6 Accounting and Statistics Servers

The function of the Accounting and Statistics Server is to collect and store data relating to the use of the Digital Library System and to send the accounting portion of it to the Billing Server. Information collected by the Accounting and Statistics Server includes not only retrieval data appropriate for billing purposes, but also statistics needed to guide operational decisions. Examples include information needed to identify capacity problems; profiles of information use (e.g., to identify the need to replicate data to reduce delay or increase transaction processing throughput); and inter-Knowbot message traffic (e.g., to determine when it would be more efficient for a Knowbot to be resident and exchange messages as opposed to moving Knowbots between a given pair of sites).

It is important to note that more than one Accounting and Statistics Server can be incorporated in the Digital Library both for redundancy and for load sharing. This means that any element of the Digital Library that produces data of interest to the Accounting and Statistics Server(s) must be configured to know to which server the data should be sent. To increase system integrity, the Accounting and Statistics Servers should be configured to accept data only from the appropriate sources and to raise alarms when data arrives from an unexpected source. Obviously, if redundancy is to be used to deal with various potential system failures, more than one Accounting and Statistics Server needs to be configured to accept data from a given source and these sources need to be configured to report to more than one Accounting and Statistics Server. This is a sensitive design area because the cost of sorting through multiple copies of accounting data collected at multiple sites is potentially very high.

The principal sources of accounting data are the Database Servers since they have direct access to querying Knowbots and their inter-Knowbot message traffic. This information is conveyed on a periodic basis (or based on the quantity of data accumulated) to the Accounting and Statistics Server. Other important sources of accounting data are the Import/Export Servers which process inter-library requests. In principle, the accounting for such queries should originate at the appropriate Database Server, but for inter-library reconciliation, the Import/Export Servers also capture traffic exchange information and pass this to the Accounting and Statistics Server.

The Registration Server is another source of accounting information since the registration of a new Library object or a new user often has accounting and pricing implications. In effect, most of the Servers in a Digital Library can be sources of accounting or statistics data, depending on the charging policy adopted by the operator of the system. An important area for agreement between two Library Systems will be their inter-library pricing and reconciliation practices.

2.7 Billing System

The Billing System generates invoices for use of the Digital Library System based on information it gets from the Accounting and Statistics Servers. The Billing System also needs to capture information about newly registered objects and users and may do this either through records sent to the Accounting and Statistics Servers by the Registration Servers or by direct exchange with the Registration Servers.

The details still need to be worked out, but it is possible that accounting data can be collected and delivered as objects like any others in the Digital Library. The mechanics of billing users and collecting revenues for service are still to be determined. By the time such a system becomes operational, direct electronic funds transfers may be the preferred collection strategy, but for the sake of backward compatibility, the system should also be capable of interfacing with a conventional lockbox service. This also implies that invoices may need to be sent either electronically (e.g., via Electronic Messaging Services) or on paper (via the postal service).

2.8 Representation Transformation Servers

The design of the Digital Library is posited on the assumption that only a few internal standard representations for library objects will be required. There will be a vast degree of heterogeneity in the actual sources of information to be placed in the Library and an equally heterogeneous collection of recipients with preferences as to the format of retrieved objects.

To avoid the need to build into the Database Servers the ability to accept or generate the entire panoply of possible object representations, the Digital Library employs Representation Transformation Servers which can accept a standard library object and convert it into any of several output representations for delivery to a user. Similarly, objects arriving at the Import/Export Server which are not in a standard library form may be converted at an appropriate Transformation Server.

It is anticipated that Transformation Services will be a lively area for competition among vendors of Digital Library products and services. Any number of such servers might operate within the context of a given Digital Library. Alternatively, the developers of such software might configure it to run in the context of a Personal Library System (see below) which would interact externally using standard object representations but could manage conversions internally using software acquired for this purpose or by means of exchanges with a Representation Transformation Server. Although the standard library representations have yet to be selected, a variety of potential representations into which or out of which it must be possible to transform already exist and will be used wherever possible.

The Association of American Publishers have adopted a version of the Standard Generalized Markup Language (SGML) as their preferred representation for the exchange of compound documents. Compound documents incorporate multi-font text and graphics in addition to raster or other bit-oriented images. If it is the case that most books and periodicals published in the U.S. will have an SGML form at some point in the process of preparation for publication, it seems reasonable that the Digital Library support this form as one of its internal standards.

In the international community, particularly in the International Standards Organization (ISO), a representation known as Office Document Architecture (ODA) is solidifying as an international standard. The National Science Foundation EXPRES project has adopted a version of ODA as its preferred representation for compound documents. This choice is compatible with the X.400 electronic messaging format recommendation of the Consultative Committee on International Telephony and Telegraphy (CCITT). Indeed, X.400 can accommodate the transport of either SGML or ODA encoded objects.

A third representation of considerable and growing popularity in the U.S. is PostScript, developed by Adobe Systems, which comprises an executable language capable of very detailed descriptions of document presentation, page layout, imagery and fonts.

A fourth representation of potential interest derives from the American National Standards Institute (ANSI) X.12C committee which is working on standards for Electronic Data Interchange (EDI), focusing particularly on business documents such as purchase requests, purchase orders, bills of lading, invoices and the like. A related set of standards have been prepared by the ANSI X.9 committee for electronic funds transfer. This representation might be important to the Personal Library System if it is applied to tracking of personal or organizational financial transactions. Electronic funds transfer mechanisms might also be invoked within and between Digital Library Systems for the purpose of achieving royalty or other compensatory payments for access to and use of the content of these systems.

A fifth representation of increasing importance is facsimile (especially Group III and Group IV). A large number of documents are now received in that form and printed on thermal paper (or plain paper), but it is not far-fetched to capture this information in digital form for storage in the Digital Library. In the long-term, one can hope for better character recognition capability so that facsimile scanned documents can be reconverted to ASCII or some multi-font encoding.

There are, in addition, a large number of different word processing formats such as those used by Wordperfect, Wordstar and Microsoft Word, to name just three. There are also numerous proprietary document representations developed by industry. The Digital Library would rely on the Representation Transformation Server to deal with these various proprietary document encodings, translating them as needed into one of the several Digital Library standards.

2.9 Personal Library System

The Personal Library System (PLS) should satisfy two distinct needs in the architecture of the Digital Library System. The first is to provide a basis for a completely stand alone instance of a library system which can operate independently from the collection of other Digital Library Systems or even components of a given DLS. The second is to interact with the other distributed components of the DLS. Both of these requirements are treated in this section.
Figure 3 illustrates an abstract view of the internal structure of a Personal Library System. The horizontal layering shown is essentially notional. There is no attempt to portray with any precision, the vertical relationships among components.

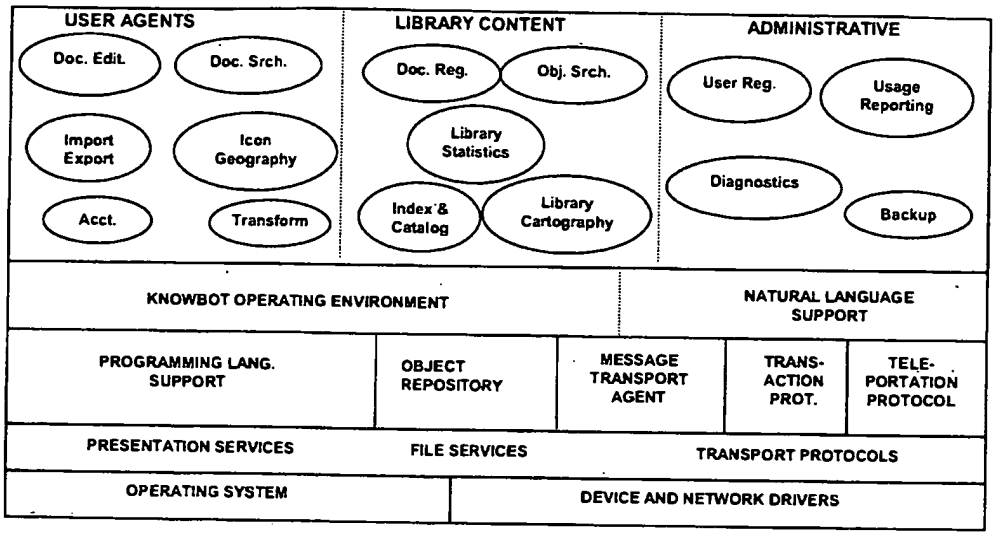


Figure 3 Personal Library System Structure

At the lowest level in the figure are the operating system and associated device and/or network drivers. It is not necessary for a PLS to be networked but it is increasingly common to find workstations interlinked on local area nets or at least capable of accessing dial-up telecommunication facilities. Although for convenience and simplicity they are not shown, included in the family of device drivers is support for common user interfacing devices such as keyboards, displays, mice and printers. These devices may eventually include audio input and output facilities and special high-resolution color displays to meet the "presentation" requirements of the contents of the library system.

The operating system will have to be capable of supporting multiple process execution. Many examples of such systems exist but the design of the Personal Library System does not impose a requirement to use one particular operating system. Whichever operating systems are selected, it is essential that they have low overhead support for interprocess communication and large scale file storage.

Transport protocols are essential when the PLS must operate as part of a larger collection of library systems within the DLS. In combination with the appropriate device drivers, the Transport Protocols enable the PLS to establish a presence in the rich networking environment and provide an avenue for access to external library services. Examples of the kinds of protocols which might be used include DOD TCP/IP/UDP, ISO TP/IP or other packet-oriented, multi-vendor protocols.

At least three application-related protocols are needed in the PLS if it is to occupy a useful place in a common networking environment. To handle electronic mail services, the PLS should support some kind of electronic message transport agent (MTA). This might be the

DOD SMTP (Simple Mail Transfer Protocol), the Multimedia Messaging Protocol (MMP) or the CCITT X.400 (Mail Handling System) protocol. To support the exchange of Knowbots between the PLS and DLS components, a "teleportation protocol" is needed. Finally, to support remote (or even local) message interaction between Knowbots, a "transaction protocol" is required. The inter-Knowbot messaging accomplished by means of the transaction protocol is distinct from the electronic mail interaction achieved using the MTA.

There is a two-fold need for the electronic mail capability in the Personal Library System. First, the PLS should be capable of assisting users in the searching, management and manipulation of their electronic mail. The PLS organization should attempt to accommodate this "under one roof;" however it is entirely reasonable for MTA functions to be provided by an electronic mail server external to the PLS but which the PLS can access to obtain copies of electronic mail intended for the PLS user. The second reason to have access to electronic mail is to provide an indirect, non-Knowbotic interface to external Digital Library Services. Distinct Digital Library Systems may not be able to share a common Knowbot Operating Environment but may want or need to exchange information. Electronic messaging technology offers one means for achieving this objective.

The Object Repository is a facility for storing the contents of the Personal Library System. The Object Repository is supported by the services of the Filing System (File Services in Fig. 3) which can be fairly conventional, but has its own organizational structure, access control mechanisms, indexing, storage and retrieval primitives. In the current design, all information stored in the Personal Library (and, in general, in the Digital Library System) is object oriented. By this it is meant that the objects have "callable interfaces." Rather than knowing the details of internal representation of an object, it is enough to be able to call on the object to supply various pieces of information (e.g., provide a bit map representation for part of a document, provide information about the content of the object such as key words, provide information about the source of an object and so on). The representation returned from such calls does have to be standardized, to permit Knowbots to manipulate arbitrary objects and their contents. The motivation for this point of view is similar to the motive for the development of object oriented languages: simplified and standardized interactions with objects while allowing substantial variation in internal representations. For information in pre-existing databases, Database Servers are used to mediate and provide arriving Knowbots with an object view of the information. The concept of Knowbots is explored in more detail in Section 3.

At the present state of design, it appears that both Knowbots and the objects they deal with can be represented using object-oriented languages. Inter-Knowbot and Knowbot-object messaging is mediated through the Transaction Protocol. Although it is still not determined, the programming language support illustrated in Fig. 3 may turn out to be identical for Knowbots and objects.

There are a number of potential object-oriented languages which might serve for the representation of objects in the Digital Library System or for the representation of Knowbots. In the most general case, even Knowbots ought to be storable in the Digital Library as objects. Examples of existing languages include Smalltalk, Common Lisp, Common LOOPS and C + +.

The selection of a methodology for building Knowbots and even the determination whether an object-oriented language is essential are two of the highest priority research questions for the Digital Library Project to resolve.

A related representation concept called "hypermedia" or "hypertext" (a term coined by Ted Nelson) also needs to be taken into account. Originating with the early work of Engelbart on the On-Line System (NLS), the notion of threading text together in multiple ways with a variety of indexing and marking mechanisms has gained currency in the late 1980's. The notion has been picked up and expanded upon by others (e.g., Xerox with its Notecards experiment and by Apple with its Hypercard product for the Macintosh).

Ultimately, the Digital Library must implement methods for the creation, maintenance and extension of a rich collection of information registered in the system. Out of this will come facilities for easy browsing and association of related information. Whether and how notions such as hypermedia are reflected in the Knowbotic paradigm of the Digital Library is one of the intriguing research areas which will be exposed by the effort to construct and use an experimental system.

The Presentation Services subsystem concerns itself with the management of user interaction facilities and includes such functions as window management; icon, graphics and multi-font text rendering; linking of displayed constructs with screen coordinates to aid mouse utilization; and sound synthesis or capture. Most of these capabilities already exist and are assumed to be available for use in the Digital Library System.

Together, the Object Repository, Programming Language Support and Protocols subsystems provide the primary support for the Knowbot Operating Environment (KNOE) which is described in Section 3. The KNOE is a collection of software which mediates the creation, cloning, destruction, scheduling and migration of Knowbots. It provides an interface to the various underlying support services, including inter-Knowbot messaging, Knowbot teleportation and access to the Object Repository. Associated with the KNOE is a Natural Language Support subsystem which is built into the environment to make more efficient the processing of natural language by Knowbots. Natural language processing requirements arise from at least two sources: the content of objects in the Library and interactions with users.

Above the level of the KNOE and its associated natural language facilities, the PLS houses a variety of Knowbots whose functions can be roughly classified into three categories: user agents, library content and administrative. The Knowbots are illustrated at the top of Fig. 3, enclosed in ellipses. The ones shown are not intended to be exhaustive but rather to suggest the kinds of functions which would be present in a stand-alone Personal Library System. Many, if not all, of these functions would also be needed for a PLS to operate in the even richer environment of multiple Digital Library Systems (or even one Digital Library System or even just another Personal Library System).

The document editing Knowbot interfaces with a user, making use of the variety of interaction support mechanisms discussed earlier. This Knowbot is capable of creating, interacting with

and altering objects in the Object Repository, presenting them or otherwise rendering ("playing" in the case of audio output) their contents. The actual implementation of a compound document editor might involve a number of Knowbots, each with specific expertise in the manipulation of different classes of information.

The document searching Knowbot has knowledge of the contents of the PLS and is capable of interacting with the user to determine what information is desired. In the context of the larger Digital Library, the document searching Knowbot must have access to knowledge about the nature and whereabouts of non-local information. Such information, contained in the Object Repository, might range from precise identification of the location of a document to information only of other Knowbots to contact to assist with the search. A consequence of program or user interaction with the document searching Knowbot may be the creation of one or more new Knowbots which can assist in carrying out the search.

One of the more interesting concepts in the user interfacing part of the Digital Library is the notion of "shared icon geography." The idea is to extend the use of icon and window style interactions to linked three dimensional models of information space which can be shared across multiple PLSs. Distributed Library contents can be visually represented and can be organized in a familiar, physical, geographic or topographic fashion. Users might travel from place to place in this space, selecting objects for examination or organizing them in a new virtual space. Object representations might be linked or stored in various places in a fictitious information space. Search Knowbots, aided by Knowbots capable of producing three dimensional renderings, could organize information in accordance with user requests. Thus, the information landscape need not be uniform or constant for all users or even for the same user.

Accounting, import/export and transformation Knowbots would provide local services to the PLS similar to those contemplated in earlier description of principal Digital Library components with similar names. The accounting Knowbot, for instance, would keep track of the usage of or reference to personal library contents and, through the usage reporting Knowbot in the administrative category (see Fig. 3), identify usage for statistical or royalty reporting purposes.

The user agent Knowbots deal largely with users or on behalf of users and interact with Knowbots in the library content and administrative categories. The library content Knowbots assist in the registration of new objects (e.g., documents), deal with searching the local object repository and capture statistics about the use or content of the library. The indexing and cataloging Knowbots are responsible for assisting in the search for or the installation of new objects in the library. As objects are added to the local library, the library cartography Knowbot keeps track of their presence. If the PLS is used to interact with other components of the Digital Library System, the cartography Knowbot captures data about the location and nature of these components and their content. Thus, the cartographic Knowbot can learn where to find objects or to find information about certain topics.

The diagnostics and backup Knowbots are tools for initiating special functional checks for proper system operation or for assuring that information stored in a Personal Library System can be reliably and redundantly archived.

In its Personal Library System mode, the user registration Knowbot is concerned with validating a local user for purposes of access control and possibly for accounting, especially in the case of access to information with associated usage fees. In the more general environment, the user registration Knowbot may be needed to validate incoming requests for information or to decide whether to host an arriving Knowbot. The Personal Library System is thus a microcosm of the larger scale Digital Library System.

3. Knowbots and Their Application

3.1 Overview

A Knowbot is an active program capable of operating in its native software environment. Knowbots are present in each of the various components of a Digital Library System. They can be cloned, replicated, created, destroyed, can be resident at a given host system or can move from one host machine to another. Knowbots communicate with each other by means of messages.

Knowbots act as the primary medium of communication and interaction between various major components of the Digital Library System. They may even transport other Knowbots. Generally, a Knowbot may be viewed as a user Knowbot or as a system Knowbot depending on whether it directly serves an individual user or not.

A user Knowbot will accept retrieval instructions from a user and determine how best to meet the stated requirements, perhaps by interacting with other Knowbots and functional elements of the Digital Library System. Knowbots then proceed to acquire the desired information by accessing the appropriate parts of the library system. In carrying out this task, they may rely on intelligent indexing services provided by other Knowbots or perform actual text searching where needed.

One set of system Knowbots specifically attend to locally available library information. They take requests from user Knowbots and actually retrieve the documents from storage (or conversely store them away). Another set of system Knowbots attend to background and administrative tasks such as diagnostics, backup and accounting.

A class of trusted Knowbots called *couriers* have the special responsibility to look after selected objects on behalf of their authors or other owners of rights in the objects. A courier may be entrusted with responsibility for an entire database or a specific document or only a portion of it. Public domain documents which may be freely transmitted, used and copied will not generally require courier services. However, we view this as a special case of a courier which is passive. For purposes of this discussion, we assume that all documents are entrusted to couriers and never appear in the library system without an accompanying courier to protect the owner's rights. The combination of a courier and its accompanying entity (e.g., paragraph, document, database) is controlled object in the system.

When a controlled object is provided to a user, all access to its contained entity is handled via its courier. If the owner of the entity originally wished to charge on a per use basis, the courier will be instructed to report such usage when it actually occurs, or to seek permission for use immediately beforehand and to deny access if it cannot be granted. Should a user wish to extract a portion of the controlled object, say for inclusion in another document, a new courier and controlled object would be created to convey the information and to represent the owner's potential interest in the user's new work.

Certain Knowbots have a permanent status within each user's system and are known as resident Knowbots. Another class of Knowbots may be spawned dynamically for the purpose of carrying out a specific task and are deleted with the task is done. These are known as transient Knowbots.

Both resident and transient Knowbots have equal status within the Digital Library System while they exist. Should a resident Knowbot need to carry out a function at another site, it will cause a transient Knowbot to be cloned for that purpose. Transient Knowbots can also be used for system updating and for populating new user systems. In this case, they might be used as templates for creating permanent resident Knowbots at the destination and then deleted.

Although the details of Knowbot construction and operation are not fully determined, the structure of a Knowbot will be refined as we explore the design of the Digital Library System. Initially, however, we envision it to behave somewhat like a cross between a Smalltalk-like object and an expert system. Thus, we expect to use many of the attributes of object-oriented programming and rule-based systems initially. As experience with this type of active programming style develops, we would expect the Knowbot concept to evolve in both structure and capability.

3.2 The Knowbot Operating Environment

Knowbots are created, destroyed and otherwise managed by a Knowbot Operating Environment called a KNOE. The KNOE provides the context in which Knowbots function within a Digital Library System. It manages the system resources needed to support them and supports inter-Knowbot communication.

A cross section of the DLS is illustrated in Figure 4. It depicts the KNOE as an annular ring and the Knowbots as circles or spheres on its periphery. Each PLS is shown as a sector or wedge containing a portion of the KNOE and some Knowbots. The principal components of the DLS are also wedges in the figure.

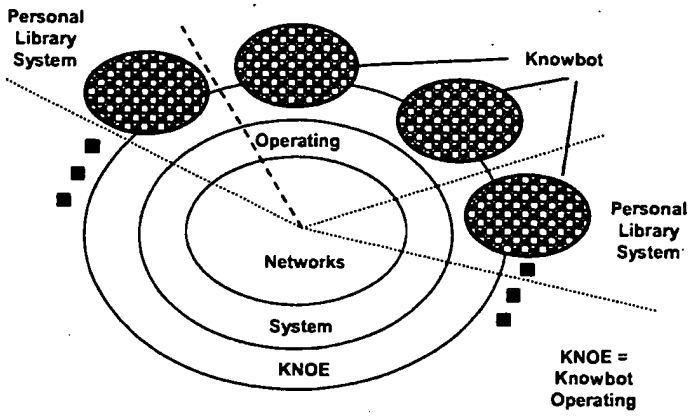


Figure 4 A Cross Section of the DLS

Each principal component of the Digital Library System contributes to and participates in the common Knowbot Operating Environment. Each local KNOE will know about all Knowbots in its local system and selected Knowbots elsewhere in the common KNOE.

Interactions between Knowbots are mediated by the KNOE. It assists in transporting messages between Knowbots in a given personal system and between systems. The KNOE will validate and authenticate messages when necessary. In a given local KNOE, any underlying message passing capabilities of the underlying operating system will be used by the KNOE in providing its layer of support.

Ideally, the KNOE could itself be created out of resident system Knowbots so that only a single architectural style is needed. However, the pragmatics of implementing the system may dictate that portions be programmed more conventionally. This aspect will be examined carefully during the early phases of the program. When detailed design and implementation choices must be made, whichever strategy (or combination) appears most desirable will be selected.

The resulting system will be designed for easy portability to other hardware and software bases. The ease of portability will depend on the extent to which the KNOE can be transported. If most of the KNOE is composed of Knowbots, then only a bootstrap version of the KNOE may be required. This is the minimum requirement on the underlying hardware and operating system. If the entire KNOE is conventionally programmed, the demands made on the underlying hardware and software may be larger as well.

3.3 Knowbots as Agents

Knowbots may themselves be nested or defined recursively by drawing on the capabilities of other Knowbots, including themselves. For example, there might be a Knowbot created to handle compound documents. This, in turn might invoke separate Knowbots for handling text, images, graphics and even electronic mail (which might itself contain a form of compound

document). It will be a design choice as to which Knowbots are visible to the user and which are hidden, in effect.

The top level of Knowbot in the system is called an *agent.* Initially, three agents are defined in the system. These are the user agent, the content agent and the administrative agent. Each agent consists of a set of resident Knowbots and each may request of the KNOE to generate transient Knowbots to assist with its work. At least initially, new agent types must be created outside the system.

The user agent consists of Knowbots for compound document generation and editing, document search and retrieval, document analysis, import/export, organizational structuring, accounting and authorization and interfacing with the user. Users deal directly with the user agent and each of its Knowbots, in turn, deals with the other agents.

The library content agent consists of Knowbots that handle document registration, indexing and cataloging, object storage and retrieval, storage management and icon geography, accounting and statistics. In addition, it contains a Knowbot to interface with other agents. The content agent is a system agent responsible for dealing directly with the object library. It receives input requests primarily from the user agent, but users do not communicate directly with it.

The administrative agent is concerned with tasks such as user registration, operations, diagnostics, backup and other similar functions such as financial analysis, billing and collection. Its main function is to support the other agents.

A typical user request for service might proceed as follows. The interface Knowbot would first determine the user's general intent and then attempt to capture what it believes is a valid user request. Let us assume the user wishes to retrieve a particular document but can only describe it generally in natural language.

The interface Knowbot would verify that the request was valid (this is largely research but simple tests can be used initially) and pass it along to the search and retrieval Knowbot to formulate a plan for satisfying the request. It might then invoke a strategy Knowbot, or a domain Knowbot to further refine its plan and then spawn one or more transient retrieval Knowbots. Each of these Knowbots might interact with other Knowbots to carry out its task.

If the requested document is not likely to be located in the user's personal database, but rather elsewhere in the system, the retrieval Knowbot is dispatched over the internet to other parts of the DLS. If the document is local, the Knowbot interacts with the storage and retrieval Knowbot in the database agent to hand off its specific task. Upon retrieval from the database, the document is supplied to the retrieval Knowbot representing the user agent after which it is ultimately made available to the user.

3.4 The User Interface

Knowbots have the primary responsibility for crafting the user's view of the library. The user conveys what he wants to see and how he would like the information presented. One or more Knowbots may then collaborate in creating the view.

The interface to the Digital Library System is essentially visual although we do not rule out other modalities such as sound. Knowbots as well as documents and controlled objects are depicted as icons and both may move dynamically in certain cases. Each Knowbot is represented by an iconic, three-dimensional symbol and its name, both of which may vary depending on the context.

The use of visual as well as logical recursion is intrinsic to the user interface. A Knowbot may be visible or invisible at the interface depending on its level of abstraction. For example, a simple search Knowbot which consists of a strategy Knowbot, an execution Knowbot and a domain Knowbot may be represented to the user as a single virtual Knowbot that does searching or as some combination of these three.

Knowbots collaborate to depict distributed objects in the DLS. Messages are used to convey the necessary information from one Knowbot to another. Multiple users will also be able to jointly participate in a joint retrieval exercise and maintain consistent views no matter which user initiates or takes an action.

The object repository may reside at multiple locations, yet the user's view should enable a single coherent logical representation of the objects independent of their location. Two users collaborating in the library system should be able to share a combination of their views as a single coherent and integrated view of the system. We refer to this aspect of the system as shared icon geography.

One of the more important concepts in the Digital Library Systems is the idea of being able to share object representations, including the details of iconic presentation, with other parts of the system. For example, if a user has a Personal Library System which contains a number of objects, it should be possible to copy the iconic representation of these objects to another Personal Library so that two users can explore the same object space together. The resulting "shared icon geography," which includes both the details of iconic presentation and the cartographic relationship among objects, would permit groups of users to work concurrently in a common information environment, coordinating the joint manipulation, examination and use of portions of the Digital Library's information space.

A particularly important issue is how to present retrieved information to the user when 1) the amount of it is inherently large or, equivalently, 2) when there are more than a few objects to be presented. This is fundamentally a research issue. We plan to seek a solution compatible with the use of shared icon geography. In addition, a simple way must be created to specify parts of the object space to browse. Electronic messages are a particularly good set of objects on which to start.

A specific request to "find the message I received 6-8 months ago about the design of the next generation workstation" may be too imprecise. Even if the system were told it was one or two pages in length with unknown sender, the Knowbot may still have to read each and every message to find the right one, if indeed it still exists (or ever did). If a few hundred messages should happen to fit the bill, the system might be unable to resolve the issue without first presenting choices to the user. The key question here is how to present this information most effectively.

As in the real world, the concept of "place" and "object" have meaning in the information space of a Digital Library. The iconic representations of places and objects in the Library are essentially multi-dimensional although they would be portrayed on a screen as two-dimensional projections of three-dimensional entities. Users interacting with iconic representations of objects should be able to use familiar, real-world paradigms to manipulate the objects and maneuver in the places that populate the Digital Library.

Objects should be able to convey the notions of containment, emptiness and fullness. It should be possible to move objects, open and close them, enter them, move about inside of them, move other objects into them and so on. It should be possible to copy all or part of an object, assuming the user has the appropriate access rights. It should also be possible to designate portions of objects to be copied and transported elsewhere. This may be achieved by some combination of highlighting and annotating the appropriate portions of the object. Two very simple examples are 1) selecting bibliographic information from a given text to be incorporated automatically in a personal data base and 2) collecting information typically found in address books such as name, address and telephone number.

The sorting and searching activities of Knowbot agents, working on behalf of one or more cooperating users, may result in the construction of three dimensional views of iconic objects found in the information space. Some of these representational ideas were originally explored by N. Negroponte of MIT in the Spatial Database Management System. Others are motivated by powerful notions of the visualization of active processes and the value of emulating common sense real-world behavior in the artificial information environment of the Digital Library System.

The realization of these ideas will require the application of leading technology in high resolution, color workstations, as well as research in powerful three-dimensional static and dynamic rendering methods, and techniques used in cinematography and television production to help "viewers" maintain context in complex visual scenes. To be effective, the user interface to the Digital Library will have to draw upon internal models of information space and options for navigating through it, techniques for animation, models of purposeful behavior and notions of goals and tasks to be accomplished. In short, the strengths of nearly every aspect of computer science, video-graphics, simulation and artificial intelligence must be marshalled to achieve the goals of the project.

3.5 Other Applications of the Digital Library System

Four possible applications of the DLS are described below. These are referred to as the Filter-Presenter, the Design Database Manager, the Researcher-Analyst and the Diagnostic Imager. Each of these uses would be implemented as an agent in the system.

The Filter-Presenter aids a user who is normally burdened by too much arriving information in the ordinary mailstream (e.g., magazines, newspapers, journals and electronic mail). If the material can be scanned electronically by the Knowbots, the user can be presented with only those aspects of the documentation he wishes to see. Of course, the user must first supply the Agent with sufficient guidance to carry out its task (including how he wants to see the results presented). Many research questions abound.

The filter may be too strong and therefore important items may be missed. Conversely, it may be too weak and the user will still be overloaded with irrelevant information. Irrelevant information may also be produced by a strong filter and relevant information missed by a weak filter, but both cases are much less likely.

The Design Database Manager couples a design program to an underlying database of relevant support information. It can also augment multiple design programs working collaboratively on a common design. In the case of VLSI design, for example, the elements in the database might be chip designs that could be used as supplements in a larger design. This could represent work underway by a team of designers. Or it could include standard designs such as simple microprocessors which may have limited use otherwise and could be used as pieces in a larger state-of-the-art chip. The advantage of this approach is that a new microprocessor design is not needed and users can be expected to be experienced in the use of existing designs for which software is already available. Alternately, the database manager could know about blueprints and bow to use them and assist a user in retrieving and interpreting them.

The Researcher-Analyst assists a person who would normally search through large collections of documentation seeking specific types of information about a particular topic. It might identify hundreds of possibly relevant items that the user would have no time to explore. This agent could search all of them and develop information for the user depending on the nature of his research. For example, if the researcher was concerned about the history of infrastructure, he might ask the system to locate as many documented examples of early uses of electricity as possible. This task might normally take weeks or months to accomplish manually but the agent using the library might accomplish it in minutes or less.

The Diagnostic Imager assists a person to find ways of binding textual or quantitative information with imagery. A reference to or selection of a given portion of an image or chart will automatically select the related textual information or vice-versa. This must be done in a well defined semantic context and not merely a geometric one.

Medical information needed to assist in patient diagnosis and evaluation covers an extreme range of modalities and levels of abstraction. From patient interviews to blood sample analyses

to X-rays, CAT scans and electrocardiograms, the Diagnostician is confronted with a rich and often perplexing array of information from which must be distilled an evaluation. In the context of the Digital Library System, the diagnostic task calls for access to a broad range of information which may range from specific information about the side effects of various drugs and chemicals to treatment protocols to indices of comparative medical imagery. The agent interacting with the user and accessing and manipulating the digital library content will rely on the use of Knowbots to transform symptomatic terms or analytic descriptions into appropriate keys for selecting useful imagery or to aid in searching for relevant treatments.

Finally, it should be noted that the structure of the DLS as a Knowbotic system makes it well suited for problems and applications that involve process control. A collection of Knowbots can be spawned to carry out a process control task and the library system architecture can be used to monitor its execution as if one were retrieving information from a more conventional library. The importance of this concept is noted, but it is not elaborated on further here.

3.6. Systems of Digital Library Systems

If an integral Digital Library System were to be constructed and placed in operation, experience predicts that evolution will result in other autonomous Digital Library Systems being generated in the future. The specifics of each system will surely differ from those of the others, and thus will arise the need for communication between these different Digital Library Systems. We call this inter-DLS communication.

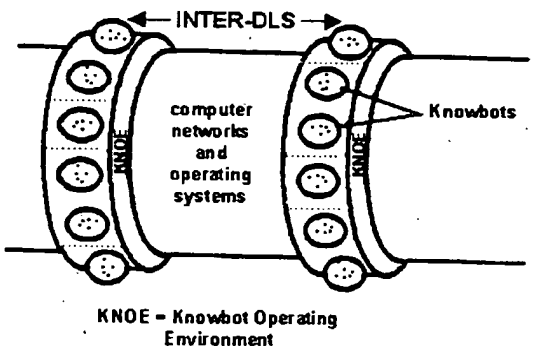


Figure 5 The KNOE Bracelet Model

We plan to define a language for inter-DLS communication which enables autonomous independent Digital Library Systems to interoperate. Most likely, this language will rely heavily on natural language as if it were supporting a normal user of either system. In this case, however, the user would be another DLS. The exact form of this language will be developed early in the project and will enable certain documents and requests to pass between systems. Inter-DLS communication is illustrated in Figure 5, where each DLS is shown as a separate bracelet on an internet-based, distributed substrate.

While a given DLS may be quite powerful in its own capabilities, the user should expect a more limited set of capabilities when multiple DLSs are involved. This may be dictated by administrative or legal restrictions dealing with enforcing copyrights as much as by technical limitations. For example, we assume Knowbots cannot be passed between autonomous DLSs and that certain documents may not be permitted to leave one DLS for another in digital form.

4. Implementation Plan

The architecture described in the previous section consists of eight significant functional component types interacting according to a set of protocols and a methodology which is observed by all the components. A three phase incremental development plan is presented here to achieve the library system program objectives. We first describe the elements which will be addressed in phase one which lasts an estimated 24 months, followed in succession by the second and third phases which last an estimated 24 and 12 months respectively. The activities in these various phases are described below.

4.1 Phase One

Four major tasks are presently envisioned for phase one. These are 1) the Knowbot Operating Environment, 2) the user interface, 3) populating the initial data bank for testing, and 4) Natural Language Text Searching. In addition, activities will be pursued in parallel to refine and further develop the overall system architecture, to explore one or more applications for the DLS and to consider matters relating to reasonable compensation for digital access and use of intellectual property.

4.1.1 The Knowbot Operating Environment (KNOE)

The description of the KNOE, the role of Knowbots, and their activities in the KNOE was presented in Section 3. This task will define the KNOE in detail, will develop a prototype system and implement three simple but functional Knowbots. Two separate subprojects are envisioned here to allow for competing views of the KNOE during this preliminary phase of the effort and to generate a total of six functional Knowbots.

The first of these two efforts, called KNOE-I, will concern itself with the issues of document creation and entry into the Digital Library System (DLS). It will focus on the user agent side of the library system and will implement Knowbots for the Document Editor, the Importer/Exporter and the Transformation Server. Using this system, documents prepared according to one standard may be manipulated by users with access to other standards or merged with documents prepared in other standards. The combined document will appear to the user as if it were a single properly merged document independent of whether the representation form of the document is changed or not. In general, the Transformation Server will convert the internal representation of a document without affecting the representation of the original copy. It will also insure that suitable notice is taken of the derivation of the work.

The Importer/Exporter will utilize a conventional database to demonstrate the issues involved with bringing new material into the Digital Library System or allowing it to migrate outside. It will also facilitate the incorporation of electronic mail or separate objects in the library system even if they were originally transmitted or received without any awareness of the Digital Library System.

The second KNOE effort, called KNOE-II, will concern itself with retrieval of documents in the library system and issues of management of the database itself It will focus on the content of the library. In particular, it will implement Knowbots for search and retrieval, indexing and cataloging and a minimal registration function. At the time of initial deposit, each document will be cataloged and indexed for future retrieval.

The search and retrieval aspect will focus primarily on interactions with the database and with the user to a lesser extent. In both of these efforts, experiments with cooperating Knowbots and multiple KNOE's will be undertaken. Knowbots will move from one operating environment to another and will cause new Knowbots to appear at the other site by message passing. These activities will be under the overall supervision and control of the KNOE. Simple document handling scenarios will be executed.

4.1.2. The User Interface

This effort will implement a prototype visual user interface consisting of a shared icon geographic view of the contents of the library system and its components. Two components of the user interface will be the icon geography system which is responsible for interacting with the user and the library cartographic system which maps the relevant contents of the library and interacts with the object repository. These two systems interact with each other directly.

Initially, the functionality of these systems will be explored and demonstrated. Ultimately, each will be represented as separate Knowbots within the system, once the concept of Knowbots has been demonstrated.

4.1.3 Populating the Database

This effort will address those key aspects of the database having to do with personal, organizational (inter-organizational) and public information. Initially, the focus will be on populating an experimental database with objects that are publicly available or of organizational interest to be used for testing. After the Database Server is developed in phase two, network connections will be made available to existing databases.

We envision this effort will involve use of representation standards already in existence (or developed in the program, if necessary) and concentrate on collection and creation of an initial database. The equipment used will include scanners, optical character readers, and facsimile devices.

Public Documents - Our initial focus here will be to collect in digital form relevant standard documents from organizations such as ANSI, CCITT, ISO, IEEE, NBS or governmental agency pronouncements.

Organizational - Here we plan to focus on selected equipment manuals that are nominally available to customers or for internal use. This might cover hardware, software, and procedures for installation, use and repair. Eventually it might include brochures, pictures and

specification sheets as well. Another organizational focus will be to collect and represent vita's of graduate researchers in the nation's colleges and universities. This will be carried out in cooperation with the university librarians who will each have responsibility for accurately representing their own school.

Personal - Finally, to make this effort interesting to the research community and to motivate some of their work, we plan to include selected research reports issued by the various universities which are not easily available otherwise. In addition, we shall include selected scientific and business publications based on their relevance and the willingness of the publishers to cooperate. Candidates are AAAI, IEEE, Scientific American, and ACM on the scientific side, and Business Week, Harvard Business Review, Fortune and Forbes on the business side.

4.1.4 Natural Language Text Search

This effort will focus on demonstration of the use of natural language for search and retrieval. Initially, electronic mail will be chosen as a candidate to demonstrate retrieval based on imprecise English requests. When an object base with actual documentation is ready in the library, the domain will be expanded to include it as well.

This task will entail dealing with ungrammatical writing (but will not necessitate building a theory of such writing) and with understanding quite a bit about messages in general. Nonetheless, the domain appears to be relatively bounded.

The Natural Language System will be structured so that it draws upon existing state of the art technology but modifies it so that the system may be handed critical information it needs to do its job. Ultimately, this information will be provided directly by the Knowbots. In general, the Natural Language System will know about everyday English words. The system will receive a lexicon of relevant specialized words and their meanings, along with any additional helpful information. The system shall be structured such that new grammars and basic vocabularies may eventually be supplied to handle other languages.

In parallel with the above activities we expect several organizations to begin work on the development of a Personal Library System which provides user access to the library. These efforts may be simply to interact with the external development efforts, provide one or more individuals to work on them or begin an internal effort.

Throughout phase one we shall need to maintain, refine and update the architectural specification of the library system. At critical junctures, preliminary protocol specification documents will be produced for the critical interfaces between the principal components that were shown in Figure 2, for representation standards as well as for normal system capabilities such as Knowbots. This will be a living document which shall serve as the "sheet music" for the entire effort.

During this phase we will explore several possible applications of the Digital Library System along the lines indicated in Section 3.3.6. Initial designs for one or more of these efforts will begin. We shall also explore concepts for handling the reasonable compensation of intellectual property owners, since fragments of their as well as entire works may be involved.

4.2 Phase Two

In the second phase, we plan to begin development of a Personal Library System based on the research efforts in the first phase. In addition, we expect several industrial organizations to actively participate in the process so that a commercial source of the technology will ultimately be available.

The initial system will be based on a powerful workstation with local disk, scanner, printer and high resolution display (plus mouse and of course keyboard). Eventually, it will be graphics capable and be equipped with facsimile, low-cost personal high density storage such as a CD-ROM, plus an acoustic subsystem (including speech) and video.

Research will continue on several fronts. First, the Knowbot research will be expanded to include exchanges between Knowbots and making Knowbots work in collaboration with the object base. This effort will include Knowbots with domain expertise and the ability to do simple domain related problem solving.

The registration Knowbot will be fully developed and outfitted to handle users and services as well as documents. In addition, it will be expanded to be knowledgeable about an organizational level as well as a personal level of objects. It will also have a mechanism for internal collection of accounting information.

Research will also be undertaken to develop a Knowbot which understands organizational structure and documents associated with it. The structure of an organizational library system will be developed and simple exchanges between organizational and personal Knowbots explored. The Personal Library System technology will provide the basis for the organizational system as well; only the contents of the two systems will differ.

The task of database population will continue as a larger and richer set of documents is added to the object base. We expect this task to focus primarily on the more sophisticated elements of the existing documents such as equations, graphics and images. However, additional documentation will be added as appropriate for the research to be conducted.

The natural language capability for text search will be improved and a user front end will be incorporated which relies on a common body of natural language software. Experimentation with sample retrieval requests will take place interactively.

A Multi-Processor Database Server will be developed to access remote Database Systems. Initial experiments will be conducted with one or more cooperative suppliers of information. Candidates are the National Library of Medicine and Dow Jones. The Database Server will

incorporate much of the Personal Library System software but be outfitted to operate several orders of magnitude faster than the personal workstation and support multiple users. The server will draw directly upon the results of the Personal Library System research and apply it to multi-processors.

Finally, an effort will be undertaken to focus on some of the practical and administrative considerations such as tools for billing and collection, diagnostics, back up, etc. Also, those aspects which allow new capabilities to be added on the system to be reconfigured over time will be included here.

Sometime during Phase Two, we expect to obtain a working prototype of the initial Digital Library System and to begin making it available to selected members in the research community for experimentation and as an object of research itself. In addition, the development of one or more of the applications will be undertaken.

4.3 Phase Three

The components of the Digital Library System (DLS) will be fully integrated and a quasi-operational DLS will be created during this phase for research purposes. The existence of the system will serve to expand the Digital Library System to more users and a larger set of documentation. We expect to gain operational experience and to incorporate several additional public and private systems such as NTIS, Westlaw, Lexis/Nexis, Compuserve or Dialog.

An organizational prototype will be created working with one or more groups and experiments with inter-organization exchanges explored. Examples of inter-organizational exchange requirements will be developed and simple interactions carried out. These might include access to manuals, student vita, electronic mail, or open memoranda. Acoustic Input and Output including speech and other audible sounds will be incorporated into the user interface and further work on the shared icon geography carried out for dealing with complex representations. One or more applications will be substantially completed.

Finally, we assume that other concepts for a Digital Library System will emerge and compatibility with them will be required. Hence, we will investigate the requirements for inter-Digital Library System exchanges and will begin experimenting with such in the context of two autonomous (but homogeneous) Digital Library Systems.

4.4 Follow-on Plans

At this stage, an experimental Digital Library System will be functioning with a small but interesting class of documents, an initial application, a nominal class of users and an expandable architecture. If the system performs effectively, as we expect it will, it is now a candidate to turn into a genuine piece of infrastructure for the entire research community. Support will be sought to expand the system and make it more widely available. A number of possible vehicles exist to do this and we expect to create several promising alternatives for evaluation. Assuming the financial basis for developing such a system can be made available,

NRI is prepared to assist in building it. If not, the technology base will be available for the sponsors to pursue the concept by independent sales of equipment for the individual user.

ROTL 1. Remote Office Test Line. Provides the capability to originate automatic inter office trunk transmission test calls under the automatic control of CAROT, from a remote location.
2. A popular online abbreviation, shorthand for "Rolling On The Floor Laughing"; an appropriate typed response to a particularly amusing online remark. Other common Net acronyms include IMHO ("In My Humble Opinion") and IMNSHO ("In My Not-So-Humble opinion").
Rotor The rotating part of a motor or other electrical machines.
ROTS Rotary Out-Trunk Switches.
Round Cutter These are used to cut cables. The blades of the cutter are curved so that there is a space between them.
Round Robin This is a method of distributing incoming calls to a bunch of people. This method selects the next agent on the list following the agent that received the last call. See also Top Down and Longest Available.
Roundtrip Propagation Delay Roundtrip propagation delay from a burst modem to a burst modem will be about 470 milliseconds to 570 milliseconds (About half a second). See Satellite Transmission Delay.
Routable Protocols Protocols, such as TCP/IP, DECnet, and XNS, that support Network Layer addressing. Packets constructed using these protocols contain information about how data should move through a network. This information, carried in the NLA (Network Layer Address) field of the packet, is used by internetworking devices to make routing decisions.
Route The path that a message takes. In telephone companyese, a route is the particular trunk group or interconnected trunk groups between two reference points used to establish a path for a call. This term (or the term routing) is also used as a verb to define the act of selecting a route or routes.
Route 66 A colloquial term for the Internet, with something interesting everywhere along the way. Route 66 used to be the way we drove across America before they put in concrete highways.
Route Advance This feature routes outgoing calls over alternate long distance lines when the first choice trunk group is busy. The phone user selects the first choice route by dialing the corresponding access code. The phone equipment automatically advances to alternate trunks and trunk groups, based on the user's class of service. Route advance is a more primitive form of least cost routing. See Least Cost Routing.
Route Daemon A program that runs under 4.2 or 4.3BSD UNIX systems (and derived operating systems) to propagate routes among machines in a local area network. Pronounced "route-dee."
Route Discovery Process through which a bruuter can learn LAN topology by passing information about its address and the LANs it connects and receiving the same information from others.
Route Diversity See Diversity.
Route Flap Frequent change in state (addition-removal) of customer Internet route announcements. Route flaps are typically caused by BGP session resets; changes in state (on-off) of router; changes in state (up-down) of private line; change in router filter list; high circuit error rate.
Route Guide A map showing how calls are to be routed at the switch: first choice group of circuits, second choice, third choice, and fourth choice.
Route Indicator An address or group of characters in the heading of a message defining the final circuit or terminal to which the message is to be delivered.
Route Length The actual length of a route, or path, between transmitter and receiver. The length of the route is one factor that determines propagation delay, i.e., the delay associated with a signal as it transverses a network. Route length is determined by measuring the actual length of a the path, rather than the distance "as the crow flies."
Route List A sequence of trunk groups that can be searched for a particular route. This list is comprised of trunk groups and configuration attributes (e.g. Class of Service) governing the use of a particular trunk group.
Route Mile Let's say that you have two sheaths of fiber, each of which contains ten fibers and runs for one mile. That is one route mile (total distance of all fibers), two sheath miles (2 sheaths running one mile), and twenty fiber miles (20 fibers running one mile).
Route Optimization Another way of saying Least Cost Routing.
Route Server An ATM term. A physical device that runs one or more network layer routing protocols, and which uses a route query protocol in order to provide network layer routing forwarding descriptions to clients.
Route Xpander Card A board manufactured by IBM for insertion into a PC which provides the PC with a wide area interface to a frame relay network, including handling all of the necessary encapsulation.
Routed Protocol A protocol that can be routed by a router. In do so a router must understand the logical internetwork as perceived by that routed protocol. Examples of routed protocols include DECnet, AppleTalk, and IP.
Router 1. As in software, router is a system level function that directs a call to an application.
2. As in hardware, routers are the central switching offices of the Internet and corporate intranets and WANs. Routers are bought by everybody
—from backbone service providers to local Internet Service Providers (ISPs), from corporations to Universities. The crown provider of routers in the world is Cisco. It has built its gigantic business on selling routers
—from small ones, connecting a simple corporate LAN to the Internet, to corporate enterprise wide networks, to huge ones connecting the largest of the largest backbone service providers. A router is, in the strictest terms, an interface between two networks.
Routers are highly intelligent devices which connect like and unlike LANs (Local Area Networks). They connect to MANs (Metropolitan Area Networks) and WANs (Wide Area Networks), such as X.25, Frame Relay and ATM. Routers are protocol-sensitive, typically supporting multiple protocols. Routers most commonly operate at the bottom 3 layers of the OSI model, using the Physical, Link and Network Layers to provide addressing and switching. Routers also may operate at Layer 4, the Transport Layer, in order to assure end-to-end reliability of data transfer.
Routers are much more capable devices than are bridges, which operate primarily at Layer 1, and switches, which operate primarily at Layer 2. Routers send their traffic based on a high level of intelligence inside themselves. This intelligence allows them to consider the network as a whole. How they route (also called routing considerations) might include destination address, packet priority level, least-cost route, minimum route delay, minimum route distance, route congestion level, and community of interest. Routers are unique in their ability to consider an enterprise network as comprising multiple physical and logical subnets (subnetworks). Thereby, they are quite capable of confining data traffic to a subnet, on the basis of privilege as defined in a policy-based routing table. In a multiple router topology, each router port defines a physical subnet, and each subnet is a separate domain. Within that domain, all connected devices share broadcast traffic; devices outside of that domain can neither see that traffic, nor can they respond to it. Contemporary routers have the ability to define subnets on a logical basis, based on logical address (IPX address or IP address) information contained within the packet header, and acted upon in consultation with a programmed routing table. In addition to standalone routers designed specifically for that purpose, server-based routers can be implemented. Such routers, in the form of high-performance PCs with routing software. As software will perform relatively and efficiently than firmware, such devices generally are considered undesirable for large enterprise-wide application, although they do serve well in smaller remote offices and less-intensive applications. Routers also are self-aware; they can communicate their existence and can learn of the existence of new routers and LAN segments. Routers constantly monitor the condition of the network, in order to dynamically adapt to changes in network conditions.
Characteristics of routers can include:
- LAN Extension
- Store & Forward
- Support for Multiple Media
- Support for Multiple LAN Segments
- Support for Disparate LAN Protocols
- Filtering
- Encapsulation
- Accommodation of Various and Large Packet Sizes
- High-Speed Internal Buses (1+ Gbps)
- Self-Learning
- Routing Based on Multiple Factors
- Route Length
- Number of Hops
- Route Congestion
- Traffic Type
- Support for a Community of Interest (VLAN)
- Redundancy
- Network Management via SNMP

Router protocols include both bridging and routing protocols.

appearance of objects when you roll over them or click on them. See also Rollover Lines.

Rollover Lines You receive many incoming calls. You don't want to miss a call, so you ask your phone company to set your phone lines up to roll over, also called hunt, also called ISG (Incoming Service Group) in telephonese. You order five lines in hunt. The calls come into the first. If the first one is busy, the second rings. If it's busy, the third rings. If they're all busy, then the caller receives a busy. The commonest types of hunting are sequential and circular hunting. Sequential hunting starts at the number dialed, keeps trying one number after another in number order and ends at the last number in the group. It's typically ascending. For example, it starts at 691-8215, goes to 691-8216, then 691-8217, etc. But it can also be ascending — from 691-8217 up. Circular hunting hunts all the lines in the hunting group, regardless of the starting point. Circular hunting, according to our understanding, circles only once (though your phone company may be able to program it circle a couple of times). The differences between sequential and circular are subtle. Circular seems to work better for large groups of numbers. You don't need consecutive phone numbers to do rollovers. Nowadays you can roll lines forwards, backwards and wrap around, for example most idle, least idle. Rollovers are now done in software. This also has its downside, since software fails. For example, theoretically if a rollover strikes a dead trunk, it should bounce to the next live trunk. But sometimes it hangs on the dead trunk and many of your incoming calls never get answered. They might ring and ring. They might hit a busy. My recommendation: Test your rollovers at least twice a day. In particular, test that your callers ultimately get a busy if all your lines are busy. Nothing worse your customer should receive a ring-no-answer or a constant busy when calling your company. See also Terminal Number.

ROLM A telephone equipment manufacturer based in Santa Clara, CA, at least once upon a time. ROLM was started in 1969 by four engineers to produce computers for the military. The company introduced one of the first digital PBXs in 1975. It was a great PBX. Later, they developed a line of KTSs (Key Telephone Systems) and hybrid PBX/KTS systems. They were not so good. IBM acquired ROLM in 1984 as part of their plan to integrate the worlds of computers and communications. It didn't work...at all. And IBM lost a lot of money with Rolm. In 1989, IBM sold ROLM to Siemens, at which time it became ROLM Company. In 1994, the name was changed to Siemens Rolm Communication Inc. In 1996, the name was changed to Siemens Business Communication Systems, Inc. Siemens really doesn't use the name ROLM (or Rolm) anymore, but there are a lot of ROLM systems still in service.

Rolodex A Trademarked product which started life as paper card based device to keep names and address on. Now it has become more of a generic name to connote software to let you look up peoples' phone numbers and addresses. Software to do this is also called PIM — for Personal Information Manager.

ROM Read Only Memory. Computer memory which can only be read from. New data cannot be entered and the existing data is non-volatile. This means it stays there even when power is turned off. A ROM is a memory device which is programmed at the factory and whose contents thereafter cannot be altered. In contrast is the device called RAM, whose contents can be altered. See Read Only Memory and Microprocessor.

ROM Font The ROM Font is your PC's type font. It consists of a set of 256 characters which cannot be edited — unless you are running in video mode, in which case you can design your own type font.

Rom Shadowing 386 and higher CPUs provide memory access on 32 & 64 bit paths. Often they will use a 16 bit data path for system ROM BIOS info. Also some adapter cards (ie. older video, network adapters etc.) with on board BIOS may use an 8 bit path to system memory. For high end computers this is a bottleneck. Like having YIELD signs out on the lanes within a freeway. ROM is very slow, 150ns-200ns. Modern RAM is 60ns or less. Therefore when the system is waiting on this data it generates wait states. For high end computers these wait states slow the entire system down. There is a system developed to transfer the contents of all the slow 8-16 bit ROM chips through out the system into 32 bit faster main memory. "This is ROM SHADOWING". This is accomplished using the MMU, the memory management unit. The MMU takes a copy of the ROM BIOS codes and places it into RAM. To the rest of the system this RAM location looks exactly like the original ROM location. This definition courtesy Charlie Irby, chsirby@foothill.net.

Roofing Filter A low-pass filter used to reduce unwanted higher frequencies.

Room Cut-Off Hotel/motel guest telephones restricted from outgoing calls when the guest room is unoccupied.

Room Status And Selection Provides the capability to store and display the occupancy and cleaning status and the type number of each guest room. This helps housekeeping management, maid locating and room selection. Also, communications between the front desk and the housekeeper are speeded up via real-time maid activity and checkout audit printouts to indicate which rooms need cleaning next. The occupancy status is normally changed by the maid or inspector dialing from the room telephone.

Root The base of a tree. The base of a hard disk. See Root Directory.

Root Directory The top-level directory of a PC disk, hard or floppy. The root directory is created when you format the disk. From the root directory, you can create files and other directories.

Root Web The FrontPage web that is provided by the server by default. To access the root web, you supply the URL of the server without specifying a page name. FrontPage is installed with a default root web named !!!root web@@@. All FrontPage webs are contained by the root FrontPage web.

ROSE Remote Operations Service Element. An application layer protocol that provides the capability to perform remote operations at a remote process. Definition from Bellcore (now Telcordia) in reference to its concept of the Advanced Intelligent Network.

Rostered Staff Factor RSF. A call center term. Alternatively called an Overlay, Shrink Factor or Shrinkage. RSF is a numerical factor that leads to the minimum staff needed on schedule over and above base staff required to achieve your service level and response time objectives. It is calculated after base staffing is determined and before schedules are organized, and accounts for things like breaks, absenteeism and ongoing training.

Rostering A call center term. The practice of rotating employees through all existing schedules in a matrix, or roster, of schedules. This "share the grief" method is prevalent in Europe and Australia, where agents work through an entire roster.

ROT13 A way to encode things that the general Internet community can't read. Each letter in a message is replaced by the letter 13 spaces away from it in the alphabet. There are online decoders to read these. For instance, Harry Newton becomes Uneel Arjgba, which sounds a lot more exotic.

ROTA A call center term. 1. An European term for a rotating shift pattern or rotating schedule, 2. Short form for roster.

Rotary Dial The circular telephone dial. As it returns to its normal position (after being turned) it opens and closes the electrical loop sent by the central office. Rotary dial telephones momentarily break the DC circuit (stop current flow) to represent the digits dialed. The circuit is broken three times for the digit 3. The CO counts these evenly-spaced breaks and determines which digit has been dialed. You can hear the "clicks". The number "seven," for example consists of seven "opens and closes," or seven clicks. You can dial on a rotary phone without using the rotary dial. Simply depress the switch hook quickly, allowing pauses in between to signify that you're about to send a new digit. It's a good party trick.

Rotary Dial Calling The telephone system will accept dialing from conventional rotary dial sets.

Rotary Hunt You buy several phone lines. Let's say 212-691-8215, 212-691-8216, 212-691-8217, 212-691-8218. Someone dials you on your main number — 212-691-8215. It's busy. (That's our number.) The central office slides the call over to 212-691-8216. If that number is busy, it slides it over to 212-691-8217, and so on. This is called rotary hunt. It hunts to the next line in the rotary group. In the old days, the phone lines you could rotary hunt to had to be in numerical sequence. But now with modern stored program control central offices, your lines in rotary hunt can be very different as long as they're all on the same exchange.

Rotary Output To Central Office Most central offices are equipped to provide tone dial service. In cases where the telephone company central office trunks are not designed to accept tone signaling, your on premise phone system (PBX, key system or single line phone) will translate the number entered by a phone in tones into rotary dial pulses which can be processed by the central office.

Rotating Cylinder (Drum) Scanner A scanning technique using a drum and a photocell scan head. The original is attached to the drum, enabling the scan head to travel along the length of the document. Reflected light from the document is concentrated on the scanner photocell, which causes an analog signal.

Rotating Helical Aperture Scanner Original is illuminated by a lamp when fed onto the platen, via a mirror and lens system, the document's image is focused first through a fixed horizontal slot, then through a rotating spiral slit disk series, and finally onto a photocell to generate an analogous electrical current.

Rotational Latency The delay time from when a disk drive's read/write head is on-track and when the requested data rotates under it.

Rotational Mailboxes Information only mailboxes whose information is automatically changed on a time sensitive or usage sensitive basis.

ROTFL I'm "Rolling on the Floor, Laughing." Used in e-mail.

...communications and data transfer. Essentially, NetBIOS is a way for application programs to talk to the network. Other applications interface as can also being used these days, such as APPC. To run an application that works with NetBIOS, a non-IBM network operating system, a network interface card must offer a NetBIOS emulator. More and more hardware and software vendors offer these emulators. They aren't always perfectly compatible, though. Many vendors either provide a version of NetBIOS to interface with their hardware or software to transport layer communications services in their network products.

NETBIOS Emulator An emulator program provided with NetWare that allows workstations to run applications that support IBM's NetBIOS calls.

NetBT NBT. NetBios over TCP/IP. A protocol supporting NetBIOS services in a TCP/IP environment, defined by RFCs 1001 and 1002.

Netcasting A method used to deliver Web content automatically to the desktop. Netcasting is referred to as push technology because content is pushed from a Web site to the user who requested receipt of the content. Content can include weather forecasts, stock market quotes, or software updates.

NETCOM A system that provides information on Customer Premises Equipment (CPE), network equipment, and contracts.

NETDDE Network Dynamic Data Exchange, a feature of Microsoft Windows.

NetFind A white pages service which enables a person to query one service and have the service search other databases for addresses matching the originally entered query.

NetFX Consortium NetFX consortium is a group of companies that have come together to provide solutions for delivering real-time multimedia over the Internet's World Wide Web. The members are Diamond Multimedia Systems www.diamondmm.com, NETCOM www.netcom.com, SAIC http://mercury.saic.com/demo), Template Graphics Software www.sd.tgs.com/~template), and Xing Technology Corp www.xingtech.com).

Netgod Also spelled NET.GOD. A person very visible on a network and who played an important role in the development of the network.

NetHead Also spelled net.head. A person who like really loves the Internet.

Nethopper A wonderfully cute name for a device that organizes data to hop from one wireless network to another. Nethoppers are also called bridges and routers.

NetHub The home of the KIAE network located in Bethesda, MD. All Super-Regional nodes on the KIAE network call the NetHub to exchange mail packets.

NETI Network Information Table.

NETIPC Network Interprocess Communications. Permits inter-CPU program sharing and lets programs running on different systems to exchange data through a set of program-callable. This peer-to-peer service is important for developing distributed applications.

Netiquette A pun on "etiquette". Contraction for "Network Etiquette". It means proper conduct on Internet. Because the recipients of text-only messages cannot see your face or hear the inflections of your voice, special care must be taken to avoid misunderstanding and to convey the "flavor" that you intended your messages to have. Some common techniques are the use of emoticons (also called emotions): punctuation that suggests sideways faces, bracketed expressions like <g> for grin and <G> for Big Grin, and careful capitalization as in I AM NOW SHOUTING AT YOU!

Netizen A citizen of the Internet.

NetKeeper See IPTC.

Netmail A private message transmission capability on FidoNet in which nodes directly communicate with one another on a point-to point (in FidoNet) or hub-routed (in PCRelay). Netmail was originally developed for use by SYSOPS to communicate with one another and is available on some BBs for regular users. See Echo.

Netmask A colloquial way of saying IP Address Mask. See IP Address Mask.

NetMeeting Microsoft's NetMeeting is a conferencing and collaboration tool designed for the Internet or Intranet. I'm giving a lot of space to it in this edition of my dictionary because I believe NetMeeting is an important product that will have a major impact on the way we communicate (voice, video, data, images, etc.) over the PSTN, over private corporate networks and over IP networks, such as the Internet and private IP networks, called Intranets. NetMeeting, according to Microsoft, has the following features: * H.323 standards-based voice and video conferencing. Real-time, point-to- point audio conferencing over the Internet or corporate Intranet enables a user to make voice calls to associates and organizations around the world. NetMeeting voice conferencing offers many features including half-duplex and full-duplex audio support for real-time conversations, automatic microphone sensitivity level setting to ensure that meeting participants hear each other clearly, and microphone muting, which lets users control the audio signal sent during calls. This voice conferencing supports network TCP/IP connections. * Support for the H.323 protocol enables interoperability between NetMeeting and other H.323-compatible voice clients. The H.323 protocol supports the ITU G.711 and G.723 audio standards and Internet Engineering Task Force (IETF) RTP and RTCP specifications for controlling audio flow to improve voice quality. On MMX-enabled computers, NetMeeting uses the MMX-enabled voice codecs to improve performance for voice compression and decompression algorithms. This will result in lower CPU use and improved voice quality during a call. With NetMeeting, a user can send and receive real-time visual images with another conference participant using any video for Windows-compatible equipment. They can share ideas and information face-to-face, and use the camera to instantly view items, such as hardware or devices, that the user chooses to display in front of the lens. Combined with the video and data capabilities of NetMeeting, a user can both see and hear the other conference participant, as well as share information and applications. This H.323 standards-based video technology is also compliant with the H.261 and H.263 video codecs. Multipoint data conferencing using T.120. Two or more users can communicate and collaborate as a group in real time. Participants can share applications, exchange information through a shared clipboard, transfer files, collaborate on a shared whiteboard, and use a text-based chat feature. Also, support for the T.120 data conferencing standard enables interoperability with other T.120-based products and services. The following features comprise multipoint data conferencing: Application sharing: A user can share a program running on one computer with other participants in the conference. Participants can review the same data or information, and see the actions as the person sharing the application works on the program (for example, editing content or scrolling through information.) Participants can share Windows-based applications transparently without any special knowledge of the application capabilities. The person sharing the application can choose to collaborate with other conference participants, and they can take turns editing or controlling the application. Only the person sharing the program needs to have the given application installed on their computer. Shared Clipboard: The shared clipboard enables a user to exchange its contents with other participants in a conference using familiar cut, copy, and paste operations. For example, a participant can copy information from a local document and paste the contents into a shared application as part of a group collaboration. File Transfer: With the file transfer capability, a user can send a file in the background to one or all of the conference participants. When one user drags a file into the main window, the file is automatically sent to each person in the conference; they can then accept or decline receipt. This file transfer capability is fully compliant with the T.127 standard. Whiteboard: Multiple users can simultaneously collaborate using the whiteboard to review, create, and update graphic information. The whiteboard is object-oriented (versus pixel-oriented), enabling participants to manipulate the contents by clicking and dragging with the mouse. In addition, they can use a remote pointer or highlighting tool to point out specific contents or sections of shared pages. Chat: A user can type text messages to share common ideas or topics with other conference participants, or record meeting dates and action items as part of a collaborative process. Also, participants in a conference can use chat to communicate in the absence of audio support. A "whisper" feature lets a user have a separate, private conversation with another person during a group chat session. TAPI 3.0 and NetMeeting both support core IP Telephony capabilities. Each platform offers unique benefits: TAPI 3.0 integrates traditional telephony with IP Telephony, providing a COM-based, protocol-independent call-control and data streaming infrastructure. NetMeeting SDK supports T.120 conferencing and application sharing in addition to IP Telephony. Applications using TAPI 3.0 and the NetMeeting API interoperate using H.323 audio and video conferencing. Because TAPI 3.0 and NetMeeting both support core IP Telephony capabilities (including support for H.323), developers may want to consider the following guidelines when choosing an API for their IP Telephony applications: TAPI 3.0. This is the API to use if you are doing IP Telephony in your application. TAPI 3.0 is especially valuable in the world of client/server computer telephony integration, for combining IP Telephony with traditional telephony, and for IP multicast of voice and video. NetMeeting API. This is the API to use if you are doing real-time collaboration and want to integrate voice, video, and data conferencing into your application. The NetMeeting API is useful for applications that want to integrate application sharing, whiteboard functionality, and multipoint file transfer with voice and video sessions. www.microsoft.com/netmeeting.

NETOPS Network Operations System.

NetPC See Network Computer and Thin Client.

Netscape Netscape was once the most famous Web Browser is Netscape. It owes its genesis to a software program called Mosaic, which Marc Andreessen and a small team of student programmers working for $6.85 an hour in 1993 wrote at the National Center for Super-

computing Applications at the University of Illinois at Urbana-Champaign. A year later, after becoming annoyed at the way the University had taken over his Mosaic creation, Mr. Andreessen proposed a "Mosaic Killer" — a new and improved version of his own creation. The team was back at work by April 1994 in a company called Netscape. And by October, they had created a new version of Mosaic, called Netscape Navigator. Netscape went public in August of 1995 in one of the most successful IPOs (Initial Public Offerings) ever. when Microsoft started giving away its Internet Explorer Browser for free, Netscape fell on hard times. And in late 1998, America OnLine (AOL) bought it.

Netserv A file server used for distributing files directly related to the BITNET network.

Netsite The term Netscape Navigator uses to refer to a URL or WWW address.

Netsploitation Flick Any one of the Hollywood films about the Internet.

Netstat A utility program used to show server connections running over TCP/IP (Transmission Control Protocol/Internet Protocol) and statistics, including current connections, failed connection attempts, reset connections, segments received, segments sent, and segments retransmitted.

Netstation In an Internet scenario, thin clients are known as NetPCs or Netstations. The NetStation is reliant on the server, which is provided by your company or a service provider (e.g., America OnLine, CompuServe, or your ISP). In addition to providing some combination of content and Internet access, the service provider's server will provide your client NetPC with access to all necessary applications (e.g., word processing and spreadsheet), will store all your personal files, will provide all significant processing power, and so on. In this Internet example, the Netstation differs from the standard thin client by virtue of the fact that it does contain a modem, a communications port and communications software, all of which are required for Internet access. See also Client, Client/Server, Client/Server Model, Fat Client, Mainframe Server, Media Server, and Thin Client.

Netview An IBM product for management of heterogeneous networks that integrates the functions of three formerly separate Communications Network Management (CNM) software programs: 1. NCCF. Network Communication Control Facility. 2. NLDM. Network Logical Data Manager, which uses functions from NCCF and helps pinpoint problems along the logical connection/path of an SNA session. 3. NPDA. Network Problem Determination Application, which displays various alerts using IBM equipment located at strategic points in the network and allows diagnostic information to be displayed. Also, NetView incorporates some of the functions from two other programs: VNCA (Virtual Telecommunications Access Method/Node Control Application) which monitors the status and current activity of all resources in a domain, and NMPF (Network Management Productivity Facility) which helps the network operator to install, learn and use many network management products. See also Network and Network Management.

NetWare NetWare is an extremely popular and extremely good operating system for a local area network from Novell, Orem, UT. NetWare is actually its own operating system. This means it is the link between machine hardware (file servers, printers, modems, etc.) and people who want it use that hardware. NetWare is neither DOS, nor OS/2 nor Windows though it can be made to look and act like them. That's part (a small part) of its popularity. See Netware MHS, Netware Workstation Files, NETx.COM and Novell.

NetWare Bindery Centralized authentication database for NetWare 3.xx LANs.

NetWare Directory Services. See NDS.

NetWare Global MHS Novell's implementation of MHS as a NetWare Loadable Module (NLM), providing powerful integration with NetWare services. This supports additional modules to connect to X.400, SNA and SMTP systems.

NetWare Loadable Module NLM. An driver that runs in a server on a local area network under Novell's NetWare operating system and can be loaded or unloaded on the fly as it's needed. In other networks, such applications could require dedicated PCs. A telephony NLM might allow a workstation on a LAN to control a PBX attached to a NetWare file server. It might also allow the workstation to control one or more voice processing cards sitting on in a NetWare server. In early 1993, AT&T became the first PBX maker to ink a deal with Novell, the creator of NetWare, to put telephony onto Novell LANs. AT&T created a PC-card resident in a Novell File server. The card connects to the ASAI (Adjunct Switch Applications Interface) BRI port on the AT&T Definity PBX. Anyone with a PC on the Novell network and an AT&T phone on their desk can use telephone features, such as auto-dialing, conference calling and message management (a new term for integrating voice, fax and e-mail on your desktop PC via your LAN). The Novell/AT&T deal intends to create open Application Programming Interfaces (APIs) that third party developers can work with. A Novell/AT&T example of what could be developed: A user could select names from a directory on his PC. He could tell the Definity PBX through the PC over the LAN to place a conference call to those names. At the same time, a program running under NetWare automatically send an e-mail to the people, alerting them to the conference call and them the agenda. All participants would have access to both the document and conference call simultaneously. See Telephony Services.

NetWare MHS Netware MHS, which is software that provides store-and-forward capability. Fax and E-mail systems that support MHS format their message transmissions according to MHS specifications. MHS reads compatible transmissions, determines the intended recipient and his location, and then sends the message to that location, regardless of the type of LAN or E-mail system at the different ends. See MHS.

NetWare Telephony Services See Telephony Services.

Network Networks are common in our lives. Think about trains and phones. Networks unities things together. Computer networks connect all types of computers and computer-related things — terminals, printers, modems, door entry sensors, temperature monitors, etc. The networks we're most familiar with are long distance ones, like phones and trains. But there are also Local Area Networks (LANs) which exist within a limited geographic area, such as a few hundred feet of a small office, an entire building or even a "campus," such as a university or industrial park. There are also Metropolitan Area Networks (MANs). See also LAN etc.

Network Access Control Electronic circuitry that determines which workstation may transmit next or when a particular workstation may transmit.

Network Access Line NAL. A communications channel between customer premises and the central office.

Network Access Point NAP. A telephone company AIN term. Software in a switch capable of recognizing a call that requires processing by AIN logic which, recognizing such a call, routes the call to an SSP or ASE switch.

Network Accounting A system or application software module that monitors and reports on packet-switched data network traffic, generally focusing on IP (Internet Protocol) traffic. Network accounting software captures data packets as they transverse the network, processes them, and stores them on a centralized data repository to which the network administrator, cost center managers, and privileged others can gain access to run various reports. Much like a call accounting system in the circuit-switched voice domain, a network accounting system captures data output from a switch or router. The SDR (Session Detail Record) is much like CDR (Call Detail Record) output from a voice PBX. Much like PBX-CDR which identifies the originating and terminating extension/telephone number, a network accounting system captures the originating and terminating IP address, and can translate it into the MAC address of the LAN-attached user workstation, and the URL (Uniform Resource Locator) of the Website the user has visited. Much like a call accounting system keeps track of duration and time of day of a voice call, a network accounting system keeps track of duration and time of day of a data network user's session. Network accounting systems are cost effective monitoring systems used by large corporations to ensure that expensive network resources are neither abused or misused. Such resources include both LAN resources (e.g., switches, servers, and routers) and high-speed (e.g., T-1 and T-3) circuits connecting to the Internet. See also Call Accounting, Electronic Communication Privacy Act, and Keystroke Logging.

Network ACD Network ACD allows ACD agent groups, at different locations (nodes), to service calls over the network independent of where the call first enters the network. NACD uses ISDN D-channel messaging to exchange information between nodes.

Network Address Every cardEvery node on an Ethernet network has one or more addresses associated with it, including at least one fixed hardware address such as "2c-16-69-f1" assigned by the device's manufacturer. Most nodes also have protocol specific addresses assigned by a network manager.

Network Address Translation Network Address Translation. A version of Port Address Translation (PAT), NAT enables a local area network (LAN) to use one set of IP addresses for internal traffic and a second set of addresses for external traffic. NAT allows a company to shield internal addresses from the public Internet. According to Cisco, NAT has several applications. You want to connect to the Internet, but not all your hosts have globally unique IP addresses. NAT enables private IP internetworks (i.e., internetworks that use nonregistered IP addresses to connect to the Internet, or another public network. NAT is configured on the router at the border of a stub domain (referred to as the inside network) and a public network such as the Internet (referred to as the outside network). NAT translates the internal local addresses to globally unique IP addresses before sending packets to the outside network. You must change your internal addresses instead of changing them, which can be a considerable amount of work, you can translate them by using NAT. You want to do basic load sharing of TCP traffic. You can map a single

to many local IP addresses by using the TCP load distribution feature. As a connectivity problem, NAT is practical only when relatively few hosts in a communicate outside of the domain at the same time. When this is the case, subset of the IP addresses in the domain must be translated into globally addresses when outside communication is necessary, and these addresses can be no longer in use. A significant advantage of NAT, according to Cisco, is that it figured without requiring changes to hosts or routers other than those few which NAT will be configured. NAT may not be practical if large numbers of hosts domain communicate outside of the domain. Furthermore, some applications embed IP addresses in such a way that it is impractical for a NAT device to trans- applications may not work transparently or at all through a NAT device. NAT the identity of hosts, which may be an advantage or a disadvantage. A router with NAT will have at least one interface to the inside and one to the outside. environment, NAT is configured at the exit router between a stub domain and When a packet is leaving the domain, NAT translates the locally significant address into a globally unique address. When a packet is entering the domain, NAT the globally unique destination address into a local address. If more than one exists, each NAT must have the same translation table. If the software cannot address because it has run out of addresses, it drops the packet and sends an unreachable packet. A router configured with NAT must not advertise the local to the outside. However, routing information that NAT receives from the outside advertised in the stub domain as usual. See also Port Address Translation.

Network Addressable Unit NAU. In IBM's SNA, a logical unit (LU), physical unit (PU) or system services control point (SSCP), which is host-based, that is the origination of information transmitted by the path control portion of an SNA network.

Network Agent A network agent is a device, such as a workstation or a router, programmed to gather network performance information to send to the network management agent. See Network Management Agent.

Network Analyzer A microwave test system that characterizes devices in terms of complex small-signal scattering parameters (S-parameters). Measurements involve measuring the ratio of magnitude and phase of input and output signals at the various ports of the network with the other ports terminated in the specified characteristic impedance (usually 50 ohms).

Network Application Architecture A generalized architecture allowing compatibility at the application level. Examples are Digital Equipment Corp.'s Network Application Support (NAS) and IBM Corp.'s Systems Application Architecture (SAA).

Network Application Support Digital Equipment Corporation's set of open software which allegedly allows its customers to integrate, port and distribute applications between different computer systems, including VMS, UNIX, MS-DOS, OS/2 and Apple Macintosh.

Network Architecture The philosophy and organizational concept for enabling communications between multiple locations and multiple organizational units. Network architecture is a structured statement of the terminal devices, switching elements and the protocols and procedures to be used for the establishment effective telecommunications. See Architecture.

Network Balancing 1. Lumped circuit elements (inductances, capacitances and resistances) connected so as to simulate the impedance of a uniform cable or open-wire circuit over a band of frequencies. 2. Moving circuits around in a multi-node switching network so that the switching loads on each of similar switching modules are roughly equal.

Network Basic Input/Output System NETBIOS. Within the context of the MS-DOS operating system, the software or software and firmware services that implement the interface between applications and a network adaptor, such as a CSMA/CD or token ring adaptor.

Network Board 1. A circuit board installed in each network station to allow stations to communicate with each other and with the file server. 2. An SCSA term. A board device designed to act as an interface between a computer-based signal processing system and a telephone network.

Network Byte Order The Internet standard way of ordering of bytes corresponding to numeric values.

Network Channel Terminating Equipment NCTE. A device or devices at the user's premises used to amplify, match impedance or match network signals to the customer's equipment connected to the network. Basically, network channel terminating equipment is a general name for equipment linking the network to a customer's premises. When NCTE connects to digital circuits, it typically consists of DSUs and CSUs. They are used for balancing of signals and providing for loop-back testing.

Network Computer NC. Larry Ellison of Oracle's idea of a $500 (or so) PC that lacks a hard disk and may lack a monitor but can be used to browse the Internet and run applications on a server on the Internet or corporate intranet. Ellison, who is Oracle's chairman, sees the NC as a "universal digital appliance." The New Yorker of September 8, 1997 discussed the implications of the network computer thus: Microsoft's worries about Ellison and NCs are not trivial. After a prolonged period of being in denial about the rise of the Internet, Gates and his team now understand that it is the central fact of the next phase of computing, and that it poses a real threat to Microsoft's power. In 1995, Sun Microsystems introduced an Internet-centric programming language called Java, which creates programs that can run on any operating system and is fast becoming the standard lingo of the Net. In a Java-fuelled future, the reign of the PC might be challenged by the NC which would let users "borrow" programs from the Net and would have no need for Microsoft's Windows — developments that would create enormous upheaval in many of the software markets that Gate's firm now dominates. See also Internet Terminal, NetPC and NetStation.

Network Computing System NCS. A RPC (Remote Procedure Call) system developed by Apollo, and used in DEC and Hewlett-Packard computer systems. The NCS protocol later was adopted by the Open Software Foundation (OSF). See also OSF.

Network Control Center A physical point within a network where various management and control functions are implemented.

Network Control Program NCP. An IBM Systems Network Architecture (SNA) term. This is the program that switches the virtual circuit connections into place, implements path control, and operates the Synchronous Data Link Control (SDLC) link. The Network Control Program is normally resident in the communications controller or the host processor.

Network Control Signaling The transmission of signals used in the telecommunications system which perform functions such as supervision, address signaling and audible tone signals to control the operation of switching machines in the telecommunication system.

Network Control Signaling Unit A telephone set that controls the transmission of signals into the telephone system which will perform supervision, number identification and control of the switching machines.

Network Controller A powerful microprocessor device designed to perform communications protocol translations between various terminals and computers and an X.25 packet switching network.

Network Data Management Protocol NDMP. An Internet draft specification from the IETF (Internet Engineering Task Force), NDMP is an open protocol for enterprise-wide, network-based data backup. NDMP is a secure backup technique which makes use of the TCP/IP protocol, running on networked file servers.

Network DDE Service A Windows NT definition. The Network DDE (dynamic data exchange) service manages shared DDE conversations. It is used by the Network DDE service.

Network Demarcation Point The network demarcation point is the point of interconnection between the local exchange carrier's facilities and the wiring and equipment at the end user's facilities. The demarcation point is located on the subscriber's side of the telephone company's protector.

Network Design and Optimization Network design and optimization is a process which balances network performance (availability) against cost. There are two fundamental tools in network design and optimization: a traffic usage recorder and software to interpret the results and make recommendations. A traffic usage recorder (TUR) is a device which connects to a network element in order to capture and record traffic statistics. Most network elements (e.g., PBXs, ACDs, data switches and routers) have special ports to which such a device can connect, usually via a RS-232 cable. As traffic flows through the network element, various information about that traffic is sent to the TUR in real time. The TUR holds that raw data in buffer memory until such time as it is polled by a centralized computer and the data is downloaded to that centralized computer. Later, the data is processed and reports are generated by traffic analysis software. That software will help you figure out which circuits you need, what speeds, to where, etc.

Network Design Order NDO. See Telephone Equipment Order.

Network Device Driver Software that coordinates communication between the network adapter card and the computer's hardware and other software, controlling the physical function of the local area network adapter cards.

Network Distribution The physical facilities of a communications carrier which are located outside the carrier's buildings. It includes cables, pole lines, conduit, etc.

Network Diversity A simply concept that says if you have a network which is important you need to have multiple ways of moving information around that network. Some of those ways should be provided by circuits from one vendor and some should be provided by circuits from several vendors. In other words, there should be diversity in both circuits and vendors. As an example of the necessity of taking network diversity very seriously, consider the year 1991. In 1991, nearly 30 million residential and business customers, more than one hundred thousand airline passengers and an alphabet soup of state and federal agencies were temporarily crippled by nine major telephone network outages. Also in a 12-month period, the FAA (Federal Aviation Commission) reported a total of 114 phone service failures that disrupted air traffic facilities around the nation. Between June 10 and July 1, 1991, six major outages in three states comprised the largest series of network outages in history.

Network Drive A disk drive that is available to multiple users and computers on a network. Network drives often store data files for many people in a work group.

Network Economy The network economy is a term economists use when they apply the law of increasing returns to our entire economy. The idea is that one fax machine is worth nothing. Connect a hundred of them together and they're now worth something. Connect one million of them and suddenly you have something truly valuable. By the end of 1998, there were 350 million email addresses worldwide. Bingo, the network economy.

Network Element Control Protocol See NECP.

Network Elements NE. Processor controlled entities of the telecommunications network that primarily provide switching and transport network functions and contain network operations functions. Examples are: Non-AIN switching systems, digital cross-connect systems, AIN switching systems, and Signaling Transfer Points (STPs). In SONET, five basic network elements are: add/drop multiplexer; broadband digital cross-connect; wideband digital cross-connect; digital loop carrier; and switch interface.

Network Entity Identifier NEI. A cellular radio term. An Internet protocol address, or ConnectionLess Networking Protocol (CLNP) address, or any other protocol addressing used by the provider in transmission and receipt of Cellular Digital Packet Data (CDPD) services. The address that identifies that a user is authorized to use the service.

Network Equalizing A device connected to a transmission path in order to alter the characteristics of that path in a specified way. Often used to equalize the frequency response characteristic of a circuit for data transmission.

Network Extension Unit NEU. An AT&T thing which sits in a telephone satellite closet (their words) which links up to 11 Starlan daisy chained clusters of PCs, etc. in a star configuration through standard telephone wiring, modular cords and plugs.

Network Externalities First seen in an article by John Cassidy in The New Yorker Jan 12, 1998 issue. In plain English, "network externalities" means that the value of a product increases along with the number of other people using it. Few people care about how many people are buying corn flakes or pizzas. But it usually applies to high-tech goods for two reasons. They have to be compatible with one and other (how useless is a Betamax videocassette today?). Second, the buyers are often linked to a network. The more people on the network, the more valuable the product becomes. Clearly, one person and one phone has no value. Two people and two phones have more value. 10,000 people and 10,000 phones have even more value, etc.

Network Fax Server An network fax server is typically a PC on LAN. The PC has one or more fax/modem cards. Its job is to send and receive faxes for everyone on the LAN. As a result it is connected to dial out phone lines. See Server.

Network File System NFS. A method developed by Sun Microsystems Inc. for distributing files within a heterogeneous network. NFS has become a de facto standard. It requires special software drivers to suit specific vendor hardware and operating system software. NFS hardware and software gateway products are also available.

Network Harm The reasoning behind the Bell System's insistence that the public could not hook its own equipment to the telephone line for fear it would produce "Irreparable Harm To The Network". A Bell System publication quoted the National Academy of Sciences as identifying four areas of potential harm. They were: Excessive signal power, hazardous voltage, improper network control signaling and line imbalance. Since the Carterfone Decision in the summer of 1968, the FCC has seen fit to allow devices, which pass an FCC Registration program, to be connected to the network, and while the quality of the stuff is occasionally poor, the fears of Network Harm have proven groundless. The fears were, of course, raised to preserve the Bell System's erstwhile almost monopoly on the manufacture of telephone equipment in the U.S.

Network Information Center NIC. Centers providing user assistance, document service, training on the Internet.

Network Information Management System NIMS. Provides customers with browsing capability in their 800 and VNET databases.

Network Interface The point of interconnection between Telephone Company communications facilities and terminal equipment, protective apparatus or wiring at a subscriber's premises. The network interface or demarcation point shall be located on the subscriber's side of the Telephone Company's protector, or the equivalent thereof in cases where a protector is not employed, as provided under the local telephone company's reasonable and nondiscriminatory standard operating practices.

Network Interface Card Also called a NIC card. A printed circuit board comprising electronic circuitry for the purpose of connecting a workstation to a LAN. A NIC usually is in the form of a card that fits into one of the expansion slots inside a PC. Alternatively, it can fit into a slot of a MAU (Multistation Access Unit), which serves multiple LAN-attached devices, such as workstations and printers. In the context of IEEE standards, NICs operate at the MAC (Medium Access Control) layer. In the context of the OSI Reference Model, NICs operate at Layers 1 (Physical Layer) and 2 (Data Link Layer). The basic job of the NIC is to take data from the transmitting workstation, form it into the specific packet format demanded by the LAN protocol you are running (e.g., Ethernet or Token Ring), and present it to the shared medium (usually a cable). On the receiving end, the process is reversed, of course. Hard-coded into the NIC at the time of manufacture is a MAC address, unique in all the world to that NIC card; the MAC address effectively identifies the LAN-attached device with which it is associated. A NIC works with the network software and computer operating system to transmit and receive messages on the network.

Network Interface Controller Same as a Network Interface Card. See above definition.

Network Interface Device 1. NID. A device wired between a telephone provider and the inside wiring to isolate the customer's equipment from the network. 2. A device that performs, functions such as code and protocol conversion, and buffering required for communications to and from a network. 3. A device used primarily within a local area network to allow a number of independent devices, with varying protocols, to communicate with each other. This communication is accomplished by converting each device protocol into a common transmission protocol.

Network Interface Module Electronic circuitry connecting a system (usually a PC) to the telephone network. Network interface modules come in as many varieties as there are ways of connecting to the telephone network. They range from a simple, single start telephone line to complex ISDN PRI circuits. Usually the network interface module slides into one of the expansion slots inside a PC. The card transmits and receives messages from the resource modules and provides access to the telephone network.

Network Interface Unit 1. A Network Interface Card (NIC) or Multistation Access Unit (MAU) that attaches to a LAN. It implements the local network protocol and provides an interface for device attachment. See also NIC and MAU for much more. 2. NIU. A semi-intelligent device that serves as the point of physical and logical demarcation between the LEC (Local Exchange Carrier) and your residential or small business line. The NIU includes a silicon-based protector that trips the circuit in the event of a lightening strike or some other form of aberrant voltage that otherwise would fry your telephone and burn down your house. The NIU also has enough intelligence to permit the carrier to conduct an automated loopback test, which tests the integrity of the copper-based, twisted-pair, local loop from the central office (CO) to your premises - and is also Loopback and NIU.

Network Interworking A Frame Relay/ATM term. Network interworking is a method of connecting two frame relay devices over an ATM backbone network. This approach leaves the entire frame relay frame intact, including header, payload and FCS. The ATM devices act to set up the connection on a cut-through basis through the ATM tunneling protocol. Network interworking is defined in the FRF.5 specification of the Frame Relay Forum, and is recognized by the ATM Forum. Contrast with Service Interworking.

Network Inward Dialing NID. A service feature of an automatic telephone network that allows a calling user to dial directly to an extension station called user facility without operator intervention.

Network Layer Third layer of the OSI model of data communications

01

CDMA Systems Engineering Handbook

Jhong Sam Lee
Leonard E. Miller

Artech House
Boston • London

Library of Congress Cataloging-in-Publication Data
Lee, Jhong S.
CDMA systems engineering handbook /Jhong S. Lee, Leonard E. Miller.
p. cm. — (Artech House mobile communications library)
Includes bibliographical references and index.
ISBN 0-89006-990-5 (alk. paper)
1. Code division multiple access. 2. Mobile communication systems.
I. Miller, Leonard E. II. Title. III. Series.
TK5103.45.L44 1998
621.382—dc21 98-33846
CIP

British Library Cataloguing in Publication Data
Lee, Jhong S.
CDMA systems engineering handbook.— (Artech House mobile communications library)
1. Code division multiple access—Handbooks, manuals, etc.
I. Title II. Miller, Leonard E.
621.3'84'56

ISBN 0-89006-990-5

Cover design by Lynda Fishbourne

International Standard Book Number: 0-89006-990-5
Library of Congress Catalog Card Number: 98-33846

10 9 8 7 6 5 4 3

To our wives, Helen Lee and Fran Miller

register, with a period of $2^{42} - 1 \approx 4.4 \times 10^{12}$ chips (lasting over 41 days at 1.2288 Mcps), that implements the recursion that is specified by the characteristic polynomial

$$f(x) = 1 + x^7 + x^9 + x^{11} + x^{15} + x^{16} + x^{17} + x^{20} + x^{21} + x^{23} + x^{24} + x^{25} + x^{26} + x^{32} + x^{35} + x^{36} + x^{37} + x^{39} + x^{40} + x^{41} + x^{42} \quad (4.4)$$

A phase offset of the original long PN code sequence that is unique to the particular paging channel and base station is obtained by combining the outputs of the shift register stages selected by a 42-bit mask, as illustrated in Figure 4.16. For paging channels, the mask is derived from the paging channel number and and the base station ID in the form of its pilot PN code offset; that is, the paging channel mask (beginning with the most significant bit) is given by

$$1100011001101000 0xxx 000000000000 yyyyyyyyy \quad (4.5)$$

where xxx is the paging channel number and yyyyyyyyy is the base station pilot PN sequence offset index.

The decimated PN sequence is some shift of the original sequence, running at 1/64 times the rate. Because the sampled PN code has the same rate as the encoded data, the data waveform is not spread in bandwidth by combining with this particular PN code. The factors affecting the selection of the long-code masks used on the forward and reverse links are discussed in Chapter 6, as is the effect of decimating a PN sequence.

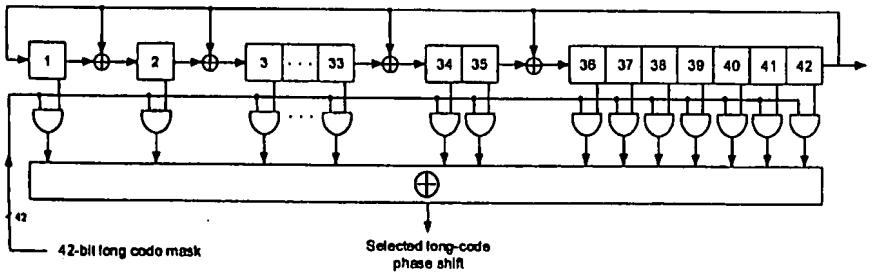


Figure 4.16 42-Stage long-code PN generator with user-distinct mask.

Each paging channel symbol data stream is spread by its designated Walsh sequence, corresponding to one of the first through seventh rows of the Hadamard matrix H_1–H_7, then spread in quadrature by the short PN codes. Paging channel modulation parameters are summarized in Table 4.2.

A paging message directed to a particular mobile unit could be broadcast periodically on one of the paging channels, requiring mobile n to scan the active paging channels for a possible message addressed to mobile n. Or, to relieve the mobile of the burden of scanning the different paging channels, the paging message could be broadcast periodically on all paging channels; however, this procedure is wasteful of paging channel capacity. In IS-95, each active paging channel has a number of periodically recurring message slots (e.g., 2,048 slots) available for transmitting pages and other base-to-mobile messages. When a message is queued up for a particular mobile, using a *hash function*, the base station pseudorandomly selects one of the paging channels and pseudorandomly selects one of the message slots in that paging channel for transmission to the particular mobile. The mobile knows exactly which paging channel and message slot to monitor for possible messages because the pseudorandom selection is based on its own identification number and known system parameters. The purpose of the hash function is to distribute the message traffic evenly among the paging channels and message slots. The IS-95 hash functions are discussed in Appendix 4B.

4.2.3.4 Traffic Channels

There are nominally up to 55 forward *traffic channels* that carry the digital voice or data to the mobile user. It is possible to use one or more of the seven Walsh sequences normally assigned to paging channels for traffic channels, for a total of up to 62 traffic channels, if the system becomes heavily loaded.

Table 4.2 Paging channel modulation parameters

Parameter	Value		Units
Data rate	9600	4800	bps
PN chip rate	1.2288	1.2288	Mcps
Code rate	1/2	1/2	bits/code symbol
Code repetition	1	2	mod symbols/code symbols
Modulation symbol rate	19,200	19,200	symbols/sec (sps)
PN chips/mod symbol	64	64	
PN chips/bit	128	256	

It is usually the case, however, that mutual interference on the reverse link limits the number of simultaneous calls to fewer than 55 calls, so the forward link's capacity is more than adequate for the traffic that can be supported by the system. Walsh sequences designated for use on the traffic channels are H_8–H_{31} and H_{33}–H_{63}. A block diagram for the forward traffic channel modulation is given in Figure 4.17. As shown in the diagram, voice data for the mth user is encoded on a frame-by-frame basis using a variable-rate voice coder, which generates data at 8.6, 4.0, 2.0, or 0.8 kbps depending on voice activity, corresponding respectively to 172, 80, 40, or 16 bits per 20-ms frame. A cyclic redundancy check (CRC) error-detecting code calculation is made at the two highest rates, adding 12 bits per frame for the highest rate and 8 bits per frame at the second highest rate. At the mobile receiver, which voice

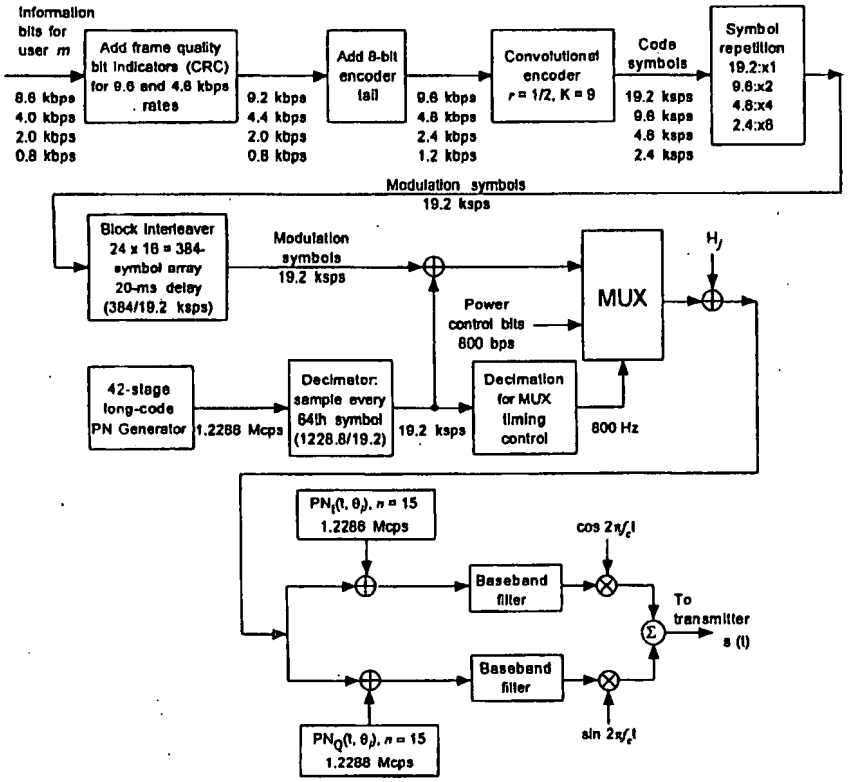


Figure 4.17 Traffic channel modulation.

data rate is being received is determined in part from performing similar CRC calculations, which also provide frame error reception statistics for forward link power control purposes. The theory and use of CRC codes are discussed in Chapter 5.

In anticipation of convolutional coding on a block basis (code symbols in one frame not affecting those in adjacent frames), a convolutional encoder "tail" of 8 bits is added to each block of data to yield blocks of 192, 96, 48, or 24 bits per frame, corresponding to the data rates of 9.6, 4.8, 2.4, and 1.2 kbps going into the encoder, which are defined as full, one-half, one-quarter, and one-eighth rates, respectively. Convolutional encoding is performed using a rate 1/2, constraint length 9 code, resulting in coded symbol rates of 19.2, 9.6, 4.8, and 2.4 ksps.

Coded symbols are repeated as necessary to give a constant number of coded symbols per frame, giving a constant symbol data rate of 19.2 ksps (i.e., 19.2 ksps × 1, 9.6 ksps × 2, 4.8 ksps × 4, 2.4 ksps × 8). The 19.2 ksps × 20 ms = 384 symbols within the same 20-ms frame are interleaved to combat burst errors due to fading, using the same interleaving scheme as on the paging channels.

Each traffic channel's encoded voice or data symbols are scrambled to provide voice privacy by a different phase offset of the long PN code, decimated to yield a code rate of 19.2 kcps. Note that different mobile users are distinguished on the forward link by the orthogonal Walsh sequence associated with the particular traffic channel, not by the user-specific long-code phase offset.

The scrambled data are punctured (overwritten) at an average rate of 800 bps by symbols that are used to control the power of the mobile station; the details of this "power control subchannel" are discussed in Section 4.4.

One of 64 possible Walsh-Hadamard periodic sequences is modulo-2 added to the data stream at 1.2288 Mcps, thus increasing the rate by a factor of 64 chips/modulation symbol (scrambled code symbol). Each symbol for a given traffic channel is represented by the same assigned 64-chip Walsh sequence for a data symbol value of 0 and the sequence's complement for a data symbol value of 1. Walsh-Hadamard sequences of order 64 have the property that all 64 of the sequences are mutually orthogonal. A unique sequence is assigned to each traffic channel so that upon reception at their respective mobile stations, the traffic channels can be distinguished (demultiplexed) based on the orthogonality of the assigned sequences. This orthogonally spread data stream is passed to the quadrature modulator for PN spreading and RF transmission.

$$\begin{array}{cccccccc|cccccccc} 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ r_1 & r_2 & r_3 & r_4 & r_5 & r_6 & r_7 & r_8 & r_9 & r_{10} & r_{11} & r_{12} & r_{13} & r_{14} & r_{15} & r_{16} \end{array}$$

$$\langle (r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8), (r_{16}, r_{15}, r_{14}, r_{13}, r_{12}, r_{11}, r_{10}, r_9) \rangle$$
$$= \langle (0, 0, 0, 1, 1, 0, 1, 0), (1, 0, 1, 0, 0, 1, 0, 1) \rangle = -6$$

Thus, $x_4 = 1$ and the decoded data sequence is now given as $X = (1, 1, 0, 1) = X_{13}$, which is the correct sequence.

The generalized decoding rule based on the fast Walsh transform can be stated as follows: In a set of Walsh sequences of order $N = 2^K$, correlate every other 2^{j-1}-tuple of symbols with the reverse order of the following 2^{j-1}-tuple. Take the sum of the correlation measures. If the sum is positive, the jth ($j = 1, 2, \ldots, K$) symbol of the index sequence is $x_j = 0$; if the sum is negative, then $x_j = 1$; and if the sum is zero, the index symbol is arbitrarily chosen; that is:

$$\sum_{i=0}^{N/2^j-1} \langle (r_{2^j \cdot i+1}, r_{2^j \cdot i+2}, \cdots, r_{2^j \cdot i+2^{j-1}}), (r_{2^j \cdot i+2^j}, r_{2^j \cdot i+2^j-1}, \cdots, r_{2^j \cdot i+2^{j-1}+1}) \rangle$$
$$\begin{cases} > 0 \Rightarrow x_j = 0 \\ = 0 \Rightarrow \text{pick } x_j = 0 \text{ or } 1 \\ < 0 \Rightarrow x_j = 1 \end{cases} \quad (5.42)$$

In summary, we have considered two methods for decoding the Walsh sequences: correlation decoding and fast Walsh transform decoding. Another possible scheme that can be employed is *matched filter decoding*.

5.6 IS-95 Data Frames

In both cellular [1] and PCS [16] CDMA systems, the forward and reverse link signals are transmitted over the channel in frames or packets. The frame stuctures vary, depending upon the channel category and data rate, such as synchronization channel, paging channel, access channel, and traffic channel.

As an example, consider the forward traffic channel frame structures shown in Figure 5.9. The 9,600-bps, 4,800-bps, 2,400-bps, and 1,200-bps data rates for the traffic channel specify different frame structures. These data rates are the input data rates for the convolutional encoder, whose output

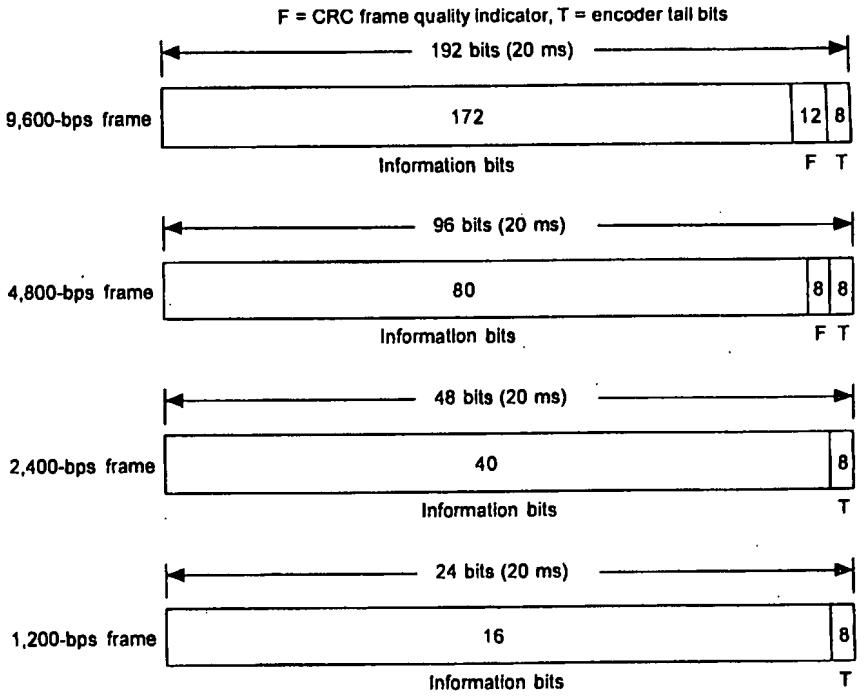


Figure 5.9 Forward traffic channel frame structures (from [1]).

is fed to the 20-ms delay block interleaver after symbol repetition to make the effective symbol rate 19.2 ksps for all four data rates, as shown previously in Figure 4.17.

For the 9,600-bps transmission rate, a total of 192 bits can be transmitted in a 20-ms frame duration. These 192 bits are composed of 172 "information" bits, followed by 12 *frame quality indicator bits* and 8 *encoder tail bits*. It is also observed from Figure 5.9 that the forward traffic frame for the 4,800-bps transmission rate consists of 96 bits that are composed of 80 information bits, 8 frame quality indicator bits, and 8 encoder tail bits. The forward traffic channel frames for the 2,400-bps and 1,200-bps transmission rates contain only information bits (40 and 16 bits, respectively) and 8 encoder tail bits each, without frame quality indicator bits. The frame quality indicator bits are "parity check" bits used in the system's error-detection

scheme, which employs CRC codes [17]. We clarify and explain in detail all these new terms later in this section.

When information bits contained within a block of bits designated as a "frame" or "packet" are transmitted, as depicted in Figure 5.9, it is necessary to determine whether the frame is received in error at the receiving end. To determine the status of "error" or "no error," a scheme is employed wherein frame quality indicator bits are used in an automatic error detection coding technique using cyclic codes. In the following section, we develop the fundamental theory of cyclic codes and proceed to the level of understanding completely the design and operation of the frame quality indicator calculations performed in the cellular CDMA system [1] as well as PCS CDMA systems [16]. In a way, the frames shown in Figure 5.9 can be looked at as codewords in the error detection coding scheme, and thus we begin with some basic concepts of block codes to accomplish our objectives.

5.7 Linear Block Codes

Assume that we have a sequence of binary bits coming out of an information source. We wish to implement a block coding scheme [18–19] for either detecting or correcting transmission errors. In any case, we must first "encode" the information sequence. The block-encoding process involves the following procedure: (1) Collect k successive information bits as a message block; (2) feed the k bits of the message block into the encoder and obtain the coded sequence of n digits, where $n > k$, as diagrammed in Figure 5.10. The result of this procedure is an (n, k) linear code with rate k/n.

Because the message block consists of k information bits, 2^k distinct message blocks are possible, and the encoder output generates 2^k possible

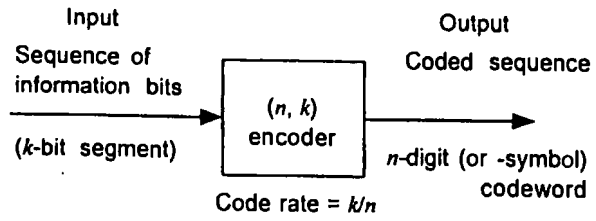


Figure 5.10 Concept of the encoding process.

coded sequences of length n digits, called *codewords*. We say that the block code consists of a set of 2^k codewords. The word "code" connotes an ensemble or a set, whereas the codewords are the elements in the set. Note that the n-digit codeword is an n-tuple, and thus it is a *code vector* in the vector space V_n of all n-tuples. The encoder is a machine (apparatus) or a mathematical rule that transforms the k information bits into an n-digit "coded" sequence.

Now, obviously the block code consisting of 2^k n-tuple codewords is a particular set of n-tuple sequences (vectors) chosen from the set of 2^n possible binary n-tuple vectors. This chosen set, based on a particular encoding rule, is the linear block code, and it is defined as a group:

Definition: A linear block code is a set of 2^k n-tuple vectors that form a subspace of the n-dimensional vector space V_n of all n-tuples.

From the definition of a linear code, the chosen code, being a subspace, must include the all-zero n-tuple, and the dimension of the subspace is k, as we defined these concepts previously in Section 5.3.4. Figure 5.11 depicts the concept of an (n, k) code in the vector space V_n of all n-tuples. The question is, how do we select 2^k n-tuple codewords from the set of all 2^n n-tuple vectors? This is the problem of encoding or designing a coding scheme.

Recall the Walsh sequences that we have defined in the first part of this chapter. The Walsh sequences of order 64 consist of $64 = 2^6$ 64-tuple vectors, and they are the codewords chosen in effect from all the 2^{64} possible 64-tuple sequences. It is in this sense that the Walsh sequences are codewords, and they form a subspace of the 64-dimensional subspace of all 64-tuples—the dimension of the subspace is only six, the number of digits in the index sequence. In fact, this is the orthogonal code that is used in the reverse link in CDMA systems [1, 16]. In Section 5.3.5, we treated the subject of generating Walsh functions using basis vectors. This is exactly the method we use in a block-encoding scheme.

The transformation of the message into a codeword is done in the encoder. The encoder's function for an (n, k) code can be fulfilled by specifying the *generator matrix* G of the code (see eq. (5.26)), which consists of k *linearly independent* n-tuple row vectors. Then all codewords can be generated by a *linear combination* of these row vectors. The coding therefore is to execute the operation of linear combining of the row vectors of the

03

Protection of Data and Delegated Keys in Digital Distribution

Masahiro Mambo[1], Eiji Okamoto[1] and Kouichi Sakurai[2]

[1] School of Information Science, Japan Advanced Institute of Science and Technology
1-1 Asahidai Tatsunokuchi Nomi Ishikawa, 923-12 Japan
Email: {mambo,okamoto}@jaist.ac.jp
[2] Dept. of Computer and Communication Engineering, Kyushu University
Hakozaki Higashi-ku Fukuoka, 812-81 Japan Email: sakurai@csce.kyushu-u.ac.jp

Abstract. A cryptography is quite effective in protecting digital information from unauthorized access. But if a receiver of information is determined after the encryption of the information, e.g. a posted encrypted news is withdrawn by an arbitrary user in open networks, we need an additional mechanism for converting the encrypted information into a form accessible only to an admissible user. Even though such a transformation is done by the consecutive execution of decryption of a ciphertext and re-encryption of a recovered plaintext, an intermediary plaintext may be stolen during the re-encryption. In this paper we examine secure digital distribution systems, information storage system and information provider system, in which encrypted information is directly transformed into a ciphertext of an admissible user. We show that the technique of a proxy cryptosystem is useful for establishing these distribution systems. Proposed protocols can be constructed base on the ElGamal cryptosystem or the RSA cryptosystem. Meanwhile, a blind decryption protocol provides privacy protection with respect to the selection of a ciphertext to be decrypted. In terms of digital distribution it also provides a secure information delivery. An information provider system using a blind decryption protocol possesses a problem such that a decrypting person computes exponentiation for a message freely selected by a requesting person. For such an oracle problem, a solution is known with use of a transformable signature. In this paper we show another measure prohibiting the abuse of the blind decryption protocol.

1 Introduction

One of the greatest advantages of an open network, e.g. Internet, is that people can obtain a large amount of information all over the world. In particular, information created at a local place can be easily read by people living at very far distance. In such a network digitized information is distributed widely and freely. During distribution the digital information passes through several sites, and it is sometimes locally stored in some of these sites. Since the open network is a decentralized insecure computer network, the digital information transmitted over the network and locally stored information should be protected from external threats like eavesdropping and illegal access. In addition to the external

threat, we should be careful about internal threats. For example, information stored in a storage of an organization may be copied by a malicious employee, and leaked out of the organization. It is a very important subject to achieve security of digital information in a distribution system over the open network.

Of course, cryptography is an effective means for protecting digital information from unauthorized access. As long as a cryptosystem used is not broken, the information is not read by anyone other than that knows a secret. Encrypted information can be securely transmitted, and put in any storage with keeping its secrecy. However, the direct use of cryptography is not good enough because a recipient of ciphertext is often not known in advance in the information distribution to the public. An information provider may collect interesting information, and try to sell it. Collected valuable information is encrypted, but since its recipient is not known at the time of encryption, the encrypted information has to be converted when its user is determined. The information provider may decrypt its ciphertext and successively re-encrypt a recovered plaintext for the user. Then there is a threat such that a malicious employee and an external entity try to obtain the intermediary plaintext. Naturally speaking, a creator of digital information wants to hide his information until it arrives at a legitimate end user. Therefore, a secure information distribution mechanism which is appropriate for the open network should be studied.

In this paper we mainly deal with two types of distribution systems. One is an information storage system and the other is an information provider system.

Information storage system: It costs to keep material products in a storage. Likewise, digital information needs disk space, and one needs to spend a certain amount of money for a storage facility. Hence, it is imaginable that information storage business emerges. A small company which lacks of funds for a storage equipment deposits its own data in a storage keeper, and withdraws the data from time to time over a network. In order to prepare for a disaster, e.g. earthquake or fire, even a large company may use this type of service as a backup of huge amount of data. If the amount of data a company needs to keep drastically fluctuates, the company can receive large benefit from the storage service by renting an adequate amount of disk space in each period. The demand for storage business will definitely increase if the communication cost becomes lower than the storage cost. In such a business, the owner of the digital information wants to hide deposited information from the keeper. Moreover, if a company retrieves information on demand of a user as in the travel agency and in the weather forecast company, it needs to show the same information to many users. That means it has to convert the information into a form accessible to admissible users.

The information storage system is expected to have a great effect not only as a business between companies but also inside a company. A company has a storage section, and files of different sections are stored in the section. It is useful both as a back up and as normal storage. Since data is gathered in one section, messages should be protected from steal by dishonest employees.

Information provider system: One of important problems in the open network is how to find useful information from large amount of data in the network. Without doubt, we can show our information to the public more easily than before the open network has been built. But it is still not an easy matter to find necessary information from the public. Therefore, one can conduct a business by collecting and providing information over the network. Superdistribution [Mori90], which is a concept on a software distribution system such as a software company can charge a user for each use of a software, also has to overcome a problem of providing information attractive enough to users. Although an electronic news system or a share-ware program system offers us an opportunity to find useful information, it does not ensure the security of posted information. Hence, we study a secure information provider system by taking an electronic news system, a newspaper system and a software distribution system as examples.

We show that the proxy cryptosystem [MO97] is quite effective in constructing secure distribution systems described above. Concrete distribution protocols can be constructed based on either the RSA cryptosystem [RSA84] or the ElGamal cryptosystem [ElG85].

In the meantime, a blind decryption system [SY96, MSO96] provides privacy protection with respect to the selection of a ciphertext to be decrypted. In terms of digital distribution it also provides a secure information delivery. But a blind decryption based on ElGamal cryptosystem has a problem such that a decrypting person computes exponentiation for any message selected by a requesting person. This type of protocol abuse is generally called an oracle problem. A person participating in a cryptographic protocol is exploited by an enemy, and plays like an oracle to the enemy. The oracle problem in the ElGamal blind decryption is solved in [MSO96] by utilizing the transformability of the ElGamal signature. In this paper we show a new solution to this problem.

After the introduction, related work is shown in Sect.2. Then two types of distribution systems are described in Sect.3. Protocol 1 in Sect.3.2 is an information storage system, and Protocol 2, Protocol 3 and Protocol 4 in Sect.3.3 are information provider systems. Protocol 4 uses a blind decryption protocol with a protective mechanism against the abuse by users. Finally, conclusion is given in Sect.4.

2 Related work

Authorization in open networks has been paid a great attentions for years, and there are much work related to it, e.g. [NS78, VAB91, Neu93]. In their schemes a user or a client computer proves that he or it is a really authorized person or computer. Once a check of user verification is passed, he can receive a service from a server. We can consider our systems are one kind of authorization protocol. Even so, our protocols are more devoted to the authorization of the power to decrypt ciphertext and delegated keys than that to prove an identity of a user.

Information provider systems are shown in [TIY95]. One of their systems, a temporary-type system, offers a user a way to use information in on-line basis, e.g. video on demand. In another system, a permanent-type system, encrypted information is delivered in advance to a user via MD, CD-ROM and so on. In both systems a session key is exchanged before information is decrypted. In our proposed systems the session key exchange is not carried out in each transmission. Instead, encrypted information is converted into a form a receiver can read.

In [OT94] an on-line shopping system protecting privacy is proposed, where three sections, customer section, intermediary section and commodity section, are established in a catalog sales company. Who buys which goods is kept unknown as long as not more than two sections collude. In our distribution systems, we do not need to consider several sections in the same company. The privacy is protected in our schemes in a sense such that a keeper cannot know the content of retrieved information in the information storage system and such that an intermediary company does not know the content of transferred information in one of the information provider systems.

Proxy decryption(Proxy decoding): The proxy decryption is a process in a proxy cryptosystem. The proxy cryptosystem [MO97] is a method by which an original decryptor D can allow a designated proxy decryptor Pxy to decrypt a ciphertext of the original decryptor. A proxy ρ is preliminarily given from the original decryptor to the proxy decryptor through a secure channel, and when the original decryptor wants to delegate the decrypting operation to the proxy decryptor, the original decryptor transforms its ciphertext $C^{(D)}$ into a ciphertext $C^{(Pxy)}$ for the proxy decryptor. Using ρ the proxy decryptor extracts a plaintext m from $C^{(Pxy)}$. This decryption by the proxy decryptor is called proxy decryption. Even the proxy decryptor given ρ cannot compute a secret s_D of the original decryptor. A process offering the same functionality as the proxy cryptosystem can be performed by first decrypting a ciphertext $C^{(D)}$ and then re-encrypting an obtained plaintext under the proxy decryptor's public key. This obvious re-encryption method is not efficient enough, and more efficient methods are shown in [MO97] based on the ElGamal cryptosystem or the RSA cryptosystem.

Blind decryption(Blind decoding): The blind decryption is a process by which a user U possessing a ciphertext $C^{(C)}$ of other user or other organization, say a company C, makes C decrypt $C^{(C)}$ without telling C which ciphertext U tries to decrypt. At the same time, C can keep hiding its secret value s_C from U. In contrast to the blind signature [Cha85] where a digital signature is created for a document unknown to a signer, the decrypting operation is executed for a document unknown to a decryptor. The blind decryption is useful for protecting privacy in software distribution in the following way. A software product is encrypted, and the key is also encrypted by C's public key. Then a set of pairs of an encrypted program and a key is delivered to a user. The user selects a favorite program and recovers it without showing which program he wants to obtain by conducting the blind decryption protocol with C. The blind decryption tech-

nique has already been used in the fair public-key cryptosystem [Mic92] based on the RSA cryptosystem and in [SY96] based on the ElGamal cryptosystem. The scheme shown in [SY96] can be estimated as a refinement of the Massey-Omura cryptosystem described in [Kob87] in the ElGamal cryptosystem. The Massey-Omura cryptosystem is also called Shamir's three-pass message transmission scheme. The scheme in [SY96] ensures secure message transmission to a receiver, either.

3 Proposed Digital Distribution Systems

3.1 General framework

In proposed systems digital information is encrypted by a secret-key cryptosystem and a ciphertext $\mathcal{C}$ is created. We can select any favorite secret-key cryptosystem. A receiver of digital information is informed what kind of secret-key cryptosystem is used, either in advance or in each transmission. The key used in the secret-key cryptosystem is randomly generated, and it is processed by the proxy cryptosystem or the blind decryption system, which is based on a public-key cryptosystem. In order to simplify the description, we do not particularly mention authentication methods of messages or users. At the same time, we do not specify a payment protocol, a receipt protocol and a simultaneous bit exchange protocol throughout the paper. Since these protocols are important by their own, we discuss them in another occasion.

3.2 Information storage system

In this subsection two different frameworks of information storage system are described. In the first framework a company C possessing data uses information storage service conducted by a keeper K. K can be an independent company or a section of C. In response to a request from a user U, C retrieves requested information from K, and sends it back to U after converting it in a form such that only U can read. U is given a proxy p for decryption in advance. The following protocol is constructed under such a framework. This protocol is based on the ElGamal cryptosystem. Denote by v_C and s_C a public key and a secret key of C, respectively. In the ElGamal cryptosystem $v_C = g^{s_C} \bmod p$, where p is a prime, g is a generator of Z_p^* and $s_C \in_R Z_{p-1}^*$. If we prepare a generator of Z_q^* s.t. $q|p-1$, we can construct the similar protocol using q in place of $p-1$ as in [Schn91].

Protocol 1

Step 0. (Preliminary)

Step 0-1. (Storage) C creates $\mathcal{C}$ by encrypting its data under a randomly generated key m, and encrypts m under v_C. Then $((g^r \bmod p, m v_C^r \bmod p), \mathcal{C})$, where $r \in_R Z_{p-1}$, together with a ciphertext number is sent to K. The ciphertext number is used for specifying a corresponding ciphertext among C and K.

Step 0-2. (Proxy delivery) C selects a random number $u \in_R Z^*_{p-1}$, and computes $\rho = us_C \bmod p-1$. ρ is given to U through a secure channel. This step needs to be done only once, for example when U registers himself for C's service. C computes $u^{-1} \bmod p-1$ and keeps the result in a database with the name of the corresponding registered user.

Step 1. (User's request) U asks C to send data he wants to look at. The choice is made after checking a list of stored information.

Step 2. (Retrieval request) C sends K a ciphertext number corresponding to the data he should return to U.

Step 3. (Return from keeper) After receiving the ciphertext number, K returns $((x_1, x_2), C)$, where $x_1 = g^r \bmod p$ and $x_2 = mv_C^r \bmod p$. If C has requested to hide the correspondence of a ciphertext in Step 0-1 and that in this step, K returns $((x_1, x_2), C) = ((g^{r+k_K} \bmod p, mv_C^{r+k_K} \bmod p), C)$ after computing $g^r g^{k_K} \bmod p$ and $(mv_C^r)v_C^{k_K} \bmod p$ for $k_K \in_R Z_{p-1}\backslash\{0\}$.

Step 4. (Transformation and return from company) C looks for $u^{-1} \bmod p-1$ of U in the database. C transforms the received (x_1, x_2) into $(x_1^{(u^{-1} \bmod p-1)} \bmod p, x_2)$ and sends $(y_1, y_2, C) = ((x_1^{(u^{-1} \bmod p-1)} \bmod p, x_2), C)$ to U. If C wants to hide correspondence of a ciphertext in Step 3 and that in this step, C sends $(y_1, y_2) = (((x_1 g^{k_C})^{(u^{-1} \bmod p-1)} \bmod p, x_2 v_C^{k_C} \bmod p), C)$ to U using $k_C \in_R Z_{p-1}\backslash\{0\}$.

Step 5. (Decryption by user) After receiving $((y_1, y_2), C)$, U obtains the key m by computing $y_2/y_1^{\rho} \bmod p \equiv m$. Using m, U decrypts C.

If the database in Step 0-2 becomes very large, C should also put the data stored in its database into K's storage in an encrypted form. In this case, U and C need extra time for retrieving data.

Discussion: Since information K stores is a ciphertext of C, K knows nothing on the plaintext. K does not know who retrieves the information, either. If C wants to check whether K has returned correct information, C should decrypt (x_1, x_2) returned from K in Step 3. As long as a secret of C is not stolen by a malicious employee, information delivered to a user is not read even in the intermediary state because a ciphertext keeps an encrypted form during the distribution.

Instead of returning (y_1, y_2, C) directly to U, C can send this triplet to U via K. In this case K takes full responsibility for the information delivery. K has to do retransmission when U claims message is not delivered.

The proxy delivery in Step 0-2, the ciphertext transformation in Step 4 and the proxy decryption in Step 5 are based on the steps of the proxy cryptosystem. Protocol 1 can also be constructed based on the RSA cryptosystem. Please refer to [MO97] for the proxy cryptosystem based on the RSA cryptosystem.

In the above framework users receiving information do not belong to the company. In contrast, every transactions are conducted inside a company C in the following framework. C organizes a storage service in its keeping section K, and employees of C are users of the information storage service. In this framework

a user U of the storage service encrypts his information by his own public key v_U. Encrypted information is transmitted to K, and stored in it. Upon request from U, K returns encrypted information to U. Inside a company employees may be required to share some of their information with keeping security against external and internal threat. Essentially, we can use the method in Protocol 1 for the secure data transmission. Suppose a user U_1 with a public key v_{U1} wants to share with another user U_2 information stored in K. Step 4 and Step 5 of Protocol 1 are performed by U_1 and U_2, respectively. That is, U_1 with a secret key s_{U1} transforms $(x_1, x_2(= mv_{U1}^r \bmod p))$ into $(x_1^{(u^{-1} \bmod p-1)} \bmod p, x_2)$ and sends $(y_1, y_2, C) = ((x_1^{(u^{-1} \bmod p-1)} \bmod p, x_2), C)$ to U_2 possessing $\rho = us_{U1} \bmod p-1$. U_2 computes $y_2/y_1^{\rho} \bmod p \equiv m$.

U_2 can share her information with U_1 without securely delivering another proxy to U_1. U_2 can use the proxy ρ given by U_1. U_2 calculates $(x_1^{\rho s_{U2}} \bmod p, x_2(= mv_{U2}^r \bmod p))$ and sends $(y_1, y_2, C) = ((x_1^{us_{U1}s_{U2}} \bmod p, x_2), C)$ to U_1. U_1 can obtain m by computing $y_2/y_1^{(u^{-1}s_{U1}^{-1} \bmod p-1)} \bmod p \equiv y_2/y_1^{(\rho^{-1} \bmod p-1)} \bmod p \equiv m$. If such a data delivery frequently occurs, U_1 should keep $\rho^{-1} \bmod p-1$ in his own database.

3.3 Information provider system

In open networks, there should be sites collecting information, then people can relatively easily find necessary information without browsing around the network. We discuss three information provider systems. As the first system, we examine how to bring encryption into a news system mentioned in the introduction. In this system each user can get information from a site he belongs to. In other two systems a company C plays the role of a site collecting information. In the latter two systems a person who releases information to the site of C either allows C to get access to the information after a conversion process, or directly executes a decryption protocol with a user U without allowing C to read the information.

A proposed news system employs the method shown in Protocol 1. There are sites $S_1, S_2, \ldots$ of the news system. A user U_1 posts a news to a site, say S_1, to which she connects. The posted news is distributed to all of relevant sites, and retrieved by a user U_2 from a site, say S_3, to which he connects. Let $s_{Si} \in_R Z_{p-1}^*$ and $v_{Si}(= g^{s_{Si}} \bmod p)$ be secret and public keys of S_i, respectively.

Protocol 2

Step 0. (Preliminary)

Step 0-1. (Proxy delivery between sites) Each site shares a proxy with each sites it connects to. S_i prepares a proxy $\rho_{SiSj}(= u_{Sj}s_{Si} \bmod p-1)$ as described in Protocol 1 and gives it to S_j through a secure channel. S_i securely stores u_{Sj} and $\rho_{SiSj}^{-1} \bmod p-1$, and S_j securely stores ρ_{SiSj} in a database with the name of the corresponding site. Suppose there are connections between S_1 and S_2 and between S_2 and S_3, then S_1 possesses u_2 and $\rho_{S1S2}^{-1} \bmod p-1$, S_2 possesses ρ_{S1S2}, u_3 and $\rho_{S2S3}^{-1} \bmod p-1$, and S_3 possesses ρ_{S2S3} in their databases.

Step 0-2. (Registration of users) When a user U_k wants to participate in the news system, she obtains a proxy from one of sites, say S_t. S_t prepares $\rho_{StUk}(= u_{Uk}s_{St} \bmod p-1)$ as described in Protocol 1 and gives it to U_k through a secure channel. S_t and U_t securely store u_{Uk} and ρ_{StUk} in a database with the name of the corresponding site or user, respectively. Suppose U_1 and U_2 connects with S_1 and S_3, respectively. Then U_1 and U_2 store ρ_{S1U1} and ρ_{S3U2} in the database, respectively. S_1 and S_3 store u_{U1} and u_{U2} in the database, respectively.

Step 1. (Posting) $U_1 \longrightarrow S_1$: U_1 creates C by encrypting a part or whole of her article under a randomly generated key m, and encrypts m under a public key v_{S1} of S_1 to which she connects. Then $((x_1, x_2), C) = ((g^r \bmod p, mv_{S1}^r \bmod p), C)$, where $r \in_R Z_{p-1}$, is delivered to S_1.

Step 2. (S_1 distributes the posted information by converting it to a ciphertext for its neighbor sites. Other sites also execute conversion successively.

Step 2-1. $S_1 \longrightarrow S_2$: S_1 computes $y_1 = x_1^{(u_{S_2}^{-1} \bmod p-1)} \bmod p$, and sends $((y_1, y_2(= x_2)), C)$ to S_2.

Step 2-2. $S_2 \longrightarrow S_3$: S_2 computes $\alpha_1 = y_1^{(\rho_{S1S2}\rho_{S2S3}^{-1} \bmod p-1)} \bmod p \equiv g^{(ru_{S_2}^{-1}u_{S_2}s_{S_1}\rho_{S2S3}^{-1} \bmod p-1)} \bmod p \equiv g^{(rs_{S_1}\rho_{S2S3}^{-1} \bmod p-1)} \bmod p$, and sends $((\alpha_1, \alpha_2(= y_2)), C)$ to S_3.

Step 3. (Retrieval) $S_3 \longrightarrow U_2$: When a user wants to read encrypted posted information, he sends a request to a site he connects to. Upon request from U_2, S_3 converts a ciphertext $((\alpha_1, \alpha_2, C)$ into $((\beta_1(= \alpha_1^{(\rho_{S2S3}u_{U_2}^{-1} \bmod p-1)}), \beta_2(= \alpha_2)), C)$. Created $((\beta_1, \beta_2), C)$ is delivered to U_2.

Step 4. (Decryption by user) U_2 obtains the key m by computing $\beta_2/\beta_1^{\rho_{S3U2}} \bmod p \equiv m$. Using m, U_2 can decrypt C.

In order not to expose the plaintext, exponents in Step 2-2 and Step 3 must be computed at first. In Step 0-1, S_1 and S_2 have created proxies. In place of these sites it is possible its partners S_2 and S_3 create proxies. Basically, sites having many neighbor sites should create a proxy and give the same proxy to its neighbor sites. Then these sites do not need to perform conversion many times for the same ciphertext.

In the following protocol a free reporter R writes articles and asks a newspaper company C to buy them. These articles are basically encrypted, and only their headline can be read. C buys the article from R when there is at least one request for purchase, or there are enough amount of requests for purchase. Once C buys the article, C can sell it to subscribers who have requested to read them. Denote by (v_i, s_i) a pair of public and secret keys of $i \in \{C, R, U\}$. $v_i = g^{s_i} \bmod p$, where $s_i \in_R Z_{p-1}^*$.

Protocol 3

Step 0. (Preliminary)

Step 0-1. (Collection) R creates a program and partially encrypts it by a secret-key cryptosystem under a randomly generated key m. A computed ciphertext is C. R also encrypts the key m using v_C. $(x_1, x_2) = (g^r \bmod p, m v_C^{s_R r} \bmod p)$, where $r \in_R Z_{p-1}$. Then $((x_1, x_2), C)$ is sent to C.

Step 0-2. (Proxy delivery) C selects a random number $u \in_R Z_{p-1}$ and gives u to U as a proxy $\rho(= u)$ through a secure channel. This step needs to be done only once, for example when U becomes a subscriber of C's newspaper. C computes $u - s_C \bmod p - 1$ and keeps the result in a database with the name of the corresponding registered user.

Step 0-3. (Partial information retrieval) U sees a headline of news of C and selects a program he wants to read.

Step 1. (User's request) U requests an article he wants to read.

Step 2. (Request for blind decryption) C sends x_1 of the requested article to R.

Step 3. (Return from programmer) After receiving x_1, R computes $y_1 = x_1^{s_R} \bmod p (\equiv g^{s_R r} \bmod p)$, and returns y_1 to C.

Step 4. (Transformation) C receives y_1. $(\alpha_1, \alpha_2) = (y_1, x_2) = (g^{s_R r} \bmod p, m v_C^{s_R r} \bmod p)$ is a ciphertext of C. C can decrypt it if C wants. If C needs to hide the correspondence of the blind decryption and the article bought from R, R gives $((g^{s_R(r+k_C)} \bmod p, m v_C^{s_R(r+k_C)} \bmod p), C)$ to U after computing $\alpha_1 v_R^{k_C} \bmod p$ and $\alpha_2 v_R^{s_C k_C} \bmod p$ for $k_C \in_R Z_{p-1} \backslash \{0\}$.

C calculates $\beta_2 = \alpha_2 \alpha_1^{(u - s_C)} \bmod p \equiv m v_R^{ru} \bmod p$, and $((\beta_1, \beta_2), C) = ((\alpha_1, \beta_2), C) = ((v_R)^r \bmod p, m v_R^{ru} \bmod p), C)$ is delivered to U.

Step 5. (Decryption by user) U obtains the key m by computing $\beta_2 / \beta_1^{\rho} \bmod p \equiv m$. Using m, U can decrypt C.

Discussion: In this protocol C can compute m. But it is difficult for a third party to extract it from communicated messages. R does not fail to get money for his article because C cannot decrypt articles without executing Step 2 and Step 3. These steps can incorporate the blind decryption technique if C wants to hide its choice of article. Meanwhile, as easily observed, information provider system can be executed together with the information storage system.

In the next protocol we exemplify the information provider system by a program distribution. Distributed programs are created by programmers P, and encrypted. A user U registered to C chooses a program from a program list. When U decides to buy a program, U executes a blind decryption protocol [SY96, MSO96] with P. Then he has permission for the use of the program. In this protocol C charges P for the use of its web site, and does not participate in the blind decryption protocol.

As studied in [MSO96] the blind decryption based on ElGamal cryptosystem has a problem that a decrypting person P computes $\sigma^{s_P} \bmod p$ for any message σ selected by a requesting person. Protocol 3 also has such an oracle problem, where a decrypting person is R. This problem is solved in [MSO96] by utilizing the transformability of the ElGamal signature. In this paper we show another solution to the oracle problem. In the following protocol P with a public and secret key pair $(v_P (= g^{s_P} \bmod p), s_P \in Z^*_{p-1})$ prepares a random number t_P such that $t_P \in_R Z^*_{p-1}$ and $t_P \neq s_P$. Then P computes $T_P = g^{t_P} \bmod p$. T_P and t_P are public and secret keys of P additional to v_P and s_P. $h(\cdot)$ is a cryptographically secure hash function.

Protocol 4

Step 0. (Preliminary)

Step 0-1. (Collection) P creates a program and partially encrypts it by a secret-key cryptosystem under a randomly generated key m. A computed ciphertext is C. P also encrypts the key m using v_P and T_P. $(x_1, x_2, x_3) = (g^r \bmod p, m v_P^r \bmod p, m T_P^r \bmod p)$, where $r \in_R Z_{p-1}$. P computes the following signature s. After selecting $w \in_R Z_{p-1}$, $h(g^w \bmod p, (v_P/T_P)^w \bmod p) = e$ is generated. s is determined by $s = w - er \bmod p - 1$. $((x_1, x_2, x_3, x_4, x_5), C)$ is sent to C, where $x_4 = s$ and $x_5 = e$.

Step 0-2. (Retrieval of encrypted information) U watches a list of programs exhibited by C and selects a program he needs. U downloads $((x_1, x_2, x_3, x_4, x_5), C)$ from C. He checks the verification equation $x_5 = h(g^{x_4} x_1^{x_5} \bmod p, (v_P/T_P)^{x_4} (x_2/x_3)^{x_5} \bmod p)$. If it is not satisfied, he requests a valid triplet to C. If satisfied, he proceeds to Step 1.

Step 1. (Request for blind decryption) U chooses $a \in_R Z^*_{p-1}$, and computes $(y_1, y_6) = (x_1^{(a^{-1} \bmod p-1)} \bmod p, (x_2/x_3)^{(a^{-1} \bmod p-1)} \bmod p)$ $\equiv (g^{(ra^{-1} \bmod p-1)} \bmod p, g^{(ra^{-1}(s_P - t_P) \bmod p-1)} \bmod p)$. (y_1, y_6) is sent to P. a is kept secret by U.

Step 2. (Return from programmer) P first checks whether a congruence $y_1^{(s_P - t_P)} \equiv y_6 \bmod p$ satisfies or not. If the check fails, P does not respond. Otherwise, P computes $z_1 = y_1^{s_P} \bmod p$, and returns z_1 to C.

Step 3. (Decryption by user) After receiving z_1, U computes $\alpha_1 = z_1^a \bmod p$. Then he has $(\alpha_1, \alpha_2) = (\alpha_1, x_2) (\equiv (v_P^r \bmod p, m v_P^r \bmod p))$. U obtains the key m by calculating $\alpha_2 \alpha_1^{-1} \bmod p \equiv m$. Using m, U decrypts C.

Discussion: Protocol 4 provides a secure message delivery to a receiver. The receiver can be determined after encryption. As stated in Sect.2, the protocol described above is closely related to the Shamir's three-pass message transmission scheme.

Because of the blind decryption protocol, U can hide which program he is going to buy. Nonetheless, P does not fail to get money for his software product by charging each execution of the blind decryption protocol.

As in information storage systems, communicated messages are kept in an encrypted form. Thus it is similarly difficult for C and a third party to extract a message from the communicated messages.

Meanwhile, it is considered to be difficult to create a pair (y_1, y_6) satisfying the verification congruence in Step 2 for a selected y_1 without knowing the secrets s_P and t_P. Therefore, P is unlikely to compute exponentiation for a message selected by U. In Protocol 4 two public keys v_P and T_P are used for detecting the abuse of the blind decryption protocol. A similar technique is used in the modified ElGamal cryptosystem shown in [Dam91].

P may try to relate a requested pair (y_1, y_6) with ciphertexts he has placed on the site of C by choosing his secret t_P as an output of a function f taking a message m as input. By checking $y_1^{(s_P - f(m))} \equiv y_6 \bmod p$ for a message m, he can find whether the requested pair corresponds with the message m. In this sense, privacy is slightly violated. However, if t_P has only one preimage, he cannot checks more than two messages because T_P compatible with P's secret t_P is made public. In case t_P has many preimages m's, he can check the correspondence to many messages. But he cannot know the precise correspondence among many messages. From this observation we believe the proposed method offers an acceptable level of privacy.

We can apply the countermeasure in Protocol 4 to Protocol 3. In this case, Protocol 3 are modified as follows. In Step 0-1, $(x_1, x_2, x_3, x_4, x_5) = (g^r \bmod p, mv_C^{s_R r} \bmod p, mv_C^{tr} \bmod p, w - er \bmod p - 1, e)$ is delivered to C. C verifies it by checking $x_5 = h(g^{x_4} x_1^{x_5} \bmod p, (v_R/T_R)^{x_C x_4} (x_2/x_3)^{x_5} \bmod p)$. In Step 2, C sends $(\widetilde{x_1}, \widetilde{x_2}) = (x_1^{v_C(a^{-1} \bmod p-1)} \bmod p, (x_2/x_3)^{(a^{-1} \bmod p-1)} \bmod p)$ to R. Verification of $(\widetilde{x_1}, \widetilde{x_2})$ is executed in Step 3 as in Step 2 of Protocol 4. Only when verification is passed, R proceeds to the next step.

4 Conclusion

In this paper we have studied several information distribution systems, information storage system and information provider system, which keep the secrecy of transmitted data and delegated keys in open networks. These distribution systems are based on either the proxy cryptosystem, the blind decryption system or both of them. The presented information provider systems can be combined with the information storage system explained in Protocol 1. Because digital information preserves an encrypted form in an intermediary organization in the proxy cryptosystem and communicated messages are independent of a ciphertext to be decrypted in the blind decryption system, digital information is not read until it arrives at an end user as long as secret keys are not compromised.

We have also shown a new method to prevent the abuse of the blind decryption protocol. By this new method users cannot obtain an exponentiated value for a message they have selected. Privacy of users participating in this protocol is preserved at an acceptable level.

References

[Dam91] I. Damgård: "Towards Practical Public Key Systems Secure against Chosen Ciphertext Attacks," Lecture Notes in Computer Science 576, Advances in Cryptology -Crypto '91, Springer-Verlag, pp.445-456 (1992).

[Cha85] D. Chaum: "Security without Identification: Transaction System to make Big Brother Obsolete," Communications of the ACM, Vol.28, No.10, pp.1030-1044 (Oct. 1985).

[ElG85] T. ElGamal: "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithm," IEEE Trans. on Information Theory, Vol.IT-31, No.4, pp.469-472 (Jul. 1985).

[Kob87] N. Koblitz: A Course in Number Theory and Cryptography, GTM 114, Springer-Verlag (1987).

[MO97] M. Mambo and E. Okamoto: "Proxy Cryptosystems: Delegation of the Power to Decrypt Ciphertexts," IEICE Transactions on Fundamentals, Vol.E80-A, No.1, pp.54-63 (Jan. 1997).

[MSO96] M. Mambo, K. Sakurai and E. Okamoto: "How to Utilize the Transformability of Digital Signatures for Solving the Oracle Problem," Lecture Notes in Computer Science 1163, Advances in Cryptology -Asiacrypt '96, Springer-Verlag, pp.322-333 (1996).

[Mic92] S. Micali: "Fair Public-Key Cryptosystems," Lecture Notes in Computer Science 740, Advances in Cryptology -Crypto '92, Springer-Verlag, pp.113-138 (1993).

[Mori90] R. Mori: "Superdistribution: The Concept and the Architecture," The Proc. of The 1990 Symposium on Cryptography and Information Security, SCIS90-6A (Jan. 1990).

[NS78] R. M. Needham and M. D. Schroeder: "Using Encryption for Authentication in Large Networks of Computers," Communications of the ACM, Vol.21, No.12, pp.993-999 (Dec. 1978).

[Neu93] B. C. Neuman: "Proxy-Based Authorization and Accounting for Distributed Systems," Proc. of the 13th International Conference on Distributed Computing Systems, pp.283-291 (May 1993).

[OT94] M. Ohmori and M. Tatebayashi: "An On-line Shopping System Protecting User's Privacy," IEICE Technical Report Vol.94, IT94-66, ISEC94-26, pp.25-32 (1995). [in Japanese]

[RSA84] R. L. Rivest, A. Shamir and L. Adleman: "A Method for Obtaining Digital Signatures and Public-key Cryptosystems," Communications of the ACM, Vol.21, No.2, pp.120-126 (1978).

[Schn91] C. P. Schnorr: "Efficient Signature Generation by Smart Cards," Journal of Cryptology, Vol.4, No.3, pp.161-174 (1991).

[TIY95] Y. Takashima, S. Ishii and K. Yamanaka: "An Intellectual Property Protection System Using a PCMCIA Card," Proc. of The 1995 Symposium on Cryptography and Information Security, SCIS95-B5.5 (Jan. 1995). [in Japanese]

[VAB91] V. Varadharajan, P. Allen and S. Black: "An Analysis of the Proxy Problem in Distributed Systems," Proc. 1991 IEEE Computer Society Symposium on Research in Security and Privacy, pp.255-275 (May 1991).

[SY96] K. Sakurai and Y. Yamane: "Blind Decoding, Blind Undeniable Signatures, and their Applications to Privacy Protection," Lecture Notes in Computer Science 1174, Information Hiding, Springer-Verlag, pp.257-264 (1996).

IEICE TRANS. FUNDAMENTALS, VOL. E80-A, NO. 1 JANUARY 1997

PAPER Special Section on Cryptography and Information Security

Proxy Cryptosystems: Delegation of the Power to Decrypt Ciphertexts

Masahiro MAMBO† *and* Eiji OKAMOTO†, *Members*

SUMMARY In this paper a new type of public-key cryptosystem, proxy cryptosystem, is studied. The proxy cryptosystem allows an original decryptor to transform its ciphertext to a ciphertext for a designated decryptor, proxy decryptor. Once the ciphertext transformation is executed, the proxy decryptor can compute a plaintext in place of the original decryptor. Such a cryptosystem is very useful when an entity has to deal with large amount of decrypting operation. The entity can actually speed-up the decrypting operation by authorizing multiple proxy decryptors. Concrete proxy cryptosystems are constructed for the ElGamal cryptosystem and the RSA cryptosystem. A straightforward construction of the proxy cryptosystem is given as follows. The original decryptor decrypts its ciphertext and re-encrypts an obtained plaintext under a designated proxy decryptor's public key. Then the designated proxy decryptor can read the plaintext. Our constructions are more efficient than such consecutive execution of decryption and re-encryption. Especially, the computational work done by the original decryptor is reduced in the proxy cryptosystems.

key words: proxy cryptosystem, proxy, proxy decryptor, ciphertext transformation

1. Introduction

Digitized information in computer networks is copied very easily. If information is encrypted, one cannot obtain the content from the copied message. Additionally, if an access control mechanism is employed with the use of user authentication protocol, an inadmissible entity cannot even make an access to digital information. Cryptography related techniques, typically cryptography itself in the former application, provides us an effective way to limit users who makes an access to digital information. One of cryptographies, public-key cryptography, which has been intensively studied after the advent of [4], is useful in an open network for transmitting information only to a specified person. In the public-key cryptography a key v is made public while a compatible key s is kept secret by its owner. This is why it is suitable for the open network. It is hard in a computational complexity sense to determine a secret s even when an attacker knows a public value v. In the similar context, if there exits a pair (s, ρ), where it is hard to compute s even with the knowledge of ρ, and a possessor of ρ can decrypt a ciphertext which is originally encrypted under v, then such a pair can be used for authorizing a user for getting an access to the encrypted digital information. Such delegation is required in the following occasions.

Suppose an organization, e.g. research company or government, plans to conduct a survey, and one section is assigned to this job. From the privacy reason, questionnaires returned from people are encrypted under a public key v of the organization, or if the questionnaire is very long, it should be encrypted by a secret-key cryptography and a key used in the secret-key cryptography is encrypted by a public-key cryptography. A president of the organization wants members of a section to decrypt the questionnaires and to analyze the survey. But he has no intention to show a secret s of the organization to them. Because it breaches the security assumption of the public-key cryptography. Moreover, the president should limit the access only to the members of the section in order to keep the user's privacy. The president may decrypt the ciphertexts and simply transmit the decrypted questionnaires to the members. Then the privacy could be violated. So, the president should encrypt the decrypted questionnaires under a public key of the members before the transmission. However, an attacker may try to see decrypted intermediate plaintexts. Such a threat is not totally overcome in this re-encryption approach. On top of that, the processes in this approach is a bit cumbersome, and a more direct and efficient method to allow a new access should be studied.

Other example is file access. Suppose a user has a file in a publicly reachable directory or in a directory which is private but possibly illegally accessed by others. In order to keep the secrecy of the file she encrypts it under her public key, or as mentioned above, both the secret-key cryptography and the public-key cryptography are used to encrypt the file. Then the access to the file is limited to her. In some occasion, the user wants to temporarily permit other user to access to the file. She first transforms the encrypted file to the form such that only a new user can read it, and after some time, the original owner revokes the access permission. Like in the above example, such a transformation is achieved by the combination of decryption and re-encryption, but we should avoid such a transformation from the following reason. A file handling system or an editor program often offers us an automatic logging system. Under such

Manuscript received March 25, 1996.
Manuscript revised July 15, 1996.

†The authors are with the School of Information Science, Japan Advanced Institute of Science and Technology, Ishikawa-ken, 923-12 Japan.

circumstances, even if we remove a decrypted plaintext, we may forget to remove its backup file. That indicates we are not totally sure whether an attacker obtains no information at all as long as we recover a plaintext from a ciphertext in a transit point. Furthermore, it is preferable that the transformation be as simple as possible.

In this paper a cryptosystem called *proxy cryptosystem*, which is well suited to the situations above, is studied. The proxy cryptosystem allows an original decryptor to delegate its decrypting operation to a designated decryptor, *proxy decryptor*. Concrete proxy cryptosystems are presented for the ElGamal cryptosystem [5] and the RSA cryptosystem [14]. In the proposed scheme the computational work is less than that in consecutive execution of decryption and re-encryption.

This paper is organized in the following way. After this introduction, related work is explained in Sect. 2. Then following the explanation on conditions of proxy cryptosystems in Sect. 3, three concrete proxy cryptosystems are proposed in Sect. 4. In Sect. 5 the efficiency of the proposed scheme is discussed, and two kinds of revocation methods are described. In addition to that, how to deal with multiple proxy decryptors is studied. Finally conclusions are given in Sect. 6.

2. Related Work

A privacy homomorphism introduced in [13] as cited in [1] is an encryption function such that the operation of its outputs for several unknown input plaintexts results in indirect operation of these plaintexts. With the use of this encryption function, a function of plaintexts is evaluated without the knowledge of the plaintexts and a decryption function corresponding to the encryption function, and an encrypted value of the output of the function is obtained. The privacy homomorphism is useful for securely evaluating a function while hiding the plaintexts. In our situation the function evaluation is not performed, and an authorized person can obtain the plaintext as the original user can.

By a directly transformed link encryption proposed in [9], one connected to a node can securely send a message to a person connected to other node through a computer network equipping a secure data-link layer. The computer network is composed of intermediate nodes and terminals to which a user has access. The user at a terminal encrypts a message under a key of a node to which the terminal connects. The ciphertext is directly transformed in each node into other form of ciphertext encrypted under a key of the next node. A trusted center generates all original keys of nodes and terminals, and computes one key for each node from both the original key of the node and that of the next node. Each node is given only this derived key, and the original key of the node is kept secret by the center. Without transforming the encrypted data into a plaintext, direct transformation is performed in this link encryption by utilizing the derived key. Diminishing the opportunity to expose the plaintext in intermediate nodes makes the data transmission very secure. Nevertheless, this method does not deal with a situation like ours, where a user diverts his ciphertext to the other user. Moreover, all communications between users require nodes and terminals performing transformation. Since keys for the transformation used in intermediate nodes are unknown to users, a receiver of a ciphertext cannot compute a diverted ciphertext a communication partner can directly read. Instead, the receiver may compute a ciphertext which would be transferred through the network, and whose plaintext would be extracted at a communication partner side. To make this possible, the receiver of a ciphertext has to either decrypt the ciphertext and re-encrypt an obtained plaintext for the terminal, or compute a secret for ciphertext transformation from the user to the terminal and behave like a node in the network with the use of the computed secret. The latter measure is closely related to our scheme, but it has not been clearly discussed in [9]. Additionally, although the idea of avoiding to recover a plaintext in transit and the transformation performed in a node are quite similar to ours, a node cannot read a message in [9], while an original decryptor can do so in our method.

Verifiable implicit asking, e.g. in [8], or server-aided secret computation, e.g. in [17], has something to do with our topic. It has a very interesting and practical framework where a relatively powerless device executes with the assistance of powerful auxiliary device(s) a polynomial time computation which exceeds the power of the device. For example, a smart card communicates with a computational center, and it executes a large amount of computation with the aid of it. In this type of computation the powerless device converts its own secret into other values by using random numbers. The powerful device receives the converted values and performs computation for assistance. After receiving the results of the computation done by the powerful device, the powerless device does the final computation and obtains a final result. The assistant powerful device cannot compute the final result by itself because it does not know the random numbers used in conversion. If it could, it would derive the secret of the powerless device. As pointed out in [16] the powerful device sometimes finds the final result after the protocol execution. For example, a digital signature computed by server-aided computation will be known to the powerful device. Even in this case, it is not the powerful device but the powerless one that computes the final result. In the proxy cryptosystem the final result is computed by the proxy decryptor or by both the original and proxy decryptors. Hence methods for verifiable implicit asking are not appropriate enough for our situation.

Proxy signatures proposed in [7] is a tool to delegate signing operation to a designated person, proxy signer. A proxy signature for partial delegation is spe-

cially important since this type of proxy signature is more efficient than signing twice, once by an original signer and once by a proxy signer. In this signature a proxy signer cannot compute a secret of an original signer from a given proxy. Similarly a secret of an original decryptor is not computed from a given proxy in the proxy cryptosystem. The signing operation is delegated in the proxy signature while the decrypting operation is delegated in the proxy cryptosystem. The proxy cryptosystem can be combined with the proxy signature. Email is an example of such an application as described in [7].

3. Conditions of Proxy Cryptosystem

In proxy cryptosystem an original decryptor asks a proxy decryptor to carry out decryption. A ciphertext is created either by the original decryptor or by an encryptor other than the original decryptor. In the latter case, the encryptor sends the created ciphertext to the original decryptor.

In the proxy signatures [7] an original signer can determine the identity of the proxy signer who has created a given proxy signature. Even when the proxy signature is created by a person who is passed the proxy from an authorized proxy signer, the original signer considers that it has been created by the originally authorized proxy signer. Proxy signatures leave an evidence of signing operation. In this sense the proxy signer bears full responsibility on signatures created from his proxy. Unlike the proxy signatures, an original decryptor has no way to detect an illegal access to a plaintext. A decryptor to which the original decryptor has never released permission but is given a proxy can decrypt a ciphertext without endangering itself. Great care should be taken for selecting faithful proxy decryptors.

A normal cryptosystem is called a proxy cryptosystem if the following conditions are satisfied. Let $C^{(U)}$ be a ciphertext for a user U.

Conditions of proxy cryptosystem:

(i) **(Transformation)** Given a ciphertext $C^{(D)}$ for an original decryptor D, only the original decryptor or only both the original decryptor and a creator of $C^{(D)}$ can transform $C^{(D)}$ into a ciphertext $C^{(P)}$ for a proxy decryptor P.

(ii) **(Authorization)** Given a ciphertext $C^{(P)}$ of a plaintext m, m is computed either from a proxy ρ, or from information computed from ρ in polynomial time. Without this information, m cannot be polynomially extracted from $C^{(P)}$.

Although no detecting method for illegal release of a proxy has been found until now, the original decryptor can have from the condition (i) control over the access to the plaintext by the proxy decryptor or possibly by others who are given ρ or information derived from ρ. The condition (ii) ensures an original decryptor that a decryptor authorized with a proxy can decrypt a transformed ciphertext.

As mentioned in the introduction, $C^{(D)}$ can be transformed into $C^{(P)}$ by first decrypting a ciphertext and then re-encrypting an obtained plaintext under the proxy decryptor's public key. This re-encryption method satisfies the conditions of the proxy cryptosystem. Obviously the re-encryption method is not efficient, and a more efficient method should be constructed.

4. Proposed Proxy Cryptosystems

Two proxy cryptosystems for the ElGamal cryptosystem and one for the RSA cryptosystem are shown in this section. In Sects. 4.1 and 4.2, two proxy cryptosystems are constructed for the ElGamal cryptosystem. It is not impossible to construct many different forms of proxy cryptosystems for one normal cryptosystem.

4.1 Proxy Cryptosystem for ElGamal Cryptosystem

Let v and s be a public key and a private key of an original decryptor, respectively, and $v \equiv g^s \bmod p$, where $s \in_R Z_{p-1}\backslash\{0\}$. T is an additional public key of the original decryptor. $t \in_R Z_{p-1}\backslash\{0\}$ is a secret key of the original decryptor, satisfying $T \equiv g^t \bmod p$. p is a prime number whose length is taken greater than 512 bits. g is a generator for Z_p^*.

[Protocol 1]

Step 1. (Proxy generation) An original decryptor computes $\rho = s - tT \bmod p-1$, where t is randomly generated from $Z_{p-1}\backslash\{0\}$.

Step 2. (Proxy delivery) The original decryptor gives ρ to a proxy decryptor in a secure way.

Step 3. (Proxy verification) The proxy decryptor checks a congruence such that

$$v \equiv g^{\rho} T^{T} \bmod p. \tag{1}$$

If (ρ, T) passes this congruence, the proxy decryptor accepts it as a valid proxy. Otherwise, it rejects it and requests the original decryptor a valid one, or it stops this protocol.

Step 4. (Encryption) A document m is encrypted into (x, y), where $r \in_R Z_{p-1}\backslash\{0\}$, $x = g^r \bmod p$ and $y = mv^r \bmod p$.

Step 5. (Ciphertext transformation) The original decryptor transforms the (x, y) into (w, x, y), where $w = x^t (\equiv T^r) \bmod p$.

Step 6. (Decryption by proxy decryptor) The proxy decryptor computes

$$\begin{aligned} y/(x^{\rho}w^{T}) &\equiv (mv^{r})/(g^{r})^{\rho}(T^{r})^{T} \bmod p \\ &\equiv m(v/(g^{\rho}T^{T}))^{r} \bmod p \\ &\qquad \text{(from the congruence (1))} \\ &\equiv m \bmod p. \end{aligned}$$

In the application for questionnaire described in the introduction the amount of computational work of the president is reduced by the following approach.

Step 1'. (Proxy generation, delivery and verification) The original proxy decryptor, the president of a research company, generates ρ as described above, and gives the same ρ to multiple proxy decryptors, members of a section assigned to the survey, in a secure way. Each proxy decryptor checks the validity of given ρ.

Step 2'. (Encryption) A document m is encrypted into (w, x, y), where $r \in_R Z_{p-1}\backslash\{0\}$, $w = T^r \bmod p$, $x = g^r \bmod p$ and $y = mv^r \bmod p$. (w, x, y) is sent to the research company.

Step 3'. (Forwarding) Each ciphertext (w, x, y) sent to the research company is transferred without any modification to one decryptor in a group of proxy decryptors.

Step 4'. (Decryption by proxy decryptors) Based on the computation in step 6 of Protocol 1, transferred (w, x, y)'s are decrypted in parallel by multiple proxy decryptors.

Due to parallel decryption, this type of decryption is faster than decryption by a single person. Moreover, the president is exempt from performing the ciphertext transformation.

The above method does not allow the president to prohibit members of the survey section, who is not designated as a proxy decryptor, to decrypt a ciphertext. Admissible proxy decryptors in a whole group of proxy decryptors can be restricted by allowing proxy decryptor to conduct a ciphertext transformation. Further discussion is given in Sect. 5.

When a three-component ciphertext is received, the original decryptor can choose one out of three cases. Either the original decryptor decrypts the ciphertext by itself, the proxy decryptor decrypts it in place of the original decryptor or both the original and proxy decryptors decrypt it. In the first and third cases where the original decryptor decrypts the ciphertext, the security of the ElGamal cryptosystem is increased by the following procedure. The original decryptor first checks $w \equiv x^t \bmod p$, and if the check is passed, it calculates further $y/x^u \bmod p$. Otherwise, it outputs nothing. This is the exactly the procedure of the cryptosystem secure against indifferently chosen ciphertext attacks proposed in [2]. In the similar context, our approach is applicable to the cryptosystem secure against adaptively chosen ciphertext attacks described in [20]. Appendix A gives the algorithm of the original cryptosystem [20] and its modification into the proxy cryptosystem.

In stead of making T public, both the original and proxy decryptors can treat it as a secret value among them. In this case, a sender of an encrypted email cannot compute a ciphertext for the proxy decryptor alone any more, and the proxy decryptor requires assistance by the original decryptor.

Security considerations: Similar to the proxy signature scheme for partial delegation [7], the security of Protocol 1 resides in the difficulty of computing ρ satisfying a congruence $g^{\rho}T^{T} \equiv v \bmod p$, given T and v. Modified ElGamal signature schemes for a signature σ, a random number K and a public key v described in [19] and [15] are based on a congruence $g^{\sigma} \equiv vK^{\{Km \bmod q\}} \bmod p$ with q satisfying $q|p-1$ and $g^{\sigma} \equiv v^{m}K^{K} \bmod p$, respectively. The congruence for Protocol 1 can be reduced to the congruence of these modified ElGamal signature schemes for a constant message, 1. To authors' knowledge no crucial attack against these modified ElGamal signature schemes has been reported up to now.

As described in Sect. 5 different t's can be assigned to multiple decryptor. In such a case two proxy decryptors may try to find s by the following method.

Step 1. Two proxy decryptors bring their proxies, ρ_1 ($\equiv s - t_1T_1 \bmod p-1$) and ρ_2 ($\equiv s - t_2T_2 \bmod p-1$).

Step 2. i_1 and i_2 are chosen from $Z_{p-1}\backslash\{0\}$ until i_1 and i_2 are found satisfying $\rho_1 + i_1T_1 \bmod p-1 \equiv \rho_2 + i_2T_2 \bmod p-1$.

Step 3. A congruence $g^{\rho_1+i_1T_1} \equiv v \bmod p$ is checked for i_1 obtained in the former step. If the check succeeds, true s is $\rho_1 + i_1T_1 \bmod p-1$. Otherwise, go to step 2.

Such a birthday attack is not enough effective in general because the step 2 is passed after about $1.17\sqrt{p-1}$ choices are made. If T_1 and/or T_2 have large common divisors with $p-1$, $\{(g^{T_1})^{i_1} \bmod p | i_1 \in Z_{p-1}\backslash\{0\}\}$ and/or $\{(g^{T_2})^{i_2} \bmod p | i_2 \in Z_{p-1}\backslash\{0\}\}$ become a small set, and the above attack may be feasible. In order to avoid such an attack, one should select T which does not have large common divisors with $p-1$, or T which satisfies $\gcd(T, p-1) = 1$. An alternative and better way is that p is selected such that $(p-1)/2$ is also a prime, or g is selected as $h^{\frac{p-1}{q}} \bmod p$ for a large q satisfying $q|p-1$ and a primitive root h of Z_p^*.

When multiple decryptors are involved, a proxy decryptor may attempt to compute other proxy, i.e. to compute ρ_2 from (ρ_1, T_1, T_2, v, p). This is the problem

of signature forgery pointed out above, and no serious attack is known.

(On transformation) Under the assumption on the difficulty of Diffie-Hellman problem, DH problem, it is hard to compute $w = T^r \bmod p$ from (x, y) without the knowledge of t, which is the secret of the original decryptor.

Even though the proxy decryptor has the proxy ρ, it is hard for the proxy decryptor to compute w. This is because both s and t are unknown values in a congruence $\rho \equiv s - tT \bmod p - 1$. Computing s of this congruence, in other words t, means breaking the modified ElGamal signature schemes [15], [19].

(On authorization) The proxy decryptor can extract m by following step 6 of Protocol 1. On the other hand, a third party observes only (g, T, p, w, x, y). (T, w) is additional to the ElGamal cryptosystem, and (T, x, w) is simply the values processed in the Diffie-Hellman key agreement between a user possessing x and the other possessing T. t of T is chosen independently of y and the message m so that (T, w) does not releases information on m.

4.2 Another Proxy Cryptosystem for ElGamal Cryptosystem

In this section another proxy cryptosystem applied for the ElGamal cryptosystem is shown. In the following protocol, s of v is randomly selected from Z^*_{p-1}.

[Protocol 2]

Step 1. (Proxy generation) An original decryptor computes $\rho = sd \bmod p - 1$, where d is randomly generated from Z^*_{p-1}.

Step 2. (Proxy delivery) The original decryptor gives ρ to a proxy decryptor in a secure way.

Step 3. (Encryption) A document m is encrypted into (x, y), where $r \in Z_{p-1}\backslash\{0\}$, $x = g^r \bmod p$ and $y = mv^r \bmod p$.

Step 4. (Ciphertext transformation) The original decryptor transforms the (x, y) into (w, y) or (w, x, y) where $w = x^e \bmod p$ and $ed \equiv 1 \bmod p - 1$.

Step 5. (Decryption by proxy decryptor) The proxy decryptor computes

$$\begin{aligned} y/w^\rho &\equiv (mv^r)/g^{xder} \bmod p \\ &\equiv m(v/g^s)^r \bmod p \\ &\equiv m \bmod p. \end{aligned}$$

Proxy verification process is not included in Protocol 2. If the original decryptor reveals $E = g^e \bmod p$ in step 2, the proxy decryptor can confirm that (ρ, E) satisfies $v \equiv E^\rho \bmod p$. But such a pair can be computed without the knowledge of s, see Lemma 1. Additionally E is not used for decryption in step 5. Thus the verification of proxy is not required in this protocol.

If an improper e is used in the ciphertext transformation, the proxy decryptor cannot recover a proper plaintext. The same trouble occurs in Protocol 1. The original decryptor should behave properly in ciphertext transformation.

Security considerations:

Lemma 1: The intractability of the problem of computing s given $(g^s \bmod p, g^r \bmod p, g^{er} \bmod p, \rho, p, g)$, where $r \in_R Z_{p-1}\backslash\{0\}$, $s, e, d \in_R Z^*_{p-1}$, $\rho \equiv sd \bmod p-1$ and $ed \equiv 1 \bmod p - 1$, is equivalent to that of the problem of computing s given $(g^s \bmod p, p, g)$, where $s \in_R Z^*_{p-1}$.

Proof: It is trivial that if the discrete logarithm problem is solved, the former problem is solved. The opposite reduction is proved as follows.

First select $\rho \in_R Z^*_{p-1}$. Then compute $E = (g^s)^{\rho^{-1}} \bmod p$, using the given $g^s \bmod p$ and the selected ρ. Select $r \in_R Z_{p-1}\backslash\{0\}$, and feed $(g^s \bmod p, g^r \bmod p, E^r \bmod p, \rho)$ in an oracle for solving the former problem. s is returned from the oracle and the difficulty of solving two problems is proven to be equivalent.

Lemma 2: The intractability of the problem of computing w $(\equiv g^{er} \bmod p)$ given $(g^s \bmod p, g^r \bmod p, \rho, p, g)$, where $r \in_R Z_{p-1}\backslash\{0\}$, $s, e, d \in_R Z^*_{p-1}$, $\rho \equiv sd \bmod p - 1$ and $ed \equiv 1 \bmod p - 1$, is equivalent to that of the problem of computing w given $(E (\equiv g^e \bmod p), g^r \bmod p, p, g)$, where $r \in_R Z_{p-1}\backslash\{0\}$ and $e \in_R Z^*_{p-1}$.

Proof: If the latter Diffie-Hellman problem [4] is solved, the former problem is solved by first computing $E = (g^s)^{\rho^{-1}} \bmod p$. Then feed $(E, g^r \bmod p, p, g)$ in a DH oracle.

If the former problem is solved, the DH problem is solved by first generating a random number $\rho \in_R Z^*_{p-1}$. Then compute $g^{e\rho} \bmod p$, and feed $(g^{e\rho} \bmod p, g^r \bmod p, \rho, p, g)$ in an oracle for the former problem.

Therefore, the difficulty of these two problems are equivalent.

(On transformation) Since d is randomly selected, e is considered to be a random number. An attacker does not know ρ. Hence, it is hard for an attacker to transform x into w.

The DH problem is regarded as hard, and it is known [3], [10] that under a certain condition it is equivalent to the discrete logarithm problem. Therefore, from the Lemma 2 even the proxy decryptor given ρ cannot compute w for ciphertext transformation.

(On authorization) The proxy decryptor can extract m by following step 5 of Protocol 2. On the other hand, a third party observes only (g, p, w, x, y). w is additional to the ElGamal cryptosystem. It is hard for

the third party to compute a correct d of the proxy decryptor corresponding to w without the knowledge of e, and the third party cannot extract m.

Unlike in Protocol 1, the proxy decryptor can compute a pair $(\rho', \bar{e})$ satisfying $\rho' \equiv \rho^{\bar{d}} \bmod p-1$ and $\bar{e}\bar{d} \equiv p-1$. With these values, (w, y) is transformed into $(w^{\bar{e}}, y)$, and m is extracted with the knowledge of ρ' by the same procedure in step 5 of Protocol 2. If the legally authorized decryptor gives other decryptor only ρ' and keeps $\bar{e}$ in secret, the second proxy decryptor has to wait until the legal proxy decryptor transforms the ciphertext (w, y) into $(w^{\bar{e}}, y)$. In this way the authorized proxy decryptor can further delegate its decrypting operation to other decryptors. Such a chain of delegation, which does not occur in Protocol 1, can be useful for an implementation in a hierarchical organization.

As in an ordinary cryptosystem, a proxy cryptosystem has a problem of illegal access. A proxy decryptor may give its proxy to an unauthorized decryptor. In the proxy signature schemes a created proxy signature is different for each proxy signer, and an original signer can identify a corresponding proxy signer from a created proxy signature. This indicates even if a proxy signer gives his proxy to other users, a signature created by the delegated user is identified as a signature of the original proxy signer. In contrast, an original decryptor is unable to detect the illegal access in the proxy cryptosystem. Let us consider two types of illegal accesses. One is an access by an person who is given ρ by the proxy decryptor but not by the original decryptor. The other is an access by a person who possesses a new proxy ρ' described above. The latter undetected illegal access is unique in the proxy cryptosystem. The original decryptor cannot distinguish or even detect both decryptions. So, a person attempting illegal access do not care which proxy, a proxy ρ or a newly generated ρ', he receives. He cannot be identified with the use of any one of them. Moreover, as easily observed, one can compute the original proxy ρ from a pair $(\rho', \bar{e})$, and vice versa. This means the possession of $(\rho', \bar{e})$ is equivalent to the possession of ρ.

Meanwhile, an illegally authorized decryptor does not have a stronger privilege than an original decryptor. The illegally authorized decryptor cannot recover the ciphertext without the ciphertext transformation from the original ciphertext (x, y) to the ciphertext (w, y) for the proxy decryptor.

4.3 Proxy Cryptosystem for RSA Cryptosystem

The following is a proxy cryptosystem for the RSA cryptosystem. An original decryptor selects two primes p and q, and computes $n = pq$. (e_s, n) and (s, p, q) are a public key and a secret key of the original decryptor, respectively, satisfying $\gcd(s, \lambda(n)) = 1$, $e_s s \equiv 1 \bmod \lambda(n)$. $\lambda(\cdot)$ is the Carmichael function.

[Protocol 3]

Step 1. (Proxy generation) An original decryptor computes $\rho = sd \bmod \lambda(n)$, where d is randomly generated from $Z^*_{\lambda(n)}$.

Step 2. (Proxy delivery) The original decryptor gives ρ to a proxy decryptor in a secure way.

Step 3. (Encryption) A document m is encrypted into x by $x = m^{e_s} \bmod n$.

Step 4. (Ciphertext transformation) The original decryptor transforms the x into w by $w = x^e \bmod n$, where $ed \equiv 1 \bmod \lambda(n)$.

Step 5. (Decryption by proxy decryptor) The proxy decryptor computes

$$\begin{aligned} w^\rho &\equiv (m^{e_s e})^{\{sd \bmod \lambda(n)\}} \bmod n \\ &\equiv m^{\{edc_s s \bmod \lambda(n)\}} \bmod n \\ &\equiv m \bmod n. \end{aligned}$$

As in Protocol 2, the proxy verification process is not included in this protocol.

Security considerations:

Lemma 3: If a random number $f \in_R Z_n \backslash \{0\}$ relatively prime to $\lambda(n)$ can be generated within a polynomial number of trials of n, the intractability of the problem of computing m given $(m^{e_s} \bmod n,\ m^{e_s e} \bmod n, e_s, n)$, where e_s, e and n meet the condition of the RSA cryptosystem, is equivalent to that of the problem of computing m given $(m^{e_s} \bmod n, e_s, n)$.

Proof: If the latter RSA problem is solved, the former is solved immediately.

If the former problem is solved, the RSA problem is solved by first generating f from $Z_n \backslash \{0\}$ at random. Then $(m^{e_s})^f \bmod n$ is computed, and feed $(m^{e_s} \bmod n,\ m^{e_s f} \bmod n, e_s, n)$ into an oracle for the former problem. If the oracle does not output anything or output an incorrect answer, repeat the above process by generating another f. This procedure is repeated until the oracle outputs a correct m.

In general, the number of integers relatively prime to an integer N is $6N/\pi^2$ in average. This means f should be selected roughly $\lceil \pi^2/6 \rceil \cong \lceil 1.64 \rceil = 2$ times in average. Usually, the RSA primes, p and q, need to satisfy several security conditions, e.g. $p-1$ has a large prime factor. Assume that $p-1 = 2\tilde{p}$ and $q-1 = 2\tilde{q}$ for primes $\tilde{p}$ and $\tilde{q}$ s.t. $|\tilde{p}| \cong |\tilde{q}|$. Then $\frac{|Z^*_{\lambda(n)}|}{|Z_n \backslash \{0\}|} = \frac{(\tilde{p}-1)(\tilde{q}-1)}{(2\tilde{p}+1)(2\tilde{q}+1)-1} \cong \frac{1}{4}$. In both cases the number of trials for generating f is within polynomial of n. Lemma 3 indicates that two problems have the same degree of difficulty under a certain condition.

(On transformation) A third party and a proxy decryptor do not know e for transformation. Even given a group of $(m^{e_s} \bmod n, m^{e_s e} \bmod n)$, the third party

with no knowledge on e cannot execute the ciphertext transformation. For the third party this problem can be regarded as a problem to solve the RSA cryptosystem without knowing both a secret value e and a public value d. In order to illegally transform a ciphertext, the proxy decryptor faces a problem to compute $(m^{e_s})^e \bmod n$ for a given $m^{e_s} \bmod n$ from (e_s, n, ρ) and a set of $(m_i^{e_s} \bmod n,\ m_i^{e_s e} \bmod n)$ for a certain amount of i. The degree of the difficulty of this problem is not totally clear, but this problem has not been solved until now.

(On authorization) As explained in Lemma 3, a third party has to solve the RSA problem in Protocol 3. So, as long as the RSA problem is not violated, the proposed scheme is considered to be secure.

5. Efficiency, Proxy Revocation and Authorization of Multiple Decryptors

In this section properties of the proposed proxy cryptosystems are examined.

Efficiency: The amount of computational work needed in the ordinary and the proxy version of ElGamal cryptosystem is shown in Tables 1, 2 and 3. Since the operation in Protocol 3 is similar to that in Protocol 2 except the inverse operations, the evaluation of computational work of Protocol 3 is omitted. The amount of computational work of the proposed schemes is roughly estimated by the number of $|p|$-bit exponentiations modulo p. $\mathrm{Inv}(|p|)$ denotes the computational work of taking inverse modulo $|p|$-bit integer.

Table 1 shows the amount of computational work required in a consecutive execution of decryption and re-encryption.

Table 2 and Table 3 show the amount of computational work required in Protocols 1 and 2. In these tables, total (case 1) shows values in a case such that an original decryptor does not decrypt the ciphertext, and total (case 2) shows values in a case such that an original decryptor decrypts the ciphertext. The computational work of proxy generation and proxy verification is shown below Table 2 and Table 3. Since the generation and verification of proxies are performed only when a proxy is given to a proxy decryptor at the first time, this computational work per one ciphertext transformation becomes negligible as the number of decrypted messages increases.

In Protocol 1 the total amount of computational work is about $31/6+\mathrm{Inv}(|p|)$ $(= 3+(13/6+2\mathrm{Inv}(|p|)))$ in the case 1 and about $5+2\mathrm{Inv}(|p|)$ $(= 3+(2+2\mathrm{Inv}(|p|)))$ in the case 2. In both cases Protocol 1 requires less amount of total computational work than the consecutive execution of decryption and re-encryption does, i.e. $6+2\mathrm{Inv}(|p|)$ $(= 4+(2+2\mathrm{Inv}(|p|)))$.

In Protocol 2 the total amount of computational work is about $4+\mathrm{Inv}(|p|)$ $(= 3+(1+\mathrm{Inv}(|p|)))$ in the case 1 and about $5+2\mathrm{Inv}(|p|)$ $(= 3+(2-2\mathrm{Inv}(|p|)))$ in the case 2. Protocol 2 also requires less amount of total computational work than the consecutive execution of decryption and re-encryption does.

A proxy decryptor needs to execute in Protocol 2 the same amount of computation as in the re-encryption method, and the proxy decryptor needs to perform a slightly more computation, i.e. 1/6, in Protocol 1 than in the re-encryption method. Even so, the original decryptor's process costs 1 $|p|$-bit exponentiation modulo p, whereas 2 $|p|$-bit exponentiations modulo p are required in consecutive processing of decryption and re-encryption. Such a property is well suited to a situation such that an original decryptor receives many ciphertexts and delegates their decryptions to multiple decryptors, e.g. survey. Concerning the computational work, the proxy signature and the proxy cryptosystem have the following relation. The computational work in verification operation is mainly reduced in the proxy signature and the computational work in ciphertext transformation is mainly reduced in the proxy cryptosystem.

Table 1 Computational work of the decryption and re-encryption method based on ElGamal cryptosystem.

	Encryption	Decryption
Total	4	$2+2\mathrm{Inv}(\|p\|)$
Sender	2	
Original decryptor	2	$1+\mathrm{Inv}(\|p\|)$
Proxy decryptor		$1+\mathrm{Inv}(\|p\|)$

Table 2 Computational work of the proxy cryptosystem for ElGamal cryptosystem (Protocol 1).

	Encryption (Transformation)	Decryption
Total (Case 1)*	3	$7/6+\mathrm{Inv}(\|p\|)$
Total (Case 2)†	3	$13/6+2\mathrm{Inv}(\|p\|)$
Sender	2	
Original decryptor	1	$1+\mathrm{Inv}(\|p\|)$
Proxy decryptor		$7/6+\mathrm{Inv}(\|p\|)$

Proxy generation: Almost 0, Proxy verification: 2

Table 3 Computational work of the proxy cryptosystem for ElGamal cryptosystem (Protocol 2).

	Encryption (Transformation)	Decryption
Total (Case 1)*	3	$1+\mathrm{Inv}(\|p\|)$
Total (Case 2)†	3	$2+2\mathrm{Inv}(\|p\|)$
Sender	2	
Original decryptor	1	$1+\mathrm{Inv}(\|p\|)$
Proxy decryptor		$1+\mathrm{Inv}(\|p\|)$

Proxy generation: Almost 0, Proxy verification: No method

* Total (case 1): Original decryptor does not decrypt the ciphertext.

† Total (case 2): Original decryptor decrypts the ciphertext.

Signing operation in the proxy signature and decrypting operation in the proxy cryptosystem require almost the same amount of computational work as an original signer and an original decryptor require, respectively, so that many signatures are created and many ciphertexts are decrypted by delegating signing power and decrypting power to multiple proxy signers and multiple proxy decryptors.

Revocation: There are two ways to prohibit proxy decryptor's access to the ciphertext after the ciphertext transformation is executed:

- To revoke the proxy of the proxy decryptor, which leads to prohibition of any further access to ciphertexts of the original decryptor.
- To disqualify the proxy decryptor for the access to a specified ciphertext.

The first method is achieved by changing the public key of the original decryptor, and accordingly update all proxies of proxy decryptors whom the original decryptor considers as qualified. There is a very simple method [7] to update proxies through an insecure channel. Thereby the owner of the proxy decryptor does not need to visit the original decryptor again, in other words no secure channel is needed. This protocol is shown in Appendix B.

The second method is useful only when the proxy decryptor makes an access to the message for the first time, or when the proxy decryptor has not stored the decrypted message at the earlier access and tries to get an access to the ciphertext again. In this revocation, the original decryptor selects a transformed ciphertext (w, x, y), and computes $x' = x \cdot g^{\bar{r}} \bmod p$ $(\equiv g^{r'} \bmod p)$ and $y' = y \cdot v^{\bar{r}} \bmod p$ $(\equiv mv^{r'} \bmod p)$, where $\bar{r}$ is a randomly selected number and $r' \equiv r + \bar{r} \bmod p - 1$. Since r of w $(\equiv T^r \bmod p)$ and r' are independent, the proxy decryptor cannot compute m from the updated ciphertext (w, x', y'). Once the form of a ciphertext is changed, an original decryptor is sure that the encrypted message is not read by a further access.

Authorization of multiple decryptors: In some cases, e.g. in file access, an original decryptor wants to restrict an access group in a group of proxy decryptors.

The original decryptor may authorize multiple proxy decryptors with different proxies for this purpose. In Protocol 1, the original decryptor gives each proxy decryptor D_i a proxy $\rho_i = s - t_i T_i \bmod p - 1$ with a distinct t_i. A transformed ciphertext for a group G of proxy decryptors is $(\{w_j | D_j \in G\}, x, y)$, where $w_j = x^{t_j} \equiv T_j^r \bmod p$. The proxy decryptor D_j with ρ_j treats (w_j, x, y) as its ciphertext, and executes decryption. This method is not enough efficient in terms of the message length and the amount of computational work since both of them are proportional to $O(\xi)$, where ξ is the number of proxy decryptors.

The original decryptor can authorize multiple proxy decryptors by combining a proxy cryptosystem with a selective broadcasting method. In this case, the multi-dimensional method for a secure broadcast communication proposed in [6] is quite useful for reducing the message length and the amount of computational work from $O(\xi)$ of the above method to $O(m \sqrt[m]{\xi})$ of m-dimensional method. In the m-dimensional method a user $U_{i_1 i_2 \cdots i_m}$ is assigned to a point $(i_1, i_2, \cdots, i_m)$ in an m-dimensional space, and public values $g^{t_{i_1}} \bmod p$, $g^{t_{i_2}} \bmod p$, $\cdots, g^{t_{i_m}} \bmod p$ are assigned to the point $(i_1, i_2, \cdots, i_m)$.

Step 1. (Proxy generation, delivery and verification) $\rho_{i_1 i_2 \cdots i_m}$ is computed by the original decryptor, and given to $D_{i_1 i_2 \cdots i_m}$. $\rho_{i_1 i_2 \cdots i_m}$ satisfies $\rho_{i_1 i_2 \cdots i_m} = s - t_{i_1 i_2 \cdots i_m} T_{i_1 i_2 \cdots i_m} \bmod p - 1$, where $t_{i_1 i_2 \cdots i_m} = t_{i_1} + t_{i_2} + \cdots + t_{i_m} \bmod p - 1$ and $T_{i_1 i_2 \cdots i_m} = g^{t_{i_1 i_2 \cdots i_m}} \bmod p$. Each $D_{i_1 i_2 \cdots i_m}$ checks the validity of given $\rho_{i_1 i_2 \cdots i_m}$.

Step 2. (Encryption) A ciphertext (x, y) is computed as in step 4 of Protocol 1.

Step 3. (Ciphertext transformation) In order to transfer (x, y) to a proxy decryptor $D_{i_1 i_2 \cdots i_m}$, the original decryptor computes $w_{i_j} = x^{t_{i_j}} \equiv T_{i_j}^r \bmod p$ for all i_j corresponding to $D_{i_1 i_2 \cdots i_m}$, where $T_{i_j} = g^{t_{i_j}} \bmod p$. A ciphertext to a group G of proxy decryptors is $(\{w_{i_j} | w_{i_j} = x^{t_{i_j}} \bmod p$ for all i_j corresponding to $D_{i_1 i_2 \cdots i_m} \in G\}, x, y)$.

Step 4. (Decryption by proxy decryptors) $D_{i_1 i_2 \cdots i_m}$ obtains m by computing step 6 of Protocol 1 by replacing ρ and T with $\rho_{i_1 i_2 \cdots i_m}$ and $T_{i_1 i_2 \cdots i_m}$, respectively.

As described in Sect. 4.1, a creator of a ciphertext can take charge of the operation for the ciphertext transformation as follows.

Step 2'. (Encryption) In addition to (x, y), all public values $T_{i_j} = g^{t_{i_j}} \bmod p$ corresponding to a group of users are selected, and these T_{i_j} are raised to rth power modulo p. A ciphertext to a group G of proxy decryptors is $(\{w_{i_j} | w_{i_j} = T_{i_j}^r \bmod p$ for all i_j corresponding to $D_{i_1 i_2 \cdots i_m} \in G\}, x, y)$.

Step 3'. (Forwarding) The original decryptor selects $\{w_{i_j}\}$ for each of $D_{i_1 i_2 \cdots i_m}$, and forwards selected $\{w_{i_j}\}$ together with (x, y) to $D_{i_1 i_2 \cdots i_m}$.

Please refer to [6] for more detail description of the m-dimensional method.

6. Conclusions

In this paper proxy cryptosystems have been proposed. By the proxy cryptosystem an original decryptor can securely transfer its ciphertext to a proxy decryptor, and

it can permit the proxy decryptor to recover the message instead of the original decryptor. Many different constructions for the proxy cryptosystem are possible for one normal cryptosystem. In this paper two proxy cryptosystems for the ElGamal cryptosystem and one proxy cryptosystem for the RSA cryptosystem have been described. In these proxy cryptosystems the transformation of ciphertexts is more efficient than consecutive processing of decryption and re-encryption, and decrypting operation requires almost the same amount of computational work as the ordinary decrypting operation. Therefore, an entity receiving a lot of encrypted messages can efficiently conduct decryption of these ciphertexts by giving the decrypting power to multiple proxy decryptors.

Acknowledgments

Authors would like to thank Dr. Renè Peralta for discussion on problems related to the discrete logarithm problem. Authors would also like to thank anonymous referees for their comments.

References

[1] E.F. Brickell and Y. Yacobi, "On privacy homomorphism," Lecture Notes in Computer Science 304, Advances in Cryptology — Eurocrypt'87, pp.117–125, Spring-Verlag, 1988.
[2] I. Damgård, "Towards practical public key systems secure against chosen ciphertext attacks," Lecture Notes in Computer Science 576, Advances in Cryptology - Crypto'91, pp.445–456, Spring-Verlag, 1992.
[3] B. den Boer, "Diffie-Hellman is as strong as discrete log for certain primes," Lecture Notes in Computer Science 403, Advances in Cryptology — Crypto'88, pp.530–539, Spring-Verlag, 1990.
[4] W. Diffie and M. Hellman, "New directions in cryptography," IEEE Trans. Inf. Theory, vol.IT-22, no.6, pp.644–645, Nov. 1976.
[5] T. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithm," IEEE Trans. Inf. Theory, vol.IT-31, no.4, pp.469–472, July 1985.
[6] M. Mambo, A. Nishikawa, E. Okamoto, and S. Tsujii, "A secure broadcast communication method with short messages," IEICE Trans. Fundamentals, vol.E77-A, no.8, pp.1319–1327, Aug. 1994.
[7] M. Mambo, K. Usuda, and E. Okamoto, "Proxy signatures: Delegation of the power to sign messages," IEICE Trans. Fundamentals, vol.E79-A, no.9, Sept. 1996. Or "Proxy signatures for delegating signing operations," Proc. of 3rd ACM Conference on Computer and Communications Security, pp.48–57, 1996.
[8] T. Matsumoto, K. Kato, and H. Imai, "Speeding up secret computation with insecure auxiliary devices," Lecture Notes in Computer Science 403, Advances in Cryptology — Crypto'88, pp.497–506, Spring-Verlag, 1990.
[9] T. Matsumoto, T. Okada, and H. Imai, "Directly transformed link encryption," IECE Trans., vol.J65-D, no.11, pp.1443–1450, Nov. 1982.
[10] U.M. Maurer, "Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms," Lecture Notes in Computer Science 839, Advances in Cryptology — Crypto'94, pp.271–281, Spring-Verlag, 1994.
[11] T. Okamoto, "Provably secure and practical identification schemes and corresponding signature schemes," Lecture Notes in Computer Science 740, Advances in Cryptology — Crypto'92, pp.31–53, Spring-Verlag, 1993.
[12] R. Rivest, "The MD5 message-digest algorithm," Request for Comments 1321, April 1992.
[13] R. Rivest, L. Adleman, and M. Dertouzos, "On databanks and privacy homomorphisms," in Foundations of Secure Computation, eds. R.A. Demillo et al., pp.168–177, Academic Press, 1978.
[14] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems," Commun. of the ACM, vol.21, pp.120–126, 1978.
[15] A. Shimbo, "Multisignature schemes based on the ElGamal scheme," Proc. 1994 Symposium on Cryptography and Information Security, SCIS94-2C, Jan. 1994.
[16] A. Shimbo and S. Kawamura, "A factorization attack against some server-aided computation protocols for the RSA secret computation," Proc. 1990 Symposium on Cryptography and Information Security, SCIS90-3B, Jan. 1990.
[17] A. Shimbo and S. Kawamura, "Performance analysis of server-aided secret computation protocols," IEICE Trans., vol.E73, no.7, pp.1073–1080, July 1990.
[18] M. Tompa and H. Woll, "Random self-reducibility and zero-knowledge interactive proofs of possession of information," Proc. of Symp. on Foundation of Computer Science, pp.472–482, 1987.
[19] S.M. Yen and C.S. Laih, "New digital signature scheme based on discrete logarithm," Electron. Letters, vol.29, no.12, pp.1120–1121, 1993.
[20] Y. Zheng and J. Seberry, "Practical approaches to attaining security against adaptively chosen ciphertext attacks," Lecture Notes in Computer Science 740, Advances in Cryptology — Crypto'92, pp.292–304, Spring-Verlag, 1993.

Appendix A

In [20] a public-key cryptosystem secure against adaptively chosen ciphertext attacks is presented. The protocol is as follows.

Let v and s be a public key and a private key of a decryptor, respectively, and $G(r)_{|1\cdots P(\eta)|}$ be a cryptographically strong pseudorandom string generator based on the difficulty of computing discrete logarithms in finite fields that outputs a $P(\eta)$-bit message, where $P(\eta)$ is an arbitrary polynomial with $P(\eta) > \eta$. $h(\cdot)$ is a one way hash function.

Algorithms by Zheng and Sheberry:
[Encryption]

Step 1. $k_1 \in_R [1, p-1]$, $k_1 \in_R [1, p-1]$, and $gcd(k_2, p-1) = 1$.

Step 2. $r = v^{k_1+k_2} \bmod p$.

Step 3. $z = G(r)_{|1\cdots P(\eta)|}$.

Step 4. $c_1 = g^{k_1} \bmod p$.

Step 5. $c_2 = g^{k_2} \bmod p$.

Step 6. $c_3 = (h(m) - k_1 r)/k_2 \bmod p - 1$.

Step 7. $c_4 = z \oplus m$

Step 8. Output (c_1, c_2, c_3, c_4) as a ciphertext.

[Decryption]

Step 1. $\tilde{r} = (c_1 c_2)^s \bmod p$.

Step 2. $\tilde{z} = G(\tilde{r})_{|1 \cdots P(\eta)|}$.

Step 3. $\tilde{m} = \tilde{z} \oplus c_4$.

Step 4. Output $\tilde{m}$ if $g^{h(\tilde{m})} \equiv c_1^{\tilde{r}} c_2^{c_3} \bmod p$. Otherwise, output $\emptyset$.

Modification to proxy cryptosystem: By the following modification a proxy cryptosystem secure against adaptively chosen ciphertext attack is constructed. This cryptosystem is based on the proposed Protocol 1. The same modification can be done based on Protocol 2.

Step 4 in decryption algorithm of an original decryptor: If a message is output, the original decryptor computes $w_1 = c_1^t \equiv g^{k_1 t} \bmod p$ and $w_2 = c_2^t \equiv g^{k_2 t} \bmod p$. The transformed ciphertext is composed of $(w_1, w_2, c_1, c_2, c_3, c_4)$.

Step 1 in decryption algorithm of a proxy decryptor: $\tilde{r}$ should be computed by $\tilde{r} = (c_1 c_2)^\rho (w_1 w_2)^\tau \equiv (t^s)^{k_1 + k_2} \bmod p$.

Appendix B

An original decryptor and a legal decryptor share a proxy, so that they can update the proxy even through an insecure channel where an eavesdropping is possibly conducted. By the following protocol a proxy of Protocol 2 is updated in on-line basis. A proxy of Protocol 1 and Protocol 3 is similarly updated [7], either.

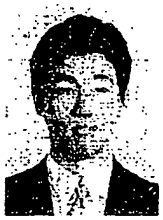
[On-line proxy updating protocol]

Step 1. (New public-key creation) An original decryptor selects its new secret $s' \in_R Z^*_{p-1}$ and computes its new public key v' by $v' = g^{s'} \bmod p$. Then the original decryptor announces the revocation to all of related proxy decryptors.

Step 2. (Identification) After the announcement, a proxy decryptor requests the update of its proxy. To this end, the proxy decryptor proves its identity by some identification protocol.

Step 3. (New proxy creation and implicit delivery) If the original decryptor is convinced of the identity of the proxy decryptor, it looks for its old secret proxy variable d in its secret proxy variable list. It calculates its old proxy $\rho = ds \bmod p - 1$ and her new proxy $\rho' = d's' \bmod p - 1$, where $d' \in_R Z^*_{p-1}$ is selected randomly. Then it can compute $\bar{\rho} = \rho' - \rho \bmod p - 1$. $\bar{\rho}$ is returned to her.

Step 4. (New proxy construction) Using the received information the proxy decryptor calculates its new proxy ρ' simply by $\rho' = \rho + \bar{\rho} \bmod p - 1$.



Masahiro Mambo received a B.Eng. degree from Kanazawa University, Japan, in 1988 and M.S.Eng. and Dr.Eng. degrees in electronic engineering from Tokyo Institute of Technology, Japan, in 1990 and 1993, respectively. He is currently a research associate in the school of information science at JAIST (Japan Advanced Institute of Science and Technology), and is involved in research on information security.

Eiji Okamoto received B.S., M.S. and Ph.D. degrees in electrical engineering from Tokyo Institute of Technology in 1973, 1975 and 1978, respectively. He worked for NEC Corporation from 1978 to 1991 and studied communication theory and information security. He has been a professor of Information Science at JAIST (Japan Advanced Institute of Science and Technology) since 1991. He was a visiting professor of Mathematics, Texas A & M University, USA, 1993–1994. He is a senior member of the IEEE Communications Society and Information Theory Group. He is also a member of the Information Processing Society of Japan, and International Association of Cryptographic Research. He received the best Paper Award of the IEICE in 1990 and the best Author Award of the IPSJ in 1993.

User's Guide

Microsoft® Word

The World's Most Popular Word Processor
Version 6.0

Microsoft Corporation

Document No. WB51157-1093
Printed in Ireland :09

Protecting Documents from Changes

Word provides several ways to restrict changes to documents. You can assign a password to prevent other users from opening a document or to keep others from saving changes to the document. You can also request or require that other users on a network open a document as read-only.

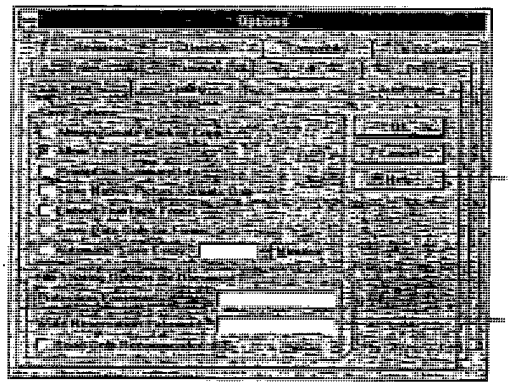
You can also assign a password so that other users can annotate a document and mark revisions. You or someone else who knows the password must open the document normally and review the changes before they become permanent. For more information, see Chapter 25, "Annotating, Revising, and Routing Documents."

If you use form fields to create a form, you can assign a password so that other users can fill in those parts of the form but cannot change anything else in the document. For more information, see Chapter 14, "Forms."

Warning If you assign a password for any of these types of protection, it's a good idea to write it down. Without the password, you cannot open the document.

Setting Passwords and Selecting Save Options

To assign a password to a document and set options that control whether changes can be saved, choose the Options button in the Save As dialog box or choose Options from the Tools menu, and then select the Save tab.



Choose the Help button for more information about these options.

Use these options to control changes to a document.

Protection Password To prevent other users from opening a document, type a password in the Protection Password box. Only users who know the password can open the document. Passwords are case-sensitive.

Write Reservation Password To prevent other users from saving changes to a document, type a password in the Write Reservation Password box, and then choose the OK button. Word will prompt you to type the password again to confirm it. Word then requires you to type the password to open the document normally. If you do not know the password, you can still open the document as read-only by choosing the Read Only button in the Password dialog box that appears when you open the document.

Read-Only Recommended To recommend, but not require, that other users open a document as read-only, select the Read-Only Recommended check box. When another user opens a document that's protected by this option, Word indicates that the document should be opened as read-only unless changes need to be saved. The user can then open the document normally or as a read-only document.

▶ **To protect a document with a password**

1. Open the document you want to protect with a password.
2. From the File menu, choose Save As.

   If you have not yet named the document, type a name in the File Name box.
3. Choose the Options button.
4. In the Protection Password box or the Write Reservation Password box, type a password, and then choose the OK button.

   A password can contain up to 15 characters and can include letters, numbers, symbols, and spaces. As you type the password, Word displays an asterisk (*) for each character you type. Note that passwords are case-sensitive.
5. When Word prompts you to confirm the password, retype it and then choose the OK button.
6. To save the document, choose the OK button.

   Make sure that you write down the document password, exactly as you typed it. You will need to type it the next time you open the document.

Tip If you want to allow other users to add only comments to a document, you can protect it by using the Protect Document command on the Tools menu. Other users can then open the document, but they can only make comments by using annotations.

---

cument, type a
v the password can

g changes to a
box, and then
ord again to
ı the document
he document as
alog box that

other users open a
eck box. When
Vord indicates that
:d to be saved. The
:ument.

File Name box.

sword box, type a

letters, numbers,
ys an asterisk (*)
ısitive.
and then choose

:tly as you typed
ent.

document, you
ools menu. Other
ıents by using

▶ **To change or delete a password**

1. Open the document whose password you want to change or delete.
2. From the File menu, choose Save As.
3. Choose the Options button.
4. In the Protection Password box or the Write Reservation Password box, select the row of asterisks that represents the existing password, and then do one of the following:
   - To change the password, type the new password.
   - To delete the password, press DELETE.
5. Choose the OK button.

   If you changed the password, Word asks you to retype the new password.
6. To save the document with the new password, choose the OK button.

Other Ways of Protecting Documents

Word offers other methods of protecting your documents.

For information about	See
Opening documents as read-only	"To open an existing document," earlier in this chapter
Preventing any changes to documents except for filling in form fields	Chapter 14, "Forms"
Preventing any changes to documents except for annotations and marked revisions	Chapter 25, "Annotating, Revising, and Routing Documents"

Note Protecting a form or locking a document for annotations or revisions does not keep another user from saving that document with a password or from setting other save options. If you want to protect a document from all types of changes, save it with a password by using one of the methods described in this chapter.

Some operating systems and networks also provide ways to protect documents. To find out if your system has these features, check with your network administrator or see the documentation for your operating system or network.

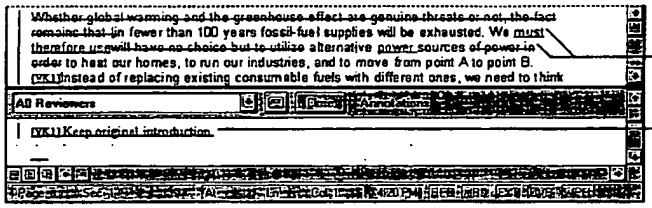
CHAPTER 25

Annotating, Revising, and Routing Documents

Word provides three features that make it easy to revise documents and incorporate comments from reviewers: annotations, revision marks, and routing.

To comment on a document rather than change it, use *annotations*—numbered comments added in a separate annotation pane. To make changes directly in the document, use *revision marks*. Revision marks show where text or graphics have been added, deleted, or moved. In addition, revision marks allow you to track changes by reviewer, date, and time.

You can use Word with Microsoft Mail or other compatible mail packages to send an online copy of a document to recipients who can comment on or add to the document. You can send multiple copies of the document to all reviewers at the same time, or you can route a single copy sequentially through a list of reviewers.

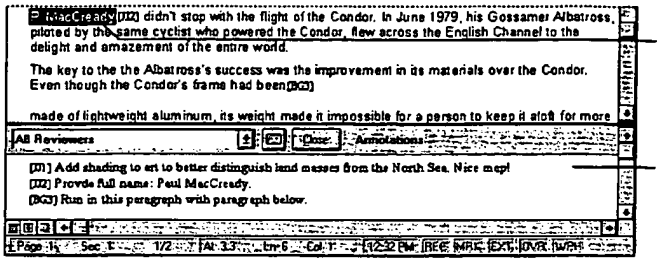


In This Chapter

QUICK START

Inserting Annotations

To comment on a document but not change the content, use annotations. To insert an annotation, select the text you want to comment on. From the Insert menu, choose Annotation. Word opens a separate annotation pane where you type your comments. To insert additional annotations, follow the same procedure, or select the text and press ALT+CTRL+A (Windows) or COMMAND+OPTION+A (Macintosh). When you finish, choose the Close button in the annotation pane.

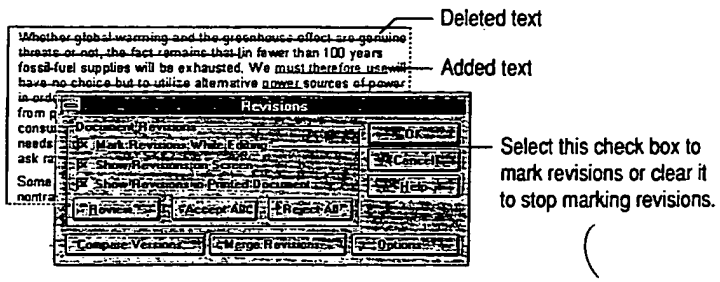


Viewing Annotations

To view annotations, choose Annotations from the View menu. The annotation pane displays comments from all reviewers. To view annotations from a single reviewer, select that person's name from the box at the top of the annotation pane. To view the range of text a particular annotation refers to, position the insertion point within the comment in the annotation pane. Word highlights the text the reviewer selected. When you finish viewing annotations, choose the Close button.

Marking Revisions

Use revision marking to track the changes that you or other reviewers make to a document. To turn on revision marking, choose Revisions from the Tools menu or double-click "MRK" in the status bar. Select the Mark Revisions While Editing check box, and then choose the OK button. From that point on, Word tracks all revisions. By default, the revision marks are displayed on the screen and in the printed document.



tations. To insert
Insert menu,
e you type your
edure, or select
l+A (Macintosh).

elected text that
ou want to
omment on

nnotation pane
here you type
omments

The annotation
s from a single
e annotation pane.
on the insertion
ts the text the
the Close button.

ewers make to a
the Tools menu or
s While Editing
Word tracks all
reen and in the

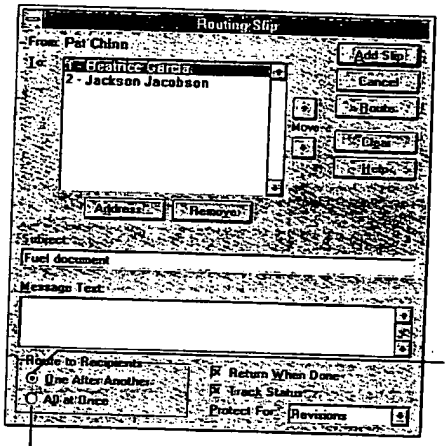
ox to
lear it
isions.

Hiding Revision Marks

If you want Word to track changes without displaying the revision marks on the screen, choose Revisions from the Tools menu. Then select the Mark Revisions While Editing check box, and clear the Show Revisions On Screen check box. To hide revision marks when printing, clear the Show Revisions In Printed Document check box, and then choose the OK button.

Routing a Document Online

You can use Word with Microsoft Mail or another compatible mail package to send a document to others. Choose Add Routing Slip from the File menu, and then choose the Address button. Select the names that you want to route the document to, choose the Add button, and then choose the OK button. Under Route To Recipients, select the distribution method. To send the document, choose the Route button. Recipients return their annotated or revised copies by using the Send command on the File menu.



To route the document to reviewers one after another, click here.

To route the document to all reviewers at once, click here.

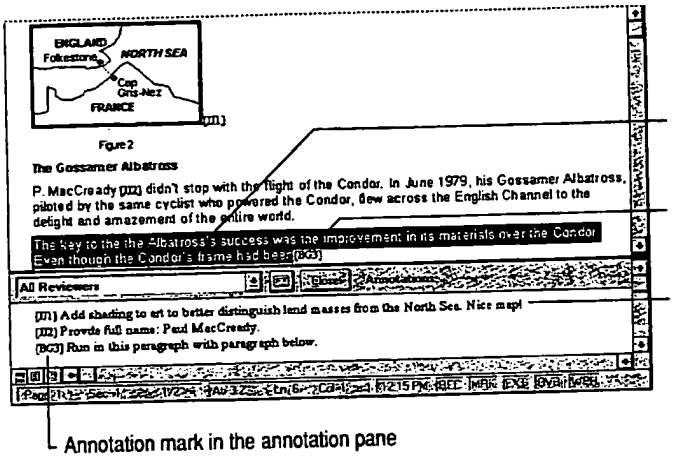
See the following pages for detailed information.

Using Annotations

When you want reviewers to comment on a document rather than make changes directly to it, have them use *annotations*. Annotations are numbered comments added in a separate annotation pane. Reviewers can include text as well as graphics in the annotation pane. With the appropriate hardware, voice and pen annotations can also be included.


Show/Hide ¶ button

Each annotation has an *annotation mark* that includes the reviewer's initials and a sequential number. Word obtains the reviewer's initials from the User Info tab in the Options dialog box (Tools menu). In the annotation pane, the annotation mark appears as normal text. In the document window, the annotation mark appears as hidden text. (You can view annotation marks in the document when the annotation pane is closed by clicking the Show/Hide ¶ button on the Standard toolbar.)



For information about routing online documents to multiple reviewers, see "Routing a Document Online," later in this chapter.

Inserting Annotations

Before you insert an annotation, it's a good idea to select the text or item that you want to comment on. That way, Word can later highlight the text or item to which the annotation refers. When you type an annotation, you can use the toolbars, the ruler, and commands on the Format menu to apply formatting to text in the annotation pane.

▶ To insert an annotation

1. Select the text or item you want to comment on, or position the insertion point at the end of the text or item you want to comment on.
2. From the Insert menu, choose Annotation.

   Word inserts an annotation mark (initials and a number formatted as hidden text) in the document and opens the annotation pane.

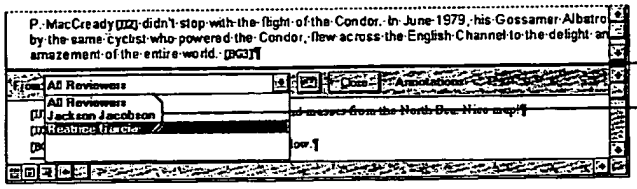
   Word uses the reviewer's initials from the User Info tab in the Options dialog box (Tools menu).
3. Type the annotation text in the annotation pane, and then do one of the following:
   - To close the annotation pane and return to the document, choose the Close button.
   - To keep the annotation pane open to add additional annotations, click in the document window, and repeat steps 1 through 3.

Using the keyboard You can also quickly insert annotations by using the keyboard. Press ALT+CTRL+A in Windows and COMMAND+OPTION+A on the Macintosh.

Tip Annotation marks are automatically displayed when the annotation pane is open. If the annotation pane is closed, you can display annotation marks and nonprinting characters in the document by clicking the Show/Hide ¶ button on the Standard toolbar. Click the button again to hide annotation marks.

Viewing and Locating Annotations

When the annotation pane is open, it shows the annotations that correspond to the part of the document displayed in the document window. As you scroll through the document, the annotation pane scrolls as well. You can change the size of the annotation pane by dragging the split box up or down on the vertical scroll bar.



▶ **To view annotations**

1. Do one of the following:
   - From the View menu, choose Annotations.
   - Double-click an annotation mark in the document window. If you don't see annotation marks, click the Show/Hide ¶ button on the Standard toolbar.
2. To view the annotations of a single reviewer, select the name of the reviewer from the Reviewers box at the top of the annotation pane. The default is All Reviewers, which displays the annotations of all reviewers.
3. When you finish viewing annotations, do one of the following:
   - To close the annotation pane and return to the document, choose the Close button.
   - To keep the annotation pane open and return to the document, click in the document window.

Note If a reviewer selected text before inserting an annotation, the selected text is highlighted when you position the insertion point within the corresponding annotation in the annotation pane. Text that is highlighted is not selected. To quickly select the highlighted text so that you can modify it, press ALT+F11 (Windows) or OPTION+F11 (Macintosh).

To locate a specific annotation, choose Go To from the Edit menu. Under Go To What, select Annotation. If you know which reviewer made the annotation, select the reviewer's name from the list. Choose the Next button until you find the annotation you want.

Incorporating and Deleting Annotations

To incorporate the text of an annotation into a document, select the annotation text or item, and then choose Copy from the Edit menu. Position the insertion point in the document where you want the text or item to appear, and then choose Paste from the Edit menu.

To delete an annotation, select the annotation mark in the document window and then press BACKSPACE or DELETE. Word automatically renumbers annotation marks each time you add, delete, or copy an annotation.

Printing Annotations

You can print annotations either separately from the rest of the document or with the document. If you print annotations only, Word prints the page number of the annotation mark, the reviewer's initials, the annotation number, and then the annotation text. If you print annotations with the document, Word prints this same information at the end of the document. Word prints hidden text in the document so that you can see the location of the annotation marks.

▶ **To print annotations only**

1. From the File menu, choose Print.
2. In the Print What box, select Annotations, and then choose the OK button.

▶ **To print a document with annotations**

1. From the File menu, choose Print.
2. Choose the Options button.
3. Under Include With Document, select the Annotations check box, and then choose the OK button.
4. In the Print dialog box, select any other printing options you want, and then choose the OK button.

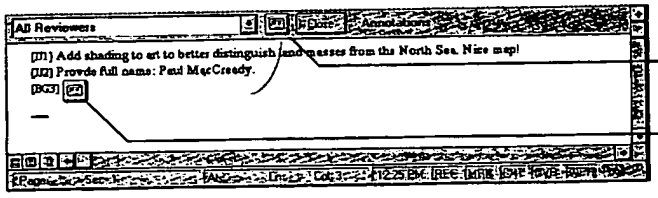
Inserting and Listening to Voice Annotations

If your computer runs Windows-based applications, you must have a sound board installed to listen to voice annotations. To record annotations, you must also have a microphone.

If you have a Macintosh computer, you can listen to voice annotations without having special sound equipment. To record annotations on a Macintosh, you must have a microphone.

You can distinguish a voice annotation from other annotations by the sound symbol—an audio cassette tape—which Word inserts in the annotation pane and in the document at the position of the insertion point.

Tip To insert a combination of text and voice annotation, insert the text annotation first. Then position the insertion point to the right of the annotation mark in the document and insert the voice annotation.



Note You can customize the marks Word uses to show document differences. For more information, see "Customizing Revision Marks," earlier in this chapter.

▶ **To compare two versions of a document**

1. Open the edited version of the document.
2. From the Tools menu, choose Revisions.
3. Choose the Compare Versions button.
4. In the Original File Name box, type or select the name of the original document, and then choose the OK button.

   Word displays the edited document marking inserted, deleted, and revised text with revision marks. The options for displaying revision marks are set on the Revisions tab in the Options dialog box (Tools menu).
5. To accept or reject the revisions, choose Revisions from the Tools menu. For more information, see "Incorporating Revisions," earlier in this chapter.

Protecting a Document for Annotations and Revisions

To allow reviewers to comment on but not make changes to a document, you can protect it for annotations. To allow reviewers to change a document and keep a record of all changes, you can protect it for revisions.

For maximum protection, you should also use a password when you protect a document for annotations or revisions. Otherwise, anyone can remove protection from the document by choosing Unprotect Document from the Tools menu.

▶ **To protect a document for annotations or revision marks**

1. Open the document you want to protect.
2. From the Tools menu, choose Protect Document.
3. Do one of the following:
   - To allow reviewers to insert annotations but not change the contents of the document, select the Annotations option button.
   - To track revisions, select the Revisions option button. The reviewers cannot turn off revision marking, and revisions cannot be accepted or rejected.
4. To ensure that a document is protected against untracked changes, type a password. This prohibits anyone who does not know the password from unprotecting the document.
5. Choose the OK button.

For more information on ways to protect a document, see Chapter 21, "Opening, Saving, and Protecting Documents."

▶ **To review and incorporate revisions**

1. From the Tools menu, choose Revisions.
2. Choose the Review button.
3. Do one of the following:
   - To move to a revision mark, click the appropriate Find button to search forward or backward in the document.
   - Click a revision mark in the document.

   For each selected revision mark, Word displays the reviewer's name, and the date and time the revision was entered.
4. Do one or more of the following:

To	Choose this button
Accept a revision	Accept
Reject a revision	Reject
Leave the revision mark unchanged and move to the next or previous revision	Find ➔ or ➔ Find
Undo the last acceptance or rejection of a revision	Undo Last

Note To automatically move to the next revision mark while reviewing the revisions, select the Find Next After Accept/Reject check box.

▶ **To accept or reject all revision marks**

1. From the Tools menu, choose Revisions.
2. Do one of the following:
   - To accept all revisions, choose the Accept All button.
   - To reject all revisions, choose the Reject All button.

   Word displays a message asking you to confirm that you want to accept or reject all revisions.
3. Choose the OK button.

Comparing Versions of a Document

To compare two versions of a document, you use the Compare Versions button in the Revisions dialog box. During the compare process, Word inserts revision marks that you can review and incorporate as described earlier in "Incorporating Revisions."

Make sure that the two documents you are comparing have different filenames, or—if they have the same name—that they are in different directories.

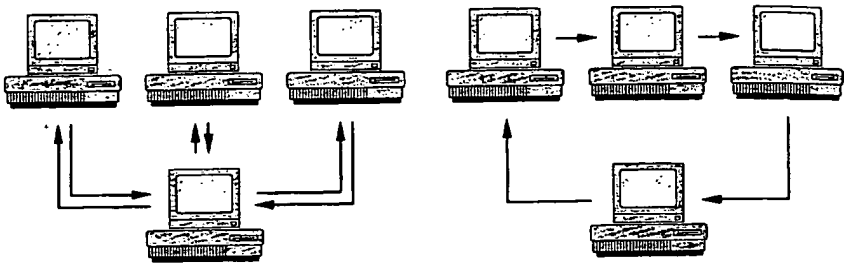
▶ **To unprotect a document for annotations or revision marks**

- From the Tools menu, choose Unprotect Document.

  If the author has protected the document with a password, you must know the password to unprotect the document.

Routing a Document Online

You can use Word and Microsoft Mail or a compatible mail program to route documents online. For example, you might want others to review an important memo before sending it out, or you might want several people to complete an online questionnaire or form.

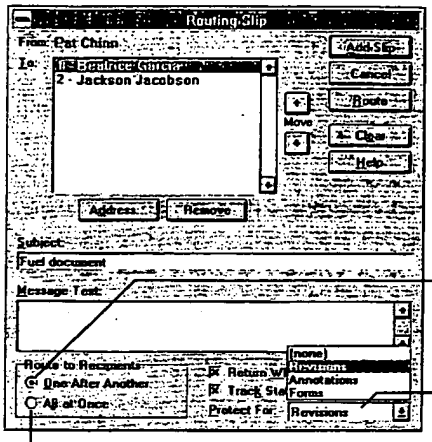


You can route a document to all reviewers at once ...

or you can route it to one reviewer after another.

You can route online copies in two ways. You can send a separate copy to all reviewers at the same time, or you can send a single copy that goes to each person on the list in turn, allowing each reviewer to see the comments of all previous reviewers.

Reviewers return their annotated or revised copies to you or the next person on the distribution list by choosing the Send command from the File menu. When all the copies have been returned, you can merge the annotations and revisions into the original document to simplify review of the comments. For more information on merging comments, see "Merging Annotations and Revisions," later in this chapter.



▶ To route a document to others

1. Open the document you want to route.
2. From the File menu, choose Add Routing Slip.
3. Choose the Address button. Select the names of the people to whom you want to route the document, choose the Add button, and then choose the OK button.

   If you want to route the document to one recipient after another, use the Move up and down arrows to put the names in the correct routing order.
4. In the Subject and Message Text boxes, type the subject and any message or instructions you want to send with the document. Each recipient will receive the same subject and message.

   Word automatically appends instructions to your message telling recipients to choose the Send command when they are finished.
5. Under Route To Recipients, do one of the following:
   - To route one copy of a document to one recipient after another, select the One After Another option button.
   - To route multiple copies of a document to all recipients at the same time, select the All At Once option button.

6. Select any other options you want, and then choose the Route button.

   If you want to continue to edit the document before you route it, choose the Add Slip button, and continue to edit the document. When you are ready to send the document, choose Send from the File menu. Word displays a message asking you to confirm that you want to route the document.

   The document is sent to the distribution list as an attached Word file. The recipients can add annotations or revisions to the document and then return the copy to you by choosing the Send command on the File menu.

   If the document is being routed to one recipient after another, the Send command automatically routes it to the next person on the list before it returns to you. You will receive all the recipients' comments in one document after it has been routed to the last person on the list.

   If you send the document to all recipients at the same time, you will receive multiple copies of the document. You can then merge all changes into one document. For more information, see the following section.

Merging Annotations and Revisions

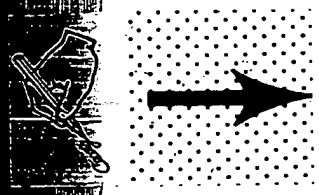
If you have given individual copies of a document to multiple reviewers, you can combine their annotations and revisions into the original document. When you merge annotations and revisions, any annotations and revisions already in the original document are preserved as additional comments are merged. Word assigns a different color to each reviewer. If there are more than eight reviewers, Word uses the colors again, so some reviewers may share the same color.

Note Annotations and revisions cannot be merged back to the original document unless they are marked. To ensure that revisions to a document are marked, you should protect the document for revisions or annotations before making the revisions. For information on protecting documents, see "Protecting Documents for Annotations and Revisions," earlier in this chapter.

▶ **To merge revision marks and annotations**

1. Open the document that has revisions you want to merge into the original document.
2. From the Tools menu, choose Revisions.
3. Choose the Merge Revisions button.

Opening a Document Created in Another Application



To convert documents to and from different file formats, you must install the appropriate *converters*. To import and export graphics contained in documents, you must install *graphics filters*. If you performed a complete installation when you installed Word, converters and graphics filters were also installed. If not, you can run the Microsoft Word Setup program again. For instructions, double-click the Help button on the Standard toolbar, and then type **setup**

Converters change the file format of a document, and they use graphics filters to import and export graphics that are within, or linked to, a document. For information about converting graphics files in such formats as TIFF and PICT, and for a list of supplied graphics filters, see Chapter 16, "Importing and Creating Graphics."

If you want to import data from a database into a Word document, you have three options. To convert an entire database file from database applications that Word can convert, such as Microsoft Excel, use the procedure under "Opening a Document," later in this chapter. To use information from a database to print form letters, mailing labels, and other types of mail merge documents, see Chapter 29, "Mail Merge: Step by Step," and Chapter 30, "Mail Merge: Advanced Techniques." To insert information from a database into a Word document as a table, see Chapter 28, "Exchanging Information with Other Applications."

To convert several documents at once, use the batch conversion macro BatchConversion in the CONVERT.DOT (Windows) or Conversion Macros (Macintosh) template. For information about using this macro, see "Converting Several Documents at Once," later in this chapter.

Supplied Converters

Word provides converters for the applications in the following list and for several plain-text file formats. For information about specific converters, and for instructions on how to obtain supplemental converters and graphics filters that were not shipped with Word, double-click the Help button on the Standard toolbar, and then type **readme**. Press ENTER twice, and then click File Conversion.

Converters Supplied with Word for Windows

Microsoft Word for Windows

Microsoft Word 4.*x* and 5.*x* for the Macintosh

Microsoft Word 3.0–6.0 for MS-DOS

WordPerfect 5.*x* for MS-DOS and Windows

Microsoft Write for Windows
Microsoft Excel BIFF 2.*x*, 3.0, 4.0*, and 5.0
Lotus 1-2-3 2.*x* and 3.*x**
RFT-DCA

* The converter can open, but not save, documents in this file format.

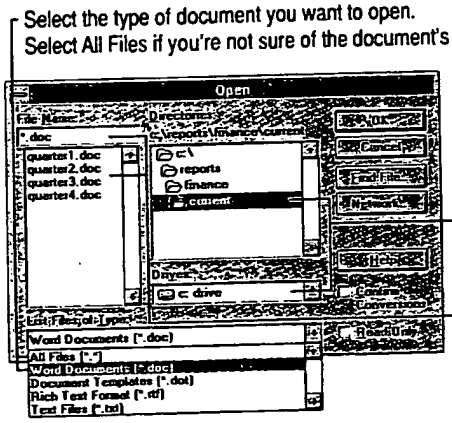
Converters Supplied with Word for the Macintosh

Microsoft Word for Windows
Microsoft Word 3.*x* for the Macintosh*
Microsoft Word 4.*x* and 5.*x* for the Macintosh
Microsoft Word 3.0–6.0 for MS-DOS
Microsoft Works 2.0 for the Macintosh
WordPerfect for MS-DOS and Windows 5.*x*
MacWrite®
MacWrite II 1.1
Microsoft Excel BIFF 3.0, 4.0*, and 5.0
RFT-DCA

* The converter can open, but not save, documents in this file format.

Opening a Document

To convert most types of documents, simply open them. Word automatically opens a copy of the document and converts the copy to Word format.



Most of the time, Word recognizes the file format, converts the document, and then displays it in a Word window. If Word cannot recognize the format, it displays the Convert File dialog box. Select the appropriate file format, and then choose the OK button. Word converts and opens the document.

If a document isn't converted correctly, close it without saving changes and then try converting again using a different converter. The original document remains unchanged until you save it in Word format or in some other format. You can also change the way Word converts some items. For more information, see "Customizing Conversions and Improving Compatibility," later in this chapter.

▶ To open a file created in another application


Open button

1. On the Standard toolbar, click the Open button.
2. In the List Files Of Type box, select the type of file you want to open. If you do not know the type of document or file format, select All Files.
3. In the File Name box, type or select the document you want to open.

   If the document you want to open does not appear in the File Name box, select a different drive, directory, or folder.
4. Choose the OK button to convert a copy of the document to Word format.

If the converter you need was not installed when you installed Word, Word displays the Convert File dialog box and asks you to choose a converter. To install additional converters, you need to run the Microsoft Word Setup program again. For online instructions, double-click the Help button on the Standard toolbar, and then type **setup**

Using the List Documents Of Type Box (Windows)

If you don't see the document you want to open in the File Name list, try selecting one of the options in the List Files Of Type box; these options are described in the following table. If you don't know the extension of the document you're looking for, or if it doesn't have an extension, select All Files.

Select	To display
All Files (*.*)	All documents in the selected directory.
Word Documents (.doc)	Word documents and other documents with the .DOC filename extension.
Document Templates (.dot)	Word templates with the .DOT filename extension.
Rich Text Format (.rtf)	Documents with the .RTF filename extension. For more information, see "Opening or Saving a Plain-Text File," later in this chapter.
Text Files (.txt)	Documents with the .TXT filename extension. Includes documents saved as ASCII text or MS-DOS text, including Text Only, Text With Line Breaks, and Text With Layout. For more information, see "Opening or Saving a Plain-Text File," later in this chapter.

locument isn't converted correctly, close it without saving changes and then
onverting again using a different converter. The original document remains
anged until you save it in Word format or in some other format. You can also
ge the way Word converts some items. For more information, see
stomizing Conversions and Improving Compatibility," later in this chapter.

pen a file created in another application

)n the Standard toolbar, click the Open button.

n the List Files Of Type box, select the type of file you want to open. If you
lo not know the type of document or file format, select All Files.

n the File Name box, type or select the document you want to open.

f the document you want to open does not appear in the File Name box, select
different drive, directory, or folder.

Choose the OK button to convert a copy of the document to Word format.

ie converter you need was not installed when you installed Word, Word
lays the Convert File dialog box and asks you to choose a converter. To
ll additional converters, you need to run the Microsoft Word Setup program
n. For online instructions, double-click the Help button on the Standard
bar, and then type **setup**

ng the List Documents Of Type Box (Windows)

ou don't see the document you want to open in the File Name list, try selectin
of the options in the List Files Of Type box; these options are described in th
owing table. If you don't know the extension of the document you're looking
or if it doesn't have an extension, select All Files.

ct	To display
Files (*.*)	All documents in the selected directory.
d Documents (.doc)	Word documents and other documents with the .DOC filename extension.
ument Templates (.dot)	Word templates with the .DOT filename extension.
Text Format (.rtf)	Documents with the .RTF filename extension. For mor information, see "Opening or Saving a Plain-Text File later in this chapter.
t Files (.txt)	Documents with the .TXT filename extension. Includes documents saved as ASCII text or MS-DOS text, including Text Only, Text With Line Breaks, and Text With Layout. For more information, see "Opening or Saving a Plain-Text File," later in this chapter.

In Windows, Word lists documents according to their three-character filename extension. For example, Word documents usually end with .DOC, and plain-text files end with .TXT. When you select a file type from the List Files Of Type box, Word inserts the appropriate filename extension in the File Name box. You can also type extensions directly in the box to see a list of documents from a specific application. For example, type ***.xls** to see a list of Microsoft Excel documents, or type ***.wps** to see a list of word-processing documents from Microsoft Works. To see a list of both types at once type ***.xls; *.wps**. After typing the extensions, press ENTER to display the list.

Note The file format does not necessarily correspond to the filename extension. For example, a WordPerfect document may have a .DOC, .TXT, or .WPS extension, or no extension at all. When you open a file in another format, Word first looks at the contents of the file to determine the file format. If Word doesn't recognize the file format, it tries to use the converters that correspond to the filename extension. If Word is still unable to recognize the file format, it asks you to choose a converter and suggests Text Only.

Using the List Documents Of Type Box (Macintosh)

If you don't see the document you want to open in the File Name box, make sure that you have selected the correct file type for the document in the List Files Of Type box. If you still don't see the document name listed, select either Readable Files (to display files Word can read with the installed converters) or All Files (to display every document in the selected folder, regardless of file type).

Saving a Converted Document

Once you have converted a document from another file format to a Word document by opening it in Word, the converted document resides only in the computer's memory; the original document remains in its original format on disk. When you save the document, Word asks you whether you want to save it in Word format or in the document's original format. If you want to retain copies in both formats, use the Save As command on the File menu and give the document a different name (or filename extension, which is usually used to denote the file format). The document will be saved with the new name in Word format.

For more information, see "Saving and Closing Converted Documents," later in this chapter.

ır on my screen instead of the text effects, equations, or charts that I

.ve selected the Picture Placeholders option. From the Tools menu, ons, and then select the View tab. Make sure that the Picture s check box is cleared.

ıle-click a text effect created in WordArt or an equation I created with itor, the menus and toolbars don't change. Instead, I see the WordArt or itor window.

ıt display the WordArt or Equation Editor toolbars and menus if you :d the Picture Placeholders option. From the Tools menu, choose d then select the View tab. Make sure that the Picture Placeholders s cleared. Also make sure that the zoom setting in the Zoom Control Standard toolbar is 100 percent.

ıe size of a WordArt text effect, equation, or chart. When I double-click it, :turns to its original size.

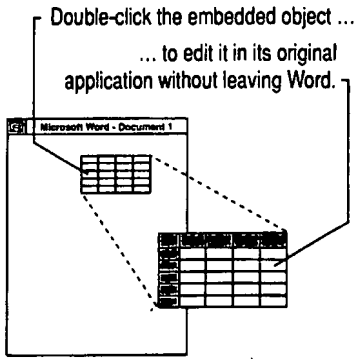
ıde for the object may contain an `\s` switch that tells Word to preserve original size. To remove the switch, choose Options from the Tools :t the View tab, and then select the Field Codes check box. If the field like this: `{EMBED WordArt \s \*mergeformat}`, delete the `\s` to the Options dialog box and clear the Field Codes check box.

CHAPTER 28

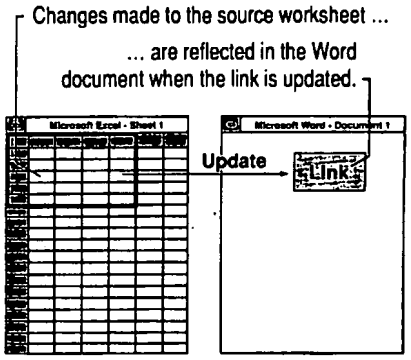
Exchanging Information with Other Applications

For online instructions, double-click the Help button on the Standard toolbar. Then type **embedding** or **linking** or **publishing**

With the linking and embedding features available in Word, you can include information, or *objects*, created in other applications. The main difference between linking and embedding is where the data is stored. Embedded objects become part of the Word document itself. Linked objects, on the other hand, are stored in the source file; the Word document stores only the location of the source file but displays a representation of the linked data.



Worksheet embedded in the Word document itself



Worksheet in a separate file linked to the Word document

If you are using Word for the Macintosh with System 7 or later, you can also use Publish and Subscribe to link a Word document to other files.

In This Chapter

QUICK START

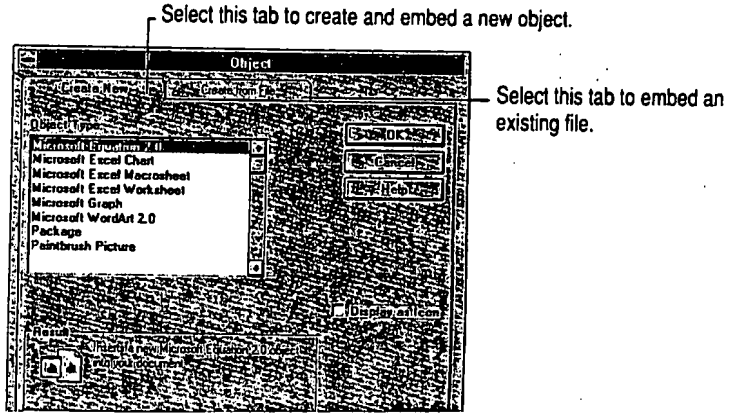
Should I embed an object or link it?

Use these guidelines to decide between embedding and linking information.

When you want to	Use this method	Comment
Include information that becomes part of the Word document and is always available, even if the original source file or the Word document is moved.	Embed the objects from another application in a Word document.	To edit the objects, all relevant applications must be installed on the computer you are using.
Include data maintained in a separate file; the Word document reflects any changes made to that file.	Create a link in the Word document to the source file.	
Include a very large file, such as a video clip or sound clip	Create a link in the Word document to the source file.	Word can store just the link; this keeps the size of the Word document manageable.
Include a file that may not always be available, such as a file stored on a network server	Embed the file from another application in a Word document.	

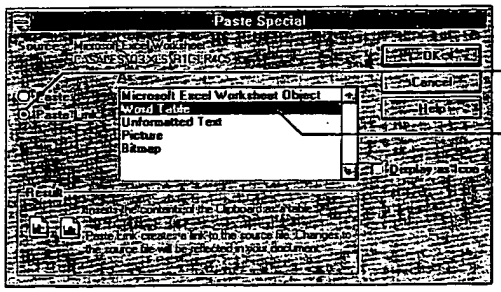
Embedding an Object

Position the insertion point where you want to embed the object. From the Insert menu, choose Object. Then, to create and embed a new object, select the Create New tab; select the type of object you want, and then choose the OK button. To embed an entire existing document, select the Create From File tab; select the filename, and then choose the OK button.



Creating a Link

Start an application that supports OLE or DDE, and then open the source file. Select the information to which you want to create a link, and then choose Copy from the source application's Edit menu. In the Word document, position the insertion point where you want the linked object, and then choose Paste Special from the Edit menu. Select the Paste Link option button, select the type of linked object in the As box, and then choose the OK button.



The Paste Special dialog box

Editing an Embedded Object

Double-click the object to edit it. Some applications open a separate window for editing. Others temporarily replace some of the Word menus and toolbars with those of the source application. If the application opens a separate window, you return to Word by choosing Exit from the application's File menu. If the application replaces the Word menus and toolbars, you return to Word by clicking outside the embedded object.

Editing a Linked Object

To edit the linked object itself, select the object and then choose the object's name from the Edit menu. For example, if you select a link to a Microsoft Excel worksheet, the command on the Edit menu is Microsoft Excel Worksheet object. When you choose the command and then choose Edit from the submenu that appears, the source application opens for editing. To update, reconnect, or break a link, choose Links from the Edit menu, make the changes, and then choose the OK button.

See the following pages for detailed information.

Embedding Objects

For an example of embedding an object, see "Examples of Embedding and Linking," later in this chapter.

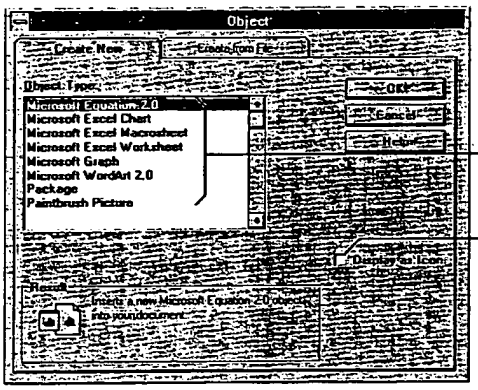
Embedding means inserting information, such as a chart, an equation, or spreadsheet data, in a Word document. Once embedded, the information—called an *object*—becomes part of the Word document. When you double-click an embedded object in Word, you open the application in which the object was created; the object is displayed and ready for editing. When you return to Word, any changes you've made to the object are reflected in the Word document.

Without leaving Word, you can embed an existing file or create and embed a new object. An embedded object increases the file size of a Word document, because the object is stored in the Word document. For example, if you embed a Microsoft Excel worksheet in a Word document, the file size of the Word document increases by approximately the file size of the worksheet. The file size increases even if you display the object as an icon in the Word document, because Word still stores the information about the file.

Creating an Embedded Object

For more information, see "Embedded Objects and Links Are Represented by Fields," later in this chapter.

Suppose you want to create a chart in a Word document. Position the insertion point where you want the chart, and then choose Object from the Insert menu. Word displays a list of the types of objects you can create and embed. Select Microsoft Graph in the Object Type box, and then choose the OK button. Word opens Microsoft Graph, in which you can create a chart. When you quit Microsoft Graph, the chart is displayed in your Word document. When you want to edit the chart, double-click it to start Microsoft Graph; you can then make changes to the chart.



You use the same process to create any other type of object, such as an equation, a Microsoft Excel worksheet, or even a Word object. The only difference is in the type of object you select in the Object dialog box. Whatever type you select, Word opens the appropriate application for creating the object.

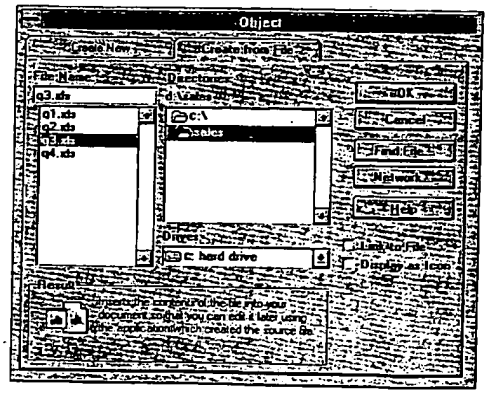
Note The applications you use to create embedded objects must have been properly installed by using their original installation programs; otherwise, they may not be listed in the Object dialog box. Supplemental applications supplied with Word (WordArt, Equation Editor, and Graph) must be installed by using the Microsoft Word Setup program.

In addition to creating embedded objects as you work, you can embed existing files or parts of files. If you embed an existing file, Word stores an independent copy of the file in the Word document. The original file remains unchanged, even if you change the embedded file. Changes to the original file don't affect the Word document.

Microsoft Excel icon

You can display an embedded object as an icon by selecting the Display As Icon check box in the Object dialog box when you insert the object. You might want to use this option when the Word document will be read on-screen and the embedded object contains supplementary information. For example, if the Word document is a summary of financial data from several Microsoft Excel worksheets, you might want to embed one of the worksheets as an icon next to the paragraph in the Word document analyzing that portion of the data. If readers want to see the data, they can double-click the icon to open the worksheet. If you print the document, the icon is printed at its current position in the Word document. Some objects created from a file—for example, ASCII text files—are always displayed as icons. You can change the icon by choosing the Change Icon button, which appears when you select the Display As Icon check box.

Note If the source application supports drag-and-drop editing, you can embed an object by selecting information in the source application and then dragging it into the Word document.



The Object dialog box (Create From File tab)

▶ To embed an object

1. Position the insertion point where you want to embed the object.
2. From the Insert menu, choose Object.
3. Do one of the following:
   - To embed a new object, select the Create New tab. In the Object Type box, select the type that describes the application in which you want to create the object, and then choose the OK button.

     The contents of the list depend on which applications installed on your computer support linking and embedding.
   - To embed an existing file, select the Create From File tab. In the File Name box, type or select the name of the file you want to embed, and then choose the OK button.

     If you do not see the file that you want to embed, select a different drive, directory, or folder, or choose the Find File button to search for the file you want.
4. To display the object in the Word document as an icon instead of as the object itself, select the Display As Icon check box.
5. To return to Word, do one of the following:
   - If the object was created in another application that is in a separate window, choose Exit from the File menu in that application. If a message appears asking if you want to update the document, choose the Yes button.
   - If the application temporarily replaces some of the Word menus and toolbars, click anywhere outside the embedded object.

Note An embedded object increases the file size of a Word document because the object is stored in the Word document. If you want to reduce the file size, see "Converting an Embedded Object to a Graphic," later in this chapter.

▶ To embed a selection from an existing file

1. In Word, position the insertion point where you want to embed the selection.
2. Switch to the source application, and then open the file from which you want to select information to embed in the Word document.
3. Select the information you want to embed in the Word document.
4. From the Edit menu in the application the selection was created in, choose Copy.

5. Switch to Word, and then choose Paste Special from the Edit menu.
6. Select the Paste option button.
7. In the As box, select the first item with the word "Object" in its name.

   To display the embedded information as an icon in the Word document, select the Display As Icon check box.
8. Choose the OK button.

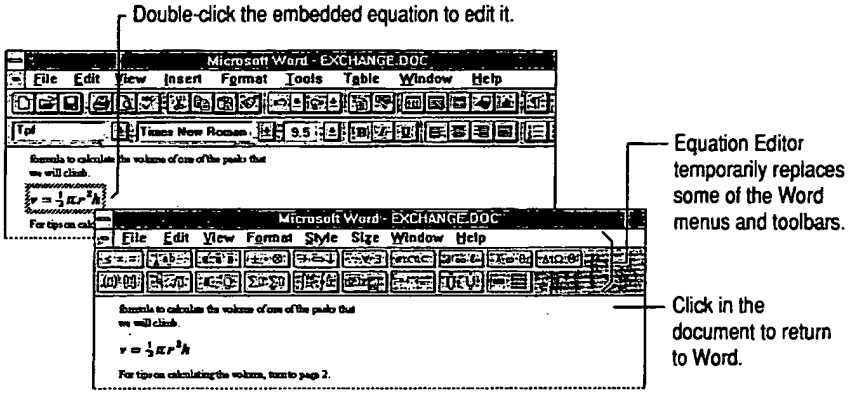
When you embed an object, Word adds an {EMBED...} field to the document. If you see this field instead of the object you embedded, select the field and press SHIFT+F9, or choose Options from the Tools menu, select the View tab, and then clear the Field Codes check box.

For information about field codes, see Chapter 32, "Inserting Information with Fields."

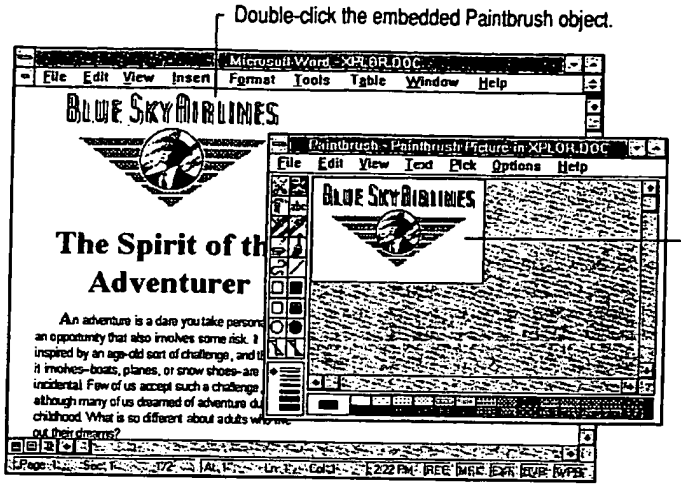
Editing an Embedded Object

In most cases, double-clicking an embedded object opens the application in which it was created. Note, however, that double-clicking some graphics brings up the Drawing toolbar rather than the original application.

For certain applications, some of the Word menus and toolbars are temporarily replaced by those of the original application. For other applications, a separate window opens, which contains the object in its original application.



Some applications, such as Equation Editor, temporarily replace the Word menus and toolbars for editing.



Some applications open a separate window for editing.

▶ To edit an embedded object directly

1. Double-click the embedded object.
2. Edit the object.
3. Do one of the following:
   - If you are editing the object in a separate application window, choose either Exit or Quit from the File menu to return to Word.
   - If you are editing the object in an application that temporarily replaces the Word menus and toolbars, click anywhere outside the embedded object to return to Word.

Tip You can use a shortcut menu to carry out common commands for editing an embedded object. Click the embedded object using the right mouse button (Windows) or hold down CTRL and click the embedded object (Macintosh), and then choose a command from the shortcut menu.

▶ To edit an embedded object by using the Object command

1. Select the object you want to edit.
2. From the bottom of the Edit menu, choose the name of the object you want to edit, and then choose Edit.

   If you see both an Edit command and an Open command, choose Open to edit the object in its own application window, or choose Edit to edit the object in the Word window (the Word toolbars and menus may change).

Note Some embedded objects, such as video and sound clips, play when you double-click them, instead of opening an application for editing. To edit one of these objects, select it, choose the name of the object from the Edit menu, and then choose Edit.

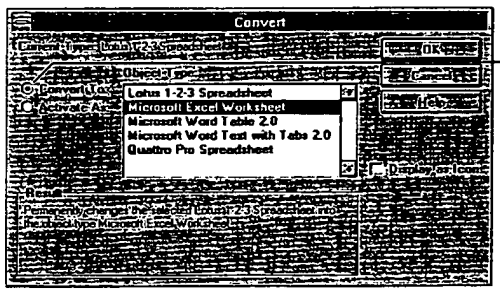
Converting an Embedded Object to a Different Format

Suppose you receive a Word document that has a Microsoft Excel worksheet embedded in it. Microsoft Excel is not installed on your computer, but you do have another spreadsheet application. When you double-click the Microsoft Excel worksheet to edit it, Word displays an error message saying that the Server could not be found. Select the worksheet, choose the Object command from the Edit menu, and then choose Convert. In the Convert dialog box, you can specify which application you'll use in the future to edit the object. The applications displayed in the list are the applications installed on your computer that are capable of converting the object to their respective file formats.

If you select the Convert To option, Word converts the embedded object to the format you choose in the Object Type box. The object remains in this format unless you specifically convert it to another format.

Now suppose you have to return the Word document to the person who created it. You know that this person uses Microsoft Excel; therefore, you don't want to permanently convert the embedded object to a different file format. However, you do want to be able to edit the embedded object on your computer. By selecting the Activate As option in the Convert dialog box, you can specify that all embedded Microsoft Excel objects are to be temporarily converted to the file format you specify. When you edit the object in question, you use the new application, but changes are saved in Microsoft Excel format. When you double-click the object, it is temporarily converted to the new format.

If you select the Activate As option, Word converts all embedded objects of the selected type to the format you specify in the Object Type box. You can edit the objects in the application you specify, but Word saves changes to the objects in their original file format.



Select this option button to convert an embedded object to a different file format.

▶ **To convert an embedded object to a different file format**

1. Select the embedded object whose source application you want to change.
2. From the Edit menu, choose the name of the object you want to convert, and then choose Convert.
3. Do one of the following:
   - To permanently convert the embedded object to the file format you specify in the Object Type box, select the Convert To option button.
   - To activate all embedded objects of the selected type in the file format you specify in the Object Type box, select the Activate As option button.
4. In the Object Type box, select the application whose file format you want to convert the embedded object to, and then choose the OK button.

Note If you upgrade to a more recent version of an application after creating an embedded object with that application and then want to edit the object, you must first convert the object to the current version of the application.

Some applications can convert files created in older versions of the application to the newer version when you open the file. When you run the application's setup program, you may be able to specify whether files are converted when you open them.

Converting an Embedded Object to a Graphic

All embedded objects are represented in Word documents as graphics. You can think of one of these graphics as a facade with information behind it. For example, a Microsoft Excel worksheet embedded in a Word document looks like an ordinary worksheet, but it is actually a graphic, or picture, of the original worksheet. Behind the graphic is all of the information necessary to open and edit the file in Microsoft Excel. When you convert the Microsoft Excel object to a graphic, you remove the information behind the facade—that is, the information that enables Word to open the object in Microsoft Excel for editing.

Converting an embedded object to a graphic reduces the file size of the Word document. If you double-click the object after converting it to a graphic, Word opens a separate window and displays the Drawing toolbar so that you can edit the graphic just as you would edit any other graphic.

▶ **To convert an embedded object to a graphic**

1. Select the object you want to convert.
2. From the Edit menu, choose the name of the object you want to convert, and then choose Convert.

3. In the Object Type box, select Picture, and then choose the OK button.

Using the keyboard You can convert objects to graphics quickly by using shortcut keys. Select the embedded object, and then press CTRL+SHIFT+F9 (Windows) or COMMAND+SHIFT+F9 (Macintosh). This method does not create an {Embed} field.

Linking to Another File

Let's say you are preparing a monthly report for the accounting department that must include up-to-date sales data. The sales department maintains this constantly changing data in a Microsoft Excel worksheet. You can link your Word document to the sales worksheet (or a portion of it) and specify that your document be automatically updated if the worksheet changes. Each time the sales department changes the worksheet, the changes are reflected in your Word document.

When you link to another file, Word stores the link in the form of a field code that indicates the source of the object. In addition, Word usually stores a visual representation of the linked information.

You can create links between two Word documents or between a Word document and a file created in another application. Once you have established a link, you can update it with a single keystroke, or you can specify that the data be updated in your Word document as soon as it changes in the source file.

For information about fields, see Chapter 32, "Inserting Information with Fields."

Embedded Objects and Links are Represented by Fields

Linked and embedded objects in Word are represented by fields. If you are working with field codes displayed, you don't see the linked or embedded object itself; you see the code that Word uses to designate the object. For example, the code for a link to a Microsoft Excel worksheet might look like the following in a Word for Windows document:

```
{LINK ExcelWorksheet "C:\\EXCEL\\SALES.XLS" "R1C1:R9C5" \a \p}
```

or like the following in a Word for the Macintosh document:

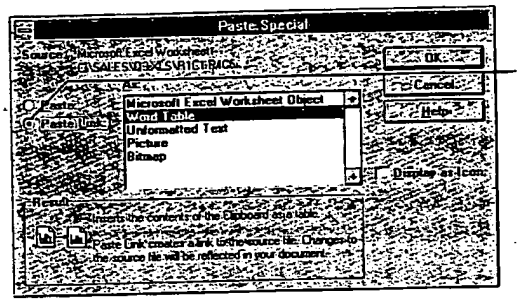
```
{LINK ExcelWorksheet "Hard Drive:Excel:Sales" "R1C1:R9C5"}
```

To specify whether the object itself or the field code is displayed, choose Options from the Tools menu, select the View tab, and then select or clear the Field Codes check box.

Creating a Link

Creating a link is as easy as copying and pasting. You copy a selection from a file (the *source*) and paste it into your Word document (the *destination*) by using the Paste Special command on the Edit menu. Before you can establish a link, you must save the source file to disk.

Important To create a link between a Word document and another application, you must be running both applications, and the other application must support dynamic data exchange (DDE) or object linking and embedding (OLE). On the Macintosh, you must be running both applications with System 7 or later, and your computer must have enough memory to run both applications at the same time.



The Paste Special dialog box

▶ **To create a link to another file or Word document**

1. Make sure that you save the source file before you link the information.
2. In the application in which the information you want to link was created, open the source file and then select the information you want to link.
3. From the Edit menu, choose Copy.
4. Switch to the Word document, and then position the insertion point where you want to insert the linked information.
5. From the Edit menu, choose Paste Special.
6. Select the Paste Link option button.
7. Under As, select the format you want, and then choose the OK button.

▶ **To create a link to another file or Word document without leaving Word**

1. From the Insert menu, choose Object.
2. Select the Create From File tab.
3. In the File Name box, type or select the name of the file to which you want to link.
4. Select the Link To File check box, and then choose the OK button.

Using this method, you can create a link only to an entire file; you cannot link to a selection in a file.

Note Word creates automatic links by default. Word updates automatic links each time you open the Word document, whereas it updates manual links only when you specify. For information, see "Updating a Link," later in this chapter.

Reducing File Size of Documents Containing Linked Graphics

When a Word document contains links to graphics files, you can reduce the file size of the Word document by storing only the links. By default, Word stores in the Word document a "picture" of the linked graphic, which increases the file size of the Word document by the number of bytes taken up by the picture. To reduce the size of the Word file, you can specify that Word store only the link itself and not the picture.

If you store only the link, the file size of the Word document will not increase appreciably. However, if the source file is not available, you will see only a rectangular placeholder in your document and the linked data will not be printed. If the source file is available, Word displays a picture of the object based on data in the source file, but it does not store the picture in the Word document. Because the picture is created from the source file itself, you may notice that it takes longer to display the picture than if it were stored in the Word document.

If the picture is stored in the Word document, Word displays a picture of the linked graphic regardless of whether or not the source file is available.

▶ **To reduce the file size of a document containing linked graphics**

1. From the Edit menu, choose Links.
2. In the Links dialog box, select the link or links to the graphics files.
3. Clear the Save Picture In Document check box, and then choose the OK button.

Reconnecting or Changing a Link

You may lose a link if you rename or move the source file. If this happens, you must reconnect the link to the original source file or redirect the link to a different file.

▶ **To reconnect or change a link**

1. From the Edit menu, choose Links.
2. From the list, select the link you want to reconnect or change. To select multiple links, hold down CTRL (Windows) or SHIFT (Macintosh) while you click each link.
3. Choose the Change Source button.
4. In the File Name box, type or select the name of the file to which you want to reconnect or change the link, and then choose the OK button.

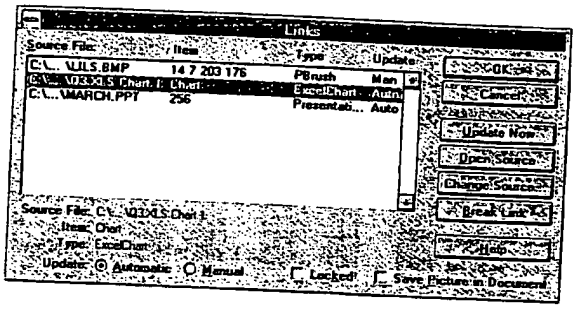
   If you do not see the file you want to open, select a different drive, directory, or folder, or choose the Find File button to search for the file you want.

If you have other links to the same source file, Word asks you to confirm that you want to change all links from the previous source file to the new source file.

Updating a Link

When information in the source document changes, Word can update the information in the Word document. You can specify either manual or automatic updating for each link. By default, newly created links are set to automatic updating, but you can easily change a link to manual updating.

- Word updates automatic links when you open the Word document and any time the source document is changed while the Word document is open.
- Word updates manual links only when you choose the Update Now button in the Links dialog box (Edit menu) or when you position the insertion point in the linked object and press F9.



The Links dialog box

If you change any editable text or numbers in the linked object while working in the Word document, the changes will be overwritten when Word updates the linked material. You can, however, apply formatting—such as bold or italic format or centered paragraph alignment—in the linked object. Word retains such formatting and reapplies it to the text or numbers when they are updated.

▶ To control how links are updated

1. From the Edit menu, choose Links.
1. From the list, select the link to the information you want to update. To select multiple links, hold down CTRL (Windows) or SHIFT (Macintosh) while you click each link.
3. Do one of the following:
   - To update linked information every time there's a change in the source file, select the Automatic option button next to Update.
   - To update linked information only when you choose, select the Manual option button next to Update.
4. Choose the OK button.

▶ To update a link manually

1. From the Edit menu, choose Links.
2. From the list, select the link to the information you want to update. To select multiple links, hold down CTRL (Windows) or SHIFT (Macintosh) while you click each link.
3. Choose the Update Now button.

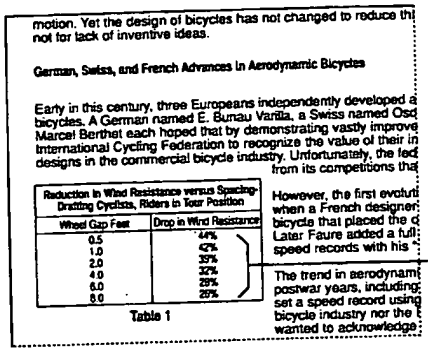
   For each selected link, the destination document reflects any changes made in the source file since the last update.

Examples of Embedding and Linking

The following two scenarios illustrate common uses of embedding and linking.

Embedding a Microsoft Excel Worksheet in Word

Embedding is a way to get fast access to the features of another application. In the following example, a table illustrates the relationship between various bicycle designs and the drop in wind resistance. You can create a Word table to present the data. However, if you embed a Microsoft Excel worksheet instead, you can use the more powerful Microsoft Excel formulas to calculate the results you want to present.


motion. Yet the design of bicycles has not changed to reduce thi
not for lack of inventive ideas.

German, Swiss, and French Advances in Aerodynamic Bicycles

Early in this century, three Europeans independently developed a
bicycles. A German named E. Bunau Varilla, a Swiss named Osc
Marcel Berthet each hoped that by demonstrating vastly improve
International Cycling Federation to recognize the value of their in
designs in the commercial bicycle industry. Unfortunately, the fed
from its competitions tha

Reduction in Wind Resistance versus Spacing-
Drafting Cyclists, Riders in Tour Position

Wheel Gap Feet	Drop in Wind Resistance
0.5	44%
1.0	42%
2.0	35%
4.0	32%
6.0	29%
8.0	25%

Table 1

However, the first evolut
when a French designer
bicycle that placed the c
Later Faure added a full
speed records with his

The trend in aerodynami
postwar years, including
set a speed record using
bicycle industry nor the
wanted to acknowledge

Use a formula in Microsoft Excel to calculate the data.

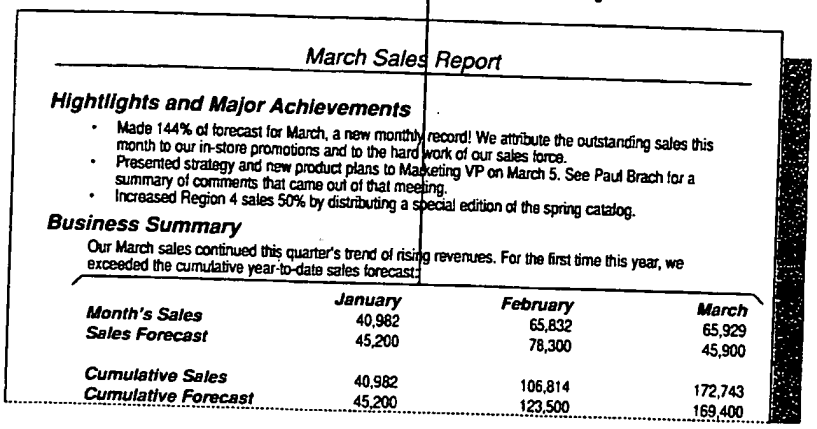
You can also click the Insert Microsoft Excel Worksheet button on the Standard toolbar to embed a worksheet.

To create and embed the worksheet, position the insertion point where you want to embed the worksheet, and then choose Object from the Insert menu. Select the Create New tab, select Microsoft Excel Worksheet in the Object Type box, and then choose the OK button. Microsoft Excel opens, and you can now create the worksheet. When you have finished, choose Exit from the File menu in Microsoft Excel. A message appears, asking if you want to update the worksheet in your document; if you choose the Yes button, the worksheet is inserted into the Word document. You can double-click the worksheet object in the Word document at any time to open Microsoft Excel and make changes.

Linking a Microsoft Excel Worksheet to a Word Document

Linking is a good way to make use of information that's stored and updated in other files. The following example shows a monthly sales report that contains data for a three-month period. The data is maintained by the sales department in a large Microsoft Excel worksheet that also contains a lot of other data. By creating a link to a specific section of the worksheet, you give yourself immediate access to the most recent data needed for the monthly report. If the sales department changes the worksheet, the Word document can be automatically updated.

The sales data can be automatically updated when the original Microsoft Excel worksheet is changed.



March Sales Report

Hightlights and Major Achievements

- Made 144% of forecast for March, a new monthly record! We attribute the outstanding sales this month to our in-store promotions and to the hard work of our sales force.
- Presented strategy and new product plans to Marketing VP on March 5. See Paul Brach for a summary of comments that came out of that meeting.
- Increased Region 4 sales 50% by distributing a special edition of the spring catalog.

Business Summary

Our March sales continued this quarter's trend of rising revenues. For the first time this year, we exceeded the cumulative year-to-date sales forecast.

	January	February	March
Month's Sales	40,982	65,832	65,929
Sales Forecast	45,200	78,300	45,900
Cumulative Sales	40,982	106,814	172,743
Cumulative Forecast	45,200	123,500	169,400

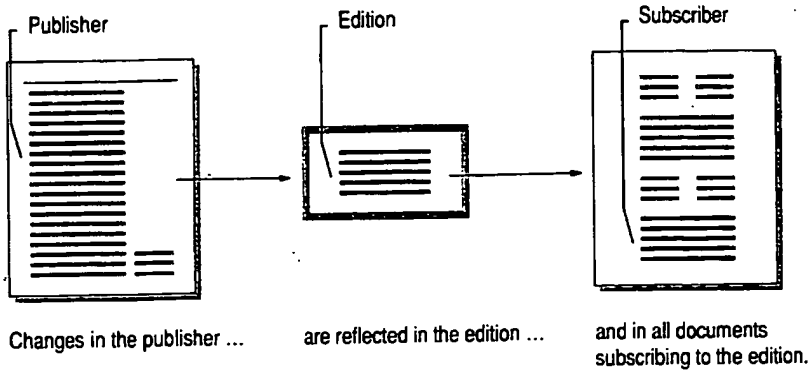
To create the link, open the sales department's Microsoft Excel worksheet, select the data you want to include in the report, and then choose Copy from the Edit menu. Open the monthly report in Word, position the insertion point where you want to insert the data, and then choose Paste Special from the Edit menu. In the Paste Special dialog box, select the Paste Link option button, select Microsoft Excel Worksheet Object in the As box, and then choose the OK button. The worksheet data appears in the Word document. By default, any changes made to the worksheet in Microsoft Excel will automatically appear in the Word document. However, you can prevent automatic updates by choosing Links from the Edit menu, selecting the link, selecting the Manual option button at the bottom of the dialog box, and then choosing the OK button.

You use this same process to link other items—such as graphics or Microsoft Excel charts—to the Word document

Publishing and Subscribing in Word for the Macintosh

With Word for the Macintosh running under System 7 or a later version, you can exchange information between documents in different applications or on different computers connected by a network. To make part of a document available for use with other applications or for other users on a network, you can create a *publisher* for that portion of the document. A publisher contains the part of the document you want to share—text, graphics, spreadsheet data, and so forth.

When you create a publisher, Word automatically creates an intermediate file called an *edition*, which contains a copy of the information that's in the publisher. An edition is a separate file that can be saved on a hard disk or a network server. When you change information in the publisher, these changes are reflected in the edition and in all documents *subscribing* (similar to linking) to the edition. You can specify how often Word sends updated information from the publisher to the edition. You can also specify how often subscribers receive updated information from the edition.



You can name and move an edition just as you would any other file. The connection is maintained even if you change the name of the edition. However, if you want Word to maintain the connection to all publishers and subscribers, do not move the edition off the server volume or hard disk.

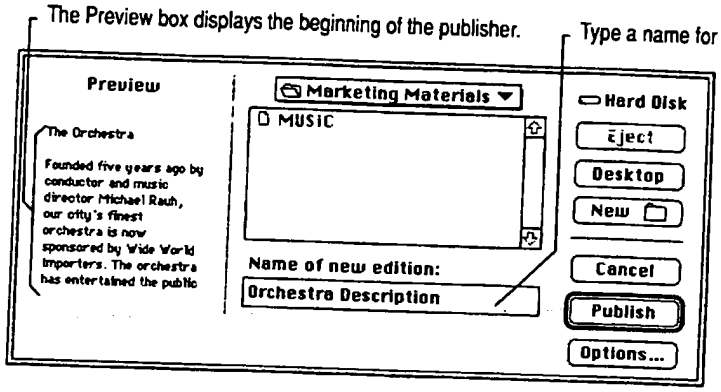
You can publish and subscribe from any application that runs under System 7 or later and supports publishing and subscribing.

Creating a Publisher and an Edition

Use the Create Publisher command to publish information from a Word document. You can edit a publisher the same way you edit other parts of a document. When you create a publisher, Word puts gray brackets around the part of the document that constitutes the publisher. You can see the brackets when you click the Show/Hide ¶ button on the Standard toolbar, but the brackets are not printed.

For information about bookmarks, see Chapter 19, "Cross-references, Captions, and Bookmarks."

Word uses a bookmark to mark a publisher. The name you give the edition when you create a publisher is the name Word uses for the bookmark. If there is already a bookmark by that name, Word adds a number to the end of the bookmark name to make it unique. If you delete the bookmark, the publisher is deleted as well.



The Create Publisher dialog box

▶ To create a publisher

1. Select the information you want to publish.

   If you publish an entire document, including the final paragraph mark, Word includes any material that's added to the document later as part of the publisher. To add information you do not want to publish to the end of the document, exclude the final paragraph mark from the selected information.
2. From the Edit menu, choose Publishing, and then choose Create Publisher.
3. Switch to the disk or open the folder in which you want to store the edition.

   If you want to store the edition on another Macintosh, choose the Desktop button. Word displays all of the available hard disks and servers. Select the computer you want, and then select the folder you want from the list.
4. In the Name Of New Edition box, type a name for the edition.
5. Choose the Publish button.

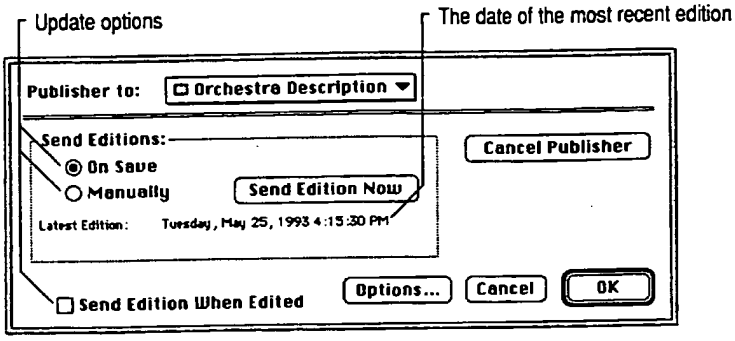
   Word encloses the published information in brackets.

   Each time you create a new publisher, Word creates a new edition. If you publish several parts of the same document, each publisher has its own separate edition.

Note If you want to make editions available to other computers on a network, you must store the editions on your hard disk and use the file sharing option; make sure to share the folder that contains the editions. To share files on the Macintosh under System 7, choose Control Panels from the Apple menu, double-click the Sharing Setup icon, and then choose the Start button under File Sharing. For more information, see your Macintosh system documentation.

Updating an Edition

Once you have created a publisher and an edition, you can specify how frequently you want to update the edition with changes you make to the publisher. Unless you specify otherwise, Word updates the edition as soon as you save changes to the publisher. There are three update options in the Publisher Options dialog box: On Save, Manually, and Send Edition When Edited. These options are described in the procedure following this illustration.



The Publisher Options dialog box

▶ To control how an edition is updated

1. Select the publisher whose update frequency you want to change.
2. From the Edit menu, choose Publishing, and then choose Publisher Options.
3. Under Send Editions, do one of the following:

To update the edition	Do this
Whenever you save the publisher	Select the On Save option button.
Only when you choose the Send Edition Now button	Clear the Send Edition When Edited check box, and then select the Manually option button.
Whenever you make changes to the publisher	Select the On Save option button, and then select the Send Edition When Edited check box.

4. Choose the OK button.

▶ **To send an edition manually**

1. Select the publisher you want to update.
2. From the Edit menu, choose Publishing, and then choose Publisher Options.
3. Choose the Send Edition Now button.

Canceling a Publisher

If you decide that you no longer want to publish information in your document, you can cancel the publisher. The contents of the publisher remain in your document. Other users can still subscribe to the edition, but it is no longer updated. To delete the edition, delete it in the Finder just as you would delete any other file.

▶ **To cancel a publisher**

1. Position the insertion point in the publisher you want to cancel.
2. From the Edit menu, choose Publishing, and then choose Publisher Options.
3. Choose the Cancel Publisher button.

   Word asks you to confirm that you want to cancel the publisher. When you cancel a publisher, Word deletes the brackets surrounding the publisher.

Subscribing to an Edition

When you subscribe to an edition, you insert a copy of the edition into your Word document. This copy is called the subscriber. Once you have inserted a subscriber into the Word document, updates received by the edition are automatically sent to the subscriber. As long as the edition remains on the same server volume or hard disk, Word maintains the connection between the edition and the subscriber, even if you change the name of either the edition or the subscriber.

▶ **To subscribe to an edition**

1. Position the insertion point in the document where you want to insert a copy of the edition.

   Make sure that the insertion point is not positioned in another subscriber in the document.

2. From the Edit menu, choose Publishing, and then choose Subscribe To.

3. From the list of files, select the edition you want to subscribe to.

   If the edition is located on another hard disk or another computer, choose the Desktop button. Word displays all available hard disks and computers. Select the hard disk or machine you want, and then select the edition from the list.

4. In the Subscribe With box, select the format you want to use for the subscriber data.
5. Choose the Subscribe button.

Updating a Subscriber

When you've subscribed to an edition, you can specify how frequently you want to receive updated information from the edition. Unless you select another option, Word updates the subscriber automatically as soon as a new edition is available—that is, any time the publisher sends changed information to the edition.

▶ **To control how a subscriber is updated**

1. Select the subscriber whose update frequency you want to change.
2. From the Edit menu, choose Publishing, and then choose Subscriber Options.
3. Under Get Editions, do one of the following:

To update the subscriber	Choose
Whenever a change is made in the edition	The Automatically button
Only when you specify	The Manually button

Note Changing the update frequency of the subscriber affects only how often the subscriber receives updates from the edition, not how frequently the publisher sends updates to the edition. You set the update frequency of the publisher independently. For information, see "Updating an Edition," earlier in this chapter.

▶ **To update a subscriber manually**

1. Select the subscriber you want to update.
2. From the Edit menu, choose Publishing, and then choose Subscriber Options.
3. Choose the Get Edition Now button.

Note You can format a subscriber, but Word overwrites the subscriber each time it receives an updated edition as long as the * MERGEFORMAT switch remains in the field. For more information, double-click the Help button on the Standard toolbar and type **format (*) switch**

Switching from a Subscriber to Its Publisher

If you need to change the contents or formatting of a subscriber, it's best to make the changes in the publisher itself. This way, the changes will be reflected in the subscriber. If you are connected to a network, you must have access to the publisher to perform this procedure.

▶ **To switch from a subscriber to its publisher**

1. Select the subscriber you want to edit.
2. From the Edit menu, choose Publishing, and then choose Subscriber Options.
3. Choose the Open Publisher button.

   Word opens the document that contains the publisher you want.
4. Make the changes in the publisher.

   When you finish making changes in the publisher, save and close the document. Each subscriber reflects the changes according to the update options you've selected in the publisher and subscriber.

Canceling a Subscriber

If you do not have access to the publisher to make changes and you do not need to receive any more updates from the edition, you can cancel the subscriber. You can then edit the information as you would edit any other text, without losing any changes when updates are sent. The contents of the subscriber remain in the document.

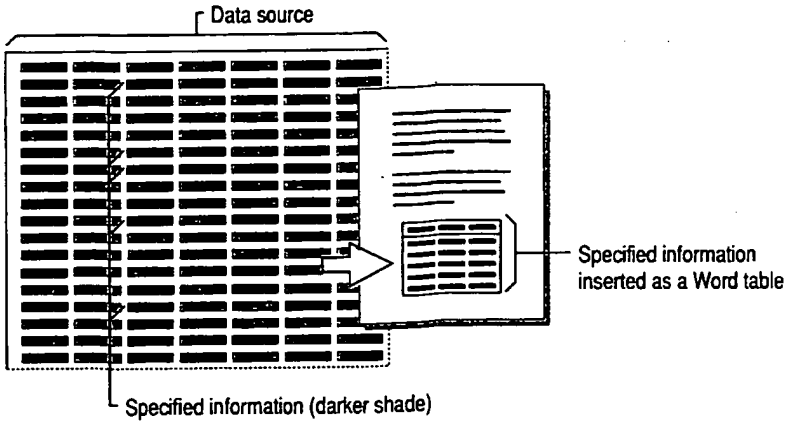
▶ **To cancel a subscriber**

1. Select the subscriber you want to cancel.
2. From the Edit menu, choose Publishing, and then choose Subscriber Options.
3. Choose the Cancel Subscriber button.

   Word asks you to confirm that you want to cancel the subscriber. When you cancel a subscriber, Word deletes the brackets surrounding the subscriber.

Inserting Tables of Information from a Database

Sometimes you may want to include in a Word document information from an existing database, a Microsoft Excel worksheet, or another source of data. By using the Database command on the Insert menu, you can specify the information you want and automatically insert it as a table in a Word document. You can screen, or "filter," the information according to criteria you select. You can also instruct Word to update the information in the Word document if the source file has changed.



Word can retrieve information from the following types of files:

- Files from the following applications that are installed on your system:

  Microsoft Access®
  Microsoft Excel

- Files from single-tier, file-based database applications for which you have an open database connectivity (ODBC) driver installed in the System subdirectory of your Windows directory. ODBC drivers for the following applications are supplied with Word:

  Microsoft Access
  Microsoft FoxPro® (or other Xbase database application such as dBASE®)
  Paradox®

For a list of file converters provided with Word, see Chapter 26, "Converting File Formats."

- Files for which you have a file converter installed. In addition to converters for ASCII text files, Word provides file converters for many applications, including:

Microsoft Word for Windows	WordPerfect 5.*x* for MS-DOS and Windows
Microsoft Word for the Macintosh versions 3.*x*,[1] 4.*x*, and 5.*x*	Microsoft Excel 2.x,[2] 3.0, 4.0,[1] and 5.0[3]
Microsoft Word for MS-DOS 3.0–6.0	Lotus 1-2-3 2.*x*[2] and 3.*x*[1]

[1] Converts only from this format.

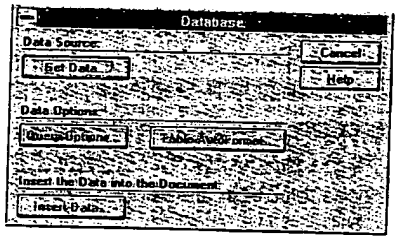
[2] Converter works only with Windows version.

[3] Converter works only with Macintosh version.

You can also insert information from another Word document. For example, you might have set up a membership directory for use as a mail merge data source. Instead of copying and pasting information from various data records, use the Database command to insert just the information you request.

Inserting the Data

When you choose the Database command from the Insert menu, Word displays the Database dialog box. Now you can locate the data source, select the information you want, and format the table in which the information is displayed.



Once you select the data source, the other buttons in the dialog box become available.

By default, Word inserts all of the information from the selected data source. In most cases, however, you'll want to use only some of the available information. For example, from a large personnel file, you might want to list only the names, departments, and hiring dates of all employees who have worked for your company 10 years or longer.

If information in the data source changes frequently and you want to keep your document up to date, you can insert the information as a Word *field*. The field is simply a "placeholder" that represents the table in your document. For more information, see "Keeping the Table Information Up to Date," later in this chapter.

▶ To insert information from a data source as a table

1. Position the insertion point where you want the new table of information to be included.
2. From the Insert menu, choose Database.
3. Choose the Get Data button.
4. In the Open Data Source dialog box, type or select the filename of the data source you want to open, and then choose the OK button.

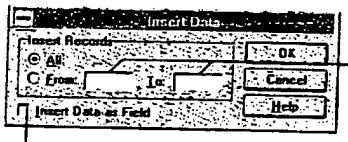
   If the data source is not listed, select the appropriate drive and directory or folder. Then select the appropriate option in the List Files Of Type box.

   If you open a Microsoft Excel worksheet, you can insert the entire worksheet or a range of cells. If you open a Microsoft Access database, you can insert records from a table or a selection of records defined by a query. For more information, see the documentation for the application you are using.
5. To insert specific information from the data source or list the information in a particular order, choose the Query Options button. Do one or more of the following, and then choose the OK button.
   - On the Filter Records tab, specify criteria to select the data records to insert.
   - On the Sort Records tab, select the data fields by which you want to sort the information.
   - On the Select Fields tab, remove any fields you don't want from the Selected Fields list. The order of the fields in the list determines the order in which the fields are inserted left to right.

     If you don't want to insert the field names from the header row with the data records, clear the Include Field Names check box.
6. To format the table, choose the Table AutoFormat button.
7. Choose the Insert Data button.

   In the Insert Data dialog box, you can specify the range of records you want to insert. The range refers only to the records that were selected by the query. If you want to be able to update the information in the table automatically, select the Insert Data As Field check box. Then choose the OK button.

Note If you insert more than 31 data fields, Word inserts tab characters to separate the columns of information.



To specify which of the selected records are inserted, type starting and ending record numbers in the From and To boxes.

Select this check box if you want to keep the table information up to date.

Modifying the table format If you don't select the Insert Data As Field check box, Word inserts the information as an ordinary table. You can resize the table columns and otherwise modify the table by using the commands on the Table menu. If you insert the information as a field, however, you must choose the Database command again to reinsert the table and update the table format by choosing the Table AutoFormat button. Otherwise, the table formatting you've applied is removed the next time you update the DATABASE field. Formatting you've applied to text in the table is also removed. For more information, see "Keeping the Table Information Up to Date," later in this chapter.

Modifying the information in the table You may want to modify the information in the table later. For example, you might want to include another column of information or select a different set of records from the data source. To do this, click in the table and then choose the Database command again to select the information you want in the table. If you insert the information as a field and then edit or format the text in the displayed table, your changes will be deleted the next time you update the DATABASE field. For more information, see "Keeping the Table Information Up to Date," later in this chapter.

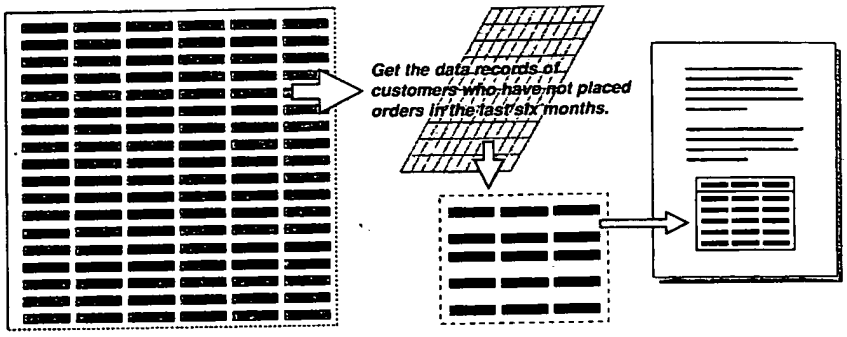
If Word can't recognize the field and record delimiters If Word can't recognize the characters used to separate data fields and data records in text-delimited files (files in which data is separated by commas, tab characters, or other characters), Word asks you to select the separating characters (delimiters). Word recognizes one data field delimiter and one data record delimiter. If a combination of two or more characters is used as a delimiter, then the remaining characters are treated as text in the data fields.

Selecting the Data

To get only the information you want from a data source, you create a *query*. A query is simply a set of instructions, or rules, that describes the information you want from the data source. You can think of the following statement as a query:

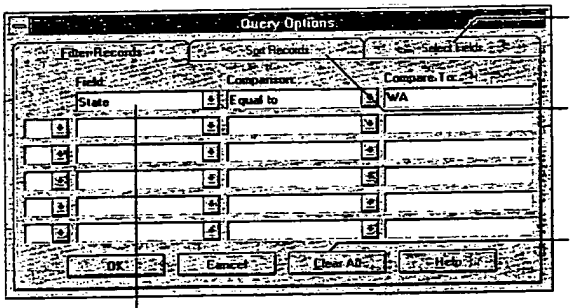
"Give me the names, addresses, and account numbers of all customers who have not placed orders in the last six months."

The first part of the statement identifies the categories of information you want—names, addresses, and account numbers. The second part of the statement indicates that you want information only for certain customers—those who have not ordered anything in the last six months.



A query tells Word which information to select from a data source.

You create queries by selecting options in the Query Options dialog box. You select data fields to specify the categories of information you want. The order in which you select the data fields determines the order of the columns of information in the table, from left to right. To get the information only from certain data records, you specify one or more rules for selecting the records. To list the rows of information in a particular order, you can sort the data records.



Specifying the Record-Selection Rules

On the Filter Records tab, you specify the rules that Word uses to retrieve the information you want, based on the contents of selected data fields. When specifying a rule, you can select any data field in the data source—even a data field you don't want to include in the table.

A record-selection rule is made up of three parts:

- A field name corresponding to a data field in the selected data source
- A comparison phrase, such as "Equal To" or "Is Not Blank"
- Text or numbers you want the contents of the data field to be compared with

If you compare text When comparing a data field that contains text, Word compares the sequence of characters based on the ANSI sorting order. The text "apple" is considered "less than" the word "berry" because, alphabetically, "apple" precedes "berry." Whether the text is uppercase or lowercase isn't significant.

For example, to retrieve data records for members of your organization whose last names begin with "A" through "L," you specify the following rule:

LastName Is Less Than M

Any name beginning with "A" through "L" is considered less than "M," so only the data records that contain those names are selected. (The last name must be contained in a separate data field, or else it must precede the first name in the field—for example, "Bendal, Maria".)

If you compare numbers mixed with text If numbers are mixed with letters, hyphens, plus or minus signs, or other nonnumeric characters, Word compares the numbers as though they were a sequence of text characters. For example, a five-digit U.S. ZIP Code is compared as a number, whereas a nine-digit "ZIP+4" code such as 99999-9099 is compared as text, as are non-U.S. postal codes that contain letters.

Comparing sequences of mixed numbers and nonnumeric characters—code numbers, for example—can have different results if some items contain more sequential numerals than others. For example, the following items are sorted in this order:

0002xy, 002, 011y, 1, 1x, 1yz, 22x, 2x

The following items, however, are sorted in this order:

0001, 0001x, 0001yz, 0002, 0002x, 0002xy, 0011y, 0022x

Specifying Multiple Rules

You can specify as many as six selection rules. Using multiple rules allows you to narrow the range of data records that are selected. When you select multiple rules, you must specify AND or OR to connect each additional rule to the preceding rule, as in the following examples.

Example 1	Example 2
State (Is) Equal To Oregon	State (Is) Equal To Oregon
AND City (Is) Equal To Portland	OR State (Is) Equal To California

The rules connected by AND select only data records that contain both "Oregon" in the State field and "Portland" in the City field. The rules connected by OR select all data records that contain either "Oregon" or "California" in the State field—a potentially larger number of records. The key difference between AND and OR is as follows:

- When you use AND to connect rules, Word selects only those records that satisfy both (or all) rules. Each rule connected by AND *eliminates* more of the records in the data source.
- When you use OR to connect rules, Word selects any record that satisfies at least one of the connected rules. Each rule connected by OR *selects more* of the records in the data source.

AND has precedence You can use AND and OR separately or in combination. In sets of rules that contain both AND and OR, rules connected by AND have precedence over rules connected by OR. This means that the set of rules connected by AND is used to select records before the set of rules connected by OR. How you connect the rules—by using AND or by using OR—affects which data records are selected.

Suppose you want to select data records of all clients who live in either Portland or Salem, Oregon. In the Query Options dialog box, you would specify the following rules to determine the contents of the data fields "City" and "State":

State (Is) Equal To Oregon
AND City (Is) Equal To Portland
OR State (Is) Equal To Oregon
AND City (Is) Equal To Salem

Using the first set of rules connected by AND, Word compares the data records to identify the clients who live in Portland, Oregon. Next, Word compares the data records with the next set of rules connected by AND. Word then selects only data records of clients in Oregon who live in either Portland or Salem.

allows you to
nultiple rules,
preceding

;on
fornia

)th "Oregon"
:d by OR
ı the State
:tween AND

cords that
:s more of the

: satisfies at
:cts *more* of

·mbination. In
ID have
rules
onnected by
affects which

ther Portland
cify the
nd "State":

data records to
)ares the data
lects only data

Notice that the following set of rules does *not* produce the same result:

State (Is) Equal To Oregon
AND City (Is) Equal To Portland
OR City (Is) Equal To Salem

Because AND takes precedence, the first set of rules connected by AND selects records of clients who live in Portland, Oregon. However, the rule connected by OR also selects records for clients in any city named Salem—including Salem, Massachusetts, for instance

Comparing a range of values You can also use AND to compare a selected field with a range of values rather than a single value. For example, given the following rules, Word selects all data records that have a value of 98001 through 98500 in the PostalCode field.

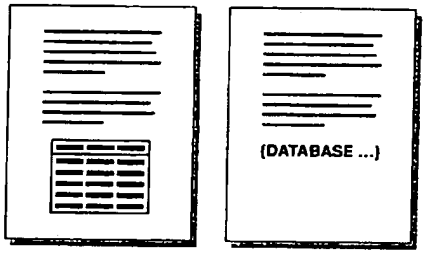
PostalCode (Is) Greater Than Or Equal (To) 98001
AND PostalCode (Is) Less Than Or Equal (To) 98500

Keeping the Table Information Up to Date

If you select the Insert Data As Field check box when you insert the information, Word does not insert an actual table; instead, it inserts a DATABASE field to represent the table.

With field codes hidden, the information is displayed as a table. With field codes displayed, the information is displayed as a DATABASE field. The field contains all information needed to locate and open the selected data source, carry out the query, and insert the information in your document.

To display or hide the field codes, press ALT+F9 (Windows) or OPTION+F9 (Macintosh).



Document with field codes hidden ...

... and with field codes displayed.

The Artech HouseUniversal Personal Communications Series

Ramjee Prasad, Series Editor

CDMA for Wireless Personal Communications, Ramjee Prasad

Universal Wireless Personal Communications, Ramjee Prasad

Wideband for Third Generation Mobile Communications, Tero Ojanperä and Ramjee Prasad

For further information on these and other Artech House titles, including previously considered out-of-print books now available through our In-Print-Forever® (IPF®) program, contact:

Artech House
685 Canton Street
Norwood, MA 02062
Phone: 781-769-9750
Fax: 781-769-6334
e-mail: artech@artechhouse.com

Artech House
46 Gillingham Street
London SW1V 1AH UK
Phone: +44 (0)20 7596-8750
Fax: +44 (0)20 7630-0166
e-mail: artech-uk@artechhouse.com

Find us on the World Wide Web at:
www.artechhouse.com

For a recent listing of titles in the *Artech House Mobile Communications Library*, turn to the back of the book.

02

Wideband CDMA for Third Generation Mobile Communications

Tero Ojanperä
Ramjee Prasad
editors

Artech House
Boston • London

Library of Congress Cataloging-in-Publication Data
Ojanperä, Tero
Wideband CDMA for third generation mobile communications / Tero Ojanperä, Ramjee Prasad, editors.
p. cm.
Includes bibliographical references and index.
ISBN 0-89006-735-X (alk. paper)
1. Code division multiple access. 2. Mobile communication systems.
3. Broadband communication systems. I. Prasad, Ramjee. II. Title.
TK5103.45.034 1998
621.3845—dc21 98-33857
CIP

British Library Cataloguing in Publication Data
Wideband CDMA for third generation mobile communications
1. Code division multiple access 2. Broadband communication systems
3. Global system for mobile communications
I. Ojanperä, Tero II. Prasad, Ramjee
6213'84'56

ISBN 0-89006-735-X

Cover design by Lynda Fishbourne

Books are the joy of life, and learning is a never-ending experience.

To my wife Tiina, and to our son Eerik
To my brother Aki, to my mother Maiju and to the memories of my brother Juha and father Iisakki
—Tero Ojanperä

To my wife Jyoti, to our daughter Neeli, and to our sons Anand and Rajeev
—Ramjee Prasad

might degrade the performance. Therefore, in some instances it might be better to implement higher bit rates with a multicode scheme.

5.9.1.3 *Forward Link Orthogonality*

Since the forward link is synchronous, it is possible to maintain orthogonality between the codes. For multicode transmission we can select a set of orthogonal codes. With VSF, it is also possible to maintain orthogonality between different spreading factors if we impose the following constraint between different rates:

$$R = R_c/2^n \qquad n = 0, 1, 2, \ldots \tag{5.3}$$

This can be achieved with the variable length Walsh sequences or with the three structured orthogonal codes [16] discussed in Section 5.6.3.

5.9.1.4 *Power Control Requirements*

In order for the receiver to be able to estimate the path loss for power control purposes, transmission power should not vary. Otherwise, the receiver has to first know the rate of a traffic channel. Another possibility is to use a fixed rate control channel for the power measurements. With multicode transmission, each of the parallel channels has a fixed power. For VSF, the power varies according to transmission rate, and thus, explicit transmission power needs to be signaled to the receiver. One further advantage of multicode transmission is that in case of several simultaneous services, the QoS can be adjusted using power control. Parallel service with similar power requirements can be time multiplexed, while code multiplexing is used for parallel services with different power requirements.

5.9.1.5 *Complexity*

Power amplifier linearity requirements should be as low as possible to allow the use of power-efficient power amplifiers. The multicode scheme results in larger envelope variations than the single code transmission. An additional drawback of the multicode scheme is that it requires as many RAKE receivers as there are codes. However, each RAKE receiver may be less complex, since a higher spreading factor might facilitate the use of fewer bits for quantization [20].

5.9.2 Granularity of Data Rates

Granularity means the minimum possible bit rate change. In order to maintain high flexibility, it is desirable to have as fine of granularity as possible. For multicode, granularity is *R/M* Kbps. If one code has the capability of smaller quantization of data rates, then better granularity can be achieved. For the VSF scheme, granularity varies according to the spreading factor and is finest for the low bit rates [20]:

$$SF = 1 \quad SF = \frac{R_c}{R} - 1 \qquad R_c = R$$

where SF is the spreading factor, R_c is the chip rate, and R is the user bit rate.

For low bit rate services, this would most likely be sufficiently fine granularity. However, if we restrict the spreading factors according to (5.2), then the granularity is too coarse. Furthermore, if we assume for multicode transmission that one code carries 10 Kbps of traffic, the basic granularity is 10 Kbps. This is too large for some services such as low rate speech codecs, which typically require better granularity. The problem of granularity can also be considered from the viewpoint that, given the coded user bit rate, it is possible to match it with the given chip rate. For example, if one code carries 32 ksymbol/s and the source data rate is 9.6 Kbps coded with a 1/3 convolutional code, it results in a gross bit rate of 28.8 Kbps. Thus we need to match the 28.8 Kbps to 32 Kbps.

One alternative to matching the bit and chip rate is the use of rate compatible punctured codes (RCPC) [33]. The basic idea of RCPC coding is to divide the bit stream of the mother code into blocks whose bits are either punctured or repeated according to a perforation/repetition matrix. The drawback of this approach is that a fixed number of rates have to be selected, since not all service rates can be directly matched to the available chip rate. Furthermore, for RCPC, good codes are only known up to the constraint length of 7.

Repetition coding, or puncturing, is another possibility for rate matching. Even unequal repetition coding (i.e., only some symbols in the frame are repeated to implement the desired symbol rate within the frame) can be used [17].

5.9.3 Transmission of Control Information

In order to vary the data rate or other service parameters, the receiver needs to know the structure of the received signal. This can be achieved either by transmitting explicit control information or by blind rate detection, for example, from the CRC information [34]. In case explicit control information is transmitted, the following issues need to be solved:

- Coding of the control information to achieve desired quality of service;
- Multiplexing of the control information;
- Position of the control information.

The control bits have to have a considerably lower error rate than the information bits, since, if a control word has an error, the whole frame is lost. Control information can either be coded together with the user data or independently from the user data. Furthermore, control information can be transmitted either code multiplexed or time multiplexed.

There are two possibilities for the position of control information: in the previous frame or in the same frame as the user data, as illustrated in Figure 5.14. Since the receiver is informed in advance of the transmission parameters, the processing of

the data can be done "on-line." However, for services with a short delay requirement the additional delay of one frame might be too long. A further drawback is that erroneous control information results in loss of the previous and the next frame, except if the next frame is transmitted with the same parameters as the previous frame.

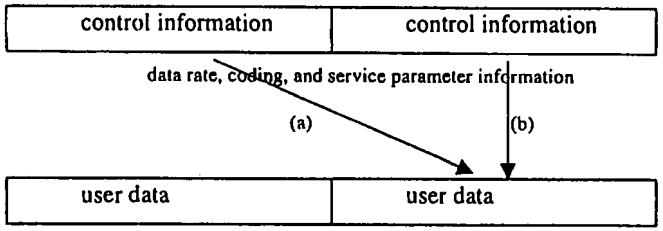


Figure 5.14 Control information transmitted (a) in the previous frame, (b) in the same frame as user data.

In case the control information is transmitted within the same frame as the user data, then the receiver needs to decode the control information first. Thus, the data need to be buffered. Memory requirements for the buffer depend on the spreading code solution. In case the spreading ratios are multiples of each other, it is possible to despread with the lowest spreading ratio and to buffer the sub-symbols after despreading, instead of the samples before the despreading, which need to be used if arbitrary spreading ratios are used. However, for high data rates buffering might be still a problem.

5.10 PACKET DATA

Since non real-time packet data services are not delay sensitive, they use the retransmission principle implemented with ARQ protocol to improve the error rate. The retransmission protocol can be either implemented in layer 2 as part of MAC and RLP or in the physical layer (layer 1). If packet data retransmission is implemented as part of the layer 2, the transmission of packet data in the physical layer does not differ from the transmission of circuit switched data. So the multirate aspects discussed above also apply to the transmission of packet data. If the physical layer ARQ is used, then the physical layer is modified depending on the ARQ scheme used (see Section 5.8 about ARQ schemes). In both cases, the access procedure and handover for packet data services have certain special implications, which are discussed in the next subsections.

5.10.1 Packet Access Procedure

The packet access procedure in CDMA should minimize the interference. Since there is no connection between the base station and the mobile station access procedure, initial access is not power controlled and thus the transmitted during this period should be minimized. There are three scenarios for packet access:

- Infrequent transmission of short packets containing little information;
- Transmission of long packets;
- Frequent transmission of short packets.

Since the establishment of a traffic channel itself requires signaling and thus consumes radio resources, it is better to transmit small packets within the random access message without power control. For long and frequent short packets, a dedicated traffic channel should be allocated.

If a dedicated channel has been reserved and there is nothing to send, the mobile station either cuts off the transmission or keeps the physical connection by transmitting power control and reference symbols only. In the former case, a virtual connection (higher layer protocols) is retained in order to rapidly re-establish the link in case of a new transmission. Selection between these two alternatives is a trade-off between resources spent for synchronization and power control information, and resources spent for random access.

5.10.2 MAC Protocol

The task of the medium access protocol is to share the transmission medium with different users in a fair and efficient way. Sometimes, multiple access protocols such as FDMA, CDMA, and TDMA are also classified as medium access protocols. However, as already discussed in the beginning of this chapter, the medium access protocol is part of the link layer, while the multiple access scheme is part of the physical layer. The medium access protocol has to resolve contention between users accessing the same physical resource. Thus, it also manages the packet access procedure described in the previous section. Since the third generation systems offer a multitude of services to customers at widely varying quality of service requirements, the MAC needs to offer capabilities to manage the access demands of different users and different service classes. This can be performed using reservation and priority schemes. Services with delay constraints can use a reservation scheme to reserve capacity to guarantee the quality of service. Priority schemes can be used to prioritize the requests from different services.

5.10.3 Packet Data Handover

Since CDMA operates with a reuse factor of one, it needs efficient and fast handover in order to avoid excessive interference with the other cells. This has been realized with soft handover in the case of circuit switched connections. Soft handover also improves performance through increased diversity. For packet connections, and especially for [illegible] there may be no need to establish soft handover even if the user is at the [illegible] however, there is still the need to route packets via the base station that [illegible] best connection. This is more important with frequent packet transmissions

Knowbots, Permissions Headers and Contract Law

paper for the conference on

Technological Strategies for Protecting Intellectual Property in the Networked Multimedia Environment

April 2-3, 1993 with revisions of 4/30/93

Henry H. Perritt, Jr.
Professor of Law
Villanova Law School
Villanova, PA 19085
(215) 645-7078
FAX (215) 645-7033, (215) 896-1723
Internet: perritt@ucis.vill.edu

Introduction

One of the ways to protect intellectual property on the NREN is through a digital library concept. Under this concept, a work would have attached to it a "permissions header," defining the terms under which the copyright owner makes the work available. The digital library infrastructure, implemented on the NREN, would match request messages from users with the permissions headers. If the request message and the permissions header match, the user would obtain access to the work. This concept encompasses major aspects of electronic contracting, which is already in wide use employing Electronic Data Interchange ("EDI") standards developed by ANSI Committee X12.1

This paper explains the relationship between the digital library concept and EDI practice, synthesizing appropriate solutions for contract law, evidence, and agency issues that arise in electronic contracting. The question of how electronic signatures should work to be legally effective is an important part of this inquiry. The paper also defines particular types of service identifiers, header descriptors, and other forms of labeling and tagging appropriate to allow copyright owners to give different levels of permission, including outright transfer of the copyright interest, use permission, copying permission, distribution permission, display permission, and permission to prepare derivative works.The paper considers how payment authorization procedures should work in conjunction with a permissions header and digital library concept in order to integrate the proposed copyright licensing procedures with existing and anticipated electronic payment authorization systems. The paper necessarily considers whether existing standards approaches related to SGML and X12 are sufficient or whether some new standards development efforts will be necessary for implementation of the concepts. The paper considers the relationship between technology and law in enforcing intellectual property, and emphasizes that the traditional adaptation of legal requirements to levels of risk is appropriate as the law is applied to new technologies.

There are certain common issues between the intellectual property question and other applications of wide area

digital network technology. The question of signatures and writings to reflect the establishment of duties and permissions and the transfer of rights is common to the intellectual property inquiry and to electronic commerce using EDI techniques. There also are common questions involving rights to use certain information channels: First Amendment privileges, and tort liability. These are common not only to technological means of protecting intellectual property but to all forms of wide area networking.

The problem

The law recognizes intellectual property because information technology permits one person to get a free ride on another person's investment in creating information value. Creative activity involving information usually is addressed by copyright, although patent has a role to play in protecting innovative means of processing information.2

Intellectual property arose in the context of letterpress printing technology. Newer technologies like xerography and more recently small computer technology and associated word processing and networking have increased the potential for free rides and accordingly increased the pressure on intellectual property.

The concern about free ride potential is especially great when people envision putting creative works on electronic publishing servers connected to wide area networks intending to permit consumers of information products to access these objects, frequently combining them and generally facilitating "publishing on demand" rather than the well known publishing just in case, typified by guessing how many copies of a work will sell, printing those in advance, and then putting them in inventory until someone wants them.

The concern is that it will be too easy to copy an entire work without detection and without paying for it. Worse, it will be easy to copy an entire work and resell it either by itself or as a part of a new derivative work or collection.

But technology is capable of protecting investment in new ways as well as gaining a free ride. Computer networks make it possible to restrict access and to determine when access occurs. Depending on how new networks are designed, they may actually reduce the potential for a free ride. The digital library is one way of realizing that potential. Professor Pamela Samuelson has observed that the digital library model replaces intellectual property with a system of technological controls.3

Digital Library Concepts

Basic Concepts

A digital library is a set of information resources ("information objects") distributed throughout an electronic network. The objects reside on servers (computers with associated disk drives connected to the network). They can be retrieved remotely by users using "client" workstations.

Origin of Concepts

The phrase "digital library" and the basic concept was first articulated in a 1989 report growing out of a workshop sponsored by the Corporation for National Research Initiatives.4 From its inception, the digital library concept envisioned retrieval of complete information

resources and not merely bibliographic information.5

The technologies of remote retrieval of complete information objects using electronic technologies is in wide use through the WESTLAW, Dialog, LEXIS, NEXIS, and National Library of Medicine databases. These remotely accessible databases, however, unlike the digital library involved a single host on which most of the data resides. The digital library concept envisions a multiplicity of hosts (servers).

Recent Developments

The remotely accessible database host concept is converging with the digital library concept as more of the electronic database vendors provide gateways to information objects actually residing on other computers. This now is commonplace with WESTLAW access to Dialog, and Dialog's gateways to other information providers.

The most explicit implementation of the digital library concept is the Wide Area Information Service ("WAIS"), which implements ANSI standard Z.39.6 WAIS permits a remote user to formulate a query that is applied to a multiplicity of WAIS servers each of which may contain information responsive to the query. The WAIS architecture permits search engines of varying degrees of sophistication, resident on WAIS information servers to apply the query against their own information objects, reporting matches back to the user.7 Future implementations of WAIS permit automatic refinement of searches according to statistical matching techniques.

The Corporation for National Research Initiatives has proposed a test bed for an electronic copyright management system.8 The proposed system would include four major elements: automated copyright recording and registration, automated, on line clearance of rights, private electronic mail and digital signatures to provide security. It would include three subsystems: a Registration and Recording System (RRS), a Digital Library System (DLS), and a Rights Management System (RMS). The RRS would provide the functions enumerated above and would be operated by the Library of Congress. It would provide "change of title" information.9 The RMS would be an interactive distributed system capable of granting rights on line and permitting the use of copyrighted material in the Digital Library System. The test bed architecture would involve computers connected to the Internet performing the RRS and RMS functions.

Digital signatures would link an electronic bibliographic record with the contents of the work, ensuring against alteration after deposit.10 Multiple RMS servers would be attached to the Internet. A user wishing to obtain rights to an electronically published work would interact electronically with the appropriate RMS. When copyright ownership is transferred, a message could be sent from the RMS to the RRS11 - creating an electronic marketplace for copyrighted material.

The EBR submitted with a new work would "identify the rights holder and any terms and conditions on the use of the document or a pointer to a designated contact for rights and permissions."12 The EBR, thus, is apparently equivalent to the permissions header discussed in this paper. Security in the transfer of rights would be provided by digital signatures using public key encryption, discussed further, infra in the section on encryption.

Basic Architectural Concepts

The digital library concept in general contemplates three basic architectural elements: a query, also called a "knowbot" in some descriptions; a permissions header attached to each information object; and a procedure for matching the query with the permissions header.

Two kinds of information are involved in all three architectural elements: information about the content of information objects desired and existing, and information about the economic terms on which an information object is made available. For example, a query desiring court opinions involving the enforcement of foreign judgments evidencing a desire to download the full text of such judicial opinions and to pay up to $1.00 per minute of search and downloading time would require that the knowbot appropriately represent the subject matter "enforcement of foreign judgments." It also requires that the knowbot appropriately represent the terms on which the user is willing to deal: downloading and the maximum price. The permissions header similarly must express the same two kinds of information. If the information object to which the permissions header is attached is a short story rather than a judicial opinion, the permissions header must so indicate. Or, if the information object is a judicial opinion and it is about enforcement of foreign judgments, the permission header may indicate that only a summary is available for downloading at a price of $10.00 per minute. The searching, matching, and retrieval procedure in the digital library system must be capable of determining whether there is a match on both subject matter and economic terms, also copying and transmitting the information object if there is a match.

Comparison to EDI

Electronic Data Interchange ("EDI") is a practice involving computer-to-computer commercial dealing without human intervention. In the most widespread implementations, computers are programmed to issue purchase orders to trading partners, and the receiving computer is programmed to evaluate the terms of the purchase order and to take appropriate action, either accepting it and causing goods to be manufactured or shipped or rejecting it and sending an appropriate message. EDI is in wide use in American and foreign commerce, using industry-specific standards for discrete commercial documents like purchase orders, invoices, and payment orders, developed through the American National Standards Institute.

There obviously are similarities between the three architectural elements of the digital library concept and EDI. There is a structured way of expressing an offer or instruction, and a process for determining whether there is a match between what the recipient is willing to do and what the sender requests.

There is also, however, an important difference. In the digital library concept, a match results in actual delivery of the desired goods and services in electronic form. In EDI practice, the performance of the contractual arrangement usually involves physical goods or performance of nonelectronic services.

Nevertheless, the digital library and EDI architectures are sufficiently similar and, it turns out the legal issues associated with both are sufficiently similar to make analogies appropriate.

Elements of Data Structure

For purposes of this paper, the interesting parts of the data structure are those elements that pertain to permission, more than those elements that pertain to content of the information object to which the header is attached. Accordingly, this section will focus on only permissions-related elements, after noting in passing that the content part of the header well might be a pointer to an inverted file to permit full text searching and matching.

The starting point conceptually for identifying the elements of the permissions header are the rights exclusively reserved to the copyright owner by 106 of the copyright statute. But these exclusive rights need not be tracked directly because the owner of an information object free to impose contractual restrictions as well as to enjoy rights granted by the Copyright Act. Accordingly, it seems that the following kinds of privileges in the requester should be addressed in the permissions header:

outright transfer of all rights

use privilege, either unrestricted or subject to restrictions

copying, either unlimited or subject to restrictions like quantitative limits

distribution, either unlimited or subject to restrictions, like geographic ones or limits on the markets to which distribution can occur

preparation of derivative works

Display and presentation rights, separately identified in 106 would be subsumed into the use element, because they are particular uses.

The simplest implementation would allow only binary values for each of these elements. But a binary approach does not permit the permissions header to express restrictions, like those suggested in the enumerated list. Elements could be defined to accept the most common kinds of restrictions on use, and quantitative limits on copying, but it would be much more difficult to define in advance the kinds of geographic or market-definition restrictions that an owner might wish to impose with respect to distribution.

In addition to these discrete privileges, the permissions header must express pricing information. The most sensible way of doing this is to have a price associated with each type of privilege. In the event that different levels of use, copying, or distribution privilege are identified, the data structure should allow a price to be associated with each level.

A complicating factor in defining elements for price is the likelihood that different suppliers would want to price differently. For example, some would prefer to impose a flat fee for the grant of a particular privilege. Others might wish to impose a volume-based fee, and still others might wish to impose a usage or connect-time based fee. The data structure for pricing terms must be flexible enough to accommodate at least these three different approaches to pricing.

Finally, the data structure must allow for a specification of acceptable payment terms and have some kind of trigger for a payment approval procedure. For example,

the permissions header might require presentation of a credit card number and then trigger a process that would communicate with the appropriate credit card database to obtain authorization. Only if the authorization was obtained would the knowbot and the permissions header "match."

There is a relationship between the data structures and legal concepts. The knowbot is a solicitation of offers. The permissions header is an offer. The matching of the two constitutes an acceptance. Mr. Linn's "envelope" could be the "contract."

There are certain aspects of the data structure design that are not obvious. One is how to link price with specific levels of permission. Another is how to describe particular levels of permission. This representation problem may benefit from the use of some deontic logic, possibly in the form of a grammar developed for intellectual property permissions. Finally, it is not clear what the acceptance should look like. Conceptually, the acceptance occurs when the knowbot matches with a permissions header, but it is unclear how this legally significant event should be represented.

Role of Encryption

The CNRI test bed proposal envisions the use of public key encryption to ensure the integrity of digital signatures and to ensure the authenticity of information objects. Public key encryption permits a person to encrypt a message - like a signature using a secret key, one known only to the sender, while permitting anyone with access to a public key to decrypt it. Use of public key cryptography in this fashion permits any user to authenticate a message, ensuring that it came from the purported sender.13 A related technology called "hashing" permits an encrypted digital signature to be linked to the content of a message. The message can be sent in plain text (unencrypted) form, but if any part of it is changed, it will not match the digital signature. The digital signature and hashing technologies thus permit not only the origin but also the content integrity of a message of arbitrary length to be authenticated without necessitating encryption of the content of the message. This technology has the advantage, among others, that it is usable by someone lacking technological access to public key encryption. An unsophisticated user not wishing to incur the costs of signature verification nevertheless can use the content of the signed information object.

It is well recognized that encryption provides higher levels of security than other approaches. But security through encryption comes at a price. Private key encryption systems require preestablished relationships and exchange of private keys in advance of any encrypted communication. The burdens of this approach have led most proponents of electronic commerce to explore public key encryption instead. But public key systems require the establishment and policing of a new set of institutions. An important infrastructure requirement for practicable public key cryptography is the establishment and maintenance of certifying entities that maintain the public keys and ensure that they are genuine ones rather than bogus ones inserted by forgers. A rough analogy can be drawn between the public key certifying entities and notaries public. Both kinds of institutions verify the authenticity of signature. Both kinds require some level of licensing by governmental entities. Otherwise the word of the "electronic notary"

(certifying entity) is no better than an uncertified, unencrypted signature. In a political and legal environment in which the limitations of regulatory programs have been recognized and have led to deregulation of major industries, it is not clear that a major new regulatory arrangement for public key encryption is practicable. Nevertheless, experimentation with the concept in support of digital library demonstration programs can help generate more empirical data as to the cost and benefits of public key encryption to reinforce electronic signatures.

On the other hand, it is not desirable to pursue approaches requiring encryption of content. No need to encrypt the contents is apparent in a network environment. Database access controls are sufficient to prevent access to the content if the permissions header terms are not matched by the knowbot. On the other hand, if the electronic publishing is effected through CDROMs or other physical media possessed by a user, then encryption might be appropriate to prevent the user from avoiding the permissions header and going directly to the content.

While encrypted content affords greater security to the owner of copyrighted material. Someone who has not paid the price to the copyright owner must incur much higher cost to steal the material. But the problem is everyone must pay a higher price to use the material. One of the dramatic lessons of the desktop computer revolution was the clear rejection of copyright protection in personal computer software. The reasons that copy protection did not survive in the market place militate against embracing encryption for content. Encryption interferes with realization of electronic markets, because producer and consumer must have the same encryption and description protocols. Encryption burdens processing of electronic information objects because it adds another layer. Some specific implementations have encryption require additional hardware at appreciable costs.

Digital libraries cannot become a reality until consumers perceive that the benefits of electronic formats outweigh the costs, compared to paper formats. Encryption interferes with electronic formats' traditional advantages of density, reusability, editability, and computer search ability and also, by impairing open architectures may perpetuate some of papers' advantages with respect with browsibility.14

The need for encryption of any kind depends upon whether security is available without it. That depends, in turn, on the kinds of free rides that may be obtainable and the legal status of various kinds of electronics transactions in the digital library system.

Legal Issues

Copyright: What legal effect is intended?

The design of the permissions header and the values in the elements of the header must be unambiguous as to whether an outright transfer of a copyright interest is intended or whether only a license is intended. If an outright transfer15 is intended, then the present copyright statute requires a writing signed by the owner of the rights conveyed.16 Recordation of the transfer with the Copyright office is not required, but provides advantages in enforcing transferee rights.17 On the other hand, non exclusive licenses need not be in writing nor registered. If the electronic transaction transfers the copyright in its entirety, then the rights of the transferor are extinguished, and the rights of the

transferee are determined by the copyright statute. The only significant legal question is whether the conveyance was effective.

On the other hand, when the copyright is not transferred outright but only certain permissions are granted or certain rights conveyed, the legal questions become more varied. Then, the rights of the transferor and the obligations of the transferee are matters of contract law. It is important to understand the degree to which the contract is enforceable and how it is to be interpreted in the event of subsequent disputes. The following sections consider briefly the first sale doctrine as a potential public policy obstacle to enforcing contractual restrictions different from those imposed by the copyright statute and then explore in greater depth whether electronic techniques satisfy the formalities traditionally required for making a contract, whether they adequately ensure against repudiation, and whether they provide sufficient information to permit predictable interpretation of contractual obligations and privileges.

First Sale Doctrine

The first sale doctrine may invalidate restrictions on use. It is impermissible for the holder of a patent to impose restrictions on the use of a patented product after the product has been sold. Restrictions may be imposed, however, on persons who merely license the product.18 The rationale for this limit on the power of the owner of the intellectual property interest is that to allow limitations on use of the product would interfere with competition beyond what the Congress - and arguably the drafters of the Constitution - intended in setting up the patent system.

The first sale doctrine applies to copyright owners.19 Indeed, because of the First Amendment's protection of informational activity, the argument against restrictions after the first sale may be even stronger in the copyright arena then in the patent arena.

The first sale doctrine is potentially important because it may invalidate restrictions imposed on the use of information beyond what is authorized by the Copyright Act and by common law trade secret. Thus, there may be serious questions about the legal efficacy of use restrictions suggested in ___, although such restrictions are common in remote database service agreements. The vendors could argue that the limitations pertain to the contractual terms for delivery of a service rather than use of information as such. The characterization avoids the overlap with copyright and thus may also avoid the conflict between federal policy and contract enforcement.20

Contract Formation Issues

The law does not enforce every promise. Instead, it focuses its power only on promises surrounded with certain formalities to make it likely that the person making the promise (the "promisor") and the person receiving the promise (the "promisee") understood that their communication had legal consequences. A threshold question for the digital library system is whether the traditional formalities for making a contract are present when the contract is made through electronic means. The digital library system considered in this paper clearly contemplates that a contract is formed when the knowbot and the permissions header achieve a match. In this respect, the digital library concept converges with EDI where trading parties contemplate that a contract to perform services or deliver goods is

formed when a match occurs either upon the receipt of a purchase order or upon the transmission of a purchase order acknowledgment.

It is not altogether clear, however, whether the match between values and computer data structures meets contract formation requirements, particularly those expressed in various statutes of frauds. Statutes of frauds require "writings" and "signatures" for certain kinds of contracts - basically those contemplating performance extending beyond a period of one year.21

In many instances, the digital library contract will be fully performed almost instantaneously upon delivery of the information object after the knowbot and the permissions header match. In such a case, the statute of frauds is not a problem and its requirements need not be satisfied. In other cases, however, as when the intent of the owner of the information object is to grant a license to do things that will extend beyond one year, the statute of frauds writing and signature requirements must be met.

Historical application of Statutes Of Frauds by the courts clearly indicates that there is flexibility in the meaning of "writing" and "signature." A signature is any mark made with the intent that it be a signature.22 Thus an illiterate person signs by making an "X," and the signature is legally effective. Another person may sign a document by using a signature stamp. Someone else may authorize an agent to sign his name or to use the signature stamp. In all three cases the signature is legally effective. There may of course be arguments about who made the X, or whether the person applying the signature stamp was the signer or his authorized agent, but these are evidentiary and agency questions, not arguments about hard and fast contract-law requirements.

Under the generally accepted legal definition of a signature, there is no legal reason why the "mark" may not be made by a computer printer, or for that matter by the write head on a computer disk drive or the data bus in a computer random access memory. The authorization to the computer agent to make the mark may be given by entering a PIN ("Personal Identification Number") on a keyboard. To extend the logic, there is no conceptual reason to doubt the legal efficacy of authority to make a mark if the signer writes a computer program authorizing the application of a PIN upon the existence of certain conditions that can be tested by the program. The resulting authority is analogous to a signature pen that can be operated only with a mechanical key attached to somebody's key ring, coupled with instructions to the possessor of the key.

Which of these various methods should be selected for particular types of transactions must depend, not on what the law requires, because the law permits any of these methods. Rather, it must depend on the underlying purposes of the legal requirement and which method best serves those purposes.

The real issue is how to prove that a particular party made the mark. In other words, the contingency to be concerned about is repudiation, not absence of formalities. Repudiation should be dealt with through usual evidentiary and fact finding processes rather than artificial distinctions between signed and unsigned documents.

Authority is skimpier on how flexible the "writing"

requirement is. The best approach is to borrow the fixation idea from the copyright statute and conclude that a writing is "embodiment in a copy . . . sufficiently permanent or stable to permit it to be perceived, reproduced, or otherwise communicated for more a period of more than transitory duration."23

The most important thing conceptually is to understand the purpose of the writing and signature requirements. They have two purposes: awareness or formality and reliability of evidence. Signature requirements, like requirements for writings and for original documents have an essentially evidentiary purpose. If there is a dispute later, they specify what kind of evidence is probative of certain disputed issues, like "who made this statement and for what purpose?" The legal requirements set a threshold of probativeness. Surely the values in a knowbot as well as the values in a permissions header constitute and "mark," and someone who knowingly sets up potential transactions in a digital library scheme can have the intent that the mark be a signature.

When a contract is made through a signed writing, it is more likely that the parties to the contract understand what they are doing. They are aware of the legal affect of their conduct because the writing in the signature involve a greater degree of formality than a simple conversation.

The awareness/formality purpose can be served by computerized contracting systems. This is so not so much because the computers are "aware" of the affect of their "conduct." Rather, it is true because the computers are agents of human principals. The programming of the computer to accept certain contract terms is the granting of authority to the computer agent to enter into a contract. The fact that a principal acts through an agent engaging in conduct at a later point and time never has been thought to defeat contract formation in the traditional evolution of agency and contract law. Nor should it when the agent is a computer.

Fulfillment of the evidentiary purpose depends on the reliability of the information retained by the computer systems making up the digital library. Such systems must be designed to permit the proponent of contract formation to establish the following propositions if the other party to the purported contract attempts to repudiate it.

1. It came from computer X

2. It accurately represents what is in computer X24 now25

3. What is in computer X now is what was in computer X at the time of the transaction

4. What was in computer X at the time of the transaction is what was received from the telecommunications channel26

5. What was received from the telecommunications channel is what was (a) sent, (b) by computer Y.

Two other questions relate to matters other than the authenticity of the message:

6 Computer Y was the agent of B

7 The message content expresses the content of the

contract (or more narrowly, the offer or the acceptance).27

Factual propositions 1-4 can be established by testimony as to how information is written to and from telecommunications channel processors, primary storage, and secondary storage. Factual proposition 5 requires testimony as to the accuracy of the telecommunications channel and characteristics of the message that associate it with computer Y. Only the last proposition (number 5) relates to signatures, because signature requirements associate the message with its source.28 The other propositions necessitate testimony as to how the basic message and database management system works. It is instructive to compare these propositions with the kinds of propositions that must be established under the business records exception to the hearsay rule when it is applied to computer information.

Those propositions may be supported with non technical evidence, presented by non programmers. A witness can lay a foundation for admission of computer records simply by testifying that the records are generated automatically and routinely in the ordinary course of business. The more inflexible the routine, and the less human intervention in the details of the computer's management of the database the better the evidence.29

The ultimate question is trustworthiness, and if the computer methods are apparently reliable, the information should be admitted unless the opponent of admissibility can raise some reasonable factual question undercutting trustworthiness.30

Contract Interpretation Issues

Assuming that the permissions header and knowbot constitute sufficient writings to permit a contract to be formed and that the signature requirement also is met, through digital signature technology or otherwise, there still are difficult contract interpretation questions. Contract interpretation questions arise not only after contractual relationships are formed, but also in connection with deciding whether there has been offer and acceptance, the prerequisites to contract formation.31 Contract interpretation always seeks to draw inferences about what the parties intended. When contract interpretation issues arise at the contract formation stage, the questions are what the offeror intended the content of the offer to be and what the offeree intended the content of the purported acceptance to be. The proposed Digital Library System envisions extremely cryptic expressions of offer and acceptance - by means of codes. The codes have no intrinsic meaning. Rather, extrinsic reference must be made to some kind of table, standard, or convention associating particular codes with the concepts they represent. Extrinsic evidence is available to resolve contract interpretation questions when the language of the contract itself is ambiguous, and perhaps at other times as well.32 The codes in the permissions header and knowbots certainly are ambiguous and become unambiguous only when extrinsic evidence is considered. So there is no problem in getting a standard or cable into evidence. The problem is whether the parties meant to assent to this standard.

In current EDI practice, this question is resolved by having parties who expect to have EDI transactions with each other to sign a paper trading partner agreement, in which the meaning of values or codes in the transaction sets is established.33 But requiring each pair of suppliers and users

of information in a digital library to have written contracts with each other in advance would defeat much of the utility of the digital library. Thus the challenge is to establish some ground rules for the meaning of permissions header and knowbot values that all participants are bound by. There are analogous situations. One is a standard credit card agreement that establishes contractual terms among credit card issuer, credit card subscriber, and merchant who accepts the credit card. The intermediary - the credit card company - unilaterally establishes contract terms to which the trading partners assent by using and accepting the credit card.34 Also, it is widely recognized that members of a private association can, through their constitution and bylaws establish contractual relationships that bind all of the members in dealing with each other.35 In the Digital Library System, similar legal arrangements can establish the standards by which electronic transactions between permissions header and knowbots will bind transferor and transferee of information.

Third Party Liability

It is not enough merely to ensure that the licensee is contractually bound. Trading partners also must ensure that the participants in funds transfers have enforceable obligations. For example, if the digital library system envisions that the information object would not be released to the purchaser without simultaneous release of a payment order, the supplier may be interested in enforcing the obligations of financial intermediaries who handle the payment order. This implicates the federal Electronic Funds Transfer Act, and Article 4A of the Uniform Commercial Code, regulating wire transfers.

Solutions

Satisfy the Business Records Exception to the Hearsay Rule

The discussion of contract formalities earlier in this paper concluded that legally enforceable contracts can be formed through electronic means and that the significant legal questions relate to reliability of proof and intent of the parties to be bound by using the electronic techniques. This section considers the reliability of proof further. Traditional evidence law permits computer records to be introduced in evidence when they satisfy the requirements of the business records exception: basically that they are made in the ordinary course of business, that they are relied on for the performance of regular business activities, and that there is no independent reason for questioning their reliability.36

The business records exception shares with the authentication concept statute of frauds and the parol evidence rule a common concern with reliability.37 The same procedural guarantees and established practices that ensure reliability for hearsay purposes also ensure reliability for the other purposes. Under the business records exception, the proponent must identify the source of a record, through testimony by one familiar with a signature on the record, or circumstantially.38 The steps in qualifying a business record under the common law, which since have been relaxed,39 were:

Proving that the record is an original entry made in the routine course of business

Proving that the entries were made upon the personal knowledge of the proponent/witness or someone reporting to him

Proving that the entries were made at or near the time of the transaction

Proving that the recorder and his informant are unavailable.40

These specific requirements are easier to understand and to adapt to electronic permissions and obligations formed in a digital library system by understanding the rationale for the business records exception. The hearsay rule excludes out of court statements because they are inherently unreliable, primarily because the maker of the statement's demeanor cannot be observed by the jury and because the maker of the statement is not subject to cross examine. On the other hand, there are some out of court statements that have other guarantees of reliability. Business records are one example. If a continuing enterprise finds the records sufficiently reliable to use them in the ordinary course of business, they should be reliable enough for a court. The criteria for the business records exception all aim at ensuring that the records really are relied upon the business to conduct its ordinary affairs.

The Manual for Multidistrict Litigation suggests steps for qualifying computer information under the business records exception:

1.The document is a business record

2.The document has probative value

3.The computer equipment used is reliable

4.Reliable data processing techniques were used41

The key in adapting the business records exception to electronic permissions in a digital library system are points 3 and 4. Establishing these propositions and the propositions set forth in section ___ of this paper requires expert testimony. Any designer of a digital library system must consult with counsel and understand what testimony an expert would give to establish these propositions. Going through that exercise will influence system design.

Reinforce the Evidentiary Reliability by Using Trusted Third Parties

The evidentiary purpose of contract formation requirements can be satisfied by using a trusted third party as an intermediary, when the third party maintains archival records of the transactions. The third party lacks any incentive for tampering with the records and when the third parties archiving system is properly designed, it can provide evidence sufficient to establish all of the propositions identified in ___.

This third party intermediary concept is somewhat different from the concept for a certifying agent in digital signature systems. To be sure, the custodian of transaction records envisioned by this section could be the same as the certifying entity for public and key encryption, but the custodian role can be played in the absence of any encryption. Indeed, the digital library itself is a good candidate for the custodian role. The library has no incentive to manipulate its records in favor of either of the producers of information value or the consumers. In order to carry out its affairs, it must use these transactional records in the ordinary course of business,

thereby making it likely that digital library records would qualify under the business records exception.

Standardization

Obviously, the digital library concept depends upon the possibility of an automated comparison between the knowbot and the permissions header. This means that potential requesters of information and suppliers of information must know in advance the data structures for representing the elements of the permissions header and the knowbot. This requires compatibility. Compatibility requires standardization. Standardization does not, however, necessarily require "Standard" in the sense that they are developed by some bureaucratic body like ANSI. It may simply imply market acceptance of a particular vendor's approach. Indeed, each digital library might use different data structures. All that is necessary is that the structure of the knowbot and the structure of the permissions header be compatible within any one digital library system. Also, as demands emerge for separate digital libraries to communicate with each other, there can be proprietary translation to assure compatibility between systems much as common word processing programs translate to and from other common formats and much as printers and word processing software communicate with each other through appropriate printer drivers. In neither of these cases has any independent standards organization developed a standard that is at all relevant in the marketplace.

Standardizing the elements of Knowbot and permissions headers involves content standardization, which generally is more challenging than format standardization.42 A permissions header/Knowbot standard is a system for representing legal concepts and for defining legal relations. As such, the standard is basically a grammar for a rule based substantive system in a very narrow domain.43 The data elements must correspond to legally meaningful relational attributes. The allowable values must correspond to legally allowable rights, obligations, privileges and powers. In other words, the standard setter must meet many of the challenges that a legal expert system designer working with Hohfeldian frameworks must meet.44 This adds a constraint to the standards setting process. Unlike setting format standards, where the participants are free to agree on an arbitrary way of expressing format attributes, participants in setting a content standard must remain within the universe of permissible content. The set of permissible values is determined by the law rather than being determined only by the imagination of format creators.

Enforcement and Bottlenecks

One of the many profound observations by Ithiel de Sola Pool was that copyright always has depended upon technological bottlenecks for its enforceability. The printing press was the original enforcement bottleneck. Now, a combination of the printing press and the practical need to inventory physical artifacts representing the work constitute the enforcement bottlenecks. As technologies change, old bottlenecks disappear and enforceability requires a search for new bottlenecks. When there are single hosts, like Westlaw, Dialog, Lexis, and CompuServe, access to that host is the bottleneck. The problem with distributed publishing on an open architecture internet is that there is no bottleneck in the middle of the distribution chain corresponding to the printer, the warehouse or the single host.

If new bottlenecks are to be found, they almost surely will be found at the origin and at the point of consumption. Encryption and decryption techniques discussed elsewhere in this volume concentrate on those bottlenecks as points of control. It also is possible that rendering software could become the new bottleneck as Mr. Linn suggests.

Even with those approaches, however, a serious problem remains in that the new technologies make it difficult or impossible to distinguish between mere use and copying. Thus the seller cannot distinguish between an end user45 and a potential competitor. On the other hand, the new technologies permit a much better audit trail, potentially producing better evidence for enforcement adjudication.

If network architectures for electronic publishing evolve in the way that Ted Nelson suggests with his Xanadu concept, the real value will be in the network and the pointers, not in the raw content. Thus, the creative and productive effort that the law should reward is the creation and productive effort that the law should reward is the creation and production and delivery of pointers, presentation, distribution, and duplication value. If this is so, then technological means will be particularly important, foreclosing access by those lacking passwords and other keys and limiting through contract what a consumer may do with the information.

In such an architecture, the law either will be relatively unimportant because technology can be counted on to prevent free riding or, the law will need to focus not on prohibiting copying or use without permission, but on preventing circumvention of the technological protections. Thus, legal approaches like that used to prevent the sale of decryption devices for television broadcasts and legal issues associated with contract enforcement may be more important than traditional intellectual property categories.

Weighing Risks and Costs

The law generally imposes sensible levels of transaction costs. Usually, transaction costs are proportional to the risk. Figure 1 shows a continuum of risk and transaction cost in traditional and new technologies. A real estate closing involves significant risks if there is some dispute later about the transaction. Therefore, the law affords much protection, including a constitutional officer called a registrar of deeds who is the custodian of records associated with the transaction. The risk level analogous to this in electronic publishing might be access to an entire library including access software as well as contents. Next, is a transaction involving a will or power of attorney. There, the risk is substantial because the maker of the instrument is not around to help interpret it. The law requires relatively high levels of assurance here, though not as great as those for real estate transactions. The law requires witnesses and attestation by a commissioned minor official called a notary public. The electronic publishing analogy of this level of risk might be the contents of an entire CDROM.

Next, in level of risk is the purchase of a large consumer durable like an automobile. The law requires somewhat less, but still significant protections for this kind of transaction: providing for the filing and enforcement of financing statements under the Uniform Commercial Code. The electronic publishing analogy might be the transfer of copyright to a complete work. Next, down

the risk continuum, is the purchase of a smaller consumer durable like a television set. Here, the law typically is reflected in written agreements of sale, but no special third party custodial mechanisms. The electronic publishing analogy might be use permission for a complete work.

Finally, is the purchase of a relatively small consumer item, say a box of diskettes. Neither the law or commercial practice involves much more than the exchange of the product for payment, with no written agreement or anything else to perform channeling, cautionary, evidentiary, or protective functions [make sure these function and the citation appears earlier]. The electronic publishing analogy might be use permission for part of a work.

Cost effectiveness = risk-proportional security

traditional transaction equivalent	institutions	electronic
real estate closing software and	registrar of deeds	entire library - contents
will/power of attorney CDROM	witnesses, notary public	contents of entire
auto purchase transfer of	UCC financing statement	complete work - copyright
television set purchase permission	written sale agreement	complete work - use
box of diskettes	-	part of a work - use permission

An encrypted object combined with rendering software is probably inconsistent with an open architecture. Because of the difficulty of setting standards for such technologies, this approach to intellectual property protection probably would be effectuated by proprietary approaches thus frustrating the vision of an open market for electronic publishing.

Conclusion

Realization of the digital library vision requires a method for collecting money and granting permission to use works protected by intellectual property. The concept of a knowbot and a permissions header attached to the work is the right way to think about such a billing and collection system. Standards for the data structures involved must be agreed to, and systems must be designed to satisfy legal formalities aimed at ensuring awareness of the legal significance of transactions and reliable proof of the terms of the transactions.

In the long run, not only must these technological issues be resolved, with appropriate attention to levels of risk and protections available under traditional legal doctrines, but also further conceptual development must be undertaken. Proponents of electronic publishing over wide area networks need to think about the appropriate metaphors: whether it is a library or a bookstore, if a library whether with or without xerox machines, if a bookstore whether it is a retail bookstore, or a mail order operation. Then,

thought must be given to how standards will be set. Finally, and most important, much more needs to be understood about the need for third party institutions. There is a good deal of enthusiasm for public key encryption. Yet the vulnerability of public key encryption systems is in the integrity of the key authority. In traditional legal protections, the third party custodians or authenticating agents like notary public and registrars of deeds receive state sanction and approval, and in the case of registrars of deeds, public funding. We must be clearer as to whether a similar infrastructure must be developed to protect against substantial risks and the use of EDI and electronic publishing technologies.

Finally, and perhaps most importantly, we must be thoughtful about what legal obligations, imposed on whom, are appropriate? The suggested 102(e) and (f) in the High Performance Computing Act looks very much like King James I's licensing of printing presses. It also looks like the FBI's proposal to prohibit the introduction of new technologies until certain conformity with past legal concepts is assured. Such approaches make the law a hurdle to new technology -- an uncomfortable position for both law and technology.

---

1 The use of EDI techniques to meter usage and determine charges for use of intellectual property is an example of billing and collection value in a typology of different types of value that can be produced in electronic marketplaces for information. See Henry H. Perritt, Jr., Market Structures for Electronic Publishing and Electronic Contracting in Brian Kahin, ed., Building Information Infrastructure: Issues in the Development of the National Research and Education Network (Harvard University and McGraw-Hill 1992) (developing typology for different types of value and explaining how market structures differ for the different types); Henry H. Perritt, Jr., Tort Liability, the First Amendment, and Equal Access to Electronic Networks, 5 Harv.J.Law & Tech. 65 (1992) (using typology of ten types of value to analyze access by competing producers of value).

2 See, e.g.U.S.Pat. No. 5,016,009, Data compression apparatus and method (May 14, 1991); U.S. Pat. No. 4,996,690, Write operator with gating capability (Feb. 26, 1991);U.S. Pat. No. 4,701,745, Data compression system (Oct. 20, 1987); Multi Tech Systems, Inc. v. Hayes Microcomputer Products, Inc., 800 F. Supp. 825 (D. Minn. 1992) (denying summary judgment on claim that patent for modem escape sequence is invalid)..

3 Comments on the 8\21 draft of "Knowbots in the Real World" from the intellectual property workshop participants at page 6 (author unknown, source unknown). Professor Samuelson also observed that the workshop, despite its title, actually did not focus much on intellectual property issues.

4 Corporation for National Research Initiatives, Workshop On The Protection Of Intellectual Property Rights In A Digital Library System: Knowbots in the Real World-May 18-19, 1989 (describing digital library system).

5 See generally Clifford A. Lynch, Visions of Electronic Libraries (libraries of future can follow acquisition-on-demand model rather than acquiring an advance of use; Z39.50 protocol will facilitate realization of that possibility, citing Robert E. Kahn & Vinton G. Serf, An Open Architecture

for a Digital Library System and a Plan for Its Development. The Digital Library Project, volume 1: The World of Knowbots (draft) (Washington D.C.: Corporation for National Research Initiatives; 1988)).

6 Clifford A. Lynch, The Z39.50 Information Retrieval Protocol: An Overview and Status Report, ACM Sigcomm Computer Communication Review at 58 (describing Z39.50 as an OSI application layer protocol that relieves clients from having to known the structure of data objects to be queried, and specifies a framework for transmitting and managing queries and results and syntax for formulating queries).

7 Brewster Kahle, Wide Area Information Server Concepts (Nov. 3, 1989 working copy; updates available from Brewster @THINK. (describing WAIS as "open protocol for connecting user interfaces on workstations and server computers") (describing information servers as including bulletin board services, shared databases, text searching and automatic indexing and computers containing current newspapers and periodicals, movie and television schedules with reviews, bulletin boards and chat lines, library catalogues, Usenet articles).

8 Robert E. Kahn, Deposit, Registration, Recordation in an Electronic Copyright Management System (August 1992) (Corporation for National Research Initiatives, Reston, Virginia).

9 Kahn 1992 at 4.

10 Kahn 1992 at 6.

11 Kahn 1992 at 10.

12 Kahn 1992 at 12.

13 Kahn 1992 at 15.

14 Browsability through techniques like the collapsible outliner function in Microsoft Word for Windows and competing products require more chunking and tagging value in the form of style and text element codes. Handling this additional formatting information through encryption and description processes is problematic.

15 " A 'transfer of copyright ownership' is an assignment, mortgage, exclusive license, or any other conveyance, alienation, or hypothecation of a copyright or of any of the exclusive rights comprised in a copyright, whether or not it is limited in time or place of effect, but not including a non-exclusive license " 17 U.S.C. 101 (1988).

16 17 U.S.C. 204(a) (1988); Valente-Kritzer Video v. Pinckney, 881 F.2d 772, 774 (9th Cir. 1989) (affirming summary judgment for author; oral agreement unenforceable under Copyright Act); Library Publications, Inc. v. Medical Economics Co., 548 F. Supp. 1231, 1233 (E.D. Pa. 1982) (granting summary judgment against trade book publisher who sought enforcement of oral exclusive distribution agreement; transfer of exclusive rights, no matter how narrow, must be in writing), aff'd mem., 714 F.2d 123 (3d Cir. 1983).

17 17 U.S.C. 205 (1988) provides constructive notice of the contents of the recorded document, determining priority as between conflicting transfers, and determines priority as between recorded transfer and non-exclusive license. The

former requirement for transfers to be recorded in order for the transferee to maintain an infringement, 17 U.S.C. 205(d), was repealed by the Berne Act Amendments 5.

18 under Adams v. Burke, 84 U.S. (17 Wall.) 453 (1873), a patentee must not attempt to exert control past the first sale. In general, use restrictions may be placed only on licensees, consistent with General Talking Pictures v. Western Elec., 304 U.S. 175 (1938). See generally Baldwin-Lima-Hamilton Corp. v. Tatnall, 169 F. Supp. 1 (E.D. Pa.1958) (applying no control after purchase rule).

19 See Red-Baron-Franklin Park, Inc. v. Taito Corp., 883 F.2d 275, 278 (4th Cir. 1989) (purchase of video game circuit boards did not create privilege to perform video game under first sale doctrine); United States v. Moore, 604 F.2d 1228, 1232 (9th Cir. 1979) (pirated sound recording not within first sale doctrine in criminal copyright infringement prosecution). But see Mirage Editions, Inc. v. Albuquerque A.R.T. Co., 856 F.2d 1341, 1344 (9th Cir. 1988) (first sale doctrine did not create privilege to prepare derivative work by transferring art in book to ceramic tiles).

20 The way in which the first sale doctrine would impact the electronically imposed use restrictions is by frustrating a breach-of-contract lawsuit by the licensor against a licensee who exceeds the use restrictions. The licensee exceeding the use restrictions would argue that it violates public policy to enforce the restrictions and therefore that state contract law may not impose liability for their violation. See generally Restatement (second) of Contracts 178 (1981) (stating general rule for determining when contract term in unenforceable on grounds of public policy).

21 In addition, as ___ of this paper notes, the Copyright Act itself requires signed writings for transfers of copyright interests. 17 U.S.C. 204(a). (1988).

22 Michael S. Baum & Henry H. Perritt, Jr., Electronic Contracting, Publishing and EDI Law ch. 6 (1991) (contract, evidence and agency issues) [hereinafter "Baum & Perritt"]. Accord, Signature Requirements Under EDGAR, Memorandum from D. Goelzer, Office of the General Counsel, SEC to Kenneth A. Fogash, Deputy Executive Director, SEC (Jan. 13, 1986) (statutory and non-statutory requirements for "signatures" may be satisfied by means other than manual writing on paper in the hand of the signatory . . . "In fact, the electronic transmission of an individual's name may legally serve as that person's signature, providing it is transmitted with the present intention to authenticate.").

23 17 U.S.C. 101 (1988). For copyright purposes, a work is created, and therefore capable of protection, when it is fixed for the first time. 17 U.S.C. 101 (1988). "[I]t makes no difference what the form, manner, or medium of fixation may be - whether it is in words, numbers, notes, sounds, pictures, or any other graphic or symbolic indicia, whether embodied in a physical object in written, printed, photographic, sculptural, punched, magnetic, or any other stable form, and whether it is capable of perception directly or by means of any machine or device 'now known or later developed.'" 1976 U.S. Code Cong. & Admin. News 5659, 5665. The legislative history further says that, "the definition of 'fixation' would exclude from the concepts purely of an evanescent or transitory nature -- reproductions such as those projected briefly on a screen

shown electronically on a television or other video display or captured momentarily in the 'memory' of a computer." 17 U.S.C. 102 note (excerpting from House Report 94-1476).

24 Or, more likely, what is on computer medium read by computer x, such as a magnetic cartridge used for archival records. Further references in the textual discussion to "what is in computer x now" should be understood to include such computer readable media.

25 Cf. Peritz, Computer Data and Reliability: A Call for Authentication of Business Records Under the Federal Rules of Evidence, 80 Nw.U.L.Rev. 956, 980 (1986) (proof that a printout accurately reflects what is in the computer is too limited a basis for authentication of computer records).

26 In some cases, the electronic transaction will be accomplished by means of a physical transfer of computer readable media. In such a case, this step in the proof would involve proving what was received physically.

27 See generally Peritz, Computer Data and Reliability: A Call for Authentication of Business Records Under the Federal Rules of Evidence, 80 Nw.U.L.Rev. 956, 979 (1986) (citing as examples of authentication Ford Motor Credit Co. v. Swarens, 447 S.W.2d 53 (Ky. 1969) (authentication by establishing relationship between computer-generated monthly summary of account activity and the customer reported on); Ed Guth Realty, Inc. v. Gingold, 34 N.Y.2d 440, 315 N.E.2d 441, 358 N.Y.S.2d 367 (1974) (authentication of summary of taxpayer liability and the taxpayer)).

28 Of course, a paper document signed at the end also is probative of the fact that no alternations have been made. In this sense, a signature requirement telescopes several steps in the inquiry outlined in the text.

29 United States v. Linn, 880 F.2d 209, 216 (9th Cir. 1989) (computer printout showing time of hotel room telephone call admissible in narcotics prosecution). See also United States v. Miller, 771 F.2d 1219, 1237 (9th Cir. 1985) (computer generated toll and billing records in price-fixing prosecution based on testimony by billing supervisor although he had no technical knowledge of system which operated from another office; no need for programmer to testify; sufficient because witness testified that he was familiar with the methods by which the computer system records information).

30 See United States v. Hutson, 821 F.2d 1015, 1020 (5th Cir. 1987) (remanding embezzlement conviction, although computer records were admissible under business records exception, despite trustworthiness challenged based on fact that defendant embezzled by altering computer files; access to files offered in evidence was restricted by special code).

31 Restatement (Second) of Contracts ___ (1981).

32 Cite for when extrinsic evidence is admissible.

33 See Baum & Perritt 2.6; The Electronic Messaging Services Task Force, The Commercial Use of Electronic Data Interchange--A Report and Model Trading Partner Agreement, 45 Bus.Law. 1645 (1990); Jeffrey B. Ritter, Scope of the Uniform Commercial Code: Computer Contracting Cases and Electronic Commercial Practices, 45 Bus.Law. 2533 (1990); Note, Legal Responses to Commercial Transactions Employing

Novel Communications Media, 90 Mich.L.Rev. 1145 (1992)

34 Garber v. Harris Trust & Savings Bank, 432 N.E.2d 1309, 1311-1312 (Ill. App. 1982) ("each use of the credit card constitutes a separate contract between the parties;" citing cases).

It is not quite this simple, because both merchant and credit card customer have separate written contracts with the credit card issuer. But there is no reason that a supplier of information to a Digital Library System and all customers of that system might not have their own contracts with the Digital Library System in the same fashion.

35 Rowland v. Union Hills Country Club, 757 P.2d 105 (Ariz. 1988) (reversing summary judgment for country club officers because of factual question whether club followed bylaws in expelling members); Straub v. American Bowling Congress, 353 N.W.2d 11 (Neb. 1984) (rule of judicial deference to private associations, and compliance with association requirements, counseled affirmance of summary judgment against member of bowling league who complained his achievements were not recognized). But see Wells v. Mobile County Board of Realtors, Inc., 387 So.2d 140 (Ala. 1980) (claim of expulsion of realtor from private association was justiciable and bylaws, rules and regulations requiring arbitration were void as against public policy; reversing declaratory judgment for defendant association).

36 F.R.E. 803(6) (excluding business records from inadmissibility as hearsay); 28 U.S.C. 1732 ("Business Records Act" permitting destruction of paper copies of government information reliably recorded by any means and allowing admission of remaining reliable record).

37 See Peritz, Computer Data and Reliability: A Call for Authentication of Business Records Under the Federal Rules of Evidence, 80 Nw.U.L.Rev. 956, 978-80, 984-85 (1986) (noting body of commentator opinion saying that business records exception and authentication are parallel ways of establishing reliability).

38 See F.R.E. 901(b)(4) (appearance, contents, substance, internal patterns, as examples of allowable authentication techniques).

39 Peritz, Computer Data and Reliability: A Call for Authentication of Business Records Under the Federal Rules of Evidence, 80 Nw.U.L.Rev. 956, 963-64 (1986) (identifying steps and trend resulting in F.R.E.).

40 Peritz, Computer Data and Reliability: A Call for Authentication of Business Records Under the Federal Rules of Evidence, 80 Nw.U.L.Rev. 956, 963 (1986).

41 Peritz, Computer Data and Reliability: A Call for Authentication of Business Records Under the Federal Rules of Evidence, 80 Nw.U.L.Rev. 956, 974 (1986) (reporting four requirements of Manual, and endorsing their use generally).

42 See Henry H. Perritt, Jr., ___, ___ Jurimetrics ___ (1993) (distinguishing between format and content standardization).

43 See Marc Lauritsen, ___ (explaining relationship between substantive legal systems and the field of artificial intelligence).

44 See Thorne, McCarty; Kevin Ashley; and Gardner.

45 It may not be particularly important to limit competition by consumers, because the consumers will never have the pointers and the rest of the network infrastructure.

Network Working Group
Request for Comments: 1510

J. Kohl
Digital Equipment Corporation
C. Neuman
ISI
September 1993

The Kerberos Network Authentication Service (V5)

Status of this Memo

This RFC specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

This document gives an overview and specification of Version 5 of the protocol for the Kerberos network authentication system. Version 4, described elsewhere [1,2], is presently in production use at MIT's Project Athena, and at other Internet sites.

Overview

This RFC describes the concepts and model upon which the Kerberos network authentication system is based. It also specifies Version 5 of the Kerberos protocol.

The motivations, goals, assumptions, and rationale behind most design decisions are treated cursorily; for Version 4 they are fully described in the Kerberos portion of the Athena Technical Plan [1]. The protocols are under review, and are not being submitted for consideration as an Internet standard at this time. Comments are encouraged. Requests for addition to an electronic mailing list for discussion of Kerberos, kerberos@MIT.EDU, may be addressed to kerberos-request@MIT.EDU. This mailing list is gatewayed onto the Usenet as the group comp.protocols.kerberos. Requests for further information, including documents and code availability, may be sent to info-kerberos@MIT.EDU.

Background

The Kerberos model is based in part on Needham and Schroeder's trusted third-party authentication protocol [3] and on modifications suggested by Denning and Sacco [4]. The original design and implementation of Kerberos Versions 1 through 4 was the work of two former Project Athena staff members, Steve Miller of Digital Equipment Corporation and Clifford Neuman (now at the Information Sciences Institute of the University of Southern California), along with Jerome Saltzer, Technical Director of Project Athena, and Jeffrey Schiller, MIT Campus Network Manager. Many other members of Project Athena have also contributed to the work on Kerberos. Version 4 is publicly available, and has seen wide use across the Internet.

Version 5 (described in this document) has evolved from Version 4 based on new requirements and desires for features not available in Version 4. Details on the differences between Kerberos Versions 4 and 5 can be found in [5].

Table of Contents

1. Introduction

Kerberos provides a means of verifying the identities of principals, (e.g., a workstation user or a network server) on an open (unprotected) network. This is accomplished without relying on authentication by the host operating system, without basing trust on host addresses, without requiring physical security of all the hosts on the network, and under the assumption that packets traveling along the network can be read, modified, and inserted at will. (Note, however, that many applications use Kerberos' functions only upon the initiation of a stream-based network connection, and assume the absence of any "hijackers" who might subvert such a connection. Such use implicitly trusts the host addresses involved.) Kerberos performs authentication under these conditions as a trusted third-party authentication service by using conventional cryptography, i.e., shared secret key. (shared secret key - Secret and private are often used interchangeably in the literature. In our usage, it takes two (or more) to share a secret, thus a shared DES key is a secret key. Something is only private when no one but its owner knows it. Thus, in public key cryptosystems, one has a public and a private key.)

The authentication process proceeds as follows: A client sends a request to the authentication server (AS) requesting "credentials" for a given server. The AS responds with these credentials, encrypted in the client's key. The credentials consist of 1) a "ticket" for the server and 2) a temporary encryption key (often called a "session key"). The client transmits the ticket (which contains the client's identity and a copy of the session key, all encrypted in the server's key) to the server. The session key (now shared by the client and server) is used to authenticate the client, and may optionally be used to authenticate the server. It may also be used to encrypt further communication between the two parties or to exchange a separate sub-session key to be used to encrypt further communication.

The implementation consists of one or more authentication servers running on physically secure hosts. The authentication servers maintain a database of principals (i.e., users and servers) and their secret keys. Code libraries provide encryption and implement the Kerberos protocol. In order to add authentication to its

transactions, a typical network application adds one or two calls to the Kerberos library, which results in the transmission of the necessary messages to achieve authentication.

The Kerberos protocol consists of several sub-protocols (or exchanges). There are two methods by which a client can ask a Kerberos server for credentials. In the first approach, the client sends a cleartext request for a ticket for the desired server to the AS. The reply is sent encrypted in the client's secret key. Usually this request is for a ticket-granting ticket (TGT) which can later be used with the ticket-granting server (TGS). In the second method, the client sends a request to the TGS. The client sends the TGT to the TGS in the same manner as if it were contacting any other application server which requires Kerberos credentials. The reply is encrypted in the session key from the TGT.

Once obtained, credentials may be used to verify the identity of the principals in a transaction, to ensure the integrity of messages exchanged between them, or to preserve privacy of the messages. The application is free to choose whatever protection may be necessary.

To verify the identities of the principals in a transaction, the client transmits the ticket to the server. Since the ticket is sent "in the clear" (parts of it are encrypted, but this encryption doesn't thwart replay) and might be intercepted and reused by an attacker, additional information is sent to prove that the message was originated by the principal to whom the ticket was issued. This information (called the authenticator) is encrypted in the session key, and includes a timestamp. The timestamp proves that the message was recently generated and is not a replay. Encrypting the authenticator in the session key proves that it was generated by a party possessing the session key. Since no one except the requesting principal and the server know the session key (it is never sent over the network in the clear) this guarantees the identity of the client.

The integrity of the messages exchanged between principals can also be guaranteed using the session key (passed in the ticket and contained in the credentials). This approach provides detection of both replay attacks and message stream modification attacks. It is accomplished by generating and transmitting a collision-proof checksum (elsewhere called a hash or digest function) of the client's message, keyed with the session key. Privacy and integrity of the messages exchanged between principals can be secured by encrypting the data to be passed using the session key passed in the ticket, and contained in the credentials.

The authentication exchanges mentioned above require read-only access to the Kerberos database. Sometimes, however, the entries in the

database must be modified, such as when adding new principals or changing a principal's key. This is done using a protocol between a client and a third Kerberos server, the Kerberos Administration Server (KADM). The administration protocol is not described in this document. There is also a protocol for maintaining multiple copies of the Kerberos database, but this can be considered an implementation detail and may vary to support different database technologies.

1.1. Cross-Realm Operation

The Kerberos protocol is designed to operate across organizational boundaries. A client in one organization can be authenticated to a server in another. Each organization wishing to run a Kerberos server establishes its own "realm". The name of the realm in which a client is registered is part of the client's name, and can be used by the end-service to decide whether to honor a request.

By establishing "inter-realm" keys, the administrators of two realms can allow a client authenticated in the local realm to use its authentication remotely (Of course, with appropriate permission the client could arrange registration of a separately-named principal in a remote realm, and engage in normal exchanges with that realm's services. However, for even small numbers of clients this becomes cumbersome, and more automatic methods as described here are necessary). The exchange of inter-realm keys (a separate key may be used for each direction) registers the ticket-granting service of each realm as a principal in the other realm. A client is then able to obtain a ticket-granting ticket for the remote realm's ticket-granting service from its local realm. When that ticket-granting ticket is used, the remote ticket-granting service uses the inter-realm key (which usually differs from its own normal TGS key) to decrypt the ticket-granting ticket, and is thus certain that it was issued by the client's own TGS. Tickets issued by the remote ticket-granting service will indicate to the end-service that the client was authenticated from another realm.

A realm is said to communicate with another realm if the two realms share an inter-realm key, or if the local realm shares an inter-realm key with an intermediate realm that communicates with the remote realm. An authentication path is the sequence of intermediate realms that are transited in communicating from one realm to another.

Realms are typically organized hierarchically. Each realm shares a key with its parent and a different key with each child. If an inter-realm key is not directly shared by two realms, the hierarchical organization allows an authentication path to be easily constructed. If a hierarchical organization is not used, it may be necessary to consult some database in order to construct an

authentication path between realms.

Although realms are typically hierarchical, intermediate realms may be bypassed to achieve cross-realm authentication through alternate authentication paths (these might be established to make communication between two realms more efficient). It is important for the end-service to know which realms were transited when deciding how much faith to place in the authentication process. To facilitate this decision, a field in each ticket contains the names of the realms that were involved in authenticating the client.

1.2. Environmental assumptions

Kerberos imposes a few assumptions on the environment in which it can properly function:

+ "Denial of service" attacks are not solved with Kerberos. There are places in these protocols where an intruder intruder can prevent an application from participating in the proper authentication steps. Detection and solution of such attacks (some of which can appear to be not-uncommon "normal" failure modes for the system) is usually best left to the human administrators and users.

+ Principals must keep their secret keys secret. If an intruder somehow steals a principal's key, it will be able to masquerade as that principal or impersonate any server to the legitimate principal.

+ "Password guessing" attacks are not solved by Kerberos. If a user chooses a poor password, it is possible for an attacker to successfully mount an offline dictionary attack by repeatedly attempting to decrypt, with successive entries from a dictionary, messages obtained which are encrypted under a key derived from the user's password.

+ Each host on the network must have a clock which is "loosely synchronized" to the time of the other hosts; this synchronization is used to reduce the bookkeeping needs of application servers when they do replay detection. The degree of "looseness" can be configured on a per-server basis. If the clocks are synchronized over the network, the clock synchronization protocol must itself be secured from network attackers.

+ Principal identifiers are not recycled on a short-term basis. A typical mode of access control will use access control lists (ACLs) to grant permissions to particular principals. If a

stale ACL entry remains for a deleted principal and the principal identifier is reused, the new principal will inherit rights specified in the stale ACL entry. By not re-using principal identifiers, the danger of inadvertent access is removed.

1.3. Glossary of terms

Below is a list of terms used throughout this document.

Authentication	Verifying the claimed identity of a principal.
Authentication header	A record containing a Ticket and an Authenticator to be presented to a server as part of the authentication process.
Authentication path	A sequence of intermediate realms transited in the authentication process when communicating from one realm to another.
Authenticator	A record containing information that can be shown to have been recently generated using the session key known only by the client and server.
Authorization	The process of determining whether a client may use a service, which objects the client is allowed to access, and the type of access allowed for each.
Capability	A token that grants the bearer permission to access an object or service. In Kerberos, this might be a ticket whose use is restricted by the contents of the authorization data field, but which lists no network addresses, together with the session key necessary to use the ticket.

Ciphertext	The output of an encryption function. Encryption transforms plaintext into ciphertext.
Client	A process that makes use of a network service on behalf of a user. Note that in some cases a Server may itself be a client of some other server (e.g., a print server may be a client of a file server).
Credentials	A ticket plus the secret session key necessary to successfully use that ticket in an authentication exchange.
KDC	Key Distribution Center, a network service that supplies tickets and temporary session keys; or an instance of that service or the host on which it runs. The KDC services both initial ticket and ticket-granting ticket requests. The initial ticket portion is sometimes referred to as the Authentication Server (or service). The ticket-granting ticket portion is sometimes referred to as the ticket-granting server (or service).
Kerberos	Aside from the 3-headed dog guarding Hades, the name given to Project Athena's authentication service, the protocol used by that service, or the code used to implement the authentication service.
Plaintext	The input to an encryption function or the output of a decryption function. Decryption transforms ciphertext into plaintext.
Principal	A uniquely named client or server instance that participates in a network communication.

Principal identifier	The name used to uniquely identify each different principal.
Seal	To encipher a record containing several fields in such a way that the fields cannot be individually replaced without either knowledge of the encryption key or leaving evidence of tampering.
Secret key	An encryption key shared by a principal and the KDC, distributed outside the bounds of the system, with a long lifetime. In the case of a human user's principal, the secret key is derived from a password.
Server	A particular Principal which provides a resource to network clients.
Service	A resource provided to network clients; often provided by more than one server (for example, remote file service).
Session key	A temporary encryption key used between two principals, with a lifetime limited to the duration of a single login "session".
Sub-session key	A temporary encryption key used between two principals, selected and exchanged by the principals using the session key, and with a lifetime limited to the duration of a single association.
Ticket	A record that helps a client authenticate itself to a server; it contains the client's identity, a session key, a timestamp, and other information, all sealed using the server's secret key. It only serves to authenticate a client when presented along with a fresh Authenticator.

2. Ticket flag uses and requests

Each Kerberos ticket contains a set of flags which are used to indicate various attributes of that ticket. Most flags may be requested by a client when the ticket is obtained; some are automatically turned on and off by a Kerberos server as required. The following sections explain what the various flags mean, and gives examples of reasons to use such a flag.

2.1. Initial and pre-authenticated tickets

The INITIAL flag indicates that a ticket was issued using the AS protocol and not issued based on a ticket-granting ticket. Application servers that want to require the knowledge of a client's secret key (e.g., a passwordchanging program) can insist that this flag be set in any tickets they accept, and thus be assured that the client's key was recently presented to the application client.

The PRE-AUTHENT and HW-AUTHENT flags provide addition information about the initial authentication, regardless of whether the current ticket was issued directly (in which case INITIAL will also be set) or issued on the basis of a ticket-granting ticket (in which case the INITIAL flag is clear, but the PRE-AUTHENT and HW-AUTHENT flags are carried forward from the ticket-granting ticket).

2.2. Invalid tickets

The INVALID flag indicates that a ticket is invalid. Application servers must reject tickets which have this flag set. A postdated ticket will usually be issued in this form. Invalid tickets must be validated by the KDC before use, by presenting them to the KDC in a TGS request with the VALIDATE option specified. The KDC will only validate tickets after their starttime has passed. The validation is required so that postdated tickets which have been stolen before their starttime can be rendered permanently invalid (through a hot-list mechanism).

2.3. Renewable tickets

Applications may desire to hold tickets which can be valid for long periods of time. However, this can expose their credentials to potential theft for equally long periods, and those stolen credentials would be valid until the expiration time of the ticket(s). Simply using shortlived tickets and obtaining new ones periodically would require the client to have long-term access to its secret key, an even greater risk. Renewable tickets can be used to mitigate the consequences of theft. Renewable tickets have two "expiration times": the first is when the current instance of the

ticket expires, and the second is the latest permissible value for an individual expiration time. An application client must periodically (i.e., before it expires) present a renewable ticket to the KDC, with the RENEW option set in the KDC request. The KDC will issue a new ticket with a new session key and a later expiration time. All other fields of the ticket are left unmodified by the renewal process. When the latest permissible expiration time arrives, the ticket expires permanently. At each renewal, the KDC may consult a hot-list to determine if the ticket had been reported stolen since its last renewal; it will refuse to renew such stolen tickets, and thus the usable lifetime of stolen tickets is reduced.

The RENEWABLE flag in a ticket is normally only interpreted by the ticket-granting service (discussed below in section 3.3). It can usually be ignored by application servers. However, some particularly careful application servers may wish to disallow renewable tickets.

If a renewable ticket is not renewed by its expiration time, the KDC will not renew the ticket. The RENEWABLE flag is reset by default, but a client may request it be set by setting the RENEWABLE option in the KRB_AS_REQ message. If it is set, then the renew-till field in the ticket contains the time after which the ticket may not be renewed.

2.4. Postdated tickets

Applications may occasionally need to obtain tickets for use much later, e.g., a batch submission system would need tickets to be valid at the time the batch job is serviced. However, it is dangerous to hold valid tickets in a batch queue, since they will be on-line longer and more prone to theft. Postdated tickets provide a way to obtain these tickets from the KDC at job submission time, but to leave them "dormant" until they are activated and validated by a further request of the KDC. If a ticket theft were reported in the interim, the KDC would refuse to validate the ticket, and the thief would be foiled.

The MAY-POSTDATE flag in a ticket is normally only interpreted by the ticket-granting service. It can be ignored by application servers. This flag must be set in a ticket-granting ticket in order to issue a postdated ticket based on the presented ticket. It is reset by default; it may be requested by a client by setting the ALLOW-POSTDATE option in the KRB_AS_REQ message. This flag does not allow a client to obtain a postdated ticket-granting ticket; postdated ticket-granting tickets can only by obtained by requesting the postdating in the KRB_AS_REQ message. The life (endtime-starttime) of a postdated ticket will be the remaining life of the ticket-

granting ticket at the time of the request, unless the RENEWABLE option is also set, in which case it can be the full life (endtime-starttime) of the ticket-granting ticket. The KDC may limit how far in the future a ticket may be postdated.

The POSTDATED flag indicates that a ticket has been postdated. The application server can check the authtime field in the ticket to see when the original authentication occurred. Some services may choose to reject postdated tickets, or they may only accept them within a certain period after the original authentication. When the KDC issues a POSTDATED ticket, it will also be marked as INVALID, so that the application client must present the ticket to the KDC to be validated before use.

2.5. Proxiable and proxy tickets

At times it may be necessary for a principal to allow a service to perform an operation on its behalf. The service must be able to take on the identity of the client, but only for a particular purpose. A principal can allow a service to take on the principal's identity for a particular purpose by granting it a proxy.

The PROXIABLE flag in a ticket is normally only interpreted by the ticket-granting service. It can be ignored by application servers. When set, this flag tells the ticket-granting server that it is OK to issue a new ticket (but not a ticket-granting ticket) with a different network address based on this ticket. This flag is set by default.

This flag allows a client to pass a proxy to a server to perform a remote request on its behalf, e.g., a print service client can give the print server a proxy to access the client's files on a particular file server in order to satisfy a print request.

In order to complicate the use of stolen credentials, Kerberos tickets are usually valid from only those network addresses specifically included in the ticket (It is permissible to request or issue tickets with no network addresses specified, but we do not recommend it). For this reason, a client wishing to grant a proxy must request a new ticket valid for the network address of the service to be granted the proxy.

The PROXY flag is set in a ticket by the TGS when it issues a proxy ticket. Application servers may check this flag and require additional authentication from the agent presenting the proxy in order to provide an audit trail.

2.6. Forwardable tickets

Authentication forwarding is an instance of the proxy case where the service is granted complete use of the client's identity. An example where it might be used is when a user logs in to a remote system and wants authentication to work from that system as if the login were local.

The FORWARDABLE flag in a ticket is normally only interpreted by the ticket-granting service. It can be ignored by application servers. The FORWARDABLE flag has an interpretation similar to that of the PROXIABLE flag, except ticket-granting tickets may also be issued with different network addresses. This flag is reset by default, but users may request that it be set by setting the FORWARDABLE option in the AS request when they request their initial ticket-granting ticket.

This flag allows for authentication forwarding without requiring the user to enter a password again. If the flag is not set, then authentication forwarding is not permitted, but the same end result can still be achieved if the user engages in the AS exchange with the requested network addresses and supplies a password.

The FORWARDED flag is set by the TGS when a client presents a ticket with the FORWARDABLE flag set and requests it be set by specifying the FORWARDED KDC option and supplying a set of addresses for the new ticket. It is also set in all tickets issued based on tickets with the FORWARDED flag set. Application servers may wish to process FORWARDED tickets differently than non-FORWARDED tickets.

2.7. Other KDC options

There are two additional options which may be set in a client's request of the KDC. The RENEWABLE-OK option indicates that the client will accept a renewable ticket if a ticket with the requested life cannot otherwise be provided. If a ticket with the requested life cannot be provided, then the KDC may issue a renewable ticket with a renew-till equal to the the requested endtime. The value of the renew-till field may still be adjusted by site-determined limits or limits imposed by the individual principal or server.

The ENC-TKT-IN-SKEY option is honored only by the ticket-granting service. It indicates that the to-be-issued ticket for the end server is to be encrypted in the session key from the additional ticket-granting ticket provided with the request. See section 3.3.3 for specific details.

3. Message Exchanges

The following sections describe the interactions between network clients and servers and the messages involved in those exchanges.

3.1. The Authentication Service Exchange

Summary

Message direction	Message type	Section
1. Client to Kerberos	KRB_AS_REQ	5.4.1
2. Kerberos to client	KRB_AS_REP or KRB_ERROR	5.4.2 5.9.1

The Authentication Service (AS) Exchange between the client and the Kerberos Authentication Server is usually initiated by a client when it wishes to obtain authentication credentials for a given server but currently holds no credentials. The client's secret key is used for encryption and decryption. This exchange is typically used at the initiation of a login session, to obtain credentials for a Ticket-Granting Server, which will subsequently be used to obtain credentials for other servers (see section 3.3) without requiring further use of the client's secret key. This exchange is also used to request credentials for services which must not be mediated through the Ticket-Granting Service, but rather require a principal's secret key, such as the password-changing service. (The password-changing request must not be honored unless the requester can provide the old password (the user's current secret key). Otherwise, it would be possible for someone to walk up to an unattended session and change another user's password.) This exchange does not by itself provide any assurance of the the identity of the user. (To authenticate a user logging on to a local system, the credentials obtained in the AS exchange may first be used in a TGS exchange to obtain credentials for a local server. Those credentials must then be verified by the local server through successful completion of the Client/Server exchange.)

The exchange consists of two messages: KRB_AS_REQ from the client to Kerberos, and KRB_AS_REP or KRB_ERROR in reply. The formats for these messages are described in sections 5.4.1, 5.4.2, and 5.9.1.

In the request, the client sends (in cleartext) its own identity and the identity of the server for which it is requesting credentials. The response, KRB_AS_REP, contains a ticket for the client to present to the server, and a session key that will be shared by the client and the server. The session key and additional information are encrypted in the client's secret key. The KRB_AS_REP message contains information which can be used to detect replays, and to

associate it with the message to which it replies. Various errors can occur; these are indicated by an error response (KRB_ERROR) instead of the KRB_AS_REP response. The error message is not encrypted. The KRB_ERROR message also contains information which can be used to associate it with the message to which it replies. The lack of encryption in the KRB_ERROR message precludes the ability to detect replays or fabrications of such messages.

In the normal case the authentication server does not know whether the client is actually the principal named in the request. It simply sends a reply without knowing or caring whether they are the same. This is acceptable because nobody but the principal whose identity was given in the request will be able to use the reply. Its critical information is encrypted in that principal's key. The initial request supports an optional field that can be used to pass additional information that might be needed for the initial exchange. This field may be used for preauthentication if desired, but the mechanism is not currently specified.

3.1.1. Generation of KRB_AS_REQ message

The client may specify a number of options in the initial request. Among these options are whether preauthentication is to be performed; whether the requested ticket is to be renewable, proxiable, or forwardable; whether it should be postdated or allow postdating of derivative tickets; and whether a renewable ticket will be accepted in lieu of a non-renewable ticket if the requested ticket expiration date cannot be satisfied by a nonrenewable ticket (due to configuration constraints; see section 4). See section A.1 for pseudocode.

The client prepares the KRB_AS_REQ message and sends it to the KDC.

3.1.2. Receipt of KRB_AS_REQ message

If all goes well, processing the KRB_AS_REQ message will result in the creation of a ticket for the client to present to the server. The format for the ticket is described in section 5.3.1. The contents of the ticket are determined as follows.

3.1.3. Generation of KRB_AS_REP message

The authentication server looks up the client and server principals named in the KRB_AS_REQ in its database, extracting their respective keys. If required, the server pre-authenticates the request, and if the pre-authentication check fails, an error message with the code KDC_ERR_PREAUTH_FAILED is returned. If the server cannot accommodate the requested encryption type, an error message with code

KDC_ERR_ETYPE_NOSUPP is returned. Otherwise it generates a "random" session key ("Random" means that, among other things, it should be impossible to guess the next session key based on knowledge of past session keys. This can only be achieved in a pseudo-random number generator if it is based on cryptographic principles. It would be more desirable to use a truly random number generator, such as one based on measurements of random physical phenomena.).

If the requested start time is absent or indicates a time in the past, then the start time of the ticket is set to the authentication server's current time. If it indicates a time in the future, but the POSTDATED option has not been specified, then the error KDC_ERR_CANNOT_POSTDATE is returned. Otherwise the requested start time is checked against the policy of the local realm (the administrator might decide to prohibit certain types or ranges of postdated tickets), and if acceptable, the ticket's start time is set as requested and the INVALID flag is set in the new ticket. The postdated ticket must be validated before use by presenting it to the KDC after the start time has been reached.

The expiration time of the ticket will be set to the minimum of the following:

+The expiration time (endtime) requested in the KRB_AS_REQ message.

+The ticket's start time plus the maximum allowable lifetime associated with the client principal (the authentication server's database includes a maximum ticket lifetime field in each principal's record; see section 4).

+The ticket's start time plus the maximum allowable lifetime associated with the server principal.

+The ticket's start time plus the maximum lifetime set by the policy of the local realm.

If the requested expiration time minus the start time (as determined above) is less than a site-determined minimum lifetime, an error message with code KDC_ERR_NEVER_VALID is returned. If the requested expiration time for the ticket exceeds what was determined as above, and if the "RENEWABLE-OK" option was requested, then the "RENEWABLE" flag is set in the new ticket, and the renew-till value is set as if the "RENEWABLE" option were requested (the field and option names are described fully in section 5.4.1). If the RENEWABLE option has been requested or if the RENEWABLE-OK option has been set and a renewable ticket is to be issued, then the renew-till field is set to the minimum of:

+Its requested value.

+The start time of the ticket plus the minimum of the two maximum renewable lifetimes associated with the principals' database entries.

+The start time of the ticket plus the maximum renewable lifetime set by the policy of the local realm.

The flags field of the new ticket will have the following options set if they have been requested and if the policy of the local realm allows: FORWARDABLE, MAY-POSTDATE, POSTDATED, PROXIABLE, RENEWABLE. If the new ticket is postdated (the start time is in the future), its INVALID flag will also be set.

If all of the above succeed, the server formats a KRB_AS_REP message (see section 5.4.2), copying the addresses in the request into the caddr of the response, placing any required pre-authentication data into the padata of the response, and encrypts the ciphertext part in the client's key using the requested encryption method, and sends it to the client. See section A.2 for pseudocode.

3.1.4. Generation of KRB_ERROR message

Several errors can occur, and the Authentication Server responds by returning an error message, KRB_ERROR, to the client, with the error-code and e-text fields set to appropriate values. The error message contents and details are described in Section 5.9.1.

3.1.5. Receipt of KRB_AS_REP message

If the reply message type is KRB_AS_REP, then the client verifies that the cname and crealm fields in the cleartext portion of the reply match what it requested. If any padata fields are present, they may be used to derive the proper secret key to decrypt the message. The client decrypts the encrypted part of the response using its secret key, verifies that the nonce in the encrypted part matches the nonce it supplied in its request (to detect replays). It also verifies that the sname and srealm in the response match those in the request, and that the host address field is also correct. It then stores the ticket, session key, start and expiration times, and other information for later use. The key-expiration field from the encrypted part of the response may be checked to notify the user of impending key expiration (the client program could then suggest remedial action, such as a password change). See section A.3 for pseudocode.

Proper decryption of the KRB_AS_REP message is not sufficient to

verify the identity of the user; the user and an attacker could cooperate to generate a KRB_AS_REP format message which decrypts properly but is not from the proper KDC. If the host wishes to verify the identity of the user, it must require the user to present application credentials which can be verified using a securely-stored secret key. If those credentials can be verified, then the identity of the user can be assured.

3.1.6. Receipt of KRB_ERROR message

If the reply message type is KRB_ERROR, then the client interprets it as an error and performs whatever application-specific tasks are necessary to recover.

3.2. The Client/Server Authentication Exchange

Summary

Message direction	Message type	Section
Client to Application server	KRB_AP_REQ	5.5.1
[optional] Application server to client	KRB_AP_REP or KRB_ERROR	5.5.2 5.9.1

The client/server authentication (CS) exchange is used by network applications to authenticate the client to the server and vice versa. The client must have already acquired credentials for the server using the AS or TGS exchange.

3.2.1. The KRB_AP_REQ message

The KRB_AP_REQ contains authentication information which should be part of the first message in an authenticated transaction. It contains a ticket, an authenticator, and some additional bookkeeping information (see section 5.5.1 for the exact format). The ticket by itself is insufficient to authenticate a client, since tickets are passed across the network in cleartext(Tickets contain both an encrypted and unencrypted portion, so cleartext here refers to the entire unit, which can be copied from one message and replayed in another without any cryptographic skill.), so the authenticator is used to prevent invalid replay of tickets by proving to the server that the client knows the session key of the ticket and thus is entitled to use it. The KRB_AP_REQ message is referred to elsewhere as the "authentication header."

3.2.2. Generation of a KRB_AP_REQ message

When a client wishes to initiate authentication to a server, it obtains (either through a credentials cache, the AS exchange, or the

TGS exchange) a ticket and session key for the desired service. The client may re-use any tickets it holds until they expire. The client then constructs a new Authenticator from the the system time, its name, and optionally an application specific checksum, an initial sequence number to be used in KRB_SAFE or KRB_PRIV messages, and/or a session subkey to be used in negotiations for a session key unique to this particular session. Authenticators may not be re-used and will be rejected if replayed to a server (Note that this can make applications based on unreliable transports difficult to code correctly, if the transport might deliver duplicated messages. In such cases, a new authenticator must be generated for each retry.). If a sequence number is to be included, it should be randomly chosen so that even after many messages have been exchanged it is not likely to collide with other sequence numbers in use.

The client may indicate a requirement of mutual authentication or the use of a session-key based ticket by setting the appropriate flag(s) in the ap-options field of the message.

The Authenticator is encrypted in the session key and combined with the ticket to form the KRB_AP_REQ message which is then sent to the end server along with any additional application-specific information. See section A.9 for pseudocode.

3.2.3. Receipt of KRB_AP_REQ message

Authentication is based on the server's current time of day (clocks must be loosely synchronized), the authenticator, and the ticket. Several errors are possible. If an error occurs, the server is expected to reply to the client with a KRB_ERROR message. This message may be encapsulated in the application protocol if its "raw" form is not acceptable to the protocol. The format of error messages is described in section 5.9.1.

The algorithm for verifying authentication information is as follows. If the message type is not KRB_AP_REQ, the server returns the KRB_AP_ERR_MSG_TYPE error. If the key version indicated by the Ticket in the KRB_AP_REQ is not one the server can use (e.g., it indicates an old key, and the server no longer possesses a copy of the old key), the KRB_AP_ERR_BADKEYVER error is returned. If the USE-SESSION-KEY flag is set in the ap-options field, it indicates to the server that the ticket is encrypted in the session key from the server's ticket-granting ticket rather than its secret key (This is used for user-to-user authentication as described in [6]). Since it is possible for the server to be registered in multiple realms, with different keys in each, the srealm field in the unencrypted portion of the ticket in the KRB_AP_REQ is used to specify which secret key the server should use to decrypt that ticket. The KRB_AP_ERR_NOKEY

error code is returned if the server doesn't have the proper key to decipher the ticket.

The ticket is decrypted using the version of the server's key specified by the ticket. If the decryption routines detect a modification of the ticket (each encryption system must provide safeguards to detect modified ciphertext; see section 6), the KRB_AP_ERR_BAD_INTEGRITY error is returned (chances are good that different keys were used to encrypt and decrypt).

The authenticator is decrypted using the session key extracted from the decrypted ticket. If decryption shows it to have been modified, the KRB_AP_ERR_BAD_INTEGRITY error is returned. The name and realm of the client from the ticket are compared against the same fields in the authenticator. If they don't match, the KRB_AP_ERR_BADMATCH error is returned (they might not match, for example, if the wrong session key was used to encrypt the authenticator). The addresses in the ticket (if any) are then searched for an address matching the operating-system reported address of the client. If no match is found or the server insists on ticket addresses but none are present in the ticket, the KRB_AP_ERR_BADADDR error is returned.

If the local (server) time and the client time in the authenticator differ by more than the allowable clock skew (e.g., 5 minutes), the KRB_AP_ERR_SKEW error is returned. If the server name, along with the client name, time and microsecond fields from the Authenticator match any recently-seen such tuples, the KRB_AP_ERR_REPEAT error is returned (Note that the rejection here is restricted to authenticators from the same principal to the same server. Other client principals communicating with the same server principal should not be have their authenticators rejected if the time and microsecond fields happen to match some other client's authenticator.). The server must remember any authenticator presented within the allowable clock skew, so that a replay attempt is guaranteed to fail. If a server loses track of any authenticator presented within the allowable clock skew, it must reject all requests until the clock skew interval has passed. This assures that any lost or re-played authenticators will fall outside the allowable clock skew and can no longer be successfully replayed (If this is not done, an attacker could conceivably record the ticket and authenticator sent over the network to a server, then disable the client's host, pose as the disabled host, and replay the ticket and authenticator to subvert the authentication.). If a sequence number is provided in the authenticator, the server saves it for later use in processing KRB_SAFE and/or KRB_PRIV messages. If a subkey is present, the server either saves it for later use or uses it to help generate its own choice for a subkey to be returned in a KRB_AP_REP message.

The server computes the age of the ticket: local (server) time minus the start time inside the Ticket. If the start time is later than the current time by more than the allowable clock skew or if the INVALID flag is set in the ticket, the KRB_AP_ERR_TKT_NYV error is returned. Otherwise, if the current time is later than end time by more than the allowable clock skew, the KRB_AP_ERR_TKT_EXPIRED error is returned.

If all these checks succeed without an error, the server is assured that the client possesses the credentials of the principal named in the ticket and thus, the client has been authenticated to the server. See section A.10 for pseudocode.

3.2.4. Generation of a KRB_AP_REP message

Typically, a client's request will include both the authentication information and its initial request in the same message, and the server need not explicitly reply to the KRB_AP_REQ. However, if mutual authentication (not only authenticating the client to the server, but also the server to the client) is being performed, the KRB_AP_REQ message will have MUTUAL-REQUIRED set in its ap-options field, and a KRB_AP_REP message is required in response. As with the error message, this message may be encapsulated in the application protocol if its "raw" form is not acceptable to the application's protocol. The timestamp and microsecond field used in the reply must be the client's timestamp and microsecond field (as provided in the authenticator). [Note: In the Kerberos version 4 protocol, the timestamp in the reply was the client's timestamp plus one. This is not necessary in version 5 because version 5 messages are formatted in such a way that it is not possible to create the reply by judicious message surgery (even in encrypted form) without knowledge of the appropriate encryption keys.] If a sequence number is to be included, it should be randomly chosen as described above for the authenticator. A subkey may be included if the server desires to negotiate a different subkey. The KRB_AP_REP message is encrypted in the session key extracted from the ticket. See section A.11 for pseudocode.

3.2.5. Receipt of KRB_AP_REP message

If a KRB_AP_REP message is returned, the client uses the session key from the credentials obtained for the server (Note that for encrypting the KRB_AP_REP message, the sub-session key is not used, even if present in the Authenticator.) to decrypt the message, and verifies that the timestamp and microsecond fields match those in the Authenticator it sent to the server. If they match, then the client is assured that the server is genuine. The sequence number and subkey (if present) are retained for later use. See section A.12 for

pseudocode.

3.2.6. Using the encryption key

After the KRB_AP_REQ/KRB_AP_REP exchange has occurred, the client and server share an encryption key which can be used by the application. The "true session key" to be used for KRB_PRIV, KRB_SAFE, or other application-specific uses may be chosen by the application based on the subkeys in the KRB_AP_REP message and the authenticator (Implementations of the protocol may wish to provide routines to choose subkeys based on session keys and random numbers and to orchestrate a negotiated key to be returned in the KRB_AP_REP message.). In some cases, the use of this session key will be implicit in the protocol; in others the method of use must be chosen from a several alternatives. We leave the protocol negotiations of how to use the key (e.g., selecting an encryption or checksum type) to the application programmer; the Kerberos protocol does not constrain the implementation options.

With both the one-way and mutual authentication exchanges, the peers should take care not to send sensitive information to each other without proper assurances. In particular, applications that require privacy or integrity should use the KRB_AP_REP or KRB_ERROR responses from the server to client to assure both client and server of their peer's identity. If an application protocol requires privacy of its messages, it can use the KRB_PRIV message (section 3.5). The KRB_SAFE message (section 3.4) can be used to assure integrity.

3.3. The Ticket-Granting Service (TGS) Exchange

Summary

Message direction	Message type	Section
1. Client to Kerberos	KRB_TGS_REQ	5.4.1
2. Kerberos to client	KRB_TGS_REP or KRB_ERROR	5.4.2 5.9.1

The TGS exchange between a client and the Kerberos Ticket-Granting Server is initiated by a client when it wishes to obtain authentication credentials for a given server (which might be registered in a remote realm), when it wishes to renew or validate an existing ticket, or when it wishes to obtain a proxy ticket. In the first case, the client must already have acquired a ticket for the Ticket-Granting Service using the AS exchange (the ticket-granting ticket is usually obtained when a client initially authenticates to the system, such as when a user logs in). The message format for the TGS exchange is almost identical to that for the AS exchange. The primary difference is that encryption and decryption in the TGS

exchange does not take place under the client's key. Instead, the session key from the ticket-granting ticket or renewable ticket, or sub-session key from an Authenticator is used. As is the case for all application servers, expired tickets are not accepted by the TGS, so once a renewable or ticket-granting ticket expires, the client must use a separate exchange to obtain valid tickets.

The TGS exchange consists of two messages: A request (KRB_TGS_REQ) from the client to the Kerberos Ticket-Granting Server, and a reply (KRB_TGS_REP or KRB_ERROR). The KRB_TGS_REQ message includes information authenticating the client plus a request for credentials. The authentication information consists of the authentication header (KRB_AP_REQ) which includes the client's previously obtained ticket-granting, renewable, or invalid ticket. In the ticket-granting ticket and proxy cases, the request may include one or more of: a list of network addresses, a collection of typed authorization data to be sealed in the ticket for authorization use by the application server, or additional tickets (the use of which are described later). The TGS reply (KRB_TGS_REP) contains the requested credentials, encrypted in the session key from the ticket-granting ticket or renewable ticket, or if present, in the subsession key from the Authenticator (part of the authentication header). The KRB_ERROR message contains an error code and text explaining what went wrong. The KRB_ERROR message is not encrypted. The KRB_TGS_REP message contains information which can be used to detect replays, and to associate it with the message to which it replies. The KRB_ERROR message also contains information which can be used to associate it with the message to which it replies, but the lack of encryption in the KRB_ERROR message precludes the ability to detect replays or fabrications of such messages.

3.3.1. Generation of KRB_TGS_REQ message

Before sending a request to the ticket-granting service, the client must determine in which realm the application server is registered [Note: This can be accomplished in several ways. It might be known beforehand (since the realm is part of the principal identifier), or it might be stored in a nameserver. Presently, however, this information is obtained from a configuration file. If the realm to be used is obtained from a nameserver, there is a danger of being spoofed if the nameservice providing the realm name is not authenticated. This might result in the use of a realm which has been compromised, and would result in an attacker's ability to compromise the authentication of the application server to the client.]. If the client does not already possess a ticket-granting ticket for the appropriate realm, then one must be obtained. This is first attempted by requesting a ticket-granting ticket for the destination realm from the local Kerberos server (using the

KRB_TGS_REQ message recursively). The Kerberos server may return a TGT for the desired realm in which case one can proceed. Alternatively, the Kerberos server may return a TGT for a realm which is "closer" to the desired realm (further along the standard hierarchical path), in which case this step must be repeated with a Kerberos server in the realm specified in the returned TGT. If neither are returned, then the request must be retried with a Kerberos server for a realm higher in the hierarchy. This request will itself require a ticket-granting ticket for the higher realm which must be obtained by recursively applying these directions.

Once the client obtains a ticket-granting ticket for the appropriate realm, it determines which Kerberos servers serve that realm, and contacts one. The list might be obtained through a configuration file or network service; as long as the secret keys exchanged by realms are kept secret, only denial of service results from a false Kerberos server.

As in the AS exchange, the client may specify a number of options in the KRB_TGS_REQ message. The client prepares the KRB_TGS_REQ message, providing an authentication header as an element of the padata field, and including the same fields as used in the KRB_AS_REQ message along with several optional fields: the enc-authorization-data field for application server use and additional tickets required by some options.

In preparing the authentication header, the client can select a sub-session key under which the response from the Kerberos server will be encrypted (If the client selects a sub-session key, care must be taken to ensure the randomness of the selected subsession key. One approach would be to generate a random number and XOR it with the session key from the ticket-granting ticket.). If the sub-session key is not specified, the session key from the ticket-granting ticket will be used. If the enc-authorization-data is present, it must be encrypted in the sub-session key, if present, from the authenticator portion of the authentication header, or if not present in the session key from the ticket-granting ticket.

Once prepared, the message is sent to a Kerberos server for the destination realm. See section A.5 for pseudocode.

3.3.2. Receipt of KRB_TGS_REQ message

The KRB_TGS_REQ message is processed in a manner similar to the KRB_AS_REQ message, but there are many additional checks to be performed. First, the Kerberos server must determine which server the accompanying ticket is for and it must select the appropriate key to decrypt it. For a normal KRB_TGS_REQ message, it will be for the

ticket granting service, and the TGS's key will be used. If the TGT was issued by another realm, then the appropriate inter-realm key must be used. If the accompanying ticket is not a ticket granting ticket for the current realm, but is for an application server in the current realm, the RENEW, VALIDATE, or PROXY options are specified in the request, and the server for which a ticket is requested is the server named in the accompanying ticket, then the KDC will decrypt the ticket in the authentication header using the key of the server for which it was issued. If no ticket can be found in the padata field, the KDC_ERR_PADATA_TYPE_NOSUPP error is returned.

Once the accompanying ticket has been decrypted, the user-supplied checksum in the Authenticator must be verified against the contents of the request, and the message rejected if the checksums do not match (with an error code of KRB_AP_ERR_MODIFIED) or if the checksum is not keyed or not collision-proof (with an error code of KRB_AP_ERR_INAPP_CKSUM). If the checksum type is not supported, the KDC_ERR_SUMTYPE_NOSUPP error is returned. If the authorization-data are present, they are decrypted using the sub-session key from the Authenticator.

If any of the decryptions indicate failed integrity checks, the KRB_AP_ERR_BAD_INTEGRITY error is returned.

3.3.3. Generation of KRB_TGS_REP message

The KRB_TGS_REP message shares its format with the KRB_AS_REP (KRB_KDC_REP), but with its type field set to KRB_TGS_REP. The detailed specification is in section 5.4.2.

The response will include a ticket for the requested server. The Kerberos database is queried to retrieve the record for the requested server (including the key with which the ticket will be encrypted). If the request is for a ticket granting ticket for a remote realm, and if no key is shared with the requested realm, then the Kerberos server will select the realm "closest" to the requested realm with which it does share a key, and use that realm instead. This is the only case where the response from the KDC will be for a different server than that requested by the client.

By default, the address field, the client's name and realm, the list of transited realms, the time of initial authentication, the expiration time, and the authorization data of the newly-issued ticket will be copied from the ticket-granting ticket (TGT) or renewable ticket. If the transited field needs to be updated, but the transited type is not supported, the KDC_ERR_TRTYPE_NOSUPP error is returned.

If the request specifies an endtime, then the endtime of the new ticket is set to the minimum of (a) that request, (b) the endtime from the TGT, and (c) the starttime of the TGT plus the minimum of the maximum life for the application server and the maximum life for the local realm (the maximum life for the requesting principal was already applied when the TGT was issued). If the new ticket is to be a renewal, then the endtime above is replaced by the minimum of (a) the value of the renew_till field of the ticket and (b) the starttime for the new ticket plus the life (endtimestarttime) of the old ticket.

If the FORWARDED option has been requested, then the resulting ticket will contain the addresses specified by the client. This option will only be honored if the FORWARDABLE flag is set in the TGT. The PROXY option is similar; the resulting ticket will contain the addresses specified by the client. It will be honored only if the PROXIABLE flag in the TGT is set. The PROXY option will not be honored on requests for additional ticket-granting tickets.

If the requested start time is absent or indicates a time in the past, then the start time of the ticket is set to the authentication server's current time. If it indicates a time in the future, but the POSTDATED option has not been specified or the MAY-POSTDATE flag is not set in the TGT, then the error KDC_ERR_CANNOT_POSTDATE is returned. Otherwise, if the ticket-granting ticket has the MAYPOSTDATE flag set, then the resulting ticket will be postdated and the requested starttime is checked against the policy of the local realm. If acceptable, the ticket's start time is set as requested, and the INVALID flag is set. The postdated ticket must be validated before use by presenting it to the KDC after the starttime has been reached. However, in no case may the starttime, endtime, or renew-till time of a newly-issued postdated ticket extend beyond the renew-till time of the ticket-granting ticket.

If the ENC-TKT-IN-SKEY option has been specified and an additional ticket has been included in the request, the KDC will decrypt the additional ticket using the key for the server to which the additional ticket was issued and verify that it is a ticket-granting ticket. If the name of the requested server is missing from the request, the name of the client in the additional ticket will be used. Otherwise the name of the requested server will be compared to the name of the client in the additional ticket and if different, the request will be rejected. If the request succeeds, the session key from the additional ticket will be used to encrypt the new ticket that is issued instead of using the key of the server for which the new ticket will be used (This allows easy implementation of user-to-user authentication [6], which uses ticket-granting ticket session keys in lieu of secret server keys in situations where such secret

keys could be easily compromised.).

If the name of the server in the ticket that is presented to the KDC as part of the authentication header is not that of the ticket-granting server itself, and the server is registered in the realm of the KDC, If the RENEW option is requested, then the KDC will verify that the RENEWABLE flag is set in the ticket and that the renew_till time is still in the future. If the VALIDATE option is rqeuested, the KDC will check that the starttime has passed and the INVALID flag is set. If the PROXY option is requested, then the KDC will check that the PROXIABLE flag is set in the ticket. If the tests succeed, the KDC will issue the appropriate new ticket.

Whenever a request is made to the ticket-granting server, the presented ticket(s) is(are) checked against a hot-list of tickets which have been canceled. This hot-list might be implemented by storing a range of issue dates for "suspect tickets"; if a presented ticket had an authtime in that range, it would be rejected. In this way, a stolen ticket-granting ticket or renewable ticket cannot be used to gain additional tickets (renewals or otherwise) once the theft has been reported. Any normal ticket obtained before it was reported stolen will still be valid (because they require no interaction with the KDC), but only until their normal expiration time.

The ciphertext part of the response in the KRB_TGS_REP message is encrypted in the sub-session key from the Authenticator, if present, or the session key key from the ticket-granting ticket. It is not encrypted using the client's secret key. Furthermore, the client's key's expiration date and the key version number fields are left out since these values are stored along with the client's database record, and that record is not needed to satisfy a request based on a ticket-granting ticket. See section A.6 for pseudocode.

3.3.3.1. Encoding the transited field

If the identity of the server in the TGT that is presented to the KDC as part of the authentication header is that of the ticket-granting service, but the TGT was issued from another realm, the KDC will look up the inter-realm key shared with that realm and use that key to decrypt the ticket. If the ticket is valid, then the KDC will honor the request, subject to the constraints outlined above in the section describing the AS exchange. The realm part of the client's identity will be taken from the ticket-granting ticket. The name of the realm that issued the ticket-granting ticket will be added to the transited field of the ticket to be issued. This is accomplished by reading the transited field from the ticket-granting ticket (which is treated as an unordered set of realm names), adding the new realm to the set,

then constructing and writing out its encoded (shorthand) form (this may involve a rearrangement of the existing encoding).

Note that the ticket-granting service does not add the name of its own realm. Instead, its responsibility is to add the name of the previous realm. This prevents a malicious Kerberos server from intentionally leaving out its own name (it could, however, omit other realms' names).

The names of neither the local realm nor the principal's realm are to be included in the transited field. They appear elsewhere in the ticket and both are known to have taken part in authenticating the principal. Since the endpoints are not included, both local and single-hop inter-realm authentication result in a transited field that is empty.

Because the name of each realm transited is added to this field, it might potentially be very long. To decrease the length of this field, its contents are encoded. The initially supported encoding is optimized for the normal case of inter-realm communication: a hierarchical arrangement of realms using either domain or X.500 style realm names. This encoding (called DOMAIN-X500-COMPRESS) is now described.

Realm names in the transited field are separated by a ",". The ",", "\", trailing "."s, and leading spaces (" ") are special characters, and if they are part of a realm name, they must be quoted in the transited field by preceding them with a "\".

A realm name ending with a "." is interpreted as being prepended to the previous realm. For example, we can encode traversal of EDU, MIT.EDU, ATHENA.MIT.EDU, WASHINGTON.EDU, and CS.WASHINGTON.EDU as:

"EDU,MIT.,ATHENA.,WASHINGTON.EDU,CS.".

Note that if ATHENA.MIT.EDU, or CS.WASHINGTON.EDU were endpoints, that they would not be included in this field, and we would have:

"EDU,MIT.,WASHINGTON.EDU"

A realm name beginning with a "/" is interpreted as being appended to the previous realm (For the purpose of appending, the realm preceding the first listed realm is considered to be the null realm ("")). If it is to stand by itself, then it should be preceded by a space (" "). For example, we can encode traversal of /COM/HP/APOLLO, /COM/HP, /COM, and /COM/DEC as:

"/COM,/HP,/APOLLO, /COM/DEC".

Like the example above, if /COM/HP/APOLLO and /COM/DEC are endpoints, they they would not be included in this field, and we would have:

"/COM,/HP"

A null subfield preceding or following a "," indicates that all realms between the previous realm and the next realm have been traversed (For the purpose of interpreting null subfields, the client's realm is considered to precede those in the transited field, and the server's realm is considered to follow them.). Thus, "," means that all realms along the path between the client and the server have been traversed. ",EDU, /COM," means that that all realms from the client's realm up to EDU (in a domain style hierarchy) have been traversed, and that everything from /COM down to the server's realm in an X.500 style has also been traversed. This could occur if the EDU realm in one hierarchy shares an inter-realm key directly with the /COM realm in another hierarchy.

3.3.4. Receipt of KRB_TGS_REP message

When the KRB_TGS_REP is received by the client, it is processed in the same manner as the KRB_AS_REP processing described above. The primary difference is that the ciphertext part of the response must be decrypted using the session key from the ticket-granting ticket rather than the client's secret key. See section A.7 for pseudocode.

3.4. The KRB_SAFE Exchange

The KRB_SAFE message may be used by clients requiring the ability to detect modifications of messages they exchange. It achieves this by including a keyed collisionproof checksum of the user data and some control information. The checksum is keyed with an encryption key (usually the last key negotiated via subkeys, or the session key if no negotiation has occured).

3.4.1. Generation of a KRB_SAFE message

When an application wishes to send a KRB_SAFE message, it collects its data and the appropriate control information and computes a checksum over them. The checksum algorithm should be some sort of keyed one-way hash function (such as the RSA-MD5-DES checksum algorithm specified in section 6.4.5, or the DES MAC), generated using the sub-session key if present, or the session key. Different algorithms may be selected by changing the checksum type in the message. Unkeyed or non-collision-proof checksums are not suitable for this use.

The control information for the KRB_SAFE message includes both a

timestamp and a sequence number. The designer of an application using the KRB_SAFE message must choose at least one of the two mechanisms. This choice should be based on the needs of the application protocol.

Sequence numbers are useful when all messages sent will be received by one's peer. Connection state is presently required to maintain the session key, so maintaining the next sequence number should not present an additional problem.

If the application protocol is expected to tolerate lost messages without them being resent, the use of the timestamp is the appropriate replay detection mechanism. Using timestamps is also the appropriate mechanism for multi-cast protocols where all of one's peers share a common sub-session key, but some messages will be sent to a subset of one's peers.

After computing the checksum, the client then transmits the information and checksum to the recipient in the message format specified in section 5.6.1.

3.4.2. Receipt of KRB_SAFE message

When an application receives a KRB_SAFE message, it verifies it as follows. If any error occurs, an error code is reported for use by the application.

The message is first checked by verifying that the protocol version and type fields match the current version and KRB_SAFE, respectively. A mismatch generates a KRB_AP_ERR_BADVERSION or KRB_AP_ERR_MSG_TYPE error. The application verifies that the checksum used is a collisionproof keyed checksum, and if it is not, a KRB_AP_ERR_INAPP_CKSUM error is generated. The recipient verifies that the operating system's report of the sender's address matches the sender's address in the message, and (if a recipient address is specified or the recipient requires an address) that one of the recipient's addresses appears as the recipient's address in the message. A failed match for either case generates a KRB_AP_ERR_BADADDR error. Then the timestamp and usec and/or the sequence number fields are checked. If timestamp and usec are expected and not present, or they are present but not current, the KRB_AP_ERR_SKEW error is generated. If the server name, along with the client name, time and microsecond fields from the Authenticator match any recently-seen such tuples, the KRB_AP_ERR_REPEAT error is generated. If an incorrect sequence number is included, or a sequence number is expected but not present, the KRB_AP_ERR_BADORDER error is generated. If neither a timestamp and usec or a sequence number is present, a KRB_AP_ERR_MODIFIED error is generated.

Finally, the checksum is computed over the data and control information, and if it doesn't match the received checksum, a KRB_AP_ERR_MODIFIED error is generated.

If all the checks succeed, the application is assured that the message was generated by its peer and was not modified in transit.

3.5. The KRB_PRIV Exchange

The KRB_PRIV message may be used by clients requiring confidentiality and the ability to detect modifications of exchanged messages. It achieves this by encrypting the messages and adding control information.

3.5.1. Generation of a KRB_PRIV message

When an application wishes to send a KRB_PRIV message, it collects its data and the appropriate control information (specified in section 5.7.1) and encrypts them under an encryption key (usually the last key negotiated via subkeys, or the session key if no negotiation has occured). As part of the control information, the client must choose to use either a timestamp or a sequence number (or both); see the discussion in section 3.4.1 for guidelines on which to use. After the user data and control information are encrypted, the client transmits the ciphertext and some "envelope" information to the recipient.

3.5.2. Receipt of KRB_PRIV message

When an application receives a KRB_PRIV message, it verifies it as follows. If any error occurs, an error code is reported for use by the application.

The message is first checked by verifying that the protocol version and type fields match the current version and KRB_PRIV, respectively. A mismatch generates a KRB_AP_ERR_BADVERSION or KRB_AP_ERR_MSG_TYPE error. The application then decrypts the ciphertext and processes the resultant plaintext. If decryption shows the data to have been modified, a KRB_AP_ERR_BAD_INTEGRITY error is generated. The recipient verifies that the operating system's report of the sender's address matches the sender's address in the message, and (if a recipient address is specified or the recipient requires an address) that one of the recipient's addresses appears as the recipient's address in the message. A failed match for either case generates a KRB_AP_ERR_BADADDR error. Then the timestamp and usec and/or the sequence number fields are checked. If timestamp and usec are expected and not present, or they are present but not current, the KRB_AP_ERR_SKEW error is generated. If the server name, along with

the client name, time and microsecond fields from the Authenticator match any recently-seen such tuples, the KRB_AP_ERR_REPEAT error is generated. If an incorrect sequence number is included, or a sequence number is expected but not present, the KRB_AP_ERR_BADORDER error is generated. If neither a timestamp and usec or a sequence number is present, a KRB_AP_ERR_MODIFIED error is generated.

If all the checks succeed, the application can assume the message was generated by its peer, and was securely transmitted (without intruders able to see the unencrypted contents).

3.6. The KRB_CRED Exchange

The KRB_CRED message may be used by clients requiring the ability to send Kerberos credentials from one host to another. It achieves this by sending the tickets together with encrypted data containing the session keys and other information associated with the tickets.

3.6.1. Generation of a KRB_CRED message

When an application wishes to send a KRB_CRED message it first (using the KRB_TGS exchange) obtains credentials to be sent to the remote host. It then constructs a KRB_CRED message using the ticket or tickets so obtained, placing the session key needed to use each ticket in the key field of the corresponding KrbCredInfo sequence of the encrypted part of the the KRB_CRED message.

Other information associated with each ticket and obtained during the KRB_TGS exchange is also placed in the corresponding KrbCredInfo sequence in the encrypted part of the KRB_CRED message. The current time and, if specifically required by the application the nonce, s-address, and raddress fields, are placed in the encrypted part of the KRB_CRED message which is then encrypted under an encryption key previosuly exchanged in the KRB_AP exchange (usually the last key negotiated via subkeys, or the session key if no negotiation has occured).

3.6.2. Receipt of KRB_CRED message

When an application receives a KRB_CRED message, it verifies it. If any error occurs, an error code is reported for use by the application. The message is verified by checking that the protocol version and type fields match the current version and KRB_CRED, respectively. A mismatch generates a KRB_AP_ERR_BADVERSION or KRB_AP_ERR_MSG_TYPE error. The application then decrypts the ciphertext and processes the resultant plaintext. If decryption shows the data to have been modified, a KRB_AP_ERR_BAD_INTEGRITY error is generated.

If present or required, the recipient verifies that the operating system's report of the sender's address matches the sender's address in the message, and that one of the recipient's addresses appears as the recipient's address in the message. A failed match for either case generates a KRB_AP_ERR_BADADDR error. The timestamp and usec fields (and the nonce field if required) are checked next. If the timestamp and usec are not present, or they are present but not current, the KRB_AP_ERR_SKEW error is generated.

If all the checks succeed, the application stores each of the new tickets in its ticket cache together with the session key and other information in the corresponding KrbCredInfo sequence from the encrypted part of the KRB_CRED message.

4. The Kerberos Database

The Kerberos server must have access to a database containing the principal identifiers and secret keys of principals to be authenticated (The implementation of the Kerberos server need not combine the database and the server on the same machine; it is feasible to store the principal database in, say, a network name service, as long as the entries stored therein are protected from disclosure to and modification by unauthorized parties. However, we recommend against such strategies, as they can make system management and threat analysis quite complex.).

4.1. Database contents

A database entry should contain at least the following fields:

Field	Value
name	Principal's identifier
key	Principal's secret key
p_kvno	Principal's key version
max_life	Maximum lifetime for Tickets
max_renewable_life	Maximum total lifetime for renewable Tickets

The name field is an encoding of the principal's identifier. The key field contains an encryption key. This key is the principal's secret key. (The key can be encrypted before storage under a Kerberos "master key" to protect it in case the database is compromised but the master key is not. In that case, an extra field must be added to indicate the master key version used, see below.) The p_kvno field is the key version number of the principal's secret key. The max_life field contains the maximum allowable lifetime (endtime - starttime) for any Ticket issued for this principal. The max_renewable_life

field contains the maximum allowable total lifetime for any renewable Ticket issued for this principal. (See section 3.1 for a description of how these lifetimes are used in determining the lifetime of a given Ticket.)

A server may provide KDC service to several realms, as long as the database representation provides a mechanism to distinguish between principal records with identifiers which differ only in the realm name.

When an application server's key changes, if the change is routine (i.e., not the result of disclosure of the old key), the old key should be retained by the server until all tickets that had been issued using that key have expired. Because of this, it is possible for several keys to be active for a single principal. Ciphertext encrypted in a principal's key is always tagged with the version of the key that was used for encryption, to help the recipient find the proper key for decryption.

When more than one key is active for a particular principal, the principal will have more than one record in the Kerberos database. The keys and key version numbers will differ between the records (the rest of the fields may or may not be the same). Whenever Kerberos issues a ticket, or responds to a request for initial authentication, the most recent key (known by the Kerberos server) will be used for encryption. This is the key with the highest key version number.

4.2. Additional fields

Project Athena's KDC implementation uses additional fields in its database:

Field	Value
K_kvno	Kerberos' key version
expiration	Expiration date for entry
attributes	Bit field of attributes
mod_date	Timestamp of last modification
mod_name	Modifying principal's identifier

The K_kvno field indicates the key version of the Kerberos master key under which the principal's secret key is encrypted.

After an entry's expiration date has passed, the KDC will return an error to any client attempting to gain tickets as or for the principal. (A database may want to maintain two expiration dates: one for the principal, and one for the principal's current key. This allows password aging to work independently of the principal's

expiration date. However, due to the limited space in the responses, the KDC must combine the key expiration and principal expiration date into a single value called "key_exp", which is used as a hint to the user to take administrative action.)

The attributes field is a bitfield used to govern the operations involving the principal. This field might be useful in conjunction with user registration procedures, for site-specific policy implementations (Project Athena currently uses it for their user registration process controlled by the system-wide database service, Moira [7]), or to identify the "string to key" conversion algorithm used for a principal's key. (See the discussion of the padata field in section 5.4.2 for details on why this can be useful.) Other bits are used to indicate that certain ticket options should not be allowed in tickets encrypted under a principal's key (one bit each): Disallow issuing postdated tickets, disallow issuing forwardable tickets, disallow issuing tickets based on TGT authentication, disallow issuing renewable tickets, disallow issuing proxiable tickets, and disallow issuing tickets for which the principal is the server.

The mod_date field contains the time of last modification of the entry, and the mod_name field contains the name of the principal which last modified the entry.

4.3. Frequently Changing Fields

Some KDC implementations may wish to maintain the last time that a request was made by a particular principal. Information that might be maintained includes the time of the last request, the time of the last request for a ticket-granting ticket, the time of the last use of a ticket-granting ticket, or other times. This information can then be returned to the user in the last-req field (see section 5.2).

Other frequently changing information that can be maintained is the latest expiration time for any tickets that have been issued using each key. This field would be used to indicate how long old keys must remain valid to allow the continued use of outstanding tickets.

4.4. Site Constants

The KDC implementation should have the following configurable constants or options, to allow an administrator to make and enforce policy decisions:

+ The minimum supported lifetime (used to determine whether the KDC_ERR_NEVER_VALID error should be returned). This constant should reflect reasonable expectations of round-trip time to the

KDC, encryption/decryption time, and processing time by the client and target server, and it should allow for a minimum "useful" lifetime.

+ The maximum allowable total (renewable) lifetime of a ticket (renew_till - starttime).

+ The maximum allowable lifetime of a ticket (endtime - starttime).

+ Whether to allow the issue of tickets with empty address fields (including the ability to specify that such tickets may only be issued if the request specifies some authorization_data).

+ Whether proxiable, forwardable, renewable or post-datable tickets are to be issued.

5. Message Specifications

The following sections describe the exact contents and encoding of protocol messages and objects. The ASN.1 base definitions are presented in the first subsection. The remaining subsections specify the protocol objects (tickets and authenticators) and messages. Specification of encryption and checksum techniques, and the fields related to them, appear in section 6.

5.1. ASN.1 Distinguished Encoding Representation

All uses of ASN.1 in Kerberos shall use the Distinguished Encoding Representation of the data elements as described in the X.509 specification, section 8.7 [8].

5.2. ASN.1 Base Definitions

The following ASN.1 base definitions are used in the rest of this section. Note that since the underscore character (_) is not permitted in ASN.1 names, the hyphen (-) is used in its place for the purposes of ASN.1 names.

```
Realm ::=           GeneralString
PrincipalName ::=   SEQUENCE {
                    name-type[0]     INTEGER,
                    name-string[1]   SEQUENCE OF GeneralString
}
```

Kerberos realms are encoded as GeneralStrings. Realms shall not contain a character with the code 0 (the ASCII NUL). Most realms will usually consist of several components separated by periods (.), in the style of Internet Domain Names, or separated by slashes (/) in

the style of X.500 names. Acceptable forms for realm names are specified in section 7. A PrincipalName is a typed sequence of components consisting of the following sub-fields:

name-type This field specifies the type of name that follows. Pre-defined values for this field are specified in section 7.2. The name-type should be treated as a hint. Ignoring the name type, no two names can be the same (i.e., at least one of the components, or the realm, must be different). This constraint may be eliminated in the future.

name-string This field encodes a sequence of components that form a name, each component encoded as a General String. Taken together, a PrincipalName and a Realm form a principal identifier. Most PrincipalNames will have only a few components (typically one or two).

```
KerberosTime ::=   GeneralizedTime
                   -- Specifying UTC time zone (Z)
```

The timestamps used in Kerberos are encoded as GeneralizedTimes. An encoding shall specify the UTC time zone (Z) and shall not include any fractional portions of the seconds. It further shall not include any separators. Example: The only valid format for UTC time 6 minutes, 27 seconds after 9 pm on 6 November 1985 is 19851106210627Z.

```
HostAddress ::=     SEQUENCE  {
                    addr-type[0]            INTEGER,
                    address[1]              OCTET STRING
}

HostAddresses ::=   SEQUENCE OF SEQUENCE {
                    addr-type[0]            INTEGER,
                    address[1]              OCTET STRING
}
```

The host adddress encodings consists of two fields:

addr-type This field specifies the type of address that follows. Pre-defined values for this field are specified in section 8.1.

address This field encodes a single address of type addr-type.

The two forms differ slightly. HostAddress contains exactly one

address; HostAddresses contains a sequence of possibly many addresses.

```
AuthorizationData ::=   SEQUENCE OF SEQUENCE {
                        ad-type[0]              INTEGER,
                        ad-data[1]              OCTET STRING
}
```

ad-data This field contains authorization data to be interpreted according to the value of the corresponding ad-type field.

ad-type This field specifies the format for the ad-data subfield. All negative values are reserved for local use. Non-negative values are reserved for registered use.

```
APOptions ::=   BIT STRING {
                reserved(0),
                use-session-key(1),
                mutual-required(2)
}

TicketFlags ::=   BIT STRING {
                  reserved(0),
                  forwardable(1),
                  forwarded(2),
                  proxiable(3),
                  proxy(4),
                  may-postdate(5),
                  postdated(6),
                  invalid(7),
                  renewable(8),
                  initial(9),
                  pre-authent(10),
                  hw-authent(11)
}

KDCOptions ::=   BIT STRING {
                 reserved(0),
                 forwardable(1),
                 forwarded(2),
                 proxiable(3),
                 proxy(4),
                 allow-postdate(5),
                 postdated(6),
```

```
                                unused7(7),
                                renewable(8),
                                unused9(9),
                                unused10(10),
                                unused11(11),
                                renewable-ok(27),
                                enc-tkt-in-skey(28),
                                renew(30),
                                validate(31)
                        }

        LastReq ::=     SEQUENCE OF SEQUENCE {
                        lr-type[0]              INTEGER,
                        lr-value[1]             KerberosTime
        }
```

lr-type This field indicates how the following lr-value field is to be interpreted. Negative values indicate that the information pertains only to the responding server. Non-negative values pertain to all servers for the realm.

If the lr-type field is zero (0), then no information is conveyed by the lr-value subfield. If the absolute value of the lr-type field is one (1), then the lr-value subfield is the time of last initial request for a TGT. If it is two (2), then the lr-value subfield is the time of last initial request. If it is three (3), then the lr-value subfield is the time of issue for the newest ticket-granting ticket used. If it is four (4), then the lr-value subfield is the time of the last renewal. If it is five (5), then the lr-value subfield is the time of last request (of any type).

lr-value This field contains the time of the last request. The time must be interpreted according to the contents of the accompanying lr-type subfield.

See section 6 for the definitions of Checksum, ChecksumType, EncryptedData, EncryptionKey, EncryptionType, and KeyType.

5.3. Tickets and Authenticators

This section describes the format and encryption parameters for tickets and authenticators. When a ticket or authenticator is included in a protocol message it is treated as an opaque object.

5.3.1. Tickets

A ticket is a record that helps a client authenticate to a service. A Ticket contains the following information:

```
Ticket ::=                  [APPLICATION 1] SEQUENCE {
                            tkt-vno[0]                  INTEGER,
                            realm[1]                    Realm,
                            sname[2]                    PrincipalName,
                            enc-part[3]                 EncryptedData
}
-- Encrypted part of ticket
EncTicketPart ::=     [APPLICATION 3] SEQUENCE {
                      flags[0]              TicketFlags,
                      key[1]                EncryptionKey,
                      crealm[2]             Realm,
                      cname[3]              PrincipalName,
                      transited[4]          TransitedEncoding,
                      authtime[5]           KerberosTime,
                      starttime[6]          KerberosTime OPTIONAL,
                      endtime[7]            KerberosTime,
                      renew-till[8]         KerberosTime OPTIONAL,
                      caddr[9]              HostAddresses OPTIONAL,
                      authorization-data[10]   AuthorizationData OPTIONAL
}
-- encoded Transited field
TransitedEncoding ::=       SEQUENCE {
                            tr-type[0]  INTEGER, -- must be registered
                            contents[1]          OCTET STRING
}
```

The encoding of EncTicketPart is encrypted in the key shared by Kerberos and the end server (the server's secret key). See section 6 for the format of the ciphertext.

tkt-vno This field specifies the version number for the ticket format. This document describes version number 5.

realm This field specifies the realm that issued a ticket. It also serves to identify the realm part of the server's principal identifier. Since a Kerberos server can only issue tickets for servers within its realm, the two will

always be identical.

sname This field specifies the name part of the server's identity.

enc-part This field holds the encrypted encoding of the EncTicketPart sequence.

flags This field indicates which of various options were used or requested when the ticket was issued. It is a bit-field, where the selected options are indicated by the bit being set (1), and the unselected options and reserved fields being reset (0). Bit 0 is the most significant bit. The encoding of the bits is specified in section 5.2. The flags are described in more detail above in section 2. The meanings of the flags are:

Bit(s)	Name	Description
0	RESERVED	Reserved for future expansion of this field.
1	FORWARDABLE	The FORWARDABLE flag is normally only interpreted by the TGS, and can be ignored by end servers. When set, this flag tells the ticket-granting server that it is OK to issue a new ticket- granting ticket with a different network address based on the presented ticket.
2	FORWARDED	When set, this flag indicates that the ticket has either been forwarded or was issued based on authentication involving a forwarded ticket-granting ticket.
3	PROXIABLE	The PROXIABLE flag is normally only interpreted by the TGS, and can be ignored by end servers. The PROXIABLE flag has an interpretation identical to that of the FORWARDABLE flag, except that the PROXIABLE flag tells the ticket-granting server that only non- ticket-granting tickets may be issued with different network addresses.

4	PROXY	When set, this flag indicates that a ticket is a proxy.
5	MAY-POSTDATE	The MAY-POSTDATE flag is normally only interpreted by the TGS, and can be ignored by end servers. This flag tells the ticket-granting server that a post- dated ticket may be issued based on this ticket-granting ticket.
6	POSTDATED	This flag indicates that this ticket has been postdated. The end-service can check the authtime field to see when the original authentication occurred.
7	INVALID	This flag indicates that a ticket is invalid, and it must be validated by the KDC before use. Application servers must reject tickets which have this flag set.
8	RENEWABLE	The RENEWABLE flag is normally only interpreted by the TGS, and can usually be ignored by end servers (some particularly careful servers may wish to disallow renewable tickets). A renewable ticket can be used to obtain a replacement ticket that expires at a later date.
9	INITIAL	This flag indicates that this ticket was issued using the AS protocol, and not issued based on a ticket-granting ticket.
10	PRE-AUTHENT	This flag indicates that during initial authentication, the client was authenticated by the KDC before a ticket was issued. The strength of the preauthentication method is not indicated, but is acceptable to the KDC.
11	HW-AUTHENT	This flag indicates that the protocol employed for initial authentication required the use of hardware expected to be possessed solely by the named

client. The hardware authentication method is selected by the KDC and the strength of the method is not indicated.

12-31 RESERVED Reserved for future use.

key This field exists in the ticket and the KDC response and is used to pass the session key from Kerberos to the application server and the client. The field's encoding is described in section 6.2.

crealm This field contains the name of the realm in which the client is registered and in which initial authentication took place.

cname This field contains the name part of the client's principal identifier.

transited This field lists the names of the Kerberos realms that took part in authenticating the user to whom this ticket was issued. It does not specify the order in which the realms were transited. See section 3.3.3.1 for details on how this field encodes the traversed realms.

authtime This field indicates the time of initial authentication for the named principal. It is the time of issue for the original ticket on which this ticket is based. It is included in the ticket to provide additional information to the end service, and to provide the necessary information for implementation of a 'hot list' service at the KDC. An end service that is particularly paranoid could refuse to accept tickets for which the initial authentication occurred "too far" in the past.

This field is also returned as part of the response from the KDC. When returned as part of the response to initial authentication (KRB_AS_REP), this is the current time on the Kerberos server (It is NOT recommended that this time value be used to adjust the workstation's clock since the workstation cannot reliably determine that such a KRB_AS_REP actually came from the proper KDC in a timely manner.).

starttime This field in the ticket specifies the time after which the ticket is valid. Together with endtime, this field specifies the life of the ticket. If it is absent from the ticket, its value should be treated as that of the

authtime field.

endtime This field contains the time after which the ticket will not be honored (its expiration time). Note that individual services may place their own limits on the life of a ticket and may reject tickets which have not yet expired. As such, this is really an upper bound on the expiration time for the ticket.

renew-till This field is only present in tickets that have the RENEWABLE flag set in the flags field. It indicates the maximum endtime that may be included in a renewal. It can be thought of as the absolute expiration time for the ticket, including all renewals.

caddr This field in a ticket contains zero (if omitted) or more (if present) host addresses. These are the addresses from which the ticket can be used. If there are no addresses, the ticket can be used from any location. The decision by the KDC to issue or by the end server to accept zero-address tickets is a policy decision and is left to the Kerberos and end-service administrators; they may refuse to issue or accept such tickets. The suggested and default policy, however, is that such tickets will only be issued or accepted when additional information that can be used to restrict the use of the ticket is included in the authorization_data field. Such a ticket is a capability.

Network addresses are included in the ticket to make it harder for an attacker to use stolen credentials. Because the session key is not sent over the network in cleartext, credentials can't be stolen simply by listening to the network; an attacker has to gain access to the session key (perhaps through operating system security breaches or a careless user's unattended session) to make use of stolen tickets.

It is important to note that the network address from which a connection is received cannot be reliably determined. Even if it could be, an attacker who has compromised the client's workstation could use the credentials from there. Including the network addresses only makes it more difficult, not impossible, for an attacker to walk off with stolen credentials and then use them from a "safe" location.

authorization-data The authorization-data field is used to pass authorization data from the principal on whose behalf a ticket was issued to the application service. If no authorization data is included, this field will be left out. The data in this field are specific to the end service. It is expected that the field will contain the names of service specific objects, and the rights to those objects. The format for this field is described in section 5.2. Although Kerberos is not concerned with the format of the contents of the subfields, it does carry type information (ad-type).

By using the authorization_data field, a principal is able to issue a proxy that is valid for a specific purpose. For example, a client wishing to print a file can obtain a file server proxy to be passed to the print server. By specifying the name of the file in the authorization_data field, the file server knows that the print server can only use the client's rights when accessing the particular file to be printed.

It is interesting to note that if one specifies the authorization-data field of a proxy and leaves the host addresses blank, the resulting ticket and session key can be treated as a capability. See [9] for some suggested uses of this field.

The authorization-data field is optional and does not have to be included in a ticket.

5.3.2. Authenticators

An authenticator is a record sent with a ticket to a server to certify the client's knowledge of the encryption key in the ticket, to help the server detect replays, and to help choose a "true session key" to use with the particular session. The encoding is encrypted in the ticket's session key shared by the client and the server:

```
-- Unencrypted authenticator
Authenticator ::=    [APPLICATION 2] SEQUENCE    {
              authenticator-vno[0]          INTEGER,
              crealm[1]                     Realm,
              cname[2]                      PrincipalName,
              cksum[3]                      Checksum OPTIONAL,
              cusec[4]                      INTEGER,
              ctime[5]                      KerberosTime,
              subkey[6]                     EncryptionKey OPTIONAL,
              seq-number[7]                 INTEGER OPTIONAL,
```

```
        authorization-data[8]        AuthorizationData OPTIONAL
              }
```

authenticator-vno This field specifies the version number for the format of the authenticator. This document specifies version 5.

crealm and cname These fields are the same as those described for the ticket in section 5.3.1.

cksum This field contains a checksum of the the application data that accompanies the KRB_AP_REQ.

cusec This field contains the microsecond part of the client's timestamp. Its value (before encryption) ranges from 0 to 999999. It often appears along with ctime. The two fields are used together to specify a reasonably accurate timestamp.

ctime This field contains the current time on the client's host.

subkey This field contains the client's choice for an encryption key which is to be used to protect this specific application session. Unless an application specifies otherwise, if this field is left out the session key from the ticket will be used.

seq-number This optional field includes the initial sequence number to be used by the KRB_PRIV or KRB_SAFE messages when sequence numbers are used to detect replays (It may also be used by application specific messages). When included in the authenticator this field specifies the initial sequence number for messages from the client to the server. When included in the AP-REP message, the initial sequence number is that for messages from the server to the client. When used in KRB_PRIV or KRB_SAFE messages, it is incremented by one after each message is sent.

For sequence numbers to adequately support the detection of replays they should be non-repeating, even across connection boundaries. The initial sequence number should be random and uniformly distributed across the full space of possible sequence numbers, so that it cannot be guessed by an attacker and so that it and the successive sequence numbers do not repeat other sequences.

authorization-data This field is the same as described for the ticket in section 5.3.1. It is optional and will only appear when additional restrictions are to be placed on the use of a ticket, beyond those carried in the ticket itself.

5.4. Specifications for the AS and TGS exchanges

This section specifies the format of the messages used in exchange between the client and the Kerberos server. The format of possible error messages appears in section 5.9.1.

5.4.1. KRB_KDC_REQ definition

The KRB_KDC_REQ message has no type of its own. Instead, its type is one of KRB_AS_REQ or KRB_TGS_REQ depending on whether the request is for an initial ticket or an additional ticket. In either case, the message is sent from the client to the Authentication Server to request credentials for a service.

The message fields are:

```
AS-REQ ::=          [APPLICATION 10] KDC-REQ
TGS-REQ ::=         [APPLICATION 12] KDC-REQ

KDC-REQ ::=         SEQUENCE {
           pvno[1]                  INTEGER,
           msg-type[2]              INTEGER,
           padata[3]                SEQUENCE OF PA-DATA OPTIONAL,
           req-body[4]              KDC-REQ-BODY
}

PA-DATA ::=         SEQUENCE {
           padata-type[1]           INTEGER,
           padata-value[2]          OCTET STRING,
                        -- might be encoded AP-REQ
}

KDC-REQ-BODY ::=    SEQUENCE {
            kdc-options[0]          KDCOptions,
            cname[1]                PrincipalName OPTIONAL,
                        -- Used only in AS-REQ
            realm[2]                Realm, -- Server's realm
                        -- Also client's in AS-REQ
            sname[3]                PrincipalName OPTIONAL,
            from[4]                 KerberosTime OPTIONAL,
            till[5]                 KerberosTime,
            rtime[6]                KerberosTime OPTIONAL,
            nonce[7]                INTEGER,
```

```
        etype[8]              SEQUENCE OF INTEGER, -- EncryptionType,
                        -- in preference order
        addresses[9]          HostAddresses OPTIONAL,
        enc-authorization-data[10]   EncryptedData OPTIONAL,
                        -- Encrypted AuthorizationData encoding
        additional-tickets[11]       SEQUENCE OF Ticket OPTIONAL
}
```

The fields in this message are:

pvno This field is included in each message, and specifies the protocol version number. This document specifies protocol version 5.

msg-type This field indicates the type of a protocol message. It will almost always be the same as the application identifier associated with a message. It is included to make the identifier more readily accessible to the application. For the KDC-REQ message, this type will be KRB_AS_REQ or KRB_TGS_REQ.

padata The padata (pre-authentication data) field contains a of authentication information which may be needed before credentials can be issued or decrypted. In the case of requests for additional tickets (KRB_TGS_REQ), this field will include an element with padata-type of PA-TGS-REQ and data of an authentication header (ticket-granting ticket and authenticator). The checksum in the authenticator (which must be collisionproof) is to be computed over the KDC-REQ-BODY encoding. In most requests for initial authentication (KRB_AS_REQ) and most replies (KDC-REP), the padata field will be left out.

This field may also contain information needed by certain extensions to the Kerberos protocol. For example, it might be used to initially verify the identity of a client before any response is returned. This is accomplished with a padata field with padata-type equal to PA-ENC-TIMESTAMP and padata-value defined as follows:

```
padata-type     ::= PA-ENC-TIMESTAMP
padata-value    ::= EncryptedData -- PA-ENC-TS-ENC

PA-ENC-TS-ENC   ::= SEQUENCE {
        patimestamp[0]              KerberosTime, -- client's time
        pausec[1]                   INTEGER OPTIONAL
}
```

with patimestamp containing the client's time and pausec containing the microseconds which may be omitted if a client will not generate more than one request per second. The ciphertext (padata-value) consists of the PA-ENC-TS-ENC sequence, encrypted using the client's secret key.

The padata field can also contain information needed to help the KDC or the client select the key needed for generating or decrypting the response. This form of the padata is useful for supporting the use of certain "smartcards" with Kerberos. The details of such extensions are beyond the scope of this specification. See [10] for additional uses of this field.

padata-type The padata-type element of the padata field indicates the way that the padata-value element is to be interpreted. Negative values of padata-type are reserved for unregistered use; non-negative values are used for a registered interpretation of the element type.

req-body This field is a placeholder delimiting the extent of the remaining fields. If a checksum is to be calculated over the request, it is calculated over an encoding of the KDC-REQ-BODY sequence which is enclosed within the req-body field.

kdc-options This field appears in the KRB_AS_REQ and KRB_TGS_REQ requests to the KDC and indicates the flags that the client wants set on the tickets as well as other information that is to modify the behavior of the KDC. Where appropriate, the name of an option may be the same as the flag that is set by that option. Although in most case, the bit in the options field will be the same as that in the flags field, this is not guaranteed, so it is not acceptable to simply copy the options field to the flags field. There are various checks that must be made before honoring an option anyway.

The kdc_options field is a bit-field, where the selected options are indicated by the bit being set (1), and the unselected options and reserved fields being reset (0). The encoding of the bits is specified in section 5.2. The options are described in more detail above in section 2. The meanings of the options are:

Bit(s)	Name	Description
0	RESERVED	Reserved for future expansion of this field.
1	FORWARDABLE	The FORWARDABLE option indicates that the ticket to be issued is to have its forwardable flag set. It may only be set on the initial request, or in a subsequent request if the ticket-granting ticket on which it is based is also forwardable.
2	FORWARDED	The FORWARDED option is only specified in a request to the ticket-granting server and will only be honored if the ticket-granting ticket in the request has its FORWARDABLE bit set. This option indicates that this is a request for forwarding. The address(es) of the host from which the resulting ticket is to be valid are included in the addresses field of the request.
3	PROXIABLE	The PROXIABLE option indicates that the ticket to be issued is to have its proxiable flag set. It may only be set on the initial request, or in a subsequent request if the ticket-granting ticket on which it is based is also proxiable.
4	PROXY	The PROXY option indicates that this is a request for a proxy. This option will only be honored if the ticket-granting ticket in the request has its PROXIABLE bit set. The address(es) of the host from which the resulting ticket is to be valid are included in the addresses field of the request.
5	ALLOW-POSTDATE	The ALLOW-POSTDATE option indicates that the ticket to be issued is to have its MAY-POSTDATE flag set. It may only be set on the initial request, or in a subsequent request if

		the ticket-granting ticket on which it is based also has its MAY-POSTDATE flag set.
6	POSTDATED	The POSTDATED option indicates that this is a request for a postdated ticket. This option will only be honored if the ticket-granting ticket on which it is based has its MAY-POSTDATE flag set. The resulting ticket will also have its INVALID flag set, and that flag may be reset by a subsequent request to the KDC after the starttime in the ticket has been reached.
7	UNUSED	This option is presently unused.
8	RENEWABLE	The RENEWABLE option indicates that the ticket to be issued is to have its RENEWABLE flag set. It may only be set on the initial request, or when the ticket-granting ticket on which the request is based is also renewable. If this option is requested, then the rtime field in the request contains the desired absolute expiration time for the ticket.
9-26	RESERVED	Reserved for future use.
27	RENEWABLE-OK	The RENEWABLE-OK option indicates that a renewable ticket will be acceptable if a ticket with the requested life cannot otherwise be provided. If a ticket with the requested life cannot be provided, then a renewable ticket may be issued with a renew-till equal to the the requested endtime. The value of the renew-till field may still be limited by local limits, or limits selected by the individual principal or server.
28	ENC-TKT-IN-SKEY	This option is used only by the ticket-granting service. The ENC-TKT-IN-SKEY option indicates that the ticket for the end server is to be

encrypted in the session key from the additional ticket-granting ticket provided.

29	RESERVED	Reserved for future use.
30	RENEW	This option is used only by the ticket-granting service. The RENEW option indicates that the present request is for a renewal. The ticket provided is encrypted in the secret key for the server on which it is valid. This option will only be honored if the ticket to be renewed has its RENEWABLE flag set and if the time in its renew till field has not passed. The ticket to be renewed is passed in the padata field as part of the authentication header.
31	VALIDATE	This option is used only by the ticket-granting service. The VALIDATE option indicates that the request is to validate a postdated ticket. It will only be honored if the ticket presented is postdated, presently has its INVALID flag set, and would be otherwise usable at this time. A ticket cannot be validated before its starttime. The ticket presented for validation is encrypted in the key of the server for which it is valid and is passed in the padata field as part of the authentication header.

cname and sname These fields are the same as those described for the ticket in section 5.3.1. sname may only be absent when the ENC-TKT-IN-SKEY option is specified. If absent, the name of the server is taken from the name of the client in the ticket passed as additional-tickets.

enc-authorization-data The enc-authorization-data, if present (and it can only be present in the TGS_REQ form), is an encoding of the desired authorization-data encrypted under the sub-session key if present in the Authenticator, or alternatively from the session key in the ticket-granting ticket, both from the padata field in the KRB_AP_REQ.

realm This field specifies the realm part of the server's principal identifier. In the AS exchange, this is also the realm part of the client's principal identifier.

from This field is included in the KRB_AS_REQ and KRB_TGS_REQ ticket requests when the requested ticket is to be postdated. It specifies the desired start time for the requested ticket.

till This field contains the expiration date requested by the client in a ticket request.

rtime This field is the requested renew-till time sent from a client to the KDC in a ticket request. It is optional.

nonce This field is part of the KDC request and response. It it intended to hold a random number generated by the client. If the same number is included in the encrypted response from the KDC, it provides evidence that the response is fresh and has not been replayed by an attacker. Nonces must never be re-used. Ideally, it should be gen erated randomly, but if the correct time is known, it may suffice (Note, however, that if the time is used as the nonce, one must make sure that the workstation time is monotonically increasing. If the time is ever reset backwards, there is a small, but finite, probability that a nonce will be reused.).

etype This field specifies the desired encryption algorithm to be used in the response.

addresses This field is included in the initial request for tickets, and optionally included in requests for additional tickets from the ticket-granting server. It specifies the addresses from which the requested ticket is to be valid. Normally it includes the addresses for the client's host. If a proxy is requested, this field will contain other addresses. The contents of this field are usually copied by the KDC into the caddr field of the resulting ticket.

additional-tickets Additional tickets may be optionally included in a request to the ticket-granting server. If the ENC-TKT-IN-SKEY option has been specified, then the session key from the additional ticket will be used in place of the server's key to encrypt the new ticket. If more than one option which requires additional tickets has been specified, then the additional tickets are used in the order specified by the ordering of the options bits (see kdc-options, above).

The application code will be either ten (10) or twelve (12) depending on whether the request is for an initial ticket (AS-REQ) or for an additional ticket (TGS-REQ).

The optional fields (addresses, authorization-data and additional-tickets) are only included if necessary to perform the operation specified in the kdc-options field.

It should be noted that in KRB_TGS_REQ, the protocol version number appears twice and two different message types appear: the KRB_TGS_REQ message contains these fields as does the authentication header (KRB_AP_REQ) that is passed in the padata field.

5.4.2. KRB_KDC_REP definition

The KRB_KDC_REP message format is used for the reply from the KDC for either an initial (AS) request or a subsequent (TGS) request. There is no message type for KRB_KDC_REP. Instead, the type will be either KRB_AS_REP or KRB_TGS_REP. The key used to encrypt the ciphertext part of the reply depends on the message type. For KRB_AS_REP, the ciphertext is encrypted in the client's secret key, and the client's key version number is included in the key version number for the encrypted data. For KRB_TGS_REP, the ciphertext is encrypted in the sub-session key from the Authenticator, or if absent, the session key from the ticket-granting ticket used in the request. In that case, no version number will be present in the EncryptedData sequence.

The KRB_KDC_REP message contains the following fields:

```
AS-REP ::=     [APPLICATION 11] KDC-REP
TGS-REP ::=    [APPLICATION 13] KDC-REP

KDC-REP ::=    SEQUENCE {
               pvno[0]                 INTEGER,
               msg-type[1]             INTEGER,
               padata[2]               SEQUENCE OF PA-DATA OPTIONAL,
               crealm[3]               Realm,
               cname[4]                PrincipalName,
               ticket[5]               Ticket,
               enc-part[6]             EncryptedData
}

EncASRepPart ::=     [APPLICATION 25[25]] EncKDCRepPart
EncTGSRepPart ::=    [APPLICATION 26] EncKDCRepPart

EncKDCRepPart ::=   SEQUENCE {
               key[0]                  EncryptionKey,
               last-req[1]             LastReq,
```

```
        nonce[2]                INTEGER,
        key-expiration[3]       KerberosTime OPTIONAL,
        flags[4]                TicketFlags,
        authtime[5]             KerberosTime,
        starttime[6]            KerberosTime OPTIONAL,
        endtime[7]              KerberosTime,
        renew-till[8]           KerberosTime OPTIONAL,
        srealm[9]               Realm,
        sname[10]               PrincipalName,
        caddr[11]               HostAddresses OPTIONAL
}
```

NOTE: In EncASRepPart, the application code in the encrypted part of a message provides an additional check that the message was decrypted properly.

pvno and msg-type These fields are described above in section 5.4.1. msg-type is either KRB_AS_REP or KRB_TGS_REP.

padata This field is described in detail in section 5.4.1. One possible use for this field is to encode an alternate "mix-in" string to be used with a string-to-key algorithm (such as is described in section 6.3.2). This ability is useful to ease transitions if a realm name needs to change (e.g., when a company is acquired); in such a case all existing password-derived entries in the KDC database would be flagged as needing a special mix-in string until the next password change.

crealm, cname, srealm and sname These fields are the same as those described for the ticket in section 5.3.1.

ticket The newly-issued ticket, from section 5.3.1.

enc-part This field is a place holder for the ciphertext and related information that forms the encrypted part of a message. The description of the encrypted part of the message follows each appearance of this field. The encrypted part is encoded as described in section 6.1.

key This field is the same as described for the ticket in section 5.3.1.

last-req This field is returned by the KDC and specifies the time(s) of the last request by a principal. Depending on what information is available, this might be the last time that a request for a ticket-granting ticket was made, or the last time that a request based on a ticket-granting ticket

was successful. It also might cover all servers for a realm, or just the particular server. Some implementations may display this information to the user to aid in discovering unauthorized use of one's identity. It is similar in spirit to the last login time displayed when logging into timesharing systems.

nonce This field is described above in section 5.4.1.

key-expiration The key-expiration field is part of the response from the KDC and specifies the time that the client's secret key is due to expire. The expiration might be the result of password aging or an account expiration. This field will usually be left out of the TGS reply since the response to the TGS request is encrypted in a session key and no client information need be retrieved from the KDC database. It is up to the application client (usually the login program) to take appropriate action (such as notifying the user) if the expira tion time is imminent.

flags, authtime, starttime, endtime, renew-till and caddr These fields are duplicates of those found in the encrypted portion of the attached ticket (see section 5.3.1), provided so the client may verify they match the intended request and to assist in proper ticket caching. If the message is of type KRB_TGS_REP, the caddr field will only be filled in if the request was for a proxy or forwarded ticket, or if the user is substituting a subset of the addresses from the ticket granting ticket. If the client-requested addresses are not present or not used, then the addresses contained in the ticket will be the same as those included in the ticket-granting ticket.

5.5. Client/Server (CS) message specifications

This section specifies the format of the messages used for the authentication of the client to the application server.

5.5.1. KRB_AP_REQ definition

The KRB_AP_REQ message contains the Kerberos protocol version number, the message type KRB_AP_REQ, an options field to indicate any options in use, and the ticket and authenticator themselves. The KRB_AP_REQ message is often referred to as the "authentication header".

```
AP-REQ ::=      [APPLICATION 14] SEQUENCE {
                pvno[0]                         INTEGER,
                msg-type[1]                     INTEGER,
```

```
                ap-options[2]           APOptions,
                ticket[3]               Ticket,
                authenticator[4]        EncryptedData
}

APOptions ::=   BIT STRING {
                reserved(0),
                use-session-key(1),
                mutual-required(2)
}
```

pvno and msg-type These fields are described above in section 5.4.1. msg-type is KRB_AP_REQ.

ap-options This field appears in the application request (KRB_AP_REQ) and affects the way the request is processed. It is a bit-field, where the selected options are indicated by the bit being set (1), and the unselected options and reserved fields being reset (0). The encoding of the bits is specified in section 5.2. The meanings of the options are:

Bit(s)	Name	Description
0	RESERVED	Reserved for future expansion of this field.
1	USE-SESSION-KEY	The USE-SESSION-KEY option indicates that the ticket the client is presenting to a server is encrypted in the session key from the server's ticket-granting ticket. When this option is not specified, the ticket is encrypted in the server's secret key.
2	MUTUAL-REQUIRED	The MUTUAL-REQUIRED option tells the server that the client requires mutual authentication, and that it must respond with a KRB_AP_REP message.
3-31	RESERVED	Reserved for future use.

ticket This field is a ticket authenticating the client to the server.

authenticator This contains the authenticator, which includes the client's choice of a subkey. Its encoding is described in section 5.3.2.

5.5.2. KRB_AP_REP definition

The KRB_AP_REP message contains the Kerberos protocol version number, the message type, and an encrypted timestamp. The message is sent in in response to an application request (KRB_AP_REQ) where the mutual authentication option has been selected in the ap-options field.

```
AP-REP ::=          [APPLICATION 15] SEQUENCE {
            pvno[0]                  INTEGER,
            msg-type[1]              INTEGER,
            enc-part[2]              EncryptedData
}

EncAPRepPart ::=    [APPLICATION 27]     SEQUENCE {
            ctime[0]                 KerberosTime,
            cusec[1]                 INTEGER,
            subkey[2]                EncryptionKey OPTIONAL,
            seq-number[3]            INTEGER OPTIONAL
}
```

NOTE: in EncAPRepPart, the application code in the encrypted part of a message provides an additional check that the message was decrypted properly.

The encoded EncAPRepPart is encrypted in the shared session key of the ticket. The optional subkey field can be used in an application-arranged negotiation to choose a per association session key.

pvno and msg-type These fields are described above in section 5.4.1. msg-type is KRB_AP_REP.

enc-part This field is described above in section 5.4.2.

ctime This field contains the current time on the client's host.

cusec This field contains the microsecond part of the client's timestamp.

subkey This field contains an encryption key which is to be used to protect this specific application session. See section 3.2.6 for specifics on how this field is used to negotiate a key. Unless an application specifies otherwise, if this field is left out, the sub-session key from the authenticator, or if also left out, the session key from the ticket will be used.

5.5.3. Error message reply

If an error occurs while processing the application request, the KRB_ERROR message will be sent in response. See section 5.9.1 for the format of the error message. The cname and crealm fields may be left out if the server cannot determine their appropriate values from the corresponding KRB_AP_REQ message. If the authenticator was decipherable, the ctime and cusec fields will contain the values from it.

5.6. KRB_SAFE message specification

This section specifies the format of a message that can be used by either side (client or server) of an application to send a tamper-proof message to its peer. It presumes that a session key has previously been exchanged (for example, by using the KRB_AP_REQ/KRB_AP_REP messages).

5.6.1. KRB_SAFE definition

The KRB_SAFE message contains user data along with a collision-proof checksum keyed with the session key. The message fields are:

```
KRB-SAFE ::=        [APPLICATION 20] SEQUENCE {
            pvno[0]                 INTEGER,
            msg-type[1]             INTEGER,
            safe-body[2]            KRB-SAFE-BODY,
            cksum[3]                Checksum
}

KRB-SAFE-BODY ::=   SEQUENCE {
            user-data[0]            OCTET STRING,
            timestamp[1]            KerberosTime OPTIONAL,
            usec[2]                 INTEGER OPTIONAL,
            seq-number[3]           INTEGER OPTIONAL,
            s-address[4]            HostAddress,
            r-address[5]            HostAddress OPTIONAL
}
```

pvno and msg-type These fields are described above in section 5.4.1. msg-type is KRB_SAFE.

safe-body This field is a placeholder for the body of the KRB-SAFE message. It is to be encoded separately and then have the checksum computed over it, for use in the cksum field.

cksum This field contains the checksum of the application data. Checksum details are described in section 6.4. The

checksum is computed over the encoding of the KRB-SAFE-BODY sequence.

user-data This field is part of the KRB_SAFE and KRB_PRIV messages and contain the application specific data that is being passed from the sender to the recipient.

timestamp This field is part of the KRB_SAFE and KRB_PRIV messages. Its contents are the current time as known by the sender of the message. By checking the timestamp, the recipient of the message is able to make sure that it was recently generated, and is not a replay.

usec This field is part of the KRB_SAFE and KRB_PRIV headers. It contains the microsecond part of the timestamp.

seq-number This field is described above in section 5.3.2.

s-address This field specifies the address in use by the sender of the message.

r-address This field specifies the address in use by the recipient of the message. It may be omitted for some uses (such as broadcast protocols), but the recipient may arbitrarily reject such messages. This field along with s-address can be used to help detect messages which have been incorrectly or maliciously delivered to the wrong recipient.

5.7. KRB_PRIV message specification

This section specifies the format of a message that can be used by either side (client or server) of an application to securely and privately send a message to its peer. It presumes that a session key has previously been exchanged (for example, by using the KRB_AP_REQ/KRB_AP_REP messages).

5.7.1. KRB_PRIV definition

The KRB_PRIV message contains user data encrypted in the Session Key. The message fields are:

```
KRB-PRIV ::=        [APPLICATION 21] SEQUENCE {
            pvno[0]                  INTEGER,
            msg-type[1]              INTEGER,
            enc-part[3]              EncryptedData
}
```

```
EncKrbPrivPart ::=   [APPLICATION 28] SEQUENCE {
            user-data[0]              OCTET STRING,
            timestamp[1]              KerberosTime OPTIONAL,
            usec[2]                   INTEGER OPTIONAL,
            seq-number[3]             INTEGER OPTIONAL,
            s-address[4]              HostAddress, -- sender's addr
            r-address[5]              HostAddress OPTIONAL
                                          -- recip's addr
}
```

NOTE: In EncKrbPrivPart, the application code in the encrypted part of a message provides an additional check that the message was decrypted properly.

pvno and msg-type These fields are described above in section 5.4.1. msg-type is KRB_PRIV.

enc-part This field holds an encoding of the EncKrbPrivPart sequence encrypted under the session key (If supported by the encryption method in use, an initialization vector may be passed to the encryption procedure, in order to achieve proper cipher chaining. The initialization vector might come from the last block of the ciphertext from the previous KRB_PRIV message, but it is the application's choice whether or not to use such an initialization vector. If left out, the default initialization vector for the encryption algorithm will be used.). This encrypted encoding is used for the enc-part field of the KRB-PRIV message. See section 6 for the format of the ciphertext.

user-data, timestamp, usec, s-address and r-address These fields are described above in section 5.6.1.

seq-number This field is described above in section 5.3.2.

5.8. KRB_CRED message specification

This section specifies the format of a message that can be used to send Kerberos credentials from one principal to another. It is presented here to encourage a common mechanism to be used by applications when forwarding tickets or providing proxies to subordinate servers. It presumes that a session key has already been exchanged perhaps by using the KRB_AP_REQ/KRB_AP_REP messages.

5.8.1. KRB_CRED definition

The KRB_CRED message contains a sequence of tickets to be sent and information needed to use the tickets, including the session key from

each. The information needed to use the tickets is encryped under an encryption key previously exchanged. The message fields are:

```
KRB-CRED        ::= [APPLICATION 22]   SEQUENCE {
                pvno[0]                INTEGER,
                msg-type[1]            INTEGER, -- KRB_CRED
                tickets[2]             SEQUENCE OF Ticket,
                enc-part[3]            EncryptedData
}

EncKrbCredPart  ::= [APPLICATION 29]   SEQUENCE {
                ticket-info[0]         SEQUENCE OF KrbCredInfo,
                nonce[1]               INTEGER OPTIONAL,
                timestamp[2]           KerberosTime OPTIONAL,
                usec[3]                INTEGER OPTIONAL,
                s-address[4]           HostAddress OPTIONAL,
                r-address[5]           HostAddress OPTIONAL
}

KrbCredInfo     ::=                    SEQUENCE {
                key[0]                 EncryptionKey,
                prealm[1]              Realm OPTIONAL,
                pname[2]               PrincipalName OPTIONAL,
                flags[3]               TicketFlags OPTIONAL,
                authtime[4]            KerberosTime OPTIONAL,
                starttime[5]           KerberosTime OPTIONAL,
                endtime[6]             KerberosTime OPTIONAL
                renew-till[7]          KerberosTime OPTIONAL,
                srealm[8]              Realm OPTIONAL,
                sname[9]               PrincipalName OPTIONAL,
                caddr[10]              HostAddresses OPTIONAL
}
```

pvno and msg-type These fields are described above in section 5.4.1. msg-type is KRB_CRED.

tickets

These are the tickets obtained from the KDC specifically for use by the intended recipient. Successive tickets are paired with the corresponding KrbCredInfo sequence from the enc-part of the KRB-CRED message.

enc-part This field holds an encoding of the EncKrbCredPart sequence encrypted under the session key shared between the sender and the intended recipient. This encrypted encoding is used for the enc-part field of the KRB-CRED message. See section 6 for the format of the ciphertext.

nonce If practical, an application may require the inclusion of a nonce generated by the recipient of the message. If the same value is included as the nonce in the message, it provides evidence that the message is fresh and has not been replayed by an attacker. A nonce must never be re-used; it should be generated randomly by the recipient of the message and provided to the sender of the mes sage in an application specific manner.

timestamp and usec These fields specify the time that the KRB-CRED message was generated. The time is used to provide assurance that the message is fresh.

s-address and r-address These fields are described above in section 5.6.1. They are used optionally to provide additional assurance of the integrity of the KRB-CRED message.

key This field exists in the corresponding ticket passed by the KRB-CRED message and is used to pass the session key from the sender to the intended recipient. The field's encoding is described in section 6.2.

The following fields are optional. If present, they can be associated with the credentials in the remote ticket file. If left out, then it is assumed that the recipient of the credentials already knows their value.

prealm and pname The name and realm of the delegated principal identity.

flags, authtime, starttime, endtime, renew-till, srealm, sname, and caddr These fields contain the values of the corresponding fields from the ticket found in the ticket field. Descriptions of the fields are identical to the descriptions in the KDC-REP message.

5.9. Error message specification

This section specifies the format for the KRB_ERROR message. The fields included in the message are intended to return as much information as possible about an error. It is not expected that all the information required by the fields will be available for all types of errors. If the appropriate information is not available when the message is composed, the corresponding field will be left out of the message.

Note that since the KRB_ERROR message is not protected by any encryption, it is quite possible for an intruder to synthesize or

modify such a message. In particular, this means that the client should not use any fields in this message for security-critical purposes, such as setting a system clock or generating a fresh authenticator. The message can be useful, however, for advising a user on the reason for some failure.

5.9.1. KRB_ERROR definition

The KRB_ERROR message consists of the following fields:

```
KRB-ERROR ::=   [APPLICATION 30] SEQUENCE {
                pvno[0]                 INTEGER,
                msg-type[1]             INTEGER,
                ctime[2]                KerberosTime OPTIONAL,
                cusec[3]                INTEGER OPTIONAL,
                stime[4]                KerberosTime,
                susec[5]                INTEGER,
                error-code[6]           INTEGER,
                crealm[7]               Realm OPTIONAL,
                cname[8]                PrincipalName OPTIONAL,
                realm[9]                Realm, -- Correct realm
                sname[10]               PrincipalName, -- Correct name
                e-text[11]              GeneralString OPTIONAL,
                e-data[12]              OCTET STRING OPTIONAL
}
```

pvno and msg-type These fields are described above in section 5.4.1. msg-type is KRB_ERROR.

ctime This field is described above in section 5.4.1.

cusec This field is described above in section 5.5.2.

stime This field contains the current time on the server. It is of type KerberosTime.

susec This field contains the microsecond part of the server's timestamp. Its value ranges from 0 to 999. It appears along with stime. The two fields are used in conjunction to specify a reasonably accurate timestamp.

error-code This field contains the error code returned by Kerberos or the server when a request fails. To interpret the value of this field see the list of error codes in section 8. Implementations are encouraged to provide for national language support in the display of error messages.

crealm, cname, srealm and sname These fields are described above in

section 5.3.1.

e-text This field contains additional text to help explain the error code associated with the failed request (for example, it might include a principal name which was unknown).

e-data This field contains additional data about the error for use by the application to help it recover from or handle the error. If the errorcode is KDC_ERR_PREAUTH_REQUIRED, then the e-data field will contain an encoding of a sequence of padata fields, each corresponding to an acceptable pre-authentication method and optionally containing data for the method:

```
METHOD-DATA ::=    SEQUENCE of PA-DATA
```

If the error-code is KRB_AP_ERR_METHOD, then the e-data field will contain an encoding of the following sequence:

```
METHOD-DATA ::=    SEQUENCE {
                   method-type[0]    INTEGER,
                   method-data[1]    OCTET STRING OPTIONAL
 }
```

method-type will indicate the required alternate method; method-data will contain any required additional information.

6. Encryption and Checksum Specifications

The Kerberos protocols described in this document are designed to use stream encryption ciphers, which can be simulated using commonly available block encryption ciphers, such as the Data Encryption Standard [11], in conjunction with block chaining and checksum methods [12]. Encryption is used to prove the identities of the network entities participating in message exchanges. The Key Distribution Center for each realm is trusted by all principals registered in that realm to store a secret key in confidence. Proof of knowledge of this secret key is used to verify the authenticity of a principal.

The KDC uses the principal's secret key (in the AS exchange) or a shared session key (in the TGS exchange) to encrypt responses to ticket requests; the ability to obtain the secret key or session key implies the knowledge of the appropriate keys and the identity of the KDC. The ability of a principal to decrypt the KDC response and present a Ticket and a properly formed Authenticator (generated with the session key from the KDC response) to a service verifies the identity of the principal; likewise the ability of the service to

extract the session key from the Ticket and prove its knowledge thereof in a response verifies the identity of the service.

The Kerberos protocols generally assume that the encryption used is secure from cryptanalysis; however, in some cases, the order of fields in the encrypted portions of messages are arranged to minimize the effects of poorly chosen keys. It is still important to choose good keys. If keys are derived from user-typed passwords, those passwords need to be well chosen to make brute force attacks more difficult. Poorly chosen keys still make easy targets for intruders.

The following sections specify the encryption and checksum mechanisms currently defined for Kerberos. The encodings, chaining, and padding requirements for each are described. For encryption methods, it is often desirable to place random information (often referred to as a confounder) at the start of the message. The requirements for a confounder are specified with each encryption mechanism.

Some encryption systems use a block-chaining method to improve the the security characteristics of the ciphertext. However, these chaining methods often don't provide an integrity check upon decryption. Such systems (such as DES in CBC mode) must be augmented with a checksum of the plaintext which can be verified at decryption and used to detect any tampering or damage. Such checksums should be good at detecting burst errors in the input. If any damage is detected, the decryption routine is expected to return an error indicating the failure of an integrity check. Each encryption type is expected to provide and verify an appropriate checksum. The specification of each encryption method sets out its checksum requirements.

Finally, where a key is to be derived from a user's password, an algorithm for converting the password to a key of the appropriate type is included. It is desirable for the string to key function to be one-way, and for the mapping to be different in different realms. This is important because users who are registered in more than one realm will often use the same password in each, and it is desirable that an attacker compromising the Kerberos server in one realm not obtain or derive the user's key in another.

For a discussion of the integrity characteristics of the candidate encryption and checksum methods considered for Kerberos, the the reader is referred to [13].

6.1. Encryption Specifications

The following ASN.1 definition describes all encrypted messages. The enc-part field which appears in the unencrypted part of messages in

section 5 is a sequence consisting of an encryption type, an optional key version number, and the ciphertext.

```
EncryptedData ::=   SEQUENCE {
                    etype[0]      INTEGER, -- EncryptionType
                    kvno[1]       INTEGER OPTIONAL,
                    cipher[2]     OCTET STRING -- ciphertext
}
```

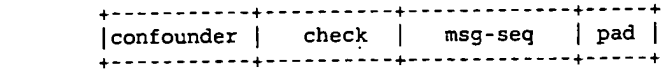
etype This field identifies which encryption algorithm was used to encipher the cipher. Detailed specifications for selected encryption types appear later in this section.

kvno This field contains the version number of the key under which data is encrypted. It is only present in messages encrypted under long lasting keys, such as principals' secret keys.

cipher This field contains the enciphered text, encoded as an OCTET STRING.

The cipher field is generated by applying the specified encryption algorithm to data composed of the message and algorithm-specific inputs. Encryption mechanisms defined for use with Kerberos must take sufficient measures to guarantee the integrity of the plaintext, and we recommend they also take measures to protect against precomputed dictionary attacks. If the encryption algorithm is not itself capable of doing so, the protections can often be enhanced by adding a checksum and a confounder.

The suggested format for the data to be encrypted includes a confounder, a checksum, the encoded plaintext, and any necessary padding. The msg-seq field contains the part of the protocol message described in section 5 which is to be encrypted. The confounder, checksum, and padding are all untagged and untyped, and their length is exactly sufficient to hold the appropriate item. The type and length is implicit and specified by the particular encryption type being used (etype). The format for the data to be encrypted is described in the following diagram:



The format cannot be described in ASN.1, but for those who prefer an ASN.1-like notation:

```
CipherText ::=   ENCRYPTED       SEQUENCE {
        confounder[0]    UNTAGGED OCTET STRING(conf_length)     OPTIONAL,
        check[1]         UNTAGGED OCTET STRING(checksum_length) OPTIONAL,
        msg-seq[2]       MsgSequence,
        pad              UNTAGGED OCTET STRING(pad_length) OPTIONAL
}
```

In the above specification, UNTAGGED OCTET STRING(length) is the notation for an octet string with its tag and length removed. It is not a valid ASN.1 type. The tag bits and length must be removed from the confounder since the purpose of the confounder is so that the message starts with random data, but the tag and its length are fixed. For other fields, the length and tag would be redundant if they were included because they are specified by the encryption type.

One generates a random confounder of the appropriate length, placing it in confounder; zeroes out check; calculates the appropriate checksum over confounder, check, and msg-seq, placing the result in check; adds the necessary padding; then encrypts using the specified encryption type and the appropriate key.

Unless otherwise specified, a definition of an encryption algorithm that specifies a checksum, a length for the confounder field, or an octet boundary for padding uses this ciphertext format (The ordering of the fields in the CipherText is important. Additionally, messages encoded in this format must include a length as part of the msg-seq field. This allows the recipient to verify that the message has not been truncated. Without a length, an attacker could use a chosen plaintext attack to generate a message which could be truncated, while leaving the checksum intact. Note that if the msg-seq is an encoding of an ASN.1 SEQUENCE or OCTET STRING, then the length is part of that encoding.). Those fields which are not specified will be omitted.

In the interest of allowing all implementations using a particular encryption type to communicate with all others using that type, the specification of an encryption type defines any checksum that is needed as part of the encryption process. If an alternative checksum is to be used, a new encryption type must be defined.

Some cryptosystems require additional information beyond the key and the data to be encrypted. For example, DES, when used in cipher-block-chaining mode, requires an initialization vector. If required, the description for each encryption type must specify the source of such additional information.

6.2. Encryption Keys

The sequence below shows the encoding of an encryption key:

```
EncryptionKey ::=   SEQUENCE {
                    keytype[0]    INTEGER,
                    keyvalue[1]   OCTET STRING
}
```

keytype This field specifies the type of encryption key that follows in the keyvalue field. It will almost always correspond to the encryption algorithm used to generate the EncryptedData, though more than one algorithm may use the same type of key (the mapping is many to one). This might happen, for example, if the encryption algorithm uses an alternate checksum algorithm for an integrity check, or a different chaining mechanism.

keyvalue This field contains the key itself, encoded as an octet string.

All negative values for the encryption key type are reserved for local use. All non-negative values are reserved for officially assigned type fields and interpretations.

6.3. Encryption Systems

6.3.1. The NULL Encryption System (null)

If no encryption is in use, the encryption system is said to be the NULL encryption system. In the NULL encryption system there is no checksum, confounder or padding. The ciphertext is simply the plaintext. The NULL Key is used by the null encryption system and is zero octets in length, with keytype zero (0).

6.3.2. DES in CBC mode with a CRC-32 checksum (des-cbc-crc)

The des-cbc-crc encryption mode encrypts information under the Data Encryption Standard [11] using the cipher block chaining mode [12]. A CRC-32 checksum (described in ISO 3309 [14]) is applied to the confounder and message sequence (msg-seq) and placed in the cksum field. DES blocks are 8 bytes. As a result, the data to be encrypted (the concatenation of confounder, checksum, and message) must be padded to an 8 byte boundary before encryption. The details of the encryption of this data are identical to those for the des-cbc-md5 encryption mode.

Note that, since the CRC-32 checksum is not collisionproof, an

attacker could use a probabilistic chosenplaintext attack to generate a valid message even if a confounder is used [13]. The use of collision-proof checksums is recommended for environments where such attacks represent a significant threat. The use of the CRC-32 as the checksum for ticket or authenticator is no longer mandated as an interoperability requirement for Kerberos Version 5 Specification 1 (See section 9.1 for specific details).

6.3.3. DES in CBC mode with an MD4 checksum (des-cbc-md4)

The des-cbc-md4 encryption mode encrypts information under the Data Encryption Standard [11] using the cipher block chaining mode [12]. An MD4 checksum (described in [15]) is applied to the confounder and message sequence (msg-seq) and placed in the cksum field. DES blocks are 8 bytes. As a result, the data to be encrypted (the concatenation of confounder, checksum, and message) must be padded to an 8 byte boundary before encryption. The details of the encryption of this data are identical to those for the descbc-md5 encryption mode.

6.3.4. DES in CBC mode with an MD5 checksum (des-cbc-md5)

The des-cbc-md5 encryption mode encrypts information under the Data Encryption Standard [11] using the cipher block chaining mode [12]. An MD5 checksum (described in [16]) is applied to the confounder and message sequence (msg-seq) and placed in the cksum field. DES blocks are 8 bytes. As a result, the data to be encrypted (the concatenation of confounder, checksum, and message) must be padded to an 8 byte boundary before encryption.

Plaintext and DES ciphtertext are encoded as 8-octet blocks which are concatenated to make the 64-bit inputs for the DES algorithms. The first octet supplies the 8 most significant bits (with the octet's MSbit used as the DES input block's MSbit, etc.), the second octet the next 8 bits, ..., and the eighth octet supplies the 8 least significant bits.

Encryption under DES using cipher block chaining requires an additional input in the form of an initialization vector. Unless otherwise specified, zero should be used as the initialization vector. Kerberos' use of DES requires an 8-octet confounder.

The DES specifications identify some "weak" and "semiweak" keys; those keys shall not be used for encrypting messages for use in Kerberos. Additionally, because of the way that keys are derived for the encryption of checksums, keys shall not be used that yield "weak" or "semi-weak" keys when eXclusive-ORed with the constant F0F0F0F0F0F0F0F0.

A DES key is 8 octets of data, with keytype one (1). This consists of 56 bits of key, and 8 parity bits (one per octet). The key is encoded as a series of 8 octets written in MSB-first order. The bits within the key are also encoded in MSB order. For example, if the encryption key is:
(B1,B2,...,B7,P1,B8,...,B14,P2,B15,...,B49,P7,B50,...,B56,P8) where B1,B2,...,B56 are the key bits in MSB order, and P1,P2,...,P8 are the parity bits, the first octet of the key would be B1,B2,...,B7,P1 (with B1 as the MSbit). [See the FIPS 81 introduction for reference.]

To generate a DES key from a text string (password), the text string normally must have the realm and each component of the principal's name appended(In some cases, it may be necessary to use a different "mix-in" string for compatibility reasons; see the discussion of padata in section 5.4.2.), then padded with ASCII nulls to an 8 byte boundary. This string is then fan-folded and eXclusive-ORed with itself to form an 8 byte DES key. The parity is corrected on the key, and it is used to generate a DES CBC checksum on the initial string (with the realm and name appended). Next, parity is corrected on the CBC checksum. If the result matches a "weak" or "semiweak" key as described in the DES specification, it is eXclusive-ORed with the constant 00000000000000F0. Finally, the result is returned as the key. Pseudocode follows:

```
string_to_key(string,realm,name) {
     odd = 1;
     s = string + realm;
     for(each component in name) {
          s = s + component;
     }
     tempkey = NULL;
     pad(s); /* with nulls to 8 byte boundary */
     for(8byteblock in s) {
          if(odd == 0)  {
              odd = 1;
              reverse(8byteblock)
          }
          else odd = 0;
          tempkey = tempkey XOR 8byteblock;
     }
     fixparity(tempkey);
     key = DES-CBC-check(s,tempkey);
     fixparity(key);
     if(is_weak_key_key(key))
          key = key XOR 0xF0;
     return(key);
}
```

6.4. Checksums

The following is the ASN.1 definition used for a checksum:

```
Checksum ::=   SEQUENCE {
               cksumtype[0]    INTEGER,
               checksum[1]     OCTET STRING
}
```

cksumtype This field indicates the algorithm used to generate the accompanying checksum.

checksum This field contains the checksum itself, encoded as an octet string.

Detailed specification of selected checksum types appear later in this section. Negative values for the checksum type are reserved for local use. All non-negative values are reserved for officially assigned type fields and interpretations.

Checksums used by Kerberos can be classified by two properties: whether they are collision-proof, and whether they are keyed. It is infeasible to find two plaintexts which generate the same checksum value for a collision-proof checksum. A key is required to perturb or initialize the algorithm in a keyed checksum. To prevent message-stream modification by an active attacker, unkeyed checksums should only be used when the checksum and message will be subsequently encrypted (e.g., the checksums defined as part of the encryption algorithms covered earlier in this section). Collision-proof checksums can be made tamper-proof as well if the checksum value is encrypted before inclusion in a message. In such cases, the composition of the checksum and the encryption algorithm must be considered a separate checksum algorithm (e.g., RSA-MD5 encrypted using DES is a new checksum algorithm of type RSA-MD5-DES). For most keyed checksums, as well as for the encrypted forms of collisionproof checksums, Kerberos prepends a confounder before the checksum is calculated.

6.4.1. The CRC-32 Checksum (crc32)

The CRC-32 checksum calculates a checksum based on a cyclic redundancy check as described in ISO 3309 [14]. The resulting checksum is four (4) octets in length. The CRC-32 is neither keyed nor collision-proof. The use of this checksum is not recommended. An attacker using a probabilistic chosen-plaintext attack as described in [13] might be able to generate an alternative message that satisfies the checksum. The use of collision-proof checksums is recommended for environments where such attacks represent a

significant threat.

6.4.2. The RSA MD4 Checksum (rsa-md4)

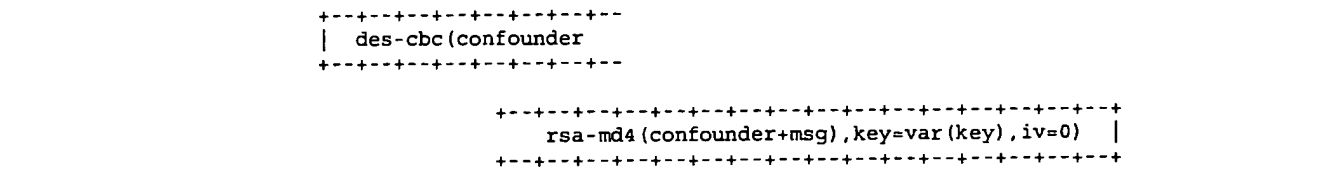
The RSA-MD4 checksum calculates a checksum using the RSA MD4 algorithm [15]. The algorithm takes as input an input message of arbitrary length and produces as output a 128-bit (16 octet) checksum. RSA-MD4 is believed to be collision-proof.

6.4.3. RSA MD4 Cryptographic Checksum Using DES (rsa-md4des)

The RSA-MD4-DES checksum calculates a keyed collisionproof checksum by prepending an 8 octet confounder before the text, applying the RSA MD4 checksum algorithm, and encrypting the confounder and the checksum using DES in cipher-block-chaining (CBC) mode using a variant of the key, where the variant is computed by eXclusive-ORing the key with the constant F0F0F0F0F0F0F0F0 (A variant of the key is used to limit the use of a key to a particular function, separating the functions of generating a checksum from other encryption performed using the session key. The constant F0F0F0F0F0F0F0F0 was chosen because it maintains key parity. The properties of DES precluded the use of the complement. The same constant is used for similar purpose in the Message Integrity Check in the Privacy Enhanced Mail standard.). The initialization vector should be zero. The resulting checksum is 24 octets long (8 octets of which are redundant). This checksum is tamper-proof and believed to be collision-proof.

The DES specifications identify some "weak keys"; those keys shall not be used for generating RSA-MD4 checksums for use in Kerberos.

The format for the checksum is described in the following diagram:



The format cannot be described in ASN.1, but for those who prefer an ASN.1-like notation:

```
rsa-md4-des-checksum ::=   ENCRYPTED        UNTAGGED SEQUENCE {
                           confounder[0]    UNTAGGED OCTET STRING(8),
                           check[1]         UNTAGGED OCTET STRING(16)
}
```

6.4.4. The RSA MD5 Checksum (rsa-md5)

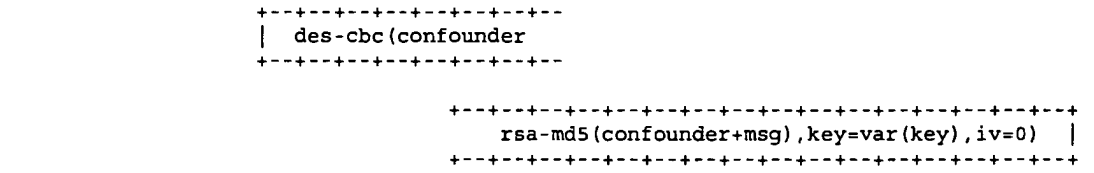
The RSA-MD5 checksum calculates a checksum using the RSA MD5 algorithm [16]. The algorithm takes as input an input message of arbitrary length and produces as output a 128-bit (16 octet) checksum. RSA-MD5 is believed to be collision-proof.

6.4.5. RSA MD5 Cryptographic Checksum Using DES (rsa-md5des)

The RSA-MD5-DES checksum calculates a keyed collisionproof checksum by prepending an 8 octet confounder before the text, applying the RSA MD5 checksum algorithm, and encrypting the confounder and the checksum using DES in cipher-block-chaining (CBC) mode using a variant of the key, where the variant is computed by eXclusive-ORing the key with the constant F0F0F0F0F0F0F0F0. The initialization vector should be zero. The resulting checksum is 24 octets long (8 octets of which are redundant). This checksum is tamper-proof and believed to be collision-proof.

The DES specifications identify some "weak keys"; those keys shall not be used for encrypting RSA-MD5 checksums for use in Kerberos.

The format for the checksum is described in the following diagram:



The format cannot be described in ASN.1, but for those who prefer an ASN.1-like notation:

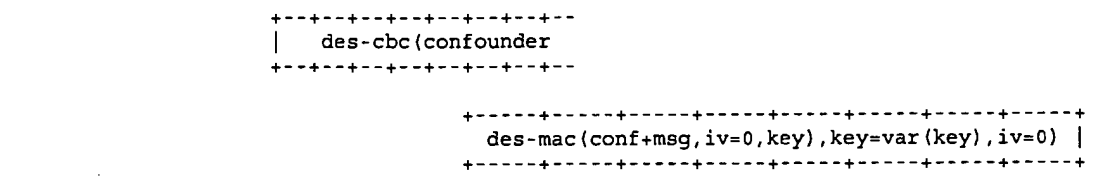
```
rsa-md5-des-checksum ::=   ENCRYPTED       UNTAGGED SEQUENCE {
                           confounder[0]   UNTAGGED OCTET STRING(8),
                           check[1]        UNTAGGED OCTET STRING(16)
}
```

6.4.6. DES cipher-block chained checksum (des-mac)

The DES-MAC checksum is computed by prepending an 8 octet confounder to the plaintext, performing a DES CBC-mode encryption on the result using the key and an initialization vector of zero, taking the last block of the ciphertext, prepending the same confounder and encrypting the pair using DES in cipher-block-chaining (CBC) mode using a a variant of the key, where the variant is computed by

eXclusive-ORing the key with the constant F0F0F0F0F0F0F0F0. The initialization vector should be zero. The resulting checksum is 128 bits (16 octets) long, 64 bits of which are redundant. This checksum is tamper-proof and collision-proof.

The format for the checksum is described in the following diagram:



The format cannot be described in ASN.1, but for those who prefer an ASN.1-like notation:

```
des-mac-checksum ::=    ENCRYPTED       UNTAGGED SEQUENCE {
                        confounder[0]   UNTAGGED OCTET STRING(8),
                        check[1]        UNTAGGED OCTET STRING(8)
}
```

The DES specifications identify some "weak" and "semiweak" keys; those keys shall not be used for generating DES-MAC checksums for use in Kerberos, nor shall a key be used whose veriant is "weak" or "semi-weak".

6.4.7. RSA MD4 Cryptographic Checksum Using DES alternative (rsa-md4-des-k)

The RSA-MD4-DES-K checksum calculates a keyed collision-proof checksum by applying the RSA MD4 checksum algorithm and encrypting the results using DES in cipherblock-chaining (CBC) mode using a DES key as both key and initialization vector. The resulting checksum is 16 octets long. This checksum is tamper-proof and believed to be collision-proof. Note that this checksum type is the old method for encoding the RSA-MD4-DES checksum and it is no longer recommended.

6.4.8. DES cipher-block chained checksum alternative (desmac-k)

The DES-MAC-K checksum is computed by performing a DES CBC-mode encryption of the plaintext, and using the last block of the ciphertext as the checksum value. It is keyed with an encryption key and an initialization vector; any uses which do not specify an additional initialization vector will use the key as both key and initialization vector. The resulting checksum is 64 bits (8 octets) long. This checksum is tamper-proof and collision-proof. Note that

this checksum type is the old method for encoding the DESMAC checksum and it is no longer recommended.

The DES specifications identify some "weak keys"; those keys shall not be used for generating DES-MAC checksums for use in Kerberos.

7. Naming Constraints

7.1. Realm Names

Although realm names are encoded as GeneralStrings and although a realm can technically select any name it chooses, interoperability across realm boundaries requires agreement on how realm names are to be assigned, and what information they imply.

To enforce these conventions, each realm must conform to the conventions itself, and it must require that any realms with which inter-realm keys are shared also conform to the conventions and require the same from its neighbors.

There are presently four styles of realm names: domain, X500, other, and reserved. Examples of each style follow:

```
     domain:   host.subdomain.domain (example)
       X500:   C=US/O=OSF (example)
      other:   NAMETYPE:rest/of.name=without-restrictions (example)
   reserved:   reserved, but will not conflict with above
```

Domain names must look like domain names: they consist of components separated by periods (.) and they contain neither colons (:) nor slashes (/).

X.500 names contain an equal (=) and cannot contain a colon (:) before the equal. The realm names for X.500 names will be string representations of the names with components separated by slashes. Leading and trailing slashes will not be included.

Names that fall into the other category must begin with a prefix that contains no equal (=) or period (.) and the prefix must be followed by a colon (:) and the rest of the name. All prefixes must be assigned before they may be used. Presently none are assigned.

The reserved category includes strings which do not fall into the first three categories. All names in this category are reserved. It is unlikely that names will be assigned to this category unless there is a very strong argument for not using the "other" category.

These rules guarantee that there will be no conflicts between the

various name styles. The following additional constraints apply to the assignment of realm names in the domain and X.500 categories: the name of a realm for the domain or X.500 formats must either be used by the organization owning (to whom it was assigned) an Internet domain name or X.500 name, or in the case that no such names are registered, authority to use a realm name may be derived from the authority of the parent realm. For example, if there is no domain name for E40.MIT.EDU, then the administrator of the MIT.EDU realm can authorize the creation of a realm with that name.

This is acceptable because the organization to which the parent is assigned is presumably the organization authorized to assign names to its children in the X.500 and domain name systems as well. If the parent assigns a realm name without also registering it in the domain name or X.500 hierarchy, it is the parent's responsibility to make sure that there will not in the future exists a name identical to the realm name of the child unless it is assigned to the same entity as the realm name.

7.2. Principal Names

As was the case for realm names, conventions are needed to ensure that all agree on what information is implied by a principal name. The name-type field that is part of the principal name indicates the kind of information implied by the name. The name-type should be treated as a hint. Ignoring the name type, no two names can be the same (i.e., at least one of the components, or the realm, must be different). This constraint may be eliminated in the future. The following name types are defined:

name-type	value	meaning
NT-UNKNOWN	0	Name type not known
NT-PRINCIPAL	1	Just the name of the principal as in DCE, or for users
NT-SRV-INST	2	Service and other unique instance (krbtgt)
NT-SRV-HST	3	Service with host name as instance (telnet, rcommands)
NT-SRV-XHST	4	Service with host as remaining components
NT-UID	5	Unique ID

When a name implies no information other than its uniqueness at a particular time the name type PRINCIPAL should be used. The principal name type should be used for users, and it might also be used for a unique server. If the name is a unique machine generated ID that is guaranteed never to be reassigned then the name type of UID should be used (note that it is generally a bad idea to reassign names of any type since stale entries might remain in access control lists).

If the first component of a name identifies a service and the remaining components identify an instance of the service in a server specified manner, then the name type of SRV-INST should be used. An example of this name type is the Kerberos ticket-granting ticket which has a first component of krbtgt and a second component identifying the realm for which the ticket is valid.

If instance is a single component following the service name and the instance identifies the host on which the server is running, then the name type SRV-HST should be used. This type is typically used for Internet services such as telnet and the Berkeley R commands. If the separate components of the host name appear as successive components following the name of the service, then the name type SRVXHST should be used. This type might be used to identify servers on hosts with X.500 names where the slash (/) might otherwise be ambiguous.

A name type of UNKNOWN should be used when the form of the name is not known. When comparing names, a name of type UNKNOWN will match principals authenticated with names of any type. A principal authenticated with a name of type UNKNOWN, however, will only match other names of type UNKNOWN.

Names of any type with an initial component of "krbtgt" are reserved for the Kerberos ticket granting service. See section 8.2.3 for the form of such names.

7.2.1. Name of server principals

The principal identifier for a server on a host will generally be composed of two parts: (1) the realm of the KDC with which the server is registered, and (2) a two-component name of type NT-SRV-HST if the host name is an Internet domain name or a multi-component name of type NT-SRV-XHST if the name of the host is of a form such as X.500 that allows slash (/) separators. The first component of the two- or multi-component name will identify the service and the latter components will identify the host. Where the name of the host is not case sensitive (for example, with Internet domain names) the name of the host must be lower case. For services such as telnet and the Berkeley R commands which run with system privileges, the first component will be the string "host" instead of a service specific identifier.

8. Constants and other defined values

8.1. Host address types

All negative values for the host address type are reserved for local use. All non-negative values are reserved for officially assigned

type fields and interpretations.

The values of the types for the following addresses are chosen to match the defined address family constants in the Berkeley Standard Distributions of Unix. They can be found in <sys/socket.h> with symbolic names AF_xxx (where xxx is an abbreviation of the address family name).

Internet addresses

Internet addresses are 32-bit (4-octet) quantities, encoded in MSB order. The type of internet addresses is two (2).

CHAOSnet addresses

CHAOSnet addresses are 16-bit (2-octet) quantities, encoded in MSB order. The type of CHAOSnet addresses is five (5).

ISO addresses

ISO addresses are variable-length. The type of ISO addresses is seven (7).

Xerox Network Services (XNS) addresses

XNS addresses are 48-bit (6-octet) quantities, encoded in MSB order. The type of XNS addresses is six (6).

AppleTalk Datagram Delivery Protocol (DDP) addresses

AppleTalk DDP addresses consist of an 8-bit node number and a 16-bit network number. The first octet of the address is the node number; the remaining two octets encode the network number in MSB order. The type of AppleTalk DDP addresses is sixteen (16).

DECnet Phase IV addresses

DECnet Phase IV addresses are 16-bit addresses, encoded in LSB order. The type of DECnet Phase IV addresses is twelve (12).

8.2. KDC messages

8.2.1. IP transport

When contacting a Kerberos server (KDC) for a KRB_KDC_REQ request using IP transport, the client shall send a UDP datagram containing only an encoding of the request to port 88 (decimal) at the KDC's IP

address; the KDC will respond with a reply datagram containing only an encoding of the reply message (either a KRB_ERROR or a KRB_KDC_REP) to the sending port at the sender's IP address.

8.2.2. OSI transport

During authentication of an OSI client to and OSI server, the mutual authentication of an OSI server to an OSI client, the transfer of credentials from an OSI client to an OSI server, or during exchange of private or integrity checked messages, Kerberos protocol messages may be treated as opaque objects and the type of the authentication mechanism will be:

```
OBJECT IDENTIFIER ::= {iso (1), org(3), dod(5),internet(1),
                       security(5), kerberosv5(2)}
```

Depending on the situation, the opaque object will be an authentication header (KRB_AP_REQ), an authentication reply (KRB_AP_REP), a safe message (KRB_SAFE), a private message (KRB_PRIV), or a credentials message (KRB_CRED). The opaque data contains an application code as specified in the ASN.1 description for each message. The application code may be used by Kerberos to determine the message type.

8.2.3. Name of the TGS

The principal identifier of the ticket-granting service shall be composed of three parts: (1) the realm of the KDC issuing the TGS ticket (2) a two-part name of type NT-SRVINST, with the first part "krbtgt" and the second part the name of the realm which will accept the ticket-granting ticket. For example, a ticket-granting ticket issued by the ATHENA.MIT.EDU realm to be used to get tickets from the ATHENA.MIT.EDU KDC has a principal identifier of "ATHENA.MIT.EDU" (realm), ("krbtgt", "ATHENA.MIT.EDU") (name). A ticket-granting ticket issued by the ATHENA.MIT.EDU realm to be used to get tickets from the MIT.EDU realm has a principal identifier of "ATHENA.MIT.EDU" (realm), ("krbtgt", "MIT.EDU") (name).

8.3. Protocol constants and associated values

The following tables list constants used in the protocol and defines their meanings.

Encryption type	etype value	block size	minimum pad size	confounder size
NULL	0	1	0	0
des-cbc-crc	1	8	4	8
des-cbc-md4	2	8	0	8
des-cbc-md5	3	8	0	8

Checksum type	sumtype value	checksum size
CRC32	1	4
rsa-md4	2	16
rsa-md4-des	3	24
des-mac	4	16
des-mac-k	5	8
rsa-md4-des-k	6	16
rsa-md5	7	16
rsa-md5-des	8	24

padata type	padata-type value
PA-TGS-REQ	1
PA-ENC-TIMESTAMP	2
PA-PW-SALT	3

authorization data type	ad-type value
reserved values	0-63
OSF-DCE	64
SESAME	65

alternate authentication type	method-type value
reserved values	0-63
ATT-CHALLENGE-RESPONSE	64

transited encoding type	tr-type value
DOMAIN-X500-COMPRESS	1
reserved values	all others

Label	Value	Meaning or MIT code
pvno	5	current Kerberos protocol version number
message types		
KRB_AS_REQ	10	Request for initial authentication
KRB_AS_REP	11	Response to KRB_AS_REQ request
KRB_TGS_REQ	12	Request for authentication based on TGT
KRB_TGS_REP	13	Response to KRB_TGS_REQ request
KRB_AP_REQ	14	application request to server
KRB_AP_REP	15	Response to KRB_AP_REQ_MUTUAL
KRB_SAFE	20	Safe (checksummed) application message
KRB_PRIV	21	Private (encrypted) application message
KRB_CRED	22	Private (encrypted) message to forward credentials
KRB_ERROR	30	Error response
name types		
KRB_NT_UNKNOWN	0	Name type not known
KRB_NT_PRINCIPAL	1	Just the name of the principal as in DCE, or for users
KRB_NT_SRV_INST	2	Service and other unique instance (krbtgt)
KRB_NT_SRV_HST	3	Service with host name as instance (telnet, rcommands)
KRB_NT_SRV_XHST	4	Service with host as remaining components
KRB_NT_UID	5	Unique ID
error codes		
KDC_ERR_NONE	0	No error
KDC_ERR_NAME_EXP	1	Client's entry in database has expired
KDC_ERR_SERVICE_EXP	2	Server's entry in database has expired
KDC_ERR_BAD_PVNO	3	Requested protocol version number not supported
KDC_ERR_C_OLD_MAST_KVNO	4	Client's key encrypted in old master key
KDC_ERR_S_OLD_MAST_KVNO	5	Server's key encrypted in old master key
KDC_ERR_C_PRINCIPAL_UNKNOWN	6	Client not found in Kerberos database
KDC_ERR_S_PRINCIPAL_UNKNOWN	7	Server not found in Kerberos database
KDC_ERR_PRINCIPAL_NOT_UNIQUE	8	Multiple principal entries in database

KDC_ERR_NULL_KEY	9	The client or server has a null key
KDC_ERR_CANNOT_POSTDATE	10	Ticket not eligible for postdating
KDC_ERR_NEVER_VALID	11	Requested start time is later than end time
KDC_ERR_POLICY	12	KDC policy rejects request
KDC_ERR_BADOPTION	13	KDC cannot accommodate requested option
KDC_ERR_ETYPE_NOSUPP	14	KDC has no support for encryption type
KDC_ERR_SUMTYPE_NOSUPP	15	KDC has no support for checksum type
KDC_ERR_PADATA_TYPE_NOSUPP	16	KDC has no support for padata type
KDC_ERR_TRTYPE_NOSUPP	17	KDC has no support for transited type
KDC_ERR_CLIENT_REVOKED	18	Clients credentials have been revoked
KDC_ERR_SERVICE_REVOKED	19	Credentials for server have been revoked
KDC_ERR_TGT_REVOKED	20	TGT has been revoked
KDC_ERR_CLIENT_NOTYET	21	Client not yet valid - try again later
KDC_ERR_SERVICE_NOTYET	22	Server not yet valid - try again later
KDC_ERR_KEY_EXPIRED	23	Password has expired - change password to reset
KDC_ERR_PREAUTH_FAILED	24	Pre-authentication information was invalid
KDC_ERR_PREAUTH_REQUIRED	25	Additional pre-authentication required*
KRB_AP_ERR_BAD_INTEGRITY	31	Integrity check on decrypted field failed
KRB_AP_ERR_TKT_EXPIRED	32	Ticket expired
KRB_AP_ERR_TKT_NYV	33	Ticket not yet valid
KRB_AP_ERR_REPEAT	34	Request is a replay
KRB_AP_ERR_NOT_US	35	The ticket isn't for us
KRB_AP_ERR_BADMATCH	36	Ticket and authenticator don't match
KRB_AP_ERR_SKEW	37	Clock skew too great
KRB_AP_ERR_BADADDR	38	Incorrect net address
KRB_AP_ERR_BADVERSION	39	Protocol version mismatch
KRB_AP_ERR_MSG_TYPE	40	Invalid msg type
KRB_AP_ERR_MODIFIED	41	Message stream modified
KRB_AP_ERR_BADORDER	42	Message out of order
KRB_AP_ERR_BADKEYVER	44	Specified version of key is not available
KRB_AP_ERR_NOKEY	45	Service key not available
KRB_AP_ERR_MUT_FAIL	46	Mutual authentication failed
KRB_AP_ERR_BADDIRECTION	47	Incorrect message direction
KRB_AP_ERR_METHOD	48	Alternative authentication method required*
KRB_AP_ERR_BADSEQ	49	Incorrect sequence number in message
KRB_AP_ERR_INAPP_CKSUM	50	Inappropriate type of checksum in

		message
KRB_ERR_GENERIC	60	Generic error (description in e-text)
KRB_ERR_FIELD_TOOLONG	61	Field is too long for this implementation

*This error carries additional information in the e-data field. The contents of the e-data field for this message is described in section 5.9.1.

9. Interoperability requirements

Version 5 of the Kerberos protocol supports a myriad of options. Among these are multiple encryption and checksum types, alternative encoding schemes for the transited field, optional mechanisms for pre-authentication, the handling of tickets with no addresses, options for mutual authentication, user to user authentication, support for proxies, forwarding, postdating, and renewing tickets, the format of realm names, and the handling of authorization data.

In order to ensure the interoperability of realms, it is necessary to define a minimal configuration which must be supported by all implementations. This minimal configuration is subject to change as technology does. For example, if at some later date it is discovered that one of the required encryption or checksum algorithms is not secure, it will be replaced.

9.1. Specification 1

This section defines the first specification of these options. Implementations which are configured in this way can be said to support Kerberos Version 5 Specification 1 (5.1).

Encryption and checksum methods

The following encryption and checksum mechanisms must be supported. Implementations may support other mechanisms as well, but the additional mechanisms may only be used when communicating with principals known to also support them: Encryption: DES-CBC-MD5 Checksums: CRC-32, DES-MAC, DES-MAC-K, and DES-MD5

Realm Names

All implementations must understand hierarchical realms in both the Internet Domain and the X.500 style. When a ticket granting ticket for an unknown realm is requested, the KDC must be able to determine the names of the intermediate realms between the KDCs realm and the requested realm.

Transited field encoding

DOMAIN-X500-COMPRESS (described in section 3.3.3.1) must be supported. Alternative encodings may be supported, but they may be used only when that encoding is supported by ALL intermediate realms.

Pre-authentication methods

The TGS-REQ method must be supported. The TGS-REQ method is not used on the initial request. The PA-ENC-TIMESTAMP method must be supported by clients but whether it is enabled by default may be determined on a realm by realm basis. If not used in the initial request and the error KDC_ERR_PREAUTH_REQUIRED is returned specifying PA-ENCTIMESTAMP as an acceptable method, the client should retry the initial request using the PA-ENC-TIMESTAMP preauthentication method. Servers need not support the PAENC-TIMESTAMP method, but if not supported the server should ignore the presence of PA-ENC-TIMESTAMP pre-authentication in a request.

Mutual authentication

Mutual authentication (via the KRB_AP_REP message) must be supported.

Ticket addresses and flags

All KDC's must pass on tickets that carry no addresses (i.e., if a TGT contains no addresses, the KDC will return derivative tickets), but each realm may set its own policy for issuing such tickets, and each application server will set its own policy with respect to accepting them. By default, servers should not accept them.

Proxies and forwarded tickets must be supported. Individual realms and application servers can set their own policy on when such tickets will be accepted.

All implementations must recognize renewable and postdated tickets, but need not actually implement them. If these options are not supported, the starttime and endtime in the ticket shall specify a ticket's entire useful life. When a postdated ticket is decoded by a server, all implementations shall make the presence of the postdated flag visible to the calling server.

User-to-user authentication

Support for user to user authentication (via the ENC-TKTIN-SKEY KDC option) must be provided by implementations, but individual realms may decide as a matter of policy to reject such requests on a per-principal or realm-wide basis.

Authorization data

Implementations must pass all authorization data subfields from ticket-granting tickets to any derivative tickets unless directed to suppress a subfield as part of the definition of that registered subfield type (it is never incorrect to pass on a subfield, and no registered subfield types presently specify suppression at the KDC).

Implementations must make the contents of any authorization data subfields available to the server when a ticket is used. Implementations are not required to allow clients to specify the contents of the authorization data fields.

9.2. Recommended KDC values

Following is a list of recommended values for a KDC implementation, based on the list of suggested configuration constants (see section 4.4).

minimum lifetime	5 minutes
maximum renewable lifetime	1 week
maximum ticket lifetime	1 day
empty addresses	only when suitable restrictions appear in authorization data
proxiable, etc.	Allowed.

10. Acknowledgments

Early versions of this document, describing version 4 of the protocol, were written by Jennifer Steiner (formerly at Project Athena); these drafts provided an excellent starting point for this current version 5 specification. Many people in the Internet community have contributed ideas and suggested protocol changes for version 5. Notable contributions came from Ted Anderson, Steve Bellovin and Michael Merritt [17], Daniel Bernstein, Mike Burrows, Donald Davis, Ravi Ganesan, Morrie Gasser, Virgil Gligor, Bill Griffeth, Mark Lillibridge, Mark Lomas, Steve Lunt, Piers McMahon, Joe Pato, William Sommerfeld, Stuart Stubblebine, Ralph Swick, Ted T'so, and Stanley Zanarotti. Many others commented and helped shape this specification into its current form.

11. References

[1] Miller, S., Neuman, C., Schiller, J., and J. Saltzer, "Section E.2.1: Kerberos Authentication and Authorization System", M.I.T. Project Athena, Cambridge, Massachusetts, December 21, 1987.

[2] Steiner, J., Neuman, C., and J. Schiller, "Kerberos: An Authentication Service for Open Network Systems", pp. 191-202 in Usenix Conference Proceedings, Dallas, Texas, February, 1988.

[3] Needham, R., and M. Schroeder, "Using Encryption for Authentication in Large Networks of Computers", Communications of the ACM, Vol. 21 (12), pp. 993-999, December 1978.

[4] Denning, D., and G. Sacco, "Time stamps in Key Distribution Protocols", Communications of the ACM, Vol. 24 (8), pp. 533-536, August 1981.

[5] Kohl, J., Neuman, C., and T. Ts'o, "The Evolution of the Kerberos Authentication Service", in an IEEE Computer Society Text soon to be published, June 1992.

[6] Davis, D., and R. Swick, "Workstation Services and Kerberos Authentication at Project Athena", Technical Memorandum TM-424, MIT Laboratory for Computer Science, February 1990.

[7] Levine, P., Gretzinger, M, Diaz, J., Sommerfeld, W., and K. Raeburn, "Section E.1: Service Management System, M.I.T. Project Athena, Cambridge, Mas sachusetts (1987).

[8] CCITT, Recommendation X.509: The Directory Authentication Framework, December 1988.

[9] Neuman, C., "Proxy-Based Authorization and Accounting for Distributed Systems," in Proceedings of the 13th International Conference on Distributed Computing Systems", Pittsburgh, PA, May 1993.

[10] Pato, J., "Using Pre-Authentication to Avoid Password Guessing Attacks", Open Software Foundation DCE Request for Comments 26, December 1992.

[11] National Bureau of Standards, U.S. Department of Commerce, "Data Encryption Standard", Federal Information Processing Standards Publication 46, Washington, DC (1977).

[12] National Bureau of Standards, U.S. Department of Commerce, "DES Modes of Operation", Federal Information Processing Standards Publication 81, Springfield, VA, December 1980.

[13] Stubblebine S., and V. Gligor, "On Message Integrity in Cryptographic Protocols", in Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland, California, May 1992.

[14] International Organization for Standardization, "ISO Information Processing Systems - Data Communication High-Level Data Link Control Procedure - Frame Structure", IS 3309, October 1984, 3rd Edition.

[15] Rivest, R., "The MD4 Message Digest Algorithm", RFC 1320, MIT Laboratory for Computer Science, April 1992.

[16] Rivest, R., "The MD5 Message Digest Algorithm", RFC 1321, MIT Laboratory for Computer Science, April 1992.

[17] Bellovin S., and M. Merritt, "Limitations of the Kerberos Authentication System", Computer Communications Review, Vol. 20(5), pp. 119-132, October 1990.

12. Security Considerations

Security issues are discussed throughout this memo.

13. Authors' Addresses

John Kohl
Digital Equipment Corporation
110 Spit Brook Road, M/S ZKO3-3/U14
Nashua, NH 03062

Phone: 603-881-2481
EMail: jtkohl@zk3.dec.com

B. Clifford Neuman
USC/Information Sciences Institute
4676 Admiralty Way #1001
Marina del Rey, CA 90292-6695

Phone: 310-822-1511
EMail: bcn@isi.edu

A. Pseudo-code for protocol processing

This appendix provides pseudo-code describing how the messages are to be constructed and interpreted by clients and servers.

A.1. KRB_AS_REQ generation

```
request.pvno := protocol version; /* pvno = 5 */
request.msg-type := message type; /* type = KRB_AS_REQ */

if(pa_enc_timestamp_required) then
        request.padata.padata-type = PA-ENC-TIMESTAMP;
        get system_time;
        padata-body.patimestamp,pausec = system_time;
        encrypt padata-body into request.padata.padata-value
                using client.key; /* derived from password */
endif

body.kdc-options := users's preferences;
body.cname := user's name;
body.realm := user's realm;
body.sname := service's name; /* usually "krbtgt",
                                 "localrealm" */
if (body.kdc-options.POSTDATED is set) then
        body.from := requested starting time;
else
        omit body.from;
endif
body.till := requested end time;
if (body.kdc-options.RENEWABLE is set) then
        body.rtime := requested final renewal time;
endif
body.nonce := random_nonce();
body.etype := requested etypes;
if (user supplied addresses) then
        body.addresses := user's addresses;
else
        omit body.addresses;
endif
omit body.enc-authorization-data;
request.req-body := body;

kerberos := lookup(name of local kerberos server (or servers));
send(packet,kerberos);

wait(for response);
if (timed_out) then
        retry or use alternate server;
endif
```

A.2. KRB_AS_REQ verification and KRB_AS_REP generation

```
decode message into req;

client := lookup(req.cname,req.realm);
server := lookup(req.sname,req.realm);
get system_time;
kdc_time := system_time.seconds;

if (!client) then
        /* no client in Database */
        error_out(KDC_ERR_C_PRINCIPAL_UNKNOWN);
endif
if (!server) then
        /* no server in Database */
        error_out(KDC_ERR_S_PRINCIPAL_UNKNOWN);
endif

if(client.pa_enc_timestamp_required and
   pa_enc_timestamp not present) then
        error_out(KDC_ERR_PREAUTH_REQUIRED(PA_ENC_TIMESTAMP));
endif

if(pa_enc_timestamp present) then
        decrypt req.padata-value into decrypted_enc_timestamp
                using client.key;
                using auth_hdr.authenticator.subkey;
        if (decrypt_error()) then
                error_out(KRB_AP_ERR_BAD_INTEGRITY);
        if(decrypted_enc_timestamp is not within allowable
                skew) then error_out(KDC_ERR_PREAUTH_FAILED);
        endif
        if(decrypted_enc_timestamp and usec is replay)
                error_out(KDC_ERR_PREAUTH_FAILED);
        endif
        add decrypted_enc_timestamp and usec to replay cache;
endif

use_etype := first supported etype in req.etypes;

if (no support for req.etypes) then
        error_out(KDC_ERR_ETYPE_NOSUPP);
endif

new_tkt.vno := ticket version; /* = 5 */
new_tkt.sname := req.sname;
new_tkt.srealm := req.srealm;
reset all flags in new_tkt.flags;
```

```
/* It should be noted that local policy may affect the  */
/* processing of any of these flags.  For example, some */
/* realms may refuse to issue renewable tickets         */

if (req.kdc-options.FORWARDABLE is set) then
        set new_tkt.flags.FORWARDABLE;
endif
if (req.kdc-options.PROXIABLE is set) then
        set new_tkt.flags.PROXIABLE;
endif
if (req.kdc-options.ALLOW-POSTDATE is set) then
        set new_tkt.flags.ALLOW-POSTDATE;
endif
if ((req.kdc-options.RENEW is set) or
    (req.kdc-options.VALIDATE is set) or
    (req.kdc-options.PROXY is set) or
    (req.kdc-options.FORWARDED is set) or
    (req.kdc-options.ENC-TKT-IN-SKEY is set)) then
        error_out(KDC_ERR_BADOPTION);
endif

new_tkt.session := random_session_key();
new_tkt.cname := req.cname;
new_tkt.crealm := req.crealm;
new_tkt.transited := empty_transited_field();

new_tkt.authtime := kdc_time;

if (req.kdc-options.POSTDATED is set) then
   if (against_postdate_policy(req.from)) then
        error_out(KDC_ERR_POLICY);
   endif
   set new_tkt.flags.INVALID;
   new_tkt.starttime := req.from;
else
   omit new_tkt.starttime; /* treated as authtime when
                              omitted */
endif
if (req.till = 0) then
        till := infinity;
else
        till := req.till;
endif

new_tkt.endtime := min(till,
                       new_tkt.starttime+client.max_life,
                       new_tkt.starttime+server.max_life,
                       new_tkt.starttime+max_life_for_realm);
```

```
if ((req.kdc-options.RENEWABLE-OK is set) and
    (new_tkt.endtime < req.till)) then
        /* we set the RENEWABLE option for later processing */
        set req.kdc-options.RENEWABLE;
        req.rtime := req.till;
endif

if (req.rtime = 0) then
        rtime := infinity;
else
        rtime := req.rtime;
endif

if (req.kdc-options.RENEWABLE is set) then
        set new_tkt.flags.RENEWABLE;
        new_tkt.renew-till := min(rtime,
        new_tkt.starttime+client.max_rlife,
        new_tkt.starttime+server.max_rlife,
        new_tkt.starttime+max_rlife_for_realm);
else
        omit new_tkt.renew-till; /* only present if RENEWABLE */
endif

if (req.addresses) then
        new_tkt.caddr := req.addresses;
else
        omit new_tkt.caddr;
endif

new_tkt.authorization_data := empty_authorization_data();

encode to-be-encrypted part of ticket into OCTET STRING;
new_tkt.enc-part := encrypt OCTET STRING
    using etype_for_key(server.key), server.key, server.p_kvno;


/* Start processing the response */

resp.pvno := 5;
resp.msg-type := KRB_AS_REP;
resp.cname := req.cname;
resp.crealm := req.realm;
resp.ticket := new_tkt;

resp.key := new_tkt.session;
resp.last-req := fetch_last_request_info(client);
resp.nonce := req.nonce;
resp.key-expiration := client.expiration;
```

```
        resp.flags := new_tkt.flags;

        resp.authtime := new_tkt.authtime;
        resp.starttime := new_tkt.starttime;
        resp.endtime := new_tkt.endtime;

        if (new_tkt.flags.RENEWABLE) then
                resp.renew-till := new_tkt.renew-till;
        endif

        resp.realm := new_tkt.realm;
        resp.sname := new_tkt.sname;

        resp.caddr := new_tkt.caddr;

        encode body of reply into OCTET STRING;

        resp.enc-part := encrypt OCTET STRING
                         using use_etype, client.key, client.p_kvno;
        send(resp);
```

A.3. KRB_AS_REP verification

```
        decode response into resp;

        if (resp.msg-type = KRB_ERROR) then
                if(error = KDC_ERR_PREAUTH_REQUIRED(PA_ENC_TIMESTAMP))
                        then set pa_enc_timestamp_required;
                        goto KRB_AS_REQ;
                endif
                process_error(resp);
                return;
        endif

        /* On error, discard the response, and zero the session key */
        /* from the response immediately */

        key = get_decryption_key(resp.enc-part.kvno, resp.enc-part.etype,
                                 resp.padata);
        unencrypted part of resp := decode of decrypt of resp.enc-part
                                 using resp.enc-part.etype and key;
        zero(key);

        if (common_as_rep_tgs_rep_checks fail) then
                destroy resp.key;
                return error;
        endif

        if near(resp.princ_exp) then
```

```
            print(warning message);
        endif
        save_for_later(ticket,session,client,server,times,flags);
```

A.4. KRB_AS_REP and KRB_TGS_REP common checks

```
        if (decryption_error() or
            (req.cname != resp.cname) or
            (req.realm != resp.crealm) or
            (req.sname != resp.sname) or
            (req.realm != resp.realm) or
            (req.nonce != resp.nonce) or
            (req.addresses != resp.caddr)) then
                destroy resp.key;
                return KRB_AP_ERR_MODIFIED;
        endif

        /* make sure no flags are set that shouldn't be, and that  */
        /* all that should be are set                              */
        if (!check_flags_for_compatability(req.kdc-options,resp.flags))
                then destroy resp.key;
                return KRB_AP_ERR_MODIFIED;
        endif

        if ((req.from = 0) and
            (resp.starttime is not within allowable skew)) then
                destroy resp.key;
                return KRB_AP_ERR_SKEW;
        endif
        if ((req.from != 0) and (req.from != resp.starttime)) then
                destroy resp.key;
                return KRB_AP_ERR_MODIFIED;
        endif
        if ((req.till != 0) and (resp.endtime > req.till)) then
                destroy resp.key;
                return KRB_AP_ERR_MODIFIED;
        endif

        if ((req.kdc-options.RENEWABLE is set) and
            (req.rtime != 0) and (resp.renew-till > req.rtime)) then
                destroy resp.key;
                return KRB_AP_ERR_MODIFIED;
        endif
        if ((req.kdc-options.RENEWABLE-OK is set) and
            (resp.flags.RENEWABLE) and
            (req.till != 0) and
            (resp.renew-till > req.till)) then
                destroy resp.key;
                return KRB_AP_ERR_MODIFIED;
```

```
        endif
```

A.5. KRB_TGS_REQ generation

```
        /* Note that make_application_request might have to     */
        /* recursivly call this routine to get the appropriate  */
        /* ticket-granting ticket                               */

        request.pvno := protocol version; /* pvno = 5 */
        request.msg-type := message type; /* type = KRB_TGS_REQ */

        body.kdc-options := users's preferences;
        /* If the TGT is not for the realm of the end-server  */
        /* then the sname will be for a TGT for the end-realm */
        /* and the realm of the requested ticket (body.realm) */
        /* will be that of the TGS to which the TGT we are    */
        /* sending applies                                    */
        body.sname := service's name;
        body.realm := service's realm;

        if (body.kdc-options.POSTDATED is set) then
                body.from := requested starting time;
        else
                omit body.from;
        endif
        body.till := requested end time;
        if (body.kdc-options.RENEWABLE is set) then
                body.rtime := requested final renewal time;
        endif
        body.nonce := random_nonce();
        body.etype := requested etypes;
        if (user supplied addresses) then
                body.addresses := user's addresses;
        else
                omit body.addresses;
        endif

        body.enc-authorization-data := user-supplied data;
        if (body.kdc-options.ENC-TKT-IN-SKEY) then
                body.additional-tickets_ticket := second TGT;
        endif

        request.req-body := body;
        check := generate_checksum (req.body,checksumtype);

        request.padata[0].padata-type := PA-TGS-REQ;
        request.padata[0].padata-value := create a KRB_AP_REQ using
                                          the TGT and checksum
```

```
        /* add in any other padata as required/supplied */

        kerberos := lookup(name of local kerberose server (or servers));
        send(packet,kerberos);

        wait(for response);
        if (timed_out) then
                retry or use alternate server;
        endif
```

A.6. KRB_TGS_REQ verification and KRB_TGS_REP generation

```
        /* note that reading the application request requires first
        determining the server for which a ticket was issued, and
        choosing the correct key for decryption.  The name of the
        server appears in the plaintext part of the ticket. */

        if (no KRB_AP_REQ in req.padata) then
                error_out(KDC_ERR_PADATA_TYPE_NOSUPP);
        endif
        verify KRB_AP_REQ in req.padata;

        /* Note that the realm in which the Kerberos server is
        operating is determined by the instance from the
        ticket-granting ticket.  The realm in the ticket-granting
        ticket is the realm under which the ticket granting ticket was
        issued.  It is possible for a single Kerberos server to
        support more than one realm. */

        auth_hdr := KRB_AP_REQ;
        tgt := auth_hdr.ticket;

        if (tgt.sname is not a TGT for local realm and is not
                req.sname) then error_out(KRB_AP_ERR_NOT_US);

        realm := realm_tgt_is_for(tgt);

        decode remainder of request;

        if (auth_hdr.authenticator.cksum is missing) then
                error_out(KRB_AP_ERR_INAPP_CKSUM);
        endif
        if (auth_hdr.authenticator.cksum type is not supported) then
                error_out(KDC_ERR_SUMTYPE_NOSUPP);
        endif
        if (auth_hdr.authenticator.cksum is not both collision-proof
            and keyed)  then
                error_out(KRB_AP_ERR_INAPP_CKSUM);
        endif
```

```
set computed_checksum := checksum(req);
if (computed_checksum != auth_hdr.authenticatory.cksum) then
        error_out(KRB_AP_ERR_MODIFIED);
endif

server := lookup(req.sname,realm);

if (!server) then
        if (is_foreign_tgt_name(server)) then
                server := best_intermediate_tgs(server);
        else
                /* no server in Database */
                error_out(KDC_ERR_S_PRINCIPAL_UNKNOWN);
        endif
endif

session := generate_random_session_key();


use_etype := first supported etype in req.etypes;

if (no support for req.etypes) then
        error_out(KDC_ERR_ETYPE_NOSUPP);
endif

new_tkt.vno := ticket version; /* = 5 */
new_tkt.sname := req.sname;
new_tkt.srealm := realm;
reset all flags in new_tkt.flags;

/* It should be noted that local policy may affect the  */
/* processing of any of these flags.  For example, some */
/* realms may refuse to issue renewable tickets         */

new_tkt.caddr := tgt.caddr;
resp.caddr := NULL; /* We only include this if they change */
if (req.kdc-options.FORWARDABLE is set) then
        if (tgt.flags.FORWARDABLE is reset) then
                error_out(KDC_ERR_BADOPTION);
        endif
        set new_tkt.flags.FORWARDABLE;
endif
if (req.kdc-options.FORWARDED is set) then
        if (tgt.flags.FORWARDABLE is reset) then
                error_out(KDC_ERR_BADOPTION);
        endif
        set new_tkt.flags.FORWARDED;
        new_tkt.caddr := req.addresses;
```

```
                resp.caddr := req.addresses;
        endif
        if (tgt.flags.FORWARDED is set) then
                set new_tkt.flags.FORWARDED;
        endif

        if (req.kdc-options.PROXIABLE is set) then
                if (tgt.flags.PROXIABLE is reset)
                        error_out(KDC_ERR_BADOPTION);
                endif
                set new_tkt.flags.PROXIABLE;
        endif
        if (req.kdc-options.PROXY is set) then
                if (tgt.flags.PROXIABLE is reset) then
                        error_out(KDC_ERR_BADOPTION);
                endif
                set new_tkt.flags.PROXY;
                new_tkt.caddr := req.addresses;
                resp.caddr := req.addresses;
        endif

        if (req.kdc-options.POSTDATE is set) then
                if (tgt.flags.POSTDATE is reset)
                        error_out(KDC_ERR_BADOPTION);
                endif
                set new_tkt.flags.POSTDATE;
        endif
        if (req.kdc-options.POSTDATED is set) then
                if (tgt.flags.POSTDATE is reset) then
                        error_out(KDC_ERR_BADOPTION);
                endif
                set new_tkt.flags.POSTDATED;
                set new_tkt.flags.INVALID;
                if (against_postdate_policy(req.from)) then
                        error_out(KDC_ERR_POLICY);
                endif
                new_tkt.starttime := req.from;
        endif


        if (req.kdc-options.VALIDATE is set) then
                if (tgt.flags.INVALID is reset) then
                        error_out(KDC_ERR_POLICY);
                endif
                if (tgt.starttime > kdc_time) then
                        error_out(KRB_AP_ERR_NYV);
                endif
                if (check_hot_list(tgt)) then
```

```
                error_out(KRB_AP_ERR_REPEAT);
        endif
        tkt := tgt;
        reset new_tkt.flags.INVALID;
endif

if (req.kdc-options.(any flag except ENC-TKT-IN-SKEY, RENEW,
                and those already processed) is set) then
        error_out(KDC_ERR_BADOPTION);
endif

new_tkt.authtime := tgt.authtime;

if (req.kdc-options.RENEW is set) then
  /* Note that if the endtime has already passed, the ticket */
  /* would have been rejected in the initial authentication  */
  /* stage, so there is no need to check again here          */
        if (tgt.flags.RENEWABLE is reset) then
                error_out(KDC_ERR_BADOPTION);
        endif
        if (tgt.renew-till >= kdc_time) then
                error_out(KRB_AP_ERR_TKT_EXPIRED);
        endif
        tkt := tgt;
        new_tkt.starttime := kdc_time;
        old_life := tgt.endttime - tgt.starttime;
        new_tkt.endtime := min(tgt.renew-till,
                               new_tkt.starttime + old_life);
else
        new_tkt.starttime := kdc_time;
        if (req.till = 0) then
                till := infinity;
        else
                till := req.till;
        endif
        new_tkt.endtime := min(till,
                           new_tkt.starttime+client.max_life,
                           new_tkt.starttime+server.max_life,
                           new_tkt.starttime+max_life_for_realm,
                           tgt.endtime);

        if ((req.kdc-options.RENEWABLE-OK is set) and
            (new_tkt.endtime < req.till) and
            (tgt.flags.RENEWABLE is set) then
                /* we set the RENEWABLE option for later  */
                /* processing                             */
                set req.kdc-options.RENEWABLE;
                req.rtime := min(req.till, tgt.renew-till);
```

```
        endif
    endif

    if (req.rtime = 0) then
        rtime := infinity;
    else
        rtime := req.rtime;
    endif

    if ((req.kdc-options.RENEWABLE is set) and
        (tgt.flags.RENEWABLE is set)) then
        set new_tkt.flags.RENEWABLE;
        new_tkt.renew-till := min(rtime,
        new_tkt.starttime+client.max_rlife,
        new_tkt.starttime+server.max_rlife,
        new_tkt.starttime+max_rlife_for_realm,
        tgt.renew-till);
    else
        new_tkt.renew-till := OMIT;
                    /* leave the renew-till field out */
    endif
    if (req.enc-authorization-data is present) then
        decrypt req.enc-authorization-data
                into    decrypted_authorization_data
                using auth_hdr.authenticator.subkey;
        if (decrypt_error()) then
                error_out(KRB_AP_ERR_BAD_INTEGRITY);
        endif
    endif
    new_tkt.authorization_data :=
    req.auth_hdr.ticket.authorization_data +
                        decrypted_authorization_data;

    new_tkt.key := session;
    new_tkt.crealm := tgt.crealm;
    new_tkt.cname := req.auth_hdr.ticket.cname;

    if (realm_tgt_is_for(tgt) := tgt.realm) then
        /* tgt issued by local realm */
        new_tkt.transited := tgt.transited;
    else
        /* was issued for this realm by some other realm */
        if (tgt.transited.tr-type not supported) then
                error_out(KDC_ERR_TRTYPE_NOSUPP);
        endif
        new_tkt.transited
           := compress_transited(tgt.transited + tgt.realm)
    endif
```

```
        encode encrypted part of new_tkt into OCTET STRING;
        if (req.kdc-options.ENC-TKT-IN-SKEY is set) then
                if (server not specified) then
                        server = req.second_ticket.client;
                endif
                if ((req.second_ticket is not a TGT) or
                    (req.second_ticket.client != server)) then
                        error_out(KDC_ERR_POLICY);
                endif

                new_tkt.enc-part := encrypt OCTET STRING using
                        using etype_for_key(second-ticket.key),
                                                    second-ticket.key;
        else
                new_tkt.enc-part := encrypt OCTET STRING
                        using etype_for_key(server.key), server.key,
                                                    server.p_kvno;
        endif

        resp.pvno := 5;
        resp.msg-type := KRB_TGS_REP;
        resp.crealm := tgt.crealm;
        resp.cname := tgt.cname;
        resp.ticket := new_tkt;

        resp.key := session;
        resp.nonce := req.nonce;
        resp.last-req := fetch_last_request_info(client);
        resp.flags := new_tkt.flags;

        resp.authtime := new_tkt.authtime;
        resp.starttime := new_tkt.starttime;
        resp.endtime := new_tkt.endtime;

        omit resp.key-expiration;

        resp.sname := new_tkt.sname;
        resp.realm := new_tkt.realm;

        if (new_tkt.flags.RENEWABLE) then
                resp.renew-till := new_tkt.renew-till;
        endif


        encode body of reply into OCTET STRING;

        if (req.padata.authenticator.subkey)
                resp.enc-part := encrypt OCTET STRING using use_etype,
```

```
                    req.padata.authenticator.subkey;
        else resp.enc-part := encrypt OCTET STRING
                              using use_etype, tgt.key;

        send(resp);
```

A.7. KRB_TGS_REP verification

```
        decode response into resp;

        if (resp.msg-type = KRB_ERROR) then
                process_error(resp);
                return;
        endif

        /* On error, discard the response, and zero the session key from
        the response immediately */

        if (req.padata.authenticator.subkey)
                unencrypted part of resp :=
                        decode of decrypt of resp.enc-part
                        using resp.enc-part.etype and subkey;
        else unencrypted part of resp :=
                        decode of decrypt of resp.enc-part
                        using resp.enc-part.etype and tgt's session key;
        if (common_as_rep_tgs_rep_checks fail) then
                destroy resp.key;
                return error;
        endif

        check authorization_data as necessary;
        save_for_later(ticket,session,client,server,times,flags);
```

A.8. Authenticator generation

```
        body.authenticator-vno := authenticator vno; /* = 5 */
        body.cname, body.crealm := client name;
        if (supplying checksum) then
                body.cksum := checksum;
        endif
        get system_time;
        body.ctime, body.cusec := system_time;
        if (selecting sub-session key) then
                select sub-session key;
                body.subkey := sub-session key;
        endif
        if (using sequence numbers) then
                select initial sequence number;
                body.seq-number := initial sequence;
        endif
```

A.9. KRB_AP_REQ generation

```
obtain ticket and session_key from cache;

packet.pvno := protocol version; /* 5 */
packet.msg-type := message type; /* KRB_AP_REQ */

if (desired(MUTUAL_AUTHENTICATION)) then
        set packet.ap-options.MUTUAL-REQUIRED;
else
        reset packet.ap-options.MUTUAL-REQUIRED;
endif
if (using session key for ticket) then
        set packet.ap-options.USE-SESSION-KEY;
else
        reset packet.ap-options.USE-SESSION-KEY;
endif
packet.ticket := ticket; /* ticket */
generate authenticator;
encode authenticator into OCTET STRING;
encrypt OCTET STRING into packet.authenticator
                    using session_key;
```

A.10. KRB_AP_REQ verification

```
receive packet;
if (packet.pvno != 5) then
        either process using other protocol spec
        or error_out(KRB_AP_ERR_BADVERSION);
endif
if (packet.msg-type != KRB_AP_REQ) then
        error_out(KRB_AP_ERR_MSG_TYPE);
endif
if (packet.ticket.tkt_vno != 5) then
        either process using other protocol spec
        or error_out(KRB_AP_ERR_BADVERSION);
endif
if (packet.ap_options.USE-SESSION-KEY is set) then
        retrieve session key from ticket-granting ticket for
         packet.ticket.{sname,srealm,enc-part.etype};
else
   retrieve service key for
   packet.ticket.{sname,srealm,enc-part.etype,enc-part.skvno};
endif
if (no_key_available) then
        if (cannot_find_specified_skvno) then
                error_out(KRB_AP_ERR_BADKEYVER);
        else
                error_out(KRB_AP_ERR_NOKEY);
        endif
```

```
        endif
        decrypt packet.ticket.enc-part into decr_ticket
                                        using retrieved key;
        if (decryption_error()) then
                error_out(KRB_AP_ERR_BAD_INTEGRITY);
        endif
        decrypt packet.authenticator into decr_authenticator
                using decr_ticket.key;
        if (decryption_error()) then
                error_out(KRB_AP_ERR_BAD_INTEGRITY);
        endif
        if (decr_authenticator.{cname,crealm} !=
            decr_ticket.{cname,crealm}) then
                error_out(KRB_AP_ERR_BADMATCH);
        endif
        if (decr_ticket.caddr is present) then
                if (sender_address(packet) is not in decr_ticket.caddr)
                        then error_out(KRB_AP_ERR_BADADDR);
                endif
        elseif (application requires addresses) then
                error_out(KRB_AP_ERR_BADADDR);
        endif
        if (not in_clock_skew(decr_authenticator.ctime,
                              decr_authenticator.cusec)) then
                error_out(KRB_AP_ERR_SKEW);
        endif
        if (repeated(decr_authenticator.{ctime,cusec,cname,crealm}))
                then error_out(KRB_AP_ERR_REPEAT);
        endif
        save_identifier(decr_authenticator.{ctime,cusec,cname,crealm});
        get system_time;
        if ((decr_ticket.starttime-system_time > CLOCK_SKEW) or
            (decr_ticket.flags.INVALID is set)) then
                /* it hasn't yet become valid */
                error_out(KRB_AP_ERR_TKT_NYV);
        endif
        if (system_time-decr_ticket.endtime > CLOCK_SKEW) then
                error_out(KRB_AP_ERR_TKT_EXPIRED);
        endif
        /* caller must check decr_ticket.flags for any pertinent */
        /* details */
        return(OK, decr_ticket, packet.ap_options.MUTUAL-REQUIRED);
```

A.11. KRB_AP_REP generation

```
        packet.pvno := protocol version; /* 5 */
        packet.msg-type := message type; /* KRB_AP_REP */
        body.ctime := packet.ctime;
        body.cusec := packet.cusec;
```

```
        if (selecting sub-session key) then
                select sub-session key;
                body.subkey := sub-session key;
        endif
        if (using sequence numbers) then
                select initial sequence number;
                body.seq-number := initial sequence;
        endif

        encode body into OCTET STRING;

        select encryption type;
        encrypt OCTET STRING into packet.enc-part;
```

A.12. KRB_AP_REP verification

```
        receive packet;
        if (packet.pvno != 5) then
                either process using other protocol spec
                or error_out(KRB_AP_ERR_BADVERSION);
        endif
        if (packet.msg-type != KRB_AP_REP) then
                error_out(KRB_AP_ERR_MSG_TYPE);
        endif
        cleartext := decrypt(packet.enc-part)
                        using ticket's session key;
        if (decryption_error()) then
                error_out(KRB_AP_ERR_BAD_INTEGRITY);
        endif
        if (cleartext.ctime != authenticator.ctime) then
                error_out(KRB_AP_ERR_MUT_FAIL);
        endif
        if (cleartext.cusec != authenticator.cusec) then
                error_out(KRB_AP_ERR_MUT_FAIL);
        endif
        if (cleartext.subkey is present) then
                save cleartext.subkey for future use;
        endif
        if (cleartext.seq-number is present) then
                save cleartext.seq-number for future verifications;
        endif
        return(AUTHENTICATION_SUCCEEDED);
```

A.13. KRB_SAFE generation

```
        collect user data in buffer;

        /* assemble packet: */
        packet.pvno := protocol version; /* 5 */
        packet.msg-type := message type; /* KRB_SAFE */
```

```
        body.user-data := buffer; /* DATA */
        if (using timestamp) then
                get system_time;
                body.timestamp, body.usec := system_time;
        endif
        if (using sequence numbers) then
                body.seq-number := sequence number;
        endif
        body.s-address := sender host addresses;
        if (only one recipient) then
                body.r-address := recipient host address;
        endif
        checksum.cksumtype := checksum type;
        compute checksum over body;
        checksum.checksum := checksum value; /* checksum.checksum */
        packet.cksum := checksum;
        packet.safe-body := body;
```

A.14. KRB_SAFE verification

```
        receive packet;
        if (packet.pvno != 5) then
                either process using other protocol spec
                or error_out(KRB_AP_ERR_BADVERSION);
        endif
        if (packet.msg-type != KRB_SAFE) then
                error_out(KRB_AP_ERR_MSG_TYPE);
        endif
        if (packet.checksum.cksumtype is not both collision-proof
                                            and keyed) then
                error_out(KRB_AP_ERR_INAPP_CKSUM);
        endif
        if (safe_priv_common_checks_ok(packet)) then
                set computed_checksum := checksum(packet.body);
                if (computed_checksum != packet.checksum) then
                        error_out(KRB_AP_ERR_MODIFIED);
                endif
                return (packet, PACKET_IS_GENUINE);
        else
                return common_checks_error;
        endif
```

A.15. KRB_SAFE and KRB_PRIV common checks

```
        if (packet.s-address != O/S_sender(packet)) then
            /* O/S report of sender not who claims to have sent it */
            error_out(KRB_AP_ERR_BADADDR);
        endif
        if ((packet.r-address is present) and
            (packet.r-address != local_host_address)) then
```

```
                /* was not sent to proper place */
                error_out(KRB_AP_ERR_BADADDR);
        endif
        if (((packet.timestamp is present) and
             (not in_clock_skew(packet.timestamp,packet.usec))) or
            (packet.timestamp is not present and timestamp expected))
                then error_out(KRB_AP_ERR_SKEW);
        endif
        if (repeated(packet.timestamp,packet.usec,packet.s-address))
                then error_out(KRB_AP_ERR_REPEAT);
        endif
        if (((packet.seq-number is present) and
             ((not in_sequence(packet.seq-number)))) or
            (packet.seq-number is not present and sequence expected))
                then error_out(KRB_AP_ERR_BADORDER);
        endif
        if (packet.timestamp not present and
            packet.seq-number not present) then
                error_out(KRB_AP_ERR_MODIFIED);
        endif

        save_identifier(packet.{timestamp,usec,s-address},
                        sender_principal(packet));

        return PACKET_IS_OK;
```

A.16. KRB_PRIV generation

```
        collect user data in buffer;

        /* assemble packet: */
        packet.pvno := protocol version; /* 5 */
        packet.msg-type := message type; /* KRB_PRIV */

        packet.enc-part.etype := encryption type;

        body.user-data := buffer;
        if (using timestamp) then
                get system_time;
                body.timestamp, body.usec := system_time;
        endif
        if (using sequence numbers) then
                body.seq-number := sequence number;
        endif
        body.s-address := sender host addresses;
        if (only one recipient) then
                body.r-address := recipient host address;
        endif
```

```
        encode body into OCTET STRING;

        select encryption type;
        encrypt OCTET STRING into packet.enc-part.cipher;
```

A.17. KRB_PRIV verification

```
        receive packet;
        if (packet.pvno != 5) then
                either process using other protocol spec
                or error_out(KRB_AP_ERR_BADVERSION);
        endif
        if (packet.msg-type != KRB_PRIV) then
                error_out(KRB_AP_ERR_MSG_TYPE);
        endif

        cleartext := decrypt(packet.enc-part) using negotiated key;
        if (decryption_error()) then
                error_out(KRB_AP_ERR_BAD_INTEGRITY);
        endif

        if (safe_priv_common_checks_ok(cleartext)) then
            return(cleartext.DATA, PACKET_IS_GENUINE_AND_UNMODIFIED);
        else
                return common_checks_error;
        endif
```

A.18. KRB_CRED generation

```
        invoke KRB_TGS; /* obtain tickets to be provided to peer */

        /* assemble packet: */
        packet.pvno := protocol version; /* 5 */
        packet.msg-type := message type; /* KRB_CRED */

        for (tickets[n] in tickets to be forwarded) do
                packet.tickets[n] = tickets[n].ticket;
        done

        packet.enc-part.etype := encryption type;

        for (ticket[n] in tickets to be forwarded) do
                body.ticket-info[n].key = tickets[n].session;
                body.ticket-info[n].prealm = tickets[n].crealm;
                body.ticket-info[n].pname = tickets[n].cname;
                body.ticket-info[n].flags = tickets[n].flags;
                body.ticket-info[n].authtime = tickets[n].authtime;
                body.ticket-info[n].starttime = tickets[n].starttime;
                body.ticket-info[n].endtime = tickets[n].endtime;
                body.ticket-info[n].renew-till = tickets[n].renew-till;
```

```
                body.ticket-info[n].srealm = tickets[n].srealm;
                body.ticket-info[n].sname = tickets[n].sname;
                body.ticket-info[n].caddr = tickets[n].caddr;
        done

        get system_time;
        body.timestamp, body.usec := system_time;

        if (using nonce) then
                body.nonce := nonce;
        endif

        if (using s-address) then
                body.s-address := sender host addresses;
        endif
        if (limited recipients) then
                body.r-address := recipient host address;
        endif

        encode body into OCTET STRING;

        select encryption type;
        encrypt OCTET STRING into packet.enc-part.cipher
        using negotiated encryption key;
```

A.19. KRB_CRED verification

```
        receive packet;
        if (packet.pvno != 5) then
                either process using other protocol spec
                or error_out(KRB_AP_ERR_BADVERSION);
        endif
        if (packet.msg-type != KRB_CRED) then
                error_out(KRB_AP_ERR_MSG_TYPE);
        endif

        cleartext := decrypt(packet.enc-part) using negotiated key;
        if (decryption_error()) then
                error_out(KRB_AP_ERR_BAD_INTEGRITY);
        endif
        if ((packet.r-address is present or required) and
           (packet.s-address != O/S_sender(packet)) then
            /* O/S report of sender not who claims to have sent it */
            error_out(KRB_AP_ERR_BADADDR);
        endif
        if ((packet.r-address is present) and
            (packet.r-address != local_host_address)) then
                /* was not sent to proper place */
                error_out(KRB_AP_ERR_BADADDR);
```

```
        endif
        if (not in_clock_skew(packet.timestamp,packet.usec)) then
                error_out(KRB_AP_ERR_SKEW);
        endif
        if (repeated(packet.timestamp,packet.usec,packet.s-address))
                then error_out(KRB_AP_ERR_REPEAT);
        endif
        if (packet.nonce is required or present) and
           (packet.nonce != expected-nonce) then
                error_out(KRB_AP_ERR_MODIFIED);
        endif

        for (ticket[n] in tickets that were forwarded) do
                save_for_later(ticket[n],key[n],principal[n],
                               server[n],times[n],flags[n]);
        return
```

A.20. KRB_ERROR generation

```
        /* assemble packet: */
        packet.pvno := protocol version; /* 5 */
        packet.msg-type := message type; /* KRB_ERROR */

        get system_time;
        packet.stime, packet.susec := system_time;
        packet.realm, packet.sname := server name;

        if (client time available) then
                packet.ctime, packet.cusec := client_time;
        endif
        packet.error-code := error code;
        if (client name available) then
                packet.cname, packet.crealm := client name;
        endif
        if (error text available) then
                packet.e-text := error text;
        endif
        if (error data available) then
                packet.e-data := error data;
        endif
```

Summary and Analysis of A13

An Overview of the Technical Goals and Concepts in "*An Open Architecture for a Digital Library System and a Plan for its Development*" by Robert E. Kahn and Vinton G. Cerf (1988)

by Mark Stefik

Research Fellow
Palo Alto Research Center
Palo Alto, California 94304

Prepared for ContentGuard
May 30, 2007

Contents

Executive Summary

In 1988 Robert Kahn and Vinton Cerf wrote a draft paper (A13) proposing an architecture and plans for a digital library system. In broad terms, they proposed an infrastructure intended to link libraries together and to enable people to share information and documents.

In setting out their proposal, they made important design choices. A crucial choice at the onset was to base their design on the metaphor of a traditional library. In contrast, digital rights management systems such as those invented in the early 1990s and deployed today were intended to support digital commerce. These DRM designs were based on the metaphor of the electronic marketplace[1]. The difference in choice of design metaphor led to different goals for what the systems were expected to do, leading in many crucial ways to essentially opposite design choices.

Traditional libraries in the United States are not commercial organizations. Traditional libraries are not concerned with regulating the particular uses to which information is put. Nor are they concerned with meeting the needs of commerce, such as pricing regimes, licensing arrangements, sales territories, or special deals for students or members of various groups. The most important legal basis for libraries is Copyright Law. In contrast, the legal basis for business agreements between parties is Contract Law.

This difference in orientation between A13 and later DRM systems is profound. In the main, A13 does not focus on the needs of commerce, where parties are often competitive and sometimes adversarial. For example, A13 does not consider any need for visible agreements reflecting the interests of parties in commercial transactions. It does not recognize a need for a threat analysis and arrangements for robust security in system architecture.

Table 1 summarizes some of the main differences in design choices between A13 and many later DRM approaches.

---

[1] For example, see the book *Internet Dreams: Archetypes, Myths, and Metaphors* by Mark Stefik (with a foreword by Vinton Cerf (one of the authors of A13), published in 1996 by The MIT Press.

Design Issue	Design Choice in A13	Design Choice in DRM systems
Main design goal.	Promote sharing of information. A13 was intended to provide friendlier services akin to traditional libraries.	Promote a viable system to enable new business models for the commercial distribution of digital content.
Mechanism for overseeing rights.	Software-only agents ("Knowbots") that are attached to documents and travel from system to system as mobile code. Potentially, each different set of requirements is met by a different program.	Digital rights may be expressed in a declarative rights language for digital contracts. These contracts are intended for use both in user interfaces and by standard and certified trusted systems that enforce usage obligations. Communication among trusted systems is often protected by encryption. The trusted systems get the parameters of usage from the digital contracts. The same trusted systems potentially work for all documents. Foundations for trust in these systems may include physical (hardware) security, communication security (e.g. encryption), and behavioral security (certified programs).
Organizational basis of trust.	Content consumers must trust the producers of content, who commission the writing of different Knowbots for each use. Content producers must trust the consumers of content, who configure the operating environments on their machines for Knowbots. There is no consideration of the practical verifiability of system correctness or security.	Producers and consumers of content agree on terms and conditions and express them in digital contracts. They do not depend on each other's code. Rather, they rely on qualified (and disinterested) third parties to develop and certify trusted systems that enforce the digital contracts.
Purpose of computer oversight.	Knowbots are intended to represent the interests of content owners. Knowbots are largely concerned with accounting for the amount of usage for each document.	Digital contracts are designed to represent *all* essential elements in the agreements between all of the parties to the transactions. Digital contracts address the terms and conditions of use – including allowed operations, fees, timing, special licenses,

		distribution, and so on.
Accommodation of existing systems.	A13 recognized that the digital library would be distributed, heterarchical, networked, and display oriented. It must have an ability to interact with Digital Library Systems that do not adhere to its internal standards and procedures.	Most DRM systems invented in the 1990s and deployed today were a departure from existing networked file systems in their foundations of trust. In order to prevent the compromise of commerce or private content, they were designed specifically to only operate with other trusted systems that could meet their standards.

The rest of this paper discusses the technical goals and choices in A13 in more detail, and shows how the design choices took opposite paths from the DRM systems that were invented in the 1990s.

1 Introduction

In 1988 Robert Kahn and Vinton Cerf wrote a visionary white paper, "An Open Architecture for a Digital Library System and A Plan for its Development" (A13). It presents a draft research and development plan for a public information infrastructure that would enable digital library services. Kahn and Cerf were not newcomers to the creation of public infrastructure. They have received prestigious awards[2] for their earlier leadership and central roles in the creation and early development of the Arpanet – which famously became the Internet. Their recognized technical contributions to the Internet were for the protocols for packet-switching – the TCP/IP protocols. These transport protocols are called "low-level" because they govern how computers robustly send bits to each other in a digital network. With their backgrounds in computer science and electrical engineering, and networking specifically, Kahn and Cerf were well prepared to define these protocols. From their positions in the Information Processing Techniques Office of ARPA, they were deeply connected to the major centers of computer science research at the time, and well positioned to guide the creation of the net.

[2] Among other awards, Robert Kahn and Vinton Cerf were the winners of the Turing Award in 2004 for their pioneering work on networking. In 2005 they were awarded the Presidential Medal of Freedom for this work. They were inducted into the National Inventors Hall of Fame in 2006.

In A13, Kahn and Cerf considered a new challenge. They chose the metaphor of the "library" to set goals and expectations for a proposed higher-level digital information infrastructure.

> The term "library" conjures a variety of different images. For some, a library is a dim and dusty place filled with out-of-date texts of limited historical interest. For others, it is a rich collection of archival quality information which may include video and audio tapes, disks, printed books, magazines, periodicals, reports and newspapers. As used in this report, a library is intended to be an extension of this latter concept to include material of current and possibly only transient interest. Seen from this new perspective, the digital library is a seamless blend of the conventional archive of current or historically important information and knowledge, along with ephemeral material such as drafts, notes, memoranda and files of ongoing activity. *[A13, Summary, page 3]*

The theme of Kahn and Cerf's earlier, seminal contribution to the Internet was to "link computers together," creating an infrastructure for sharing data. In A13, they develop the related theme to "link libraries together" in an infrastructure that would enable people to share information and documents. In their view, users would participate with their personal computers acting as personal digital libraries.

> In its broadest sense, a DLS is made up of many Digital Libraries sharing common standards and methodologies. It involves many geographically distributed users and organizations, each of which has a digital library which contains information of both local and/or widespread interest. *[A13, page 3]*

Ultimately, the success and scope of A13 were limited by the library metaphor itself, which has proven unsuitable for fully characterizing commerce and information sharing in human organizations. This problem is that the normal activities in information exchange among competitive and sometimes adversarial human organizations are outside the scope of what normally happens in libraries. When A13 left behind the relatively simple world of computers that exchange bits, it entered the complex world of human organizations where activities involve ownership, competition, collaboration, and variations in law and agreement. This broader world of human commerce has many complexities and requirements that are not significant in libraries. Nor are these requirements comprehended or addressed in the goals and technologies described in A13.

- ***A13 Goals.*** In the U.S., the most influential libraries are public libraries and university libraries. The needs and experiences of these cooperative, non-

commercial institutions have proven to be somewhat misleading when used to set goals for a large-scale, commercially-oriented information infrastructure. For example, the library-oriented vision lacks a realistic consideration of information security and regulation of how information is used, which play central roles in the activities of online commercial systems.

- ***A13 Technology***. For their leadership of the digital library project, Kahn and Cerf had to reach well beyond the EE/CS research that they knew best. For example, the technical projections on which elements of their approach depended (such as progress in natural language understanding by 2003) have proven to be overly optimistic at least in time-scale. Furthermore, some of the main technical ideas in their proposal have so far proven unworkable, and have been superseded by very different approaches. For example, A13 suggests transmitting mobile code ("Knowbots") on behalf of a user to run at distant sites in order to search through their information files. The technology approaches that have come to dominate searching services in the World Wide Web are a study in contrast to A13's proposals. For example, Web search services such as Google employ web-crawlers and indexing engines with massive resources[3] for computation and storage[4]. Restated, the technical approaches that have proven most practical for massive search transmit content rather than transmitting programs ("Knowbots").

The issues around goals and technologies in A13 are elaborated in the following sections.

2. Setting Goals for Digital Libraries

In the United States, most major libraries are either public libraries or are affiliated with higher educational institutions. Although there are also corporate research libraries and some collections in private ownership, most of these libraries are connected to academic libraries, especially for purposes of obtaining access to collections through inter-library loans. Kahn and Cerf also mention databases – for information such as scientific data, public records, law records and medical data. Such databases are mostly outside of the

[3] Computation facilities known as "server farms" are employed by search companies to index the web. Server farms can cover several acres and have upwards of tens of thousands of computers.
[4] Although web sites can have their own search services, their information is part of what is sometimes called the "dark web" if the information is not open for search by the main search engines.

library system and have not been freely accessible to the public. In their proposed "digital library project," however, Kahn and Cerf sweep all of these sources and kinds of information into the familiar library pattern.

The mainstream activities of public and academic libraries establish key expectations. For example, libraries are not organized as commercial enterprises and their services are not for hire. Public libraries serve a public good, and academic libraries prioritize the particular needs of their academic communities. Libraries do not charge for information. Monetary transactions in libraries are mainly about record keeping and cost recovery for inter-library loans. Libraries do not advertise or suggest that people should buy certain sources of information. Librarians often provide facilities for making free copies of materials and are at least generally aware of copyright laws and the principle of fair use, whereby library clients may copy certain materials for their own scholarly purposes. Most libraries don't house secret information such as private commercial data or data for which there are privacy concerns. Their goal is to make all of their information holdings available to all of their clients rather than making selected information differentially available to different people.

The mission of traditional libraries is to make information available to the public. In the main, libraries are not concerned with regulating who has access to information or what people can do with information once they have it. These library-centered assumptions about information and its use are reflected in the design goals and assumptions of the digital library proposal described in A13. Specifically, they affect how A13 sets expectations and goals about regulated uses, about commerce and business models, and about security.

2.1 Setting Goals for Regulated Access

In the world of competitive and adversarial commercial activities, organizations create trust boundaries that regulate access to information. For example, the accounting department in an enterprise keeps its files of information within a trust boundary where access is limited to people with particular authorizations. The ability to read or update accounting records is limited to those who have authorization to operate inside the trust boundary. A company often has multiple trust boundaries for different parts of its

operations. For example, human resource records are used by different people than engineering plans for future products. For another example, law firms representing multiple clients must keep the client information separate. Two competing companies do not generally have access to each others' customer, payroll, sales, or strategic planning information.

A diagram of the information infrastructure for a commercial environment would have elements supporting information use and elements supporting security. For example, it would include firewalls and other security systems to detect intrusions and to keep malicious software out. It would have cryptographic services that support codes to protect information from prying eyes and, as part of a communication subsystem, to robustly identify communicating parties and to protect the integrity of content. It would include authentication services to identify and authorize particular parties to access information.

Figure 1 reproduces a diagram from A13 showing the structure of the proposed Digital Library System. The figure shows that personal and organizational digital libraries are linked together on the Internet together with various databases. Boxes in the figure indicate services for registering documents, importing documents into the system, indexing and cataloging. There are also accounting and billing services – suitable for handling library fees. In the figure, multiple personal library systems co-exist with multiple organizational library systems without trust boundaries and few provisions for security[5].

[5] For example, page 23 of A13 describes a registration server as responsible for "registering new users, sources of information (databases) or other components newly added to the system." In the further description about this, the main concern is about assigning unique identifiers as an aid to indexing and cataloging. No where are issues of verifying the authenticity of users or content discussed. Like an open library, the users are expected to behave in a kind of honor system. In contrast, more is at stake in an adversarial commercial environment. Digital systems supporting commerce and information access in such an environment would need robust and automatic means for checking the credentials of users and content before authorizing either access or changes to information.

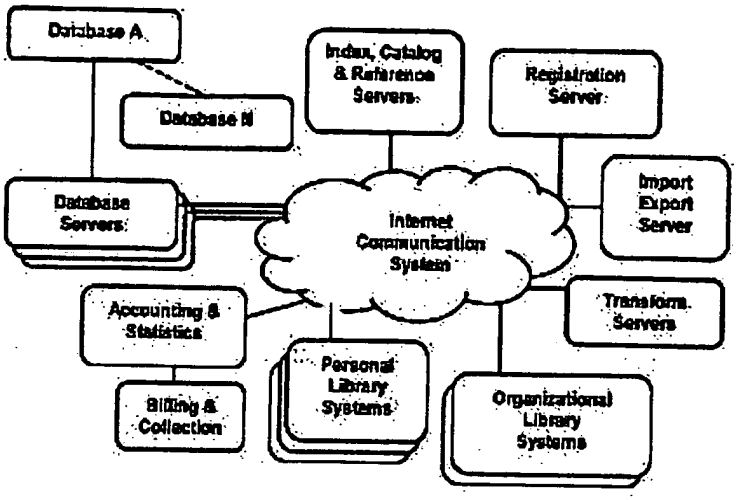


Figure 1. Reproduction of Figure 2 from A13 – "Structure of the Digital Library System"

In understanding the scope and limitations of A13, what is interesting is what is missing. None of the normal commercial considerations involving security, authorization and trust are indicated in the diagram or elsewhere in the text of A13. There is no system "threat analysis" to characterize security requirements. Such provisions are outside the scope of libraries and they are outside the scope of the proposed digital library design[6]. Overall, A13 pays scant attention to commerce or security.

2.2 Setting Goals for Commercial Use

The early government-supported computer networks in the United States allowed no commerce. In the 1970s, if someone sent an email that could be construed as a "commercial message," it would often set off a flurry of discussion about inappropriate use of the network. This "non-commercial" sensibility – coupled with the context of libraries – is part of the background context for A13.

Among the commercial businesses that use information, publishing stands out for its dependence on the regulation of uses of content in order to sustain its business. In publishing, authors and people in other creative roles create new content. Publishers and

[6] On page 26 there is the following sentence: "To increase system integrity, the Accounting and Statistics Servers should be configured to accept data only from the appropriate sources and to raise alarms when data arrives from an unexpected source." In this way the design focuses on accounting reliability, in contrast to security as it relates to regulated use of content.

distributors publish, distribute, and sell content. Consumers purchase and consume content for their information or entertainment purposes -- such as playing music, watching a movie, or reading a book. People carry out different activities around information according to their roles as creators, owners, distributors and consumers. For example, consumers do not typically write in or modify books that they buy, make copies, and then distribute them in competition with the original authors or publishers.

Various arrangements are routinely employed to promote commerce in the sale and distribution of content. For example, content may be offered at a temporary discount to encourage sales. Tiered pricing is used to maximize profits by enabling different pricing in different markets. For example, price may depend on the time of sale, the location of the sale, or the affiliations of the purchaser. Special discounts may be offered for students, for members of a particular organization, for senior citizens, or for handicapped persons. For another example, businesses distinguish between the making of additional copies of content and the serial re-use of a single copy. Rental services circulate content without increasing the number of copies. Volume discounts are offered – reflecting differences in the costs of sales when large numbers of copies are sold at once. These sometimes socially-aware concepts arise in commercial publishing business practices. Successful use of these approaches in a digital publishing regime requires that information systems have adequate means for regulating the distribution and use of content.

In contrast, such an elaboration of requirements for success of commercial arrangements in a digital content regime is outside the scope of A13. A13 reflects the idea that a library is a place where clients get information. A13 has a model of owner's interests which is largely based on some accounting of levels of library usage *without* regulation of how information is used.

To their credit, Kahn and Cerf recognize that when information is digital, there are natural concerns about intellectual property protection. After all, compared to the substantial effort involved in reproducing (say) books as bound volumes, it is often very easy to copy (unprotected) digital information regardless of the amounts. Kahn and Cerf enumerate some of the problems and also acknowledge that solutions were not known at the time of their writing.

> At present, the basis for intellectual property protection in the U.S. is Patent and Copyright law. The large scale aggregations of information found on CD-ROMs and the selective access to information found in on-line databases may require substantial re-thinking of the ways in which the creators and owners of such information are compensated for its use. There are many issues at stake in this area, not the least of which relate to the ease with which information can be replicated once in digital form and the rapidity with which large quantities of information can be processed (accessed, transferred, analyzed, integrated, etc.). Concepts of value and pricing and royalty for use of information could require considerable revision if the cost of such use is to remain within reason. One does not now pay an author a royalty each time a book is read. However, a royalty may be earned each time a song is played in public, though not in private. If a thousand books are combined on a single CD-ROM and the acquirer of the CD-ROM only intends to read one of them, what sort of royalty arrangement is appropriate to compensate the copyright owners? How would compensation be extended for cases in which electronic copies are provided to users? In fact, the concept of copying or duplicating a work may no longer be the essential factor in calculating royalties since far more complex actions may now be taken on digital information.
>
> These questions are not trivial in nature nor have many workable solutions been proposed thus far. *[A13, pages 11-12]*

In this way, Kahn and Cerf encounter difficulties that are inherent in the library metaphor. Libraries do not themselves make copies of books (for example) in order to save on purchasing costs from publishers. They sometimes provide copiers so that clients can make their own limited copies of information for personal use. Such activities are governed by the provisions of U.S. copyright law. In the preceding quotation, Kahn and Cerf acknowledge that the kinds of activities likely to take place in a digital information infrastructure go beyond the routine activities in a library. When they suggest that "the concept of copying or duplicating a work may no longer be the essential factor in calculating royalties" they are in effect acknowledging that the economic agreements that form the basis of the publishing business break down when people make many (unauthorized and unrecorded) copies. In a digital network, provisions of Copyright Law apply, but enforcement can be intractable.

Kahn and Cerf recognized that solutions to this problem are needed. In terms of the legal basis for regulating use of content, they did not anticipate the possibility of using

Contract Law where Copyright Law falls short. Not recognizing the relevance of Contract Law, they also did not recognize the value of having a machine-understandable declarative language in which "contracts" can be expressed – so that their terms could be explicitly presented both to people and to computer systems. They did not anticipate that secure computer systems could play a practical role in the enforcement of the sort of declarative "digital contracts" that could robustly support the range of human agreements and activities typical in commerce.

In the 1990's, other people developed concepts and technologies that addressed these problems[7]. These approaches led to today's digital rights management (DRM) systems. In contrast to the library metaphor, DRM approaches recognize that there are different and distinguished uses or operations on content, such as making copies, modifying content, printing it, distributing it, selling it, and so on. DRM systems provide means for regulating these different uses in order to sustain an economy in which content is produced and consumed. Conditions are associated with the operations. These conditions must be satisfied before the operations can be performed. For example, a person might need to pay a fee, be a certain age, live in a certain jurisdiction, or belong to a particular group, and so on.

It is not surprising that these distinguished operations and conditions are outside the scope of the library metaphor. Libraries do not create or sell content. They are concerned mainly with making information available to the public. Once a client "gets" information from a library – such as checking out a book—the library is not involved with what the client does with it. Libraries generally do not regulate how information is used or who has access[8].

---

[7] See for example, Stefik, Mark. "Letting Loose the Light: Igniting Commerce in Electronic Publication." In *Internet Dreams*, The MIT Press, Cambridge, Ma., 1996, pages 219-253. The foreword of this book was written by Vint Cerf, one of the authors of A13. The book proposes four metaphors for understanding the origins and directions of the Internet: digital libraries, electronic mail, electronic markets, and digital worlds. In contrasting digital libraries to electronic markets, it highlights the different capabilities that are suggested by the different metaphors. Specifically, this book elaborates how the metaphor of the electronic market usefully covers activities that are not associated with libraries.

[8] The more recent controversies about regulating access by children to certain materials on the Internet, and the reporting of which people have had access to certain information, has been controversial in part because of the conflict with the over all mission of libraries to provide the public with access to information with a minimum of barriers.

Discussion of these issues in A13 adheres to the expectations set by libraries. When Kahn and Cerf imagine what people will do with the information in digital libraries, they say people will "register, store, catalog, search, retrieve, and manipulate digital information in the library" (A13, page 8). These are the same kinds of activities that people could already do in traditional libraries. Notably absent from this list are commercial activities like selling or distributing information, or re-publishing information in revised forms.

In summary, A13 does not investigate the commercial situations where different people have different roles and sanctioned activities around information. A13 characterizes operations on information in library terms – relating to catalogs and retrieval and so on. It does not analyze requirements of a publishing business or requirements among competitive enterprises around regulated use of information.

3. Technology Choices

A13 proposes a high-level draft architecture for a digital library infrastructure. Analogous to the network architecture of the Internet, Kahn and Cerf proposed a network of computers where the nodes of the network are digital libraries. The architecture also included specialized servers for indexing, cataloging, registration, and other specialized functions.

Figure 2 reproduces a diagram from A13 illustrating its high-level system elements for a personal library system (PLS)—a node in the network of libraries. The diagram shows the structure of the PLS in terms of "layers" with an operating system and its device drivers in the bottom layer and application elements in the top layer. The second layer from the bottom contains file services, presentation (display) services, and network transport services. The top layer of the diagram divides elements into ones that serve the user (user interface), ones that access library content, and administrative functions.

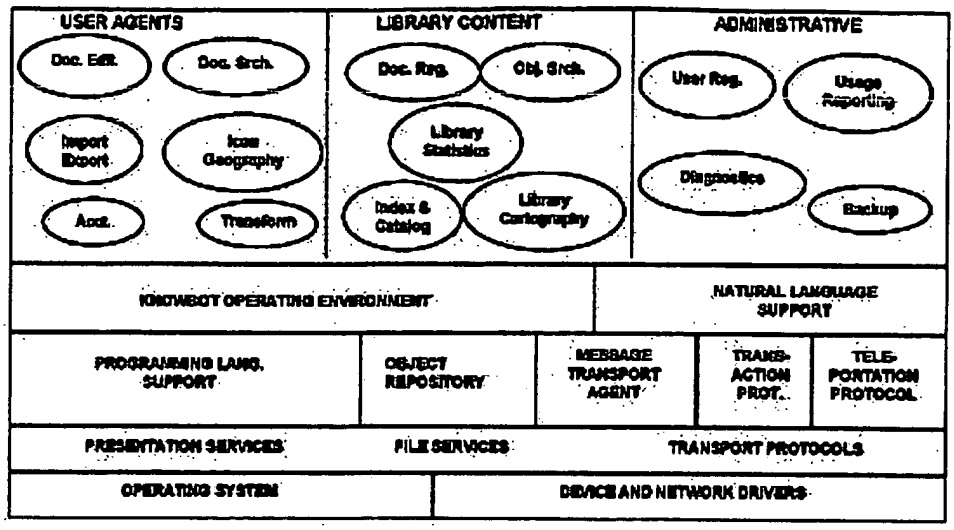


Figure 2. Reproduction of Figure 3 from A13 – "Personal Library System Structure"

The fourth layer of the architecture is of particular interest and contains the most novel aspects of the design. It contains a Knowbot operating environment (KNOE) and natural language support. Knowbots are central in Kahn and Cerf's design for many of the capabilities of the digital library infrastructure. A13 describes Knowbots as follows:

> A Knowbot is an active program capable of operating in its native software environment. Knowbots are present in each of the various components of a Digital Library System. They can be cloned, replicated, created, destroyed, can be resident at a given host system or can move from one host machine to another. Knowbots communicate with each other by means of messages.
>
> Knowbots act as the primary medium of communication and interaction between various major components of the Digital Library System. They may even transport other Knowbots. Generally, a Knowbot may be viewed as a user Knowbot or as a system Knowbot depending on whether it directly serves an individual user or not. [A13, page 34]

Knowbots are more than an incidental part of the proposal for a digital library infrastructure. Unlike the other system architectural concepts, A13 devotes a full section for describing Knowbots (Section 3: Knowbots and Their Application). As Kahn and Cerf noted:

> The selection of a methodology for building Knowbots and even the determination whether an object-oriented language is essential are two

> of the highest priority research questions for the Digital Library Project to resolve. [A13, page 31].

Many of the issues with the architecture in A13 arise from fundamental difficulties with Knowbots, especially security issues[9]. The use of Knowbots as mobile code in the system architecture was an aggressive design choice. Although concepts of object-oriented programming had been developed in the Computer Science community for over a decade[10], the research focused on systems for single computers[11]. Object approaches for distributed and mobile code applications were less explored. Nor had object-oriented applications been deeply explored involving mobile objects or their security considerations across multiple organizations.

The following sections examine Knowbot-related problems with the design in A13 involving trust among multiple parties, security vulnerabilities, and practical problems in using opaque ("black box") mobile code to represent agreements between producers and consumers of content.

3.1 Trust among Multiple Parties

Knowbots are employed in the proposed digital library architecture to carry out many functions. As described in A13, Knowbots can be asked to retrieve or file documents. They also watch over information objects on behalf of owners.

> One set of system Knowbots specifically attend to locally available library information. They take requests from user Knowbots and actually retrieve the documents from storage (or conversely store them away). Another set of system Knowbots attend to background and administrative tasks such as diagnostics, backup and accounting. [A13, page 34]

---

[9] Since the focus of A13 is on libraries rather than commercial organizations, it does not focus on security issues. Nor does A13 explore security requirements for Knowbots.

[10] On page 30, A13 cites Smalltalk, Common Lisp, Common LOOPS and C++ as examples of existing object languages. Smalltalk and Common LOOPS were both largely developed at PARC where I work. I was one of the contributors to the LOOPS and Common LOOPS specifications at the time and was an active researcher in the object-oriented programming community.

[11] For example, see Stefik, M. Bobrow, D.G. Object-oriented programming: Themes and Variations. *AI Magazine* 6:4, pp. 40-62, Winter 1986. (Reprinted in Peterson, G.E. (ed), *Object-Oriented Computing*, Volume 1: Concepts, IEEE Computer Society Press, pp. 182-204, 1987. Also reprinted in Richer, M.H. (ed.) *AI Tools and Techniques*, pp. 3-45, Ablex Publishing Corporation, Norwood, New Jersey.)

In carrying out these functions, Knowbots would have to travel through the network of libraries. When we consider this architecture in the context of commercial activities, a problem of trust arises. Suppose that Company A has a computer system. A document and its Knowbot arrives from Company B. How does Company A know that the Knowbot can be trusted?

A13 only partially addresses the issue of trust. In the context of courier Knowbots, it says the following:

> A class of trusted Knowbots called *couriers* have the special responsibility to look after selected objects on behalf of their authors or other owners of rights in the objects. [A13, page 34]

In the example, the courier is expected to look after the interests of Company B. The problem is that the Knowbot consists of mobile code that is being enabled to run on the computers of Company A. What assurances does Company A have about what the Knowbot will do? Will it correctly enforce the agreements that Company A has with Company B about use of the document? When the Knowbot reports back to Company B, will it also send back any additional information that compromises the business interests of Company A? Will it confine its activities to the document from Company B, or will it read or modify any other sensitive documents in the library of Company A? Even if the intentions of Company B are completely legitimate, what if there is a bug in the mobile code that inadvertently leads to commercial losses for Company A? After all, the coders of the Knowbot have presumably not been able to test the Knowbot on the computers of Company A. Allegorically, these scenarios are akin to "turning a fox loose in the hen house."

Such concerns about mobile code are not unrealistic. The most familiar examples of mobile code today are computer viruses. Analogous to Knowbots, viruses travel from system to system "carrying out the wishes of their creators" – which are generally disruptive. Most businesses invest heavily in virus detection and firewalls in order to prevent unwanted infections that can compromise their computers and disrupt their

businesses. Even today nearly two decades since A13 was written, mobile code has had very limited use beyond support for user interfaces in web pages[12].

Nor are all of the risks associated with Company A. Suppose that the KNOE (Knowbot Operating Environment) in the computers of Company A has been altered. It could potentially separate a Knowbot from its document, and provide its own altered-Knowbot which acts differently. It could under-report the usage statistics in order to reduce fees. It could alter the Knowbot from Company B and send it on to Company C in a way that intercepts a "funding stream" to the coffers of company A rather than Company B. Allegorically, these scenarios are akin to "sending a lamb to a den of wolves."

A fundamental problem in A13's conception of Knowbots is that they are designed to serve the interests of *one* party. However, commercial transactions *inherently* involve multiple parties with differing interests, such as a producer and a consumer. A13 does not discuss any requirement for robust, verifiable and accountable trust across multiple parties. Furthermore, since programs and programming environments are so complex, the concepts of Knowbots and KNOEs does not appear to be a workable approach for achieving such security[13].

DRM systems like those invented in the 1990s address multi-party security issues in a way more appropriate for commercial use: DRM systems like those invented in the 1990s use host computers configured as trusted systems, rather than as untrusted hosts. Each trusted system is designed and verified by a third party to guarantee to two contracting parties that the trusted system can be relied upon to act as an impartial agent to enforce the terms of any digital contract that has been agreed to by those parties. In this way, DRM approaches address multi-party security issues in a way more appropriate for commercial use.

In summary, the Knowbot architecture of A13 does not recognize that trust in a commercial setting requires a basis of trust across multiple parties. The Knowbot-based

[12] Even the use of mobile code in the limited context of "JavaScript" in web pages introduces security issues.

[13] Even if the "source code" of Knowbots and KNOEs were made available to both parties, since there would be so much variation in Knowbots and environments it would not be practical to prove their correctness. It would also be difficult to prove (for example) that Knowbots were not transported to different environments than the ones that were tested.

architecture contains no provisions for this. As suggested by scenarios above, it appears that the Knowbot approach is not a sound basis for building such systems.

3.2 Communication and Code Security

Libraries generally do not often have active adversaries and their every day operations do not give much attention on security in communication. For example, when a library loans a client a book, there is typically no concern about whether the book has been altered to contain false information. In contrast, commercial organizations have private and critical communication. It is normal security practice to use encryption, redundant check sums, and other techniques in commercial communication systems to assure secrecy and integrity. For example, regular web users are familiar with providing passwords and with the "encrypted page" messages that they get in their web browsers when critical information is requested. Such concepts, however, are outside the activities of traditional libraries. Furthermore, they are not considered or even mentioned in A13.

The lack of attention to cryptography, certification and other means of securing communication has widespread implications for the system design and its applications. For example, consider again the transmission of mobile Knowbots. Knowbots are proposed for many functions in the digital libraries – used for retrieval and search, to billing, and even as couriers. When a Knowbot is being transmitted from one system to another, it is represented by bits in packets over the communications network. Today, several years of experience with computer viruses has made designers of computer systems aware of a myriad of possible "attacks" on communications systems[14]. By various means, communications can be intercepted, faked, copied, altered, and blocked. Lacking such understanding, A13 does not recognize that its Knowbots are vulnerable to being intercepted, faked, copied, altered, or blocked.

To restate the design issue, A13 proposes Knowbots as the security mechanism to act on behalf of owners. However, Knowbots are software-only entities. They are expected to travel over networks where they are vulnerable to modification, and to run on many

[14] For example, see Chapter 3, "The Digital Wallet and the Copyright Box: The Coming Arms Race in Trusted Systems" in Stefik, Mark. *The Internet Edge: Social, Technical, and Legal Challenges for a Networked World.* Cambridge, Ma. The MIT Press, 1999. Robert Kahn and Vinton Cerf, the authors of A13, both provided endorsements of the book to the publisher. These appear on the back cover of this book.

different computer systems where they potentially could be modified. This leaves the foundations of security in A13 – the infrastructure that is supposed to look after the interests of content owners – profoundly vulnerable and insecure. In contrast, modern security designs have design features that address security in communication (communication integrity), authentication (behavioral integrity), and hardware (physical integrity).

A13 specifically proposes to leverage the Internet but to integrate the digital library infrastructure with existing technologies.

> Before describing specific features of the Digital Library System, it will be helpful to review some of the fundamental assumptions which strongly affect its design. Perhaps the most dominant of these assumptions are that the system is distributed, heterarchical, hierarchical, networked and strongly display-oriented. In addition, it must have an ability to interact with other autonomous Digital Library Systems that do not adhere to its internal standards and procedures. [A13, page 19]

From a computer security point-of-view, a chain is as strong as its weakest link. Every subsystem that does not meet security standards creates a vulnerability. This is the opposite of the trusted system approach of DRM systems of the 1990s, which require that every system in a transaction be certified as trustworthy.

In summary, the approach taught in A13 is vulnerable to communications attacks which would need to be addressed in a context of commercial use.

3.3 Representing Agreements

When people engage in financial transactions, they often formalize their agreements with contracts. For example, contracts and warranties are common when people rent an apartment, buy a home, take out a loan, or buy an expensive appliance. There are many variations in contracts – in terms of what provisions are included. At the same time, there are often a few key issues, such as the loan amount, fees, and the interest rate and term of a loan. Contracts explain the terms and conditions of the agreements, and lay out the rights of the parties. The substance of an agreement is not hidden. Rather it is written openly in the contracts. Contracts are intended to express the terms and conditions so that any of the parties to a transaction can examine them and understand them.

Contracts do not play a highly visible role in traditional libraries. For example, there are very few variations in the terms and conditions of book loans. There may be some books that cannot be checked out, and there may be some books that have to be returned more quickly than others. However, contracts are not a major focus for a traditional library. When Kahn and Cerf chose the library metaphor for their proposed digital library infrastructure in A13, they did not mention contracts as a design element.

A13 proposes Knowbots to take the responsibility to look after the interests of users and others in the digital library. Absent any other means, Knowbots are A13's only vehicle for representing and enforcing agreements. When there are multiple, different agreements between parties, multiple, different Knowbots would be required.

The problem with this approach is that Knowbots are mobile computer code and computer code is notoriously non-transparent and hard to understand for most people. Most of the code in a program is generally about its internal bookkeeping and managing relations with other objects in its operating environment. To understand what code does, a programmer must understand not only the computer language, but also the operating environment in which the code is operating.

Using Knowbots to represent a myriad of business agreements is not practical. The important and salient elements of human agreements would be buried and essentially obfuscated by putting them into a program. The opaqueness would make it very difficult to understand what agreement is represented by a Knowbot, or even to tell the difference between a genuine Knowbot and a rogue Knowbot.

The DRM systems that were invented in the early 1990s addressed the problem of representing agreements through the use of a declarative digital rights language[15]. These languages were designed to express the kinds of operations, terms and conditions that are salient for practical contracts about the use of digital content. For example, there were specific operations for copying, loaning, printing and other common things. Terms and conditions could express a range of requirements for payments, times of use, and so on.

[15] For example, see "Letting Loose the Light" in Stefik's *Internet Dreams* book for an example of a digital rights language. See "The Bit and the Pendulum" in Stefik's *The Internet Edge* book for a discussion of digital contracts.

Digital contracts could be expressed in a grammar. From there they could be presented to people in clear user interfaces. They could also be interpreted and enforced by computers.

In summary, the Knowbot approach is not practical for supporting the negotiation or understanding of agreements. The subsequently-developed rights language approach side-steps the main difficulties of the Knowbots approach by making it unnecessary for people to try to discern the meaning of an agreement from the programming code of a Knowbot.

4. Concluding Remarks

A13 was a visionary proposal for a digital library system. The design was guided by the metaphor of a traditional library as a system to enable people to share information. Like traditional libraries – A13's focus is on exchange of information. True to the purpose of libraries, A13 is not much concerned with an infrastructure for commerce in digital content, which would require much more attention to mechanisms for commerce and to security requirements for transactions among potentially competitive parties. A13 made very different (and often opposite) design choices from many of the DRM systems that were invented in the 1990's and the systems that are deployed today.

Appendix A. Qualifications

Education and Research

My university education was at Stanford University, both undergraduate and graduate. I received my Bachelors degree in Mathematics in 1970 and my doctorate in Computer Science in 1980. I am a Fellow in the American Association for the Advancement of Science (AAAS) and also in the American Association for Artificial Intelligence (AAAI).

I work at the Palo Alto Research Center (PARC), where I am a research fellow. Since I started at PARC in 1980 I have taken several tours of duty in research management, leading three technical areas and one of PARC's laboratories for several years. I occasionally teach courses and give lectures at Stanford University and U.C. Berkeley. I have been an external thesis advisor and dissertation committee member for Ph.D. students at Stanford, U.C. Berkeley, and the University of Maryland.

I have published five technical books including *The Internet Edge: Social, Technical, and Legal Challenges for a Networked World* (MIT Press, 1999), *Internet Dreams: Archetypes, Myths, and Metaphors* (MIT Press, 1996), and *Introduction to Knowledge Systems* (Morgan Kaufmann Press, 1995). I have published over forty technical papers.

Some of my technical work is mentioned in the *Open Architecture for a Digital Library System* paper (A13), which cites Common LOOPS among current object languages. I was one of the main creators of the Loops system and a contributor to the Common LOOPS specification. PARC was a center for much of the research that inspired Kahn and Cerf during this period – including the technologies for distributed computing, the SmallTalk language, and the NoteCards hypermedia system which are mentioned in the paper.

As a computer scientist, I am somewhat of a generalist and have switched my area of focus every few years. A unifying goal in my work has been to enhance the creation and sharing of knowledge. My dissertation work on an expert system for experiment planning included a frame-based knowledge representation system. A version of this was later commercialized by Intellicorp. My research on collaboration in electronic meeting rooms ("Colab") included creating an infrastructure for distributed objects. The Colab research

led to collaboration with Bob Kahn and others in the creating of the "National Collaboratory" projects in the U.S.

My current research on "sensemaking systems" is about technology to help people facing information overload to master and understand large amounts of information in carrying out their work. Our sensemaking projects at PARC are multi-disciplinary, involving computer scientists and cognitive psychologists as well as specialists in natural language technology, user interfaces, and distributed systems. The current directions in this involve what we call "augmented social cognition," which is the technology that aggregates and combines the sensemaking contributions of large groups of people.

Relationship to Robert Kahn and Vinton Cerf

Bob Kahn was a fairly frequent visitor to the Heuristic Programming Project at Stanford University in the mid 1970s when I was a graduate student there. I got to know him since he worked closely with Edward Feigenbaum, who was one of my faculty advisors. In the early 1980's after I had graduated, I was a participant together with about a dozen others in some weekend workshops led by Bob Kahn at Stanford when the early ideas for a "digital library project" were being developed. Some of the other participants at the workshop were professors from Stanford (Edward Feigenbaum, Joshua Lederberg, John McCarthy) and MIT (Marvin Minsky). In this way I was able to contribute in a small way to the early stages of the Digital Library Project, even before the Open Architecture paper was written.

Bob and I have worked together on various projects over the years. I typically see him two or three times a year. When I first developed the concepts of digital rights management in the early 1990s, Bob signed a non-disclosure agreement with the Palo Alto Research Center so that we could discuss the ideas in some depth and also plan participation in some coordinated activities, such as the "digital object identifier" project.

Vinton Cerf was a professor at Stanford in the Computer Science Department when I started there as a graduate student. Initially, my research plan was to do a dissertation in the Systems area and he was my graduate advisor. Within a year or so, however, he decided to leave Stanford to work on developing the Internet (then ARPANET) at DARPA. I switched research areas in Computer Science to artificial intelligence,

focusing on expert systems. We have kept in occasional contact over the years. When I published the book *Internet Dreams* with MIT Press in 1995, he graciously wrote the foreword to the book. This book contained my first publication (beyond patents) of the ideas for digital rights management in the chapter titled "Letting Loose the Light: Igniting Commerce in Electronic Publication." The book also includes an excerpt from Kahn and Cerf's paper *The World of Knowbots*.

In 1999 I wrote a second Internet book – *The Internet Edge* – that provided a deeper analysis of the trends in networks, and discussed social, technical, and legal challenges. Both Robert Kahn and Vinton Cerf provided endorsements of the book to the publisher that appeared on the jacket of the book.

Mark Stefik 30 May 2007

O10 An Open Architecture for Digital Library System and a Plan for its Development - CNRI (1998)

Corporation for National Research Initiatives

Programs/Activities List

Research Programs

- CORDRA
- D-Lib® and D-Lib® Magazine
- DVIA
- Digital Object Architecture
- Digital Object Identifier
- Digital Object Store
- Handle System
- Knowbot Programs
- MEMS Exchange
- Repository Architecture
- Speech and Language

Digital Object Architecture Project

Note: the Digital Object Architecture Project includes the CNRI Repository.

CNRI's program of research and development in digital libraries has a number of inter-related activities that overlap and build upon each other. The work includes development of core technology that is usec in several testbeds and implementation projects, with funding from a variety of sources.

The **Digital Object Architecture Project** continues the architectural work of the DARPA-funded **Computer Science Technical Reports Project** (CSTR).

The project focuses on the development of an infrastructure of services that provide access to distributed and secure digital objects. Digital objects are networked objects that are instantiated by an infrastructure service we call a repository. Digital objects provide access to their content using an extensible and secure dissemination mechanism. Disseminations can be thought of as high level types that are uniquely distinguished by a combination of operations, and types of data the latter are performed on. Disseminations consist of mobile code called Servlet that can be designed, implemented, and registered with the digital object infrastructure by anyone with the proper permissions. Any digital object with the appropriate rights can automatically use registered servlets. This extensible dissemination mechanism enables digital objects to accommodate a wide variety of possible content, from complex to simple, static or dynamic, and from permanent to real time data. Disseminations have few operational limits and enable digital objects to dynamically generate or acquire their content.

Current ongoing research includes the development of dissemination registry, infrastructure searching, security and scalability.

Support for the Digital Object Architecture project is provided by DARPA, the Library of Congress, and the Defense Technical Information Center (DTIC), through DARPA grant MDA972-92-J-1029.

Technology

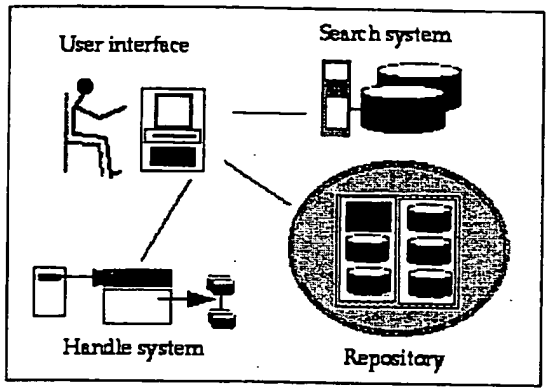


Figure 1

Figure 1 shows the principal system components. CNRI's research concentrates on the concept of digital objects, the Handle System for identifying digital objects, and the Repository for storing them and making them available over the Internet. The Registry is a specialized repository that is used to authenticate digital objects.

The Handle System is a system for providing persistent names for Internet resources. It is a highly reliable, high performance, distributed system.
The Repository provides network based storage and access to digital objects. All access to digital objects passes uses a simple repository access protocol and is subject to access controls established by the manager of the repository.
The Registry is a specialized repository that provides secure registration and authentication of digital objects.

Partners

CORDRA/ADL. CNRI is designing a registration system for the ADL CORDRA project, to be known as the ADL Registry or ADL-R, and deliver it for use at the Defense Technical Information Center (DTIC). This effort will enable the discovery and reuse of learning content held in repositories distributed across the DoD.

Defense Virtual Information Architecture. The Defense Technical Information Center (DTIC), the Defense Advanced Research Projects Agency (DARPA), and CNRI are working to extend and transfer CNRI's Digital Object Architecture work into a prototype digital library implementation.

Additional Information

- Kahn, Robert & Wilensky, Robert. "A framework for distributed digital object services"; *International Journal on Digital Libraries* (2006) 6(2). [**doi:10.1007/s00799-005-0128-x**]. **Reproduced** with permission of the publisher.

- Henry Jerez, Giridhar Manepalli, Christophe Blanchi, and Larry Lannom, "ADL-R: The First Instance of a CORDRA Registry", *D-Lib Magazine*, February 2006. [**doi:10.1045/february2006-**

jerez]

- Giridhar Manepalli, Henry Jerez, and Michael L. Nelson, "FeDCOR: An Institutional CORDRA Registry", *D-Lib Magazine*, February 2006. [**doi:10.1045/february2006-manepalli**]
- Talk by Dr. Robert Kahn on the **Digital Object Architecture and it's importance to the future of the Internet**. [Windows Media Format, 240 MB]
- Christophe Blanchi, Jason Petrone, "**Distributed Interoperable Metadata Registry**", *D-Lib Magazine*, December 2001. [doi:10.1045/december2001-blanchi]
- Christophe Blanchi, Jason Petrone, "**An Architecture for Digital Object Typing**", White Paper, CNRI, September 2001. [hdl:4263537/4096]
- Robert E. Kahn and Vinton G. Cerf, "**What is the Internet (And What Makes It Work)**", prepared by the authors at the request of the *Internet Policy Institute*, December 1999.
- Sun, Sam, "**Handle System Namespace and Service Definition**", TWIST '99, Irvine, California, August 19, 1999.
- "**Managing Access to Digital Information**" Cross Industry Working Team (XIWT), July 1999.
- Sandra Payette, Cornell University; Christophe Blanchi, CNRI; Carl Lagoze, Cornell University; Edward A. Overly, CNRI, "**Interoperability for Digital Objects and Repositories: The Cornell/CNRI Experiments**, *D-Lib Magazine*, May 1999. [doi:10.1045/may99-payette]
- Laurence Lannom. "**Handle System Overview**". ICSTI Forum, No. 30, April 1999.
- William Y. Arms, "A national library for undergraduate science, mathematics, engineering, and technology education: needs, options, and feasibility (technical considerations)". In: **National Research Council, "Developing a national digital library for undergraduate science, mathematics, engineering, and technology education".** Washington D.C.: National Academy Press. 1998.
- "**Implementing Policies for Access Management**" by William Y. Arms, *D-Lib Magazine*, February 1998. [doi:10.1045/february98-arms]
- William Y. Arms, "**Digital Object Identifiers (DOIs) and Clifford Lynch's five questions on identifiers**". ARL Newsletter October 1997.
- "**An Architecture for Information in Digital Libraries**" by William Y. Arms, Christophe Blanchi, Edward A. Overly. *D-Lib Magazine*, February 1997. [doi:10.1045/february97-arms]
- "**Uniform Resource Names: A Progress Report**" by the URN

Implementors. *D-Lib Magazine*, February 1996.
[hdl:cnri.dlib/february96-urn_implementors]

- "A Design for Inter-Operable Secure Object Stores (ISOS)" by Carl Lagoze, Robert McGrath, Ed Overly, Nancy Yeager. Cornell Computer TR95-1558.

- "Implementation Issues in an Open Architecture Framework for Digital Object Services" by Carl Lagoze and David Ely. Cornell Computer Science Technical Report TR95-1540.

- Lagoze, Carl, "**A Secure Repository Design for Digital Libraries**", *D-Lib Magazine*, Decmeber 1995.
[doi:10.1045/december95-lagoze]

- "**Key Concepts in the Architecture of the Digital Library**" by William Y. Arms, *D-Lib Magazine*, July 1995.
[doi:10.1045/july95-arms]

- "**A Framework for Distributed Digital Object Services**" by Robert Kahn and Robert Wilensky, May 1995.
[hdl:4263537/5001]

TechEncyclopedia More than 20,000 IT terms

Results found for: inverted file

inverted file

In data management, a file that is indexed on many of the attributes of the data itself. For example, in an employee file, an index could be maintained for all secretaries, another for managers. It is faster to search the indexes than every record. Also known as "inverted lists," inverted file indexes use a lot of disk space; searching is fast, updating is slower.

TERMS SIMILAR TO YOUR ENTRY

Entries before inverted file

- inverse addressing
- inverse kinematics
- inverse multiplexor
- inverse telecine
- inverse video

Entries after inverted file

- inverted list
- inverter
- InVircible
- invisible GIF
- invisible Web

DEFINE ANOTHER IT TERM

Define

Or get a random definition

Brian Cryer's Glossary of IT Terms with Links

Inverted File Index

Inverted File Index

A type of Inverted Index where each index provides a mapping from words to the documents that contain them. cf Fully Inverted Index.

Can you add to this definition? If so please Report an Observation. Do you know of a relevant link to add under this definition? If so please Add a Link.

Add a Term | Add a Link | Report an Observation | Glossary | Home

thefreedictionary.com

(database, information science) **Inverted index** - A sequence of (key, pointer) pairs where each pointer points to a record in a database which contains the key value in some particular field. The index is sorted on the key values to allow rapid searching for a particular key value, using e.g. binary search. The index is "inverted" in the sense that the key value is used to find the record rather than the other way round. For databases in which the records may be searched based on more than one field, multiple indices may be created that are sorted on those keys.

An index may contain gaps to allow for new entries to be added in the correct sort order without always requiring the following entries to be shifted out of the way.

NIST
National Institute of Standards and Technology

inverted file index

(data structure)

Definition: An *inverted index* which only indicates the text in which a word appears, not where the word appears within the text.

See also *full inverted index*, *block addressing index*.

Note: See the example at inverted index.

Author: PEB

More information

Nivio Ziviani, Edleno Silva de Moura, Gonzalo Navarro, Ricardo Baeza-Yates, *Compression: A Key for Next-Generation Text Retrieval Systems*, IEEE Computer, 33(11):37-44, November 2000, (page 42).

Go to the Dictionary of Algorithms and Data Structures home page.

If you have suggestions, corrections, or comments, please get in touch with Paul E. Black.

Entry modified 17 December 2004.
HTML page formatted Wed Apr 19 12:22:45 2006.

Cite this as:
Paul E. Black, "inverted file index", in *Dictionary of Algorithms and Data Structures* [online], Paul E. Black, ed., U.S. National Institute of Standards and Technology. 17 December 2004. (accessed TODAY) Available from: http://www.nist.gov/dads/HTML/invertedFileIndex.html

NIST
HOME

Inverted index

From Wikipedia, the free encyclopedia

An **inverted index** is an index structure storing a mapping from words to their locations in a document or a set of documents, giving full text search. An inverted index is the most important data structure used in search engines. Such an associative array is a multimap, and can be implemented in many ways. It could be a hash table, where the keys are words (strings), and the values are arrays of locations.

There are two main variants of inverted indexes: An *inverted file index* contains for each word a list of references to all the documents in which it occurs. A *full inverted index* additionally contains information about where in the documents the words appear. This could be implemented in several ways. The simplest is perhaps a list of all pairs of document IDs and local positions. An *inverted file index* needs less space, but also has less functionality. You can do *term search* (what you usually do in a search engine), but not *phrase search* (what you usually get when you put quotes around your search query).

Contents

Example

Given the texts T_0 = `"it is what it is"`, T_1 = `"what is it"` and T_2 = `"it is a banana"`, we have the following *inverted file index*:

```
"a":      {2}
"banana": {2}
"is":     {0, 1, 2}
"it":     {0, 1, 2}
"what":   {0, 1}
```

A *term search* for the terms `"what"`, `"is"` and `"it"` would give the set $\{0,1\} \cap \{0,1,2\} \cap \{0,1,2\} = \{0,1\}$.

With the same texts, we get the following *full inverted index*, where the pairs are document numbers and local word numbers. Like the document numbers, local word numbers also begin with zero. So, `"banana": {(2, 3)}` means the word "banana" is in the third document (T_2), and it is the fourth word in that document (position 3).

```
"a":      {(2, 2)}
"banana": {(2, 3)}
"is":     {(0, 1), (0, 4), (1, 1), (2, 1)}
"it":     {(0, 0), (0, 3), (1, 2), (2, 0)}
"what":   {(0, 2), (1, 0)}
```

If we run a *phrase search* for `"what is it"` we get hits for all the words in both document 0 and 1. But the terms occur only consecutively in document 1.

References

- Donald Knuth. *The Art of Computer Programming*, Volume 3: *Sorting and Searching*, Third Edition. Addison-Wesley, 1997. ISBN 0-201-89685-0. Pages 560–563 of section 6.5: Retrieval on Secondary Keys.
- Justin Zobel, Alistair Moffat and Kotagiri Ramamohanarao, *Inverted files versus signature files for text indexing*. ACM Transactions on Database Systems (TODS), Volume 23, Issue 4 (December 1998), Pages: 453 - 490.

See also

Vector space model

External links

- NIST's Dictionary of Algorithms and Data Structures: inverted index (http://www.nist.gov/dads/HTML/invertedIndex.html)
- Inverted index for search engine (http://www.rankcount.com/reverse-keyword-search.php)

Retrieved from "http://en.wikipedia.org/wiki/Inverted_index"

Categories: Data structures | Algorithms on strings

About CNI

Task Force Meetings

Conferences

Projects

CNI Collaborations

Site Map

Proceedings: Technological Strategies for Protecting Intellectual Property in the Networked Multimedia Environment

Coalition for Networked Information

Interactive Multimedia Association

John F. Kennedy School of Government

Science, Technology & Public Policy Program

Massachusetts Institute of Technology

Program on Digital Open High-Resolution Systems

Rather than put a trademark symbol in every occurrence of other trademarked names, we state that we are using the names only in an editorial fashion, and to the benefit of the trademark owner, with no intention of infringement of the trademark.

Published by:

Interactive Multimedia Association
Intellectual Property Project
3 Church Circle
Suite 800
Annapolis, MD 21401-1933
Phone: (410) 626-1380
FAX: (410) 263-0590

Table of Contents

Consortium for University Printing and Information Distribution (CUPID)

A Publishing and Royalty Model for Networked Documents

Theodor Holm Nelson

Acronyms List

Last updated Monday, July 2, 2001.

tiscali

From: www.tiscali.co.uk/reference/

Index

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

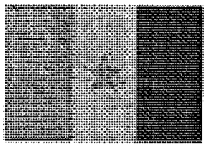
Interactive Multimedia Association
Organization founded in 1987 to promote the growth of the multimedia industry. Based in Anapolis, Maryland, USA, the IMA runs special interest groups, summit meetings, conferences, and trade shows for its member companies.

Dictionary Search

Search for:

Go

Senegal Flag



The star represents Islam and expresses peace, harmony, hope, and socialism. The tricolour is reminiscent of the flag of France, the former colonial power. The pan-African colours express unity with other African nations. Effective date: 25 August 1960. >>

Sponsored by: ARL

Coalition for Networked Information

About CNI

Task Force Meetings

Conferences

Presentations/ Publications

Projects

CNI Collaborations

Site Map

CNI Membership and Task Force Information

Frequently Asked Questions about CNI Membership

List of CNI Members

Becoming a Member
The Coalition for Networked information (CNI) is an institutional membership organization. Membership dues are the primary financial resource for CNI's programs. Dues for the year (July 1, 2006-June 30, 2007) are $6,200, and membership is renewable on a year-by-year basis.

CNI's members include:

- Higher Education Institutions
- Publishers
- Network & Telecommunications Companies
- Scholarly Organizations
- Professional Organizations
- Libraries
- Information Technology Companies
- Government Agencies

To request more information or a membership application, write to Jackie Eudell at <jackie@cni.org>.

The Task Force
Each of CNI's member institutions appoints two representatives to the CNI Task Force, which holds semi-annual meetings. Higher education institutions are encouraged to appoint their heads of libraries and information technology. Other types of institutions generally appoint top administrators or directors of electronic publishing. All members of the Task Force have equal status; for example, there are no separate membership categories for corporate members. A Steering Committee guides the Coalition's activities.

Last updated Tuesday, May 2, 2006.

Matt Blaze's Technical Papers

Last updated 6 August 2006

Many of my technical papers are available here. Newer papers are usually in Adobe PDF format; like it or not, PDF is the de facto standard format for scientific papers these days. Most of the older papers are in PostScript format; you'll need a PostScript printer or viewer (such as GhostView) to read them. Most of these files have also been converted to Adobe PDF format (using ps2pdf) and can be viewed or printed with a PDF viewer such as Acrobat, acroread4, or xpdf. If you have a choice, you'll probably find the PostScript version looks and works better than the PDF version does (ps2pdf doesn't do particularly well with some of the fonts). A few papers are available as plain ASCII text or LaTeX source.

Wiretapping, Surveillance and Countermeasures

The Trustworthy Network Eavesdropping and Countermeasures (TNEC) project studies the reliability of communications interception systems and technologies. A better understanding of the limitations of eavesdropping techniques could lead to more trustworthy law enforcement wiretap evidence (or at least more appropriate treatment of electronic evidence), networks with properties that inherently frustrate (or facilitate) interception, and new techniques for achieving communications security.

One of our first efforts is a comprehensive analysis of the wiretapping technologies used by law enforcement (for both voice and data). We have found serious exploitable weaknesses in fielded interception systems. For details, including audio demos of novel eavesdropping countermeasures, see the wiretapping web page here.

- M. Sherr. E. Cronin, S. Clark and M. Blaze. http://www.crypto.com/papers/wiretapping/ web page.
- M. Sherr, E. Cronin, S. Clark and M. Blaze. "Signaling Vulnerabilities in Wiretapping Systems." *IEEE Security and Privacy.* November/December 2005. [PDF].

Similar vulnerabilities exist in digital Internet eavedropping systems as well:

- E. Cronin, M. Sherr, and M. Blaze. "The Eavedsdropper's Dilemma." Technical Report MS-CIS-05-24. University of Pennsylvania. 2005. [PDF].

Another focus of the TNEC project examines local host-based surveillance. The *JitterBug* demonstrates a novel eavesdropping threat against typed keyboard input. Commercially-available hardware keyboard "sniffers" can easily capture and store an unsuspecting user's keystrokes. Because a subverted keyboard has no direct network connection, sniffer attacks are generally assumed to require either support software on the host or periodic in-person access by the attacker to retrieve the data. We show that this need not be the case. A new technique based on "JitterBugs" can exfiltrate captured data entirely through subtle perturbations in the precise times at which typed keystrokes are passed to the host. Whenever a user runs an interactive network application (such as SSH), an attacker can derive previously captured keystrokes entirely by observing the timing of network packets, even from across the

Internet or via encrypted wireless traffic. The JitterBug demonstrates that input devices must be scrutinized as part of any trusted computing base and, more generally, that simple "supply chain attacks" can represent a practical and serious threat to data confidentiality. (Gaurav Shah and Andres Molina won the Best Student Paper award at USENIX Security 2006 for this work.)

- G. Shah, A. Molina, and M. Blaze. "Keyboards and Covert Channels." *Proc. 15th USENIX Security Symposium.* Vancouver, BC. August 2006. [PDF].

- Gaurav Shah's JitterBug page: http://www.cis.upenn.edu/~gauravsh/jitterbug.html. JitterBug prototype code and PCB template.

Physical and "Human-Scale" Security

Cryptologic techniques can be applied outside of computers and networks, Perhaps surprisingly, the abstractions used in analyzing secure computing and communications systems turn out also to be useful for understanding mechnical locks and their keyspaces. Indeed, modeling master keyed locks as online authentication oracles leads directly to efficient solutions for what might naively seem like exponential problems for the attacker. In fact, it seems like almost a textbook example, as if master keying practices for locks were designed specifically to illustrate this class of weakness. We sometimes assume that hardware-based security is inherently superior to that based in software, but even the humble mechanical lock can be just as insecure as complex computing systems, and can fail in similar ways.

A widely circulated paper of mine describes attacks against master keyed mechanical locks. For an overview of the attack, which was described in the January 23rd 2003 *New York Times*, click here. For a brief commentary on the reaction to this paper, see my essay, "Keep it secret, stupid!" (click here), which was originally posted to comp.risks.

(Warning: there are embedded photos in this paper; they make the PS and PDF files very large. The GZIPed PostScript version is 5.7MB long (uncompresses to 14MB), and the PDF version is 4MB long.)

- M. Blaze. "Cryptology and Physical Security: Rights Amplification in Master-Keyed Mechanical Locks." March 2003. *IEEE Security and Privacy.* March/April 2003. [GZIPed PostScript], [PDF].

- My *Notes on Picking Pin Tumbler Locks*, intended primarily for use by students in my security seminar, can be found here [HTML].

While the security metrics and mechanical safeguards used in safes and vaults may not relay on the latest technology, they are often quite ingenious. They may have much to teach computer security. Some of what I understand about the subject is in the survey paper below (warning -- heavily illustrated 2.5MB .pdf file). And for a brief commentary on the reaction to *this* paper, see my essay, "the second sincerest form of flattery" (click here), which was originally posted to `interesting-people`.

- M. Blaze. "Safecracking for the Computer Scientist." *U. Penn CIS Department Technical Report.* 7 December 2004 (revised 20 December 2004). [PDF].

This position paper, presented at the Cambridge Security Protocols Workshop 2004, introduces and advocates the "Human Scale Security Project," which supports the above work.

- M. Blaze. "Toward a broader view of security protocols." *12th Cambridge International Workshop on Security Protocols*. Cambridge, UK. April 2004. [PDF].

Trust Management

These papers introduce the "trust management" approach to specifying and enforcing security policy.

- The Trust Management Web Page, updated regularly.
- M. Blaze, J. Ioannidis, A. Keromytis. "Offline Micropayments without Trusted Hardware." *Financial Cryptography 2001*. Grand Cayman, February 2001. [PostScript], [PDF].
- M. Blaze, J. Ioannidis, A. Keromytis. "Trust Management for IPSEC." *NDSS 2001*. San Diego, February 2001. [PDF].
- M. Blaze, J. Feigenbaum, J. Ioannidis, A. Keromytis. The KeyNote Trust Management System, Version 2. *RFC-2704*. IETF, September 1999. [ASCII Text].
- M. Blaze, J. Ioannidis, A. Keromytis. "Compliance Checking and IPSEC Policy Management." *Internet Draft*. `draft-blaze-ipsp-trustmgt-00.txt`. IETF, March 2000. [ASCII Text].
- M. Blaze, J. Ioannidis, A. Keromytis. "DSA and RSA Key and Signature Encoding for the KeyNote Trust Management System." *RFC-2792*. IETF, March 2000. [ASCII Text]
- M. Blaze, J. Ioannidis, and A. Keromytis. " Trust Management and Network-Layer Security Protocols." *1999 Cambridge Protocols Workshop*. Cambridge, April 1999. [PostScript], [PDF], [LaTeX Source].
- M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. "The Role of Trust Management in Distributed Systems Security." Chapter in *Secure Internet Programming: Security Issues for Mobile and Distributed Objects*, (Vitek and Jensen, eds.) Springer-Verlag, 1999. [PostScript], [PDF].
- M. Blaze, J. Feigenbaum, M. Strauss. "Compliance-Checking in the PolicyMaker Trust-Management System." *Proc. 2nd Conference on Financial Cryptography*. Anguilla 1998. LNCS 1465, pp 251-265, Springer-Verlag, 1998. [PostScript], [PDF].
- M. Blaze, J. Feigenbaum and J. Lacy. "Decentralized Trust Management." *IEEE Symposium on Security and Privacy*, Oakland, CA. May 1996. [PostScript], [PDF].

Angelos Keromytis's KeyNote Trust Management toolkit and open-source reference

implementation is available here as a GZIPed TAR archive. The toolkit runs under most Unix-like (BSD, linux, etc.) platforms, with limited support for Win32 platforms.

Also see Angelos Keromytis' KeyNote web page for the latest details on the KeyNote implementation.

Remotely-Keyed Encryption

These papers introduce and formalize the notion of "remotely-keyed" encryption, in which a low-bandwidth, but trusted device (such as a smart card) assists a high-bandwidth, but untrusted host with bulk encryption.

- M. Blaze, J. Feigenbaum, and M. Naor. "A Formal Treatment of Remotely Keyed Encryption (Extended Abstract)". Eurocrypt '98, Helsinki. LNCS 1403 pp. 251-265. [PostScript], [PDF].
- M. Blaze. "High-Bandwidth Encryption with Low-Bandwidth Smartcards." January 18, 1996. *Cambridge Workshop on Fast Software Encryption*, February 1996. [PostScript], [PDF].

Key Escrow

These papers describe and evaluate various key escrow proposals, from a technical (as opposed to political) perspective.

- *The Risks of Key Recovery, Key Escrow, and Trusted Third-Party Encryption* (second edition). June 1998. [HTML], [PDF].
- *The Risks of Key Recovery, Key Escrow, and Trusted Third-Party Encryption* (first edition). May 1997. (OBSOLETE: superseded by second edition, above). [ASCII Text], [PDF], [PostScript].
- M. Blaze. "Oblivious Key Escrow." *First Cambridge Workshop on Information Hiding* May 1996. Springer 1997. [PostScript], [PDF], [LaTeX source].
- Memo from NSA regarding key length report, with comments from M. Blaze and W. Diffie. July 18, 1996. [ASCII Text].
- M. Blaze, W. Diffie, R. Rivest, B. Schneier, T. Shimomura, E. Thompson and M. Wiener. "Minimal Key Lengths for Symmetric Ciphers to Provide Adequate Commercial Security". Report of ad hoc panel of cryptographers and computer scientists. January 1996. [ASCII Text], [PDF], [PostScript].
- M. Blaze, J. Feigenbaum and F.T. Leighton. "Master-Key Cryptosytems." Abstract presented at *Crypto '95 (rump session)*, Santa Barbara, CA, August 1995. [PostScript], [PDF]
- M. Blaze. "Protocol Failure in the Escrowed Encryption Standard." *Proceedings of Second ACM Conference on Computer and Communications Security*, Fairfax, VA, November 1994. [PostScript], [PDF].

Network-Layer Security

These papers describe the design and implementation network-layer and related security protocols, including JFK, a secure key exchange protocol, and swIPe, a predecessor to the IPSEC standard. (At this point, swIPe is of primarily historical interest, although the USENIX paper should be of some value to IPSEC implementors. JFK is a useful key exchange protocol that should be especially valuable for IPSEC and network security key management).

- W. Aiello, S. M. Bellovin, M. Blaze, R. Canetti, J. Ioannidis, A. D. Keromytis, and O. Reingold. "Efficient, DoS-Resistant, Secure Key Exchange for Internet Protocols." In Proc. ACM Computer and Communications Security (CCS) Conference. November 2002, Washington, DC. (pp 48-58). [PDF].

- J. Ioannidis and M. Blaze. "The swIPe IP Security Protocol." *Internet Draft*. December 1993. [ASCII Text].

- J. Ioannidis and M. Blaze. "Architecture and Implementation of Network Layer Security Under UNIX." *Proceedings of the Fourth USENIX Security Workshop*, October 1993. [PostScript], [PDF].

Cryptographic Applications

- R. Levein, L. McCarthy, M. Blaze. "Transparent Internet E-mail Security (DRAFT)". August 9, 1996. [PostScript], [PDF].

- M. Blaze and S.M. Bellovin. "Session-Layer Encryption." *Proceedings of the USENIX Security Workshop*, June 1995. [PostScript].

- M. Blaze. "Key Management in an Encrypting File System." *USENIX Summer 1994 Technical Conference*, Boston, MA, June 1994. [PostScript], [PDF].

- M. Blaze. "A Cryptographic File System for Unix." *Proceedings of the First ACM Conference on Computer and Communications Security*, Fairfax, VA, November 1993. [PostScript], [PDF].

  The latest CFS code can be found here.

Ciphers and Algorithms

- S. M. Bellovin, M. Blaze. "Cryptographic Modes of Operation for the Internet." NIST Workshop on AES Modes. Santa Barbara, CA. August 2001. [PDF].

- M. Blaze, M. Strauss. "Atomic Proxy Cryptography." Full version of our *EuroCrypt '98* paper. May 1997. [PostScript], [PDF].

- M. Blaze. "Efficient Symmetric-Key Ciphers Based on an NP-Complete Subproblem (DRAFT)". October 2, 1996. [PostScript], [PDF]

- M. Blaze and B. Schneier. "The MacGuffin Block Cipher Algorithm." *Leuven Workshop on Cryptographic Algorithms*, Leuven, Belgium, December 1994.

[PostScript], [PDF].

Cryptography Policy, Export Regulations, and Politics

- M. Blaze. Declaration in Felten, et al v. RIAA. 13 August 2001. [ASCII Text].
- S. Bellovin, M. Blaze, D. Farber, P. Neumann, E. Spafford. "Comments on the Carnivore System Technical Review." Formal comments to the US Department of Justice. 3 December 2000. [HTML].
- M. Blaze & S. M. Bellovin. "Tapping, Tapping on my Network Door." INSIDE RISKS 124. *CACM*, October 2000. [HTML].
- M. Blaze. "Cryptography Policy and the Information Economy." Draft. 17 December 1996. [PostScript], [PDF], [ASCII Text].
- My prepared testimony before the Senate Commerce Committee subcommittee on Science, Technology, and Space. June 26, 1996 [ASCII Text].
- M. Blaze. "My Life as an International Arms Courier." January, 1995. Adapated from posting to *comp.risks* [ASCII Text]

Peer-to-Peer Networking

My dissertation work, over ten years ago, anticipated and analyzed what we would now call "Peer-to-Peer" file distribution.

- M. Blaze. Caching in Large-Scale Distributed File Systems. PhD thesis. Princeton University Department of Computer Science. November 1992. [PostScript].

Other People's Papers

From time to time, I make available papers from other researchers that I didn't write myself but that are of wide interest and don't otherwise have a home. Here's what's available now:

- S. Fluhrer, I. Mantin and A. Shamir. Weaknesses in the Key Scheduling Algorithm of RC4. Preliminary Draft, July 25, 2001. [PostScript].
- A. Biryukov and A. Shamir. Real Time Cryptanalysis of the Alleged A5/1 on a PC. Preliminary Draft, December 9, 1999. [PostScript].

Click here to return to the crypto.com home page.

ElGamal encryption

From Wikipedia, the free encyclopedia

The **ElGamal algorithm** is an asymmetric key encryption algorithm for public key cryptography which is based on Diffie-Hellman key agreement. It was described by Taher Elgamal in 1984. The ElGamal algorithm is used in the free GNU Privacy Guard software, recent versions of PGP, and other cryptosystems. The Digital Signature Algorithm is a variant of the ElGamal signature scheme, which should not be confused with the ElGamal algorithm.

ElGamal can be defined over any cyclic group G. Its security depends upon the difficulty of a certain problem in G related to computing discrete logarithms (see below).

Contents

The algorithm

ElGamal consists of three components: the key generator, the encryption algorithm, and the decryption algorithm.

The key generator works as follows:

- Alice generates an efficient description of a cyclic group G of order q with generator g. See below for specific examples of how this can be done.
- Alice chooses a random x from $\{0, \ldots, q-1\}$.
- Alice computes $h = g^x$.
- Alice publishes h, along with the description of G,q,g, as her public key. Alice retains x as her secret key.

The encryption algorithm works as follows: to encrypt a message m to Alice under her public key (G,q,g,h),

- Bob converts m into an element of G.
- Bob chooses a random y from $\{0, \ldots, q-1\}$, then calculates $c_1 = g^y$ and $c_2 = m \cdot h^y$.
- Bob sends the ciphertext (c_1,c_2) to Alice.

The decryption algorithm works as follows: to decrypt a ciphertext (c_1,c_2) with her secret key x,

- Alice computes $\frac{c_2}{c_1^x}$ as the plaintext message.

The decryption algorithm produces the intended message, since

$$\frac{c_2}{c_1^x} = \frac{m \cdot h^y}{g^{xy}} = \frac{m \cdot g^{xy}}{g^{xy}} = m$$

If the space of possible messages is larger than the size of G, then the message can be split into several pieces and each piece can be encrypted independently. Typically, however, a short key to a symmetric-key cipher is first encrypted under ElGamal, and the (much longer) intended message is encrypted more efficiently using the symmetric-key cipher — this is termed *hybrid encryption*.

Security

ElGamal is a simple example of a semantically secure asymmetric key encryption algorithm (under reasonable assumptions). It is probabilistic, meaning that a single plaintext can be encrypted to many possible ciphertexts, with the consequence that a general ElGamal encryption produces a 2:1 expansion in size from plaintext to ciphertext.

ElGamal's security rests, in part, on the difficulty of solving the discrete logarithm problem in G. Specifically, if the discrete logarithm problem could be solved efficiently, then ElGamal would be broken. However, the security of ElGamal actually relies on the so-called Decisional Diffie-Hellman (DDH) assumption. This assumption is often stronger than the discrete log assumption, but is still believed to be true for many classes of groups.

Generating the group G

As described above, ElGamal can be defined over any cyclic group G, and is secure if a certain computational assumption (the "DDH Assumption") about that group is true. Unfortunately, the straightforward use of $G = Z_p$ for a prime p is **insecure**, because the DDH Assumption is false in this group. In contrast, computing discrete logs is believed to be hard in Z_p, but this is not enough for the security of ElGamal.

The two most popular types of groups used in ElGamal are *subgroups* of Z_p and groups defined over certain elliptic curves. Here is one popular way of choosing an appropriate subgroup of Z_p which is believed to be secure:

- Choose a random large prime p such that $p - 1 = kq$ for some small integer k and large prime q. This can be done, for example with $k = 2$, by first choosing a random large prime q and checking if $p = 2q + 1$ is prime.
- Choose a random element $g \in Z_p$ such that $g \neq 1$ and $g^q = 1 \bmod p$, i.e. such that g is of order q.
- The group G is the subgroup of Z_p generated by g, i.e. the set of kth residues mod p.

When encrypting, care must be taken to properly encode the message m as an element of G, and not, say, as just an arbitrary element of Z_p.

Efficiency

Encryption under ElGamal requires two exponentiations; however, these exponentiations are independent of the message and can be computed ahead of time if need be. The ciphertext is twice as long as the plaintext, which is a disadvantage as compared to some other algorithms. Decryption only requires one exponentiation (instead of division, exponentiate c_1 to $q - x$). Unlike in the RSA and Rabin systems, ElGamal decryption *cannot* be sped up via the Chinese remainder theorem.

Miscellaneous

ElGamal is malleable in an extreme way: for example, given an encryption (c_1,c_2) of some (possibly unknown) message m, one can easily construct an encryption $(c_1, 2 \cdot c_2)$ of the message $2m$. Therefore ElGamal is not secure under chosen ciphertext attack. On the other hand, the Cramer-Shoup system (which is based on ElGamal) *is* secure under chosen ciphertext attack.

See also

- ElGamal Signature scheme

References

- Taher ElGamal, "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Transactions on Information Theory, v. IT-31, n. 4, 1985, pp469–472 *or* CRYPTO 84, pp10–18, Springer-Verlag.
- Handbook of Applied Cryptography (http://www.cacr.math.uwaterloo.ca/hac/), contains a detailed description of ElGamal Algorithm in Chapter 8 (http://www.cacr.math.uwaterloo.ca/hac/about/chap8.pdf) (PDF file).

Public-key cryptography

Algorithms: Cramer-Shoup | DH | DSA | ECDH | ECDSA | EKE | **ElGamal** | GMR | IES | Lamport | MQV | NTRUEncrypt | NTRUSign | Paillier | Rabin | Rabin-Williams | RSA | Schnorr | SPEKE | SRP | XTR

Theory: Discrete logarithm | Elliptic curve cryptography | RSA problem

Standardization: ANS X9F1 | CRYPTREC | IEEE P1363 | NESSIE | NSA Suite B **Misc:** Digital signature | Fingerprint | PKI | Web of trust | Key size

Cryptography

History of cryptography | Cryptanalysis | Cryptography portal | Topics in cryptography

Symmetric-key algorithm | Block cipher | Stream cipher | Public-key cryptography | Cryptographic hash function | Message authentication code | Random numbers

Category: Asymmetric-key cryptosystems

Atomic Proxy Cryptography

Matt Blaze

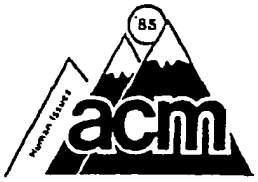
AT&T Shannon Laboratory

This talk introduces *atomic proxy cryptography*, in which an atomic proxy function, in conjunction with a public proxy key, converts ciphertext (messages in a public key encryption scheme or signatures in a digital signature scheme) for one key (k_1) into ciphertext for another (k_2). Proxy keys, once generated, may be made public and proxy functions applied in untrusted environments. Various kinds of proxy functions might exist; {\em symmetric} atomic proxy functions assume that the holder of k_2 unconditionally trusts the holder of k_1, while {\em asymmetric} proxy functions do not. It is not clear whether proxy functions exist for previous public-key cryptosystems. Several new public-key cryptosystems with symmetric proxy functions are described: an encryption scheme, which is at least as secure as Diffie-Hellman, an identification scheme, which is at least as secure as the discrete log, and a signature scheme derived from the identification scheme via a hash function.

Full paper available.

This is joint work with Martin Strauss.

O12 Evidence with regard to computer network architecture, satellite communications and digital television distribution



EXTENDED ABSTRACT

A Secure Distributed Capability Based System

Howard L. Johnson
Information Intelligence Sciences, Inc.
University of Denver, New College

John F. Koegel, Rhonda M. Koegel
Department of Mathematics and Computer Science
University of Denver
Denver, Colorado 80210

A novel design for a secure distributed system is described and evaluated. A capability based computer architecture is combined with cryptographic network security techniques to protect global objects and preserve access rights across system boundaries. The resulting architecture is evaluated against several criteria, including the DoD Computer System Evaluation Criteria. The strengths and weaknesses of the approach are presented.

key words: computer security; distributed system security; capability architecture; network encryption

1. Introduction

A distributed system connects various computing entities in several locations so resources can be shared by users. Distributed computing offers the advantage of flexibility so that each facility can be locally controlled and configured for a specific application. It also offers incremental growth so that additional features can be easily added, usually at a lower cost than upgrading a central host. The connection of distributed systems facilitates information sharing. The physical network can be implemented by point-to-point or multi-point links, LAN's or WAN's.

In a single centralized computing facility, system security is achieved through physical, operational, and system controls. System controls include operating system functions such as login passwords, file system protection, and memory management. In a distributed environment, these controls can still be effective for securing each specific system. However, additional problems arise because of the interconnection of systems and the information flows between systems.

There are two areas of concern in securing a distributed system. The first, that of securing the network facilities, has received greater attention in the literature. This need stems from the fact that physical facilities in most prevalent use today as communication media (land lines, microwave links, and satellite channels) offer little protection for themselves [1]. To secure these facilities, some type of cryptography is employed. The user who wishes to obtain an off-the-shelf solution to the problem can use a conventional substitution-permutation algorithm, such as the NBS's DES [2] or a public key algorithm such as RSA [3]. Although there is active research in both breaking and strengthening these techniques, for many applications currently available methods will suffice.

Even with encryption, a network is still vulnerable to certain types of threats against the communications protocol being employed [4]. Conventional link-level protocols only allow the data field to be encrypted, while control and address fields are transmitted unencrypted. This leaves a network open to such attacks as message modification and message replay.

The second area of concern, that has received relatively little attention in the literature, is the control of information protection across system boundaries. Within a given computer facility, the operating system can be used to enforce uniform and constant protection of information. However, once the information is removed from the computer, these controls no longer apply. Protection of information can only be maintained in a local environment. It would be preferable if access rights could be enforced across system boundaries. This would produce a secure distributed system and protect proprietary software and data.

Consider the case of a remote database user who has purchased read access to certain information in the database. If the user accesses the

1985 ACM 0-89791-170-9/85/1000-0392 $00.75

database with a personal computer, it is a straightforward step for the user to read the database and store the information in the PC. Once the user has a local copy, then he/she is free to distribute this data to any other party, regardless of whether that party has purchased access to the database. Thus, the access protection of a single system is easily violated by availability of distributed computing.

The database owner could protect his/her investment by requiring the user to purchase a proprietary interface program to access and manipulate the data. Not only does this restrict the user and provide an economic deterrent to the sale of information, it also makes this protection dependent on the copy protection of the interface program.

Another example is if the host is used as a central distribution point for software, possibly for a CAI application. Once a module is removed from the host, it is very difficult to limit the production of duplicates. Encryption of the key elements of a program has been proposed as a solution [5]. However, not only does this place additional burden on the applications programmer, but also requires a design that may not be met by many programs.

Most secure network strategies deal only with encryption of data as it is transmitted across network facilities, and not at all with the management of protection across system boundaries. However, there are numerous instances of distributed information system security and proprietary software protection not solved by network encryption. The authors believe that an integrated solution involving both capability-based computers and network protection using encryption and a secure protocol can provide distributed system security.

After discussion of network security and capability architectures in a distributed environment, we present an integrated design of a secure distributed capability based system. The resulting architecture is evaluated against several criteria, including the DoD Computer System Evaluation Criteria.

2. Network Security

2.1 General

This paper pertains to hardware (and a few hardware/software) data system security protection mechanisms, but design must be accomplished in the context of existing or proposed physical security, personnel security, operations security, emanations security, and communications security. The implementation of each guides implementation of the network data system security. A key criteria is minimized degradation in throughput and response.

A brief summary of network security follows. The term "association" is used to refer to a (potentially bidirectional) end-to-end data path through the network. The reader is directed to Voydock and Kent [4] and to Davies and Price [6] for a more complete treatment of these topics.

2.2 Threat

From [4], passive attacks to network security are intended to bring about the unauthorized release of information or authorized release of information sufficient to perform a traffic analysis. Passive attacks usually cannot be detected but can be prevented.

Active attacks include unauthorized modification of information, unauthorized resource use denial, and attempts to initiate spurious associations. Active attacks cannot be prevented, but can usually be detected. In a network environment we are equally concerned with threats internal to the system as those outside.

2.3 Protection Principles

2.3.1 Encryption Techniques

Rushby and Randell [7] observed that separation is one of the key elements in enforcing a secure system, and that four separation methods exist: physical, temporal, cryptographical and logical. In a communication system, physical separation is the most desirable, but unless the system is completely contained in a secure building environment or in a specially constructed tunnel vault, the distances involved leave too much line unprotected.

Some transmission media are more secure than others, such as fiber optics, directional satellite links, and exotic military communications systems, but each has a reasonable vulnerability to capture or disruption of data flow. Within a secure environment, either logical or cryptographical means can be employed to protect data authenticity. Methods analogous to periods processing can make use of transmission links for different levels of control at different isolated periods of protection, performing necessary cleansing of storage registers or buffers, if any exist.

Data encryption is the primary means by which communicated data are protected. It directly prevents passive attacks by preventing an intruder from seeing data in the clear. Data patterns can be masked by using a unique key for each association, employing cipher block chaining which causes each encrypted value to be a complex function of previously encrypted data, and appropriately selecting the proper initialization vector for chaining.

There are three ways to incorporate cryptography into a communications system: link, node and end-to-end encryption. In link encryption, cryptographic devices bracket a communication line between two nodes. Node encryption uses a protected security module to absolutely protect data at the node. In end-to-end encryption, data are deciphered only at their final destination, requiring several keys at each origin and destination.

There are several tradeoff variables in choosing between link, node, and end-to-end encryption:

- the number of encryption units required (and therefore the potential response degradation)
- the number of keys required by each node originating and receiving data
- the complexity required in specifying a routing path independent of the specification of data, or alternately the overhead in interim decryption attempts.

The number of security devices are fewer in end-to-end encryption, but number of keys required is greater. Addressing information must be developed independent of the data, or interim decryption attempts must be made. Both create a difficult design problem. Link and node encryption are normally transparent to the user, but so is end-to-end encryption if initiated by system services. The message and its header can both be encrypted with node encryption; however, with link encryption and end-to-end encryption normally both message and header are encrypted. The exception is a technique whereby each node attempts to decrypt the message and passes it if unsuccessful or if the successfully decrypted message indicates another addressee. If not all nodes have encryption facilities or if encryption of only selected messages is desired due to overhead, an additional mechanism is required to enable and disable the encryption function.

Voydock and Kent [4] observed that a communication network can also be viewed as providing a medium for establishing associations between protocol entities. An association oriented approach constitutes a refinement to end-to-end measures. It not only protects the path, but reduces the probability of undetected cross talk, whether induced by hardware or software.

2.3.2 Detection Techniques

If the communications header is in clear form, transmitting bogus messages helps prevent traffic analysis. The protocol layer selection determines the precision with which traffic analysis can be done. If encryption is performed in the presentation layer, an intruder could determine which presentation, session, and transport entities were involved. Performing encryption in the transport, network, or link layers limits the intruder to observing patterns at the network address levels. Contradistinctively, the higher the layer, the more of the path protected.

To prevent message stream modification, there are measures that ensure message integrity. Measures that ensure message authenticity rely on the integrity measures. Measures that ensure message ordering rely on both of the previous measures. Countermeasures involve use of unique keys, sequence numbers, and error detection codes.

Denial of service attacks often can be detected by message stream modification countermeasures. If the attacks begin when an association is quiescent, a request response mechanism must be employed.

For spurious association attacks, hierarchic or public key systems can defeat attempts to establish an association under a false identity. Timestamp, checksums, and/or random challenge-response mechanisms detect playing back of a previously legitimate association-initiation.

A covert channel allows a process to transfer information in a manner that violates the systems security policy. A covert timing channel is a covert channel in which one process signals information to another by modulating its own use of system resource (e.g., CPU time) in such a way that this manipulation affects the real response time observed by the second process. Covert channels with low bandwidths represent a lower threat than those with high bandwidths. In any complex system there are a number of relatively low-bandwidth covert channels whose existence is deeply ingrained in the system design. Faced with the large potential cost of reducing the bandwidths of such covert channels, it is felt that those with a maximum bandwidth of less than one bit per second are acceptable in most applications environments [8]. The channel bandwidth can be reduced by introducing noise, or complicated traffic patterns, making it difficult to detect and extract deliberate modulation.

These measures provide security only in a probabilistic sense, providing a high probability that the intruder cannot subvert the encryption algorithm and that active attacks will be detected. The goal is to make it more difficult for the intruder to break the system than to create the information through other means.

2.4 Protection Mechanisms

2.4.1 Reference Monitors

A reference monitor [9] must be tamperproof, must always be invoked, and must be small enough to be subject to analysis and tests, the completeness of which can be assured. The reference monitor is the most popular type of authentication mechanism. Interaction is generally only with the message header, whereas cryptographic compatibility serves to authenticate an entire message. Further, data can remain encrypted for continued protection while in buffers, storage, and internal communications. The reference monitor allows such things as separate encryption of the message without the header and requires neither the time and cost spent in encryption nor the cost of a key management and distribution system.

2.4.2 Authentication and Secrecy

Cryptography can not only be used for security, but can also be employed for authenticity. Solutions using encryption are equally applicable to local area networks as they are to large long-haul communications networks. Different applications lead to different solutions, as do design tradeoffs based on changing technologies (e.g., fiber optics), speed, cost, and level of protec-

tion. The following are key topics associated with cryptography.

Secrecy and Authentication - Secrecy exists when it is computationally infeasible to determine the deciphering transformation. Authenticity exists when it is computationally infeasible to determine the enciphering transformation. The latter establishes the validity of a claimed identity (e.g., of the sender in a digital signature or user verification application).

Substitution-Permutation Ciphers (e.g., the DES) - Information theory has allowed theoretical data protection to any degree desired, based on the length of the key and repeated application of the algorithm steps; even when the algorithm is known to the perpetrator. This class of cipher has been implemented into a very fast chip. As cryptanalysis capability increases, the dimensionality of the implementation can be increased, with a corresponding loss in efficiency, (unless microcircuit technology makes up the difference). A DES block cipher breaks the message into blocks and enciphers each with the same key. A stream cipher breaks the message into characters or bits and enciphers them with successive elements of a key stream (which might be the prior encrypted text as in the cipher block chaining mode and the cipher feedback mode of the DES).

Public Key Ciphers (e.g., the RSA scheme) - These methods of protection provide both secrecy and authenticity. Several public key ciphers have fallen prey to cryptanalysts, but the RSA cipher stands a good chance of surviving these attacks based on the mathematical history of factorization of large numbers (although a surprisingly large number was factored on the Cray at Sandia Laboratories recently). Keys are large and computation still relatively complex. Technologies such as gallium arsenide and parallel bit stream implementations should solve immediate speed problems, however, as cryptanalysis comes closer, the size of the prime numbers must be increased.

One-way Ciphers - These virtually unknown, but simply implemented ciphers are important to design because once data are encrypted they cannot be simply decrypted, even by the originator. They are useful in applications, for example, where authentication of passwords can be accomplished by comparing pairs of encrypted data values. Certain simple functions such as comparison can be accomplished in encryption space.

2.4.3 Key Management Design

The responsibility for key management depends on the security policy and the choice of implementation. Unless keys are given at least the same level of protection as the data, they will be the weak link. Once the penetrator has gained access to the key (generally a very small piece of data) he has gained access to all data. Techniques of generating, transmitting and protecting keys include host keys, hierarchical key protection, partitioning of keys for different protection levels, and diverse means by which the key management system interfaces with the rest of the system [7].

In the normal implementation of public key systems, the public key is published with no protection whatsoever. The private key is originated and held by only one person. Certain implementations require distribution of the private key under a protected key distribution scheme, especially where the private key is used within the processor as a means of both secrecy and authentication of source or another system variable.

A third party or a host can provide the authentication necessary for key distribution. There are several established approaches for the implementation of a distributed session key system, appropriate to network communications. The public key system has the property that two parties can establish a secret key for use in a unique session between them, obviating involvement of a third party. The strategy can be repeated often for a greater degree of protection. Prolonged use of a single key makes a system more vulnerable to cryptanalysis. The degree of added vulnerability depends on the cryptographic technique used, which in turn is related to the nature of data transmission, intercommunication requirements, and security inherent to the communications system.

2.5 Network Protocol Considerations

In late 1970's, the International Standards Organization adopted a network architecture known as the reference model for open system interconnection, ISO/OSI. Layers 1 to 3 are concerned with data transmission/routing and deal respectively with physical, data link, and network concerns. Layer 4 provides end-to-end control of data transport. Layers 5 and 7 are the session, presentation, and application layers. Some of the possible approaches to implementing security under ISO/OSI are as follows:

Layer	Protocol	Security
7	Applications Services	User identification, encryption of stored data, key distribution.
6	Presentation Formats	User controlled use of encryption for secrecy and identification including a user request for encryption.
5	User Session Control	Establishing secrecy and authentication during the conduct of a session between system users (people and programs). The most desirable encryption point in high level protocols [4].

4	Transport Flow Control	
3	Network Routing	
2	Data Link Control	Security control entirely in the communications systems such as link encryption where the data are protected between adjacent network nodes and are decrypted and re-encrypted at each node. Security control entirely in the network communications use of node encryption schemes where data are not in the clear at an intermediate node, but are rather decrypted and re-encrypted by a special security module.
1	Physical Connection	

Design should be such that acceptance of data or requests into the memory associated with a node should be based on the assurance that the transaction is legitimate and does not violate the security policy. An example of protocol layer 2 (data link) encryption is provided in [10], in which source and destination subnets and trusted interface units are designated in the packet formats for the carrier sense multiple access with collision detection (CSMA/CD) protocol. The protocol also specifies the data security level.

Popek and Kline [11] identified the important issues to be addressed in defining secure protocols:

- establishing initial cleartext/ciphertext/cleartext channel from sender to receiver
- passing cleartext addresses without providing a leakage path
- determining error recovery and resynchronization mechanisms to be employed
- performing flow control
- closing channels
- interaction of the encryption protocols with the rest of the protocols
- dependence on software in implementation.

3. Capability Architectures

3.1 Description

A capability-based computer uses an architecture in which objects are addressed by means of a two-component entity called a capability. One component of the capability is a unique object identification number which is translated by the hardware into an actual machine address. The other component of the capability can be viewed as an access rights field which identifies to the hardware the operations that the owner of the capability may perform on the object.

Capability architectures have been promoted for a number of reasons including their hardware support for object-based programming [12] and system security [13]. A capability-based computer offers greater generality than does a conventional computer architecture. This generality includes hardware support for object identification and management which allows the user to approach the machine interface at a higher level of abstraction. By encapsulating objects and defining unique object identification numbers, the system can provide a more secure hardware base on which to place the operating system.

To maintain system security and integrity, it is typical for a capability-based computer to use hardware tagging of capabilities stored in memory [14,15]. When a user attempts to use a capability to reference an object, the hardware tag indicates that use of the capability is a legal one. The capability itself will be further compared with the operation that the user is attempting to ensure its validity. Since the tag controlled by hardware, the user is not able to arbitrarily modify the tag bits associated with a memory address. If the user attempts to modify a capability, the hardware will reset the associated tag bits.

Another feature of capability architectures is that the machine interface is usually implemented at a higher level than that of a conventional architecture. This higher level includes functions that relate to object addressing and object management. By placing greater functionality in the firmware, the goal is to improve the performance of the architecture while ensuring that the object related operations can not be interrupted and possibly altered by another process. Thus, the security of a capability-based computer follows the precept that hardware is inherently more secure than software.

3.2 Design Issues

There are a number of issues to be faced by the designer of a capability machine. These include:

- generating and maintaining unique object id's for a large number of objects
- managing objects, including object deletion and the dangling pointer problem

- controlling the copying of capabilities for object sharing

- defining object categories

- speeding-up object address translation

- permitting called programs to have more access rights than their callers for operating system functions

- providing object encapsulation to promote object protection.

The resolution of these issues can take various forms. Levy [16] surveys many of these in his book on capability based systems.

3.3 Goals

For the purposes of a discussion of capability system goals, we assume that the network facilities for the distributed system have already been secured using encryption and secure high-level protocols as described in the previous section. By employing capabilities for defining and protecting objects in a distributed environment, the following goals can be achieved:

- Objects can be transferred across system boundaries while preserving access rights across these boundaries. This is accomplished by forcing any object transfer between systems to be accompanied by the transfer of the capability needed to access the object. Without this capability, the object can not be accessed.

  The process performing the copy operation must possess the original capability on the source computer to effect the copy operation. The capability which results on the destination computer must uniquely identify the copied object and must have access rights equal to or less than those of the original capability. The network interfaces for each host are responsible for checking the validity of the operation. The network interface at the destination must generate a unique object id (possibly using already existing firmware for object creation) and must translate the source capability accordingly. At the same time it must preserve or decrease the access rights of the translated capability.

- Capabilities for objects can be transferred across system boundaries. This allows capabilities to be used to reference remote objects. This requires that the capability contain a field which identifies the network node containing the object. Alternatively, the capability could reference a local "network reference object" which would contain the information needed by the operating system and network interface to address the remote object.

- Objects can be referenced across system boundaries using either user-local or user-remote capabilities for these objects. This is analogous to a distributed file system, but is generalized to all the object categories defined in a given architecture.

  A user-local capability is one which is contained in the user's capability list in the local host from which the object reference is being made. Similarly, a user-remote capability is one that is contained in the user's capability list on the remote host that contains the object being referenced. Capabilities used to access objects created remotely are derived from the capability generated by the system where the object was created.

In describing these goals, it is assumed that object identification and addressing are defined locally. When a capability is transferred between systems, a new object id will be created by the destination host automatically. This object id will have meaning only in the context of this host. This will preclude the need for designing a universal object identification scheme that would be impractical both in terms of the size of the id needed and the overhead to coordinate the use of id's. It is also assumed that a capability can be safely and accurately transmitted between systems. The network interface for the capability-based computer controls the encryption and protocols needed to effect secure communications.

To support the preceding goals, a number of issues need to be addressed. First, in keeping with the fine granularity of capability access rights, it would be beneficial to define additional access rights that deal with network operations. These might include the capability to copy an object or the capability itself across the network interface. Access rights for remote operations on capabilities or objects might also be defined this way. Controlling the copying of a capability across a network interface has the same implication as controlling it between users on a single system.

Second, in some systems, an object can be given its own capability list for accessing whatever objects are needed in its operations. When the object is copied from one system to another, is this capability list also preserved? Although it may be desirable to define a network copy operation for capability lists, it does not seem advisable to automatically copy this list and translate it when the object itself is copied. This should be a separate operation, if done at all.

In translating a capability copied from one system to another, there are a number of conditions to be observed. First, the translated capability should never be greater than the original capability. This would violate the basic security principles of capability-based architectures. Second, the process receiving the copied capability should not be able to increase its access over any other objects by means of the copy operation. The situation where the copying of a capability gives the owner greater privilege must be avoided. Finally, if the two computers do not define their objects in the same fashion

(heterogeneous distributed capability system case), the host receiving the capability must translate it to an equivalent or lower object and access rights pair, or else reject the operation.

4. A Secure Distributed Capability System

4.1 Integrated Design

In this paper we deal with distributed systems of user terminals, processing hosts, storage elements, and other resources. The processors and terminals may be heterogeneous or of a compatible family. Our goal is to consider a design based on a combination of cryptography and a capability based control to provide network security.

There is a strong desire in a distributed system for the system to be transparent to the user. Rushby and Randell [7] established that network transparency is most easily achieved if all system components have a common interface. The "recursive structuring" principle for the design of distributed systems states: each component of a distributed system should be functionally equivalent to the entire system of which it is a part. This does not preclude heterogeneous sub-elements, since each system interface must contain provisions for exception conditions to be returned when a requested operation cannot be carried out. The value of the recursive structuring of a system is that, by definition, it is indefinitely extensible.

To use the capability approach in a distributed environment, additional capability categories are needed. These include definitions that protect the network interface and that validate specific network operations:

- network interface to a specific node can be used
- network parameters can be modified, examined, or tested
- capability can be copied across network
- object can be copied across network
- object can be used remotely
- object can be deleted remotely if user has delete capability
- capability can be translated (needed by network interface)
- network object (for referencing remote objects) can be created, managed, or deleted
- audit trail enable.

The network interface design should follow the standard seven-layer ISO OSI model. It will be subject to the same protection that the operating system is given on a capability machine, plus additional protection provided by whatever capabilities are required to use the interface. The various network protocol layers should be designed to promote detection of active network attacks. Data encryption can be built into the user session layer.

All network operations which require capability checking for validation are passed by the network interface to the operating system and/or firmware. Outgoing network transactions are checked in the normal way by comparing the attempted operation with the capability list of the agent process. Incoming transactions that involve the copying of a capability from a remote system will also involve the translation of the object identification within the capability and the object encapsulation to a valid object identification for the destination host. This translation will also be a firmware function that most closely resembles object creation.

4.2 Multilevel Considerations

If a distributed capability system were used in a multilevel security environment, both network security mechanisms and the capability architecture would need to be enhanced to recognize and protect objects of different classification levels.

Here we review some of the characteristics of a multilevel secure system and then discuss its relation to the one proposed. Users are assigned levels, some resources are assigned maximum levels and one must keep track of the high watermark (highest level received since cleansing) of the device. Objects have levels indicated by labels. A process keeps track of the high watermark of objects used in a current period. Users can specify the level of an object created and a process can specify the level of the objects it creates (which must dominate, i.e., be greater than or equal to, the current high watermark). There are several other details that pertain to specific implementations that will not be dealt with here, such as the principals that control the flow of data based on dominance rules.

The protection domain extends across the network, encompassing its nodes. Capabilities are used to determine transmission of objects across nodes, the same as they are within a node. The transmission is not allowed if the process does not possess the capability (e.g., the high watermark is greater than the security level of the destination). At the receiving node the processes cannot have access to the object without the appropriate capability.

Encryption for authenticity, key passing, and secrecy protection is within the encapsulated portion of the capability protocol, implemented in firmware. Also, detection techniques such as those discussed earlier -- unique transmission key, sequence numbers, error detection, request response, and time stamps -- are implemented and initiated at that level.

Encryption is at the user session protocol (layer 5), so that there is end-to-end encryption between geographically separate parts of the protection domain. The capability system would communicate the necessary protocol information to the

transport and other lower layers, providing the necessary protocol parameters.

Modifications to the capability hardware would consist of additional types of capabilities and additional bits to the object identification field of the capability. When a user account is created on a system, the profile of that user would be given capabilities to read, write, create and delete objects of specific classification levels. The capability to perform an operation at one classification level would allow the same operation to be performed at a lower level, provided that an indirect data leakage did not result. The user could also be given the capability to create objects, which could also be given the capability to read, write, create and delete sub-objects of different levels, all of which must be dominated by the user's own capabilities.

When an object is created, it would be created at a given classification level. This level could be economically encoded in the object identification field (2 bits provides 4 levels), which would also be encapsulated with the object itself. Thus, when any data transfer operation is performed on a given object, the object's classification level is used to insure that a legal data flow is occurring.

Additional capabilities would be needed to permit the changing of an object's classification level. Both the classification checks and capability tests would be performed by firmware. The rules governing legal and illegal data movements between levels would also be stored in firmware.

5. Evaluation

Just as the user community is slow to accept some of the most obviously beneficial computing improvements, it is felt that part of the task in portraying an unfamiliar way of thinking is to show consistency with present approaches. Rushby and Randell [7] have described a distributed computing system composed of small trustworthy security mechanisms linked together to provide multilevel security in such a way that the entire system appears as single system to its users. A prototype has been successfully demonstrated. Key to this system are separate security processors, operating in parallel with the general purpose processors, and a software subsystem "the Newcastle Connection," that links multiple UNIX systems, and does not require applications programs or operating system to be changed.

The Department of Defense Trusted Computer System Evaluation Criteria [8] will serve as a standard for the accreditation of commercial systems, at least in the near term, thus it was considered important to compare this system against those criteria. We have also considered Saltzer and Schroeder's [17] principles of design.

5.1 Definitions [8]

"Trusted Computing Base - All protection mechanisms within a computer system (including hardware, firmware, and software) the combination of which is responsible for enforcing the security policy." The cryptographic capabilities network can be considered a trusted computer base, but has an unusually large scope in that it encompasses a network.

"Domain - The set of objects that a subject has the ability to access." An object is defined here as a passive entity that contains or receives information, for which access potentially implies access to the information it contains. The capabilities system considers domain in the same context, however, it further specifies and controls resources and enforces the extent and type of access.

"Dominate - Security level S1 is said to dominate security level S2 if the hierarchical classification of S1 is greater than or equal to that of S2 and the non-hierarchical categories of S1 include those of S2 as a subset." A dominant capability can be enforced categorizing object id's into the appropriate classifications. Another approach would be to define a capabilities base at each independent level. In either case, the capabilities system can further restrict usage to what is required by a task.

"Reference Monitor Concept - An access control concept that refers to an abstract machine that mediates all accesses to objects by subjects." The hardware, firmware, and software elements of a Trusted Computing Base that implement the reference monitor concept are referred to as the security kernel. The capabilities based system employs and enforces a reference monitor type of control, independent of special hardware (although special hardware may be required to enhance performance).

"Star Property - A Bell-LaPadula security model [18] rule allowing a subject write access to an object only if the security level of an subject is dominated by the security level of the object." This rule can be enforced in a capabilities based system, but the implementation must place capabilities in control of the system and not the user.

5.1.2 Requirements [8]

"Discretionary access control - The trusted computer base (TCB) shall define and control access between named users and named objects. The enforcement mechanism shall allow users to specify and control sharing of those objects." Capability access control involves restricting access to objects or resources based on the possession of a ticket that unconditionally authorizes the possessor (user or process) access to the named object with specific rights, where objects include both resources and data. The list is actually inverted from the normal access control list, but contains at least the same information. It can be used by the operating system to emulate the discre-

tionary access model. If the system places the user "in charge", he can establish his own policy with respect to the capabilities possessed by him. In most DoD implementations, however, only a special user (the security officer) can pass capabilities to a user that has not previously possessed them at that level.

"Object Reuse - When a storage object is initially assigned, allocated, or reallocated to a subject from the TCB's pool of unused storage objects, the TCB shall assure that the object contains no data for which the subject is not authorized." This requires cleansing of the resource upon reallocation.

"Labels - Sensitivity labels associated with each ADP system resource that is directly or indirectly accessible by subjects external to the TCB shall be maintained by the TCB and shall be used as the basis for mandatory access control decisions." The assignment of capabilities can be based on the sensitivity of resources. The sensitivity labels can be built directly into the encapsulation scheme as a standard part of the object control. The resources are assigned virtually with the security manager having ownership of the assignment table with the right of revocation and reassignment.

"Label Integrity - Sensitivity labels shall accurately represent security levels of the specific subjects or objects with which they are associated. When exported by the TCB, sensitivity labels shall accurately and unambiguously represent the internal labels and shall be associated with the information being exported." As stated before, the sensitivity labels can be inherent to the definition of the capabilities and become part of the encapsulation scheme. The capability system enforces the authorization for exportation.

"Exportation of Label Information - The TCB shall designate each communications channel and I/O device as either single-level or multilevel, with changes done manually and any changes auditable. When the TCB exports an object to an I/O device, the sensitivity label associated with that object shall also be exported and, in the case of multilevel devices, shall reside on the same physical medium as the exported information and shall be in the same form (i.e., machine readable or human readable form). When the TCB exports or imports an object over a multilevel communication channel, the protocol used on that channel shall provide for the unambiguous pairing between the sensitivity labels and the associated information that is sent or received." This functionality can be incorporated in the capability system. The capability system enforces the transfer request, whereas a conventional system may not.

"Device Labels - The TCB shall support the assignment of minimum and maximum security levels to all attached physical devices to enforce the constraints imposed by the physical environments in which the devices are located." This is indirectly accomplished by the assignment of capabilities. This corresponds better with non data processing information control.

"Mandatory Access Control - The TCB shall enforce a mandatory access control policy over all resources (i.e., subjects, storage objects, and I/O devices) that are directly or indirectly accessible by subjects external to the TCB." External subjects become internally controlled by the capabilities list when they are given the capability of access, otherwise they possess none.

"Identification and Authentication - The TCB shall require users to identify themselves to it before beginning to perform any other actions that the TCB is expected to mediate. Furthermore, the TCB shall maintain authentication data that includes information for verifying the identity of individual users as well as maximum security levels to all attached physical devices." The identification must be part of the issuing of capabilities. The association with devices is more restrictive than simple security levels.

"Trusted Path - The TCB shall support a trusted communications path between itself and users for use when a positive TCB-to-user connection is required. Communications via this trusted path shall be activated exclusively by the user or the TCB and shall be logically isolated and unmistakably distinguishable from other paths." Since user consoles are resources, and because of the cryptographic requirements of this system, this requirement is rigidly enforced.

"Audit - The TCB shall be able to create, maintain, and protect from modification or unauthorized access or destruction an audit trail of access to the object it protects." The audit trail will be a capability assigned solely to the security control function.

5.2 Principles of Design

Saltzer and Schroeder [17] identified several design principles for protection mechanisms. Following is an evaluation of this approach against those criteria:

Least privilege - The capability system enforces this principle to a greater extent than existing implementations.

Economy of mechanism - This architecture supports security control to a far greater degree than general architectures and therefore should be verifiable. In general, hardware is simpler to verify than software or software/hardware mechanisms.

Complete mediation - This requirement is a basic design principle.

Open Design - The design is completely open and does not depend on any secret parts.

Separation of privilege - Satisfaction of this requirement is moot, although the implementation depends on the technique for allocation of capabilities and identification when logging on the system. The implementation of labels and a consistency check against user identification should satisfy this requirement.

Least common mechanism - The mechanism is protected and each user has a separate virtual capability. The concept of distributed control in physically distributed elements tends to support this principle, but certainly not to its ultimate intent.

Psychological acceptability - The mechanism cannot be bypassed and is transparent to the user.

5.3 Advantages and Disadvantages

A capability approach to distributed system security offers strong object protection in both local and distributed contexts. This strength derives from firmware support of access rights at the machine addressing level. In addition, the design offers greater granularity of access rights than is found in a conventional operating system.

A distributed capability system is not without its complications. One potential problem is the vulnerability of capabilities as they are transmitted across the network. This is analogous to the problem of password transmission across a network in a conventional system. Both can be solved by encryption.

Another possible problem is the translation of capabilities in an environment of heterogeneous capability machines. Because object categories may vary from machine to machine, the difficulty is in preserving the meaning of the capability when it is translated. From a security standpoint, security is not compromised if the original capability dominates the translated capability.

A more difficult situation is the linking of a conventional computer to a network of capability systems. Since conventional operating systems do not support the same granularity of protection, meaningful sharing and strong security will probably not be compatible goals. The conventional computer will be the Achille's heel of the distributed capability network if remote object references are uncontrolled.

A final issue is the translation of the capability list for an object that is being copied from one system to another. For efficiency reasons, we have considered it advantageous for the copy operation to copy only the object and the capability for its use, and ignore the capability lists belonging to the object and any of its creations.

6. Summary

The meshing of capability characteristics and a cryptographically supported network is natural. Cryptography will support network communications and detection functions using public key systems or trusted interface modules to provide satisfaction of security protection from the outside world, as well as authentication functions. The capability based resource control provides a simpler environment than that dealt with by a discretionary kernelized system. There is a natural checking mechanism for determination of system misuse and simpler recovery in the event of a malicious internal attack. The system can be changed as the security policy changes without hardware/software modification.

A capability approach can provide a distributed system where data originators or some central authority determine the data, program, and sharing policy. The distributed capability system described here solves the problem of preserving access rights across system boundaries, since an object can not be referenced or copied across the network interface without processing the capability for a specific operation. In comparison to a conventional operating system, a capability based design offers greater protection and more granularity.

With proper implementation, the system also appears to be capable of supporting the DoD trusted system requirements under the unique DoD security policy implementation. Further, a properly architected capability machine and network interface could provide a secure multilevel distributed system. The DoD security requirements could be met by a design including the following provisions:

- Star property should be enforced by the system through assignment of high water mark levels to capabilities, objects, and resources.
- Sensitivity labels need to be integrated into the capabilities protection mechanism, and then be supported accordingly.
- User identification and authentication must be part of the capability issue and usage mechanism
- End-to-end encryption needs to be integrated and network protocol interfaces need to be developed

References:

1. Grayson, W.C., "Vulnerabilities of Data Telecommunications," in Advances in Computer Security Management, V2, ed. by M.M. Wofsey, John Wiley, 1983, 161-172.

2. "Data Encryption Standard," FIPS PUB 46, National Bureau of Standards, Washington D.C., January 1977.

3. Rivest, R.L., A. Shamir, and L. Adelman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," Communications ACM, V21(2), 120-126, February 1978.

4. Voydock, V.L., and S.T. Kent, "Security Mechanisms in High-Level Network Protocols," ACM Computing Surveys, Vol. 15, No.2, June 1983.

5. DeMillo, R., R. Lipton, and L. McNeil, "Proprietary Software Protection," in Foundations of Secure Communication, ed. by R. A. DeMillo, et al, Orlando, FL: Academic Press, 1978, 115-132.

6. Davies, D.W., and W.L. Price, Security for Computer Networks, John Wiley and Sons, 1984

7. Rushby, J.M., and B. Randell, "A Distributed Secure System," Computing Laboratory, University of Newcastle upon Tyne, December 1982.

8. Department of Defense Trusted Computer System Evaluation Criteria, CSC-STD-01-83, 15 August 1983

9. Anderson, J.P., Computer Security Technology Planning Study, ESD-TR-73-51, Vol. I, AD-758 206, ESD/AFSC, Hanscom AFB, Bedford, Mass., October 1972.

10. Gasser, M., and D.P. Sidhu, "A Multilevel Secure Local Area Network," Symposium on Security and Privacy, 1982

11. Popek, G.J., and C.S. Kline, "Encryption Protocols, Public Key Algorithms, and Digital Signatures in Computer Networks," in Foundations of Secure Communication, ed. by R. A. DeMillo, et al, Orlando, FL: Academic Press, 1978, 133-154.

12. Organick, I. E., A Programmer's View of the Intel 432 System, New York: McGraw-Hill. 1983

13. Routh, Capt. R. L., "A Proposal for an Architectural Approach Which Apparently Solves All Known Software-Based Internal Computer Security Problems," ACM Operating Systems Review, (Sept 1984), 31-39.

14. Lorriore, L., "Capability Based Tagged Architectures," IEEE Transaction on Computer, vol C-33, no. 9. (Sept 1984), 786-803

15. Houdek, M. E., F,G, Soltis, and R. L. Hoffman, "IBM System/38 Support for Capability-Based Addressing," Proc. 8th Annual Symposium on Computer Architecture, Minneapolis, MN, 1981, 341-348

16. Levy, H.M., Capability-Based Computer Systems, Digital Press, 1984.

17. Saltzer, J.H., and M.D. Schroeder, "The Protection of Information in Computer Systems," Proceedings IEEE, Vol. 63(9) pp 1278-1308, September 1975.

18. Bell, D.E., and L.S. LaPadula, Secure Computer Systems: Unified Exposition and Multics Interpretation, MTR-2997 Rev. 1, Mitre Corporation, Bedford, Mass., March 1976.

19. Rauch-Hindon, W., "Distributed Databases," Systems and Software, September 1983.

What is the ElGamal Cryptosystem?

The ElGamal system is a public-key cryptosystem based on the discrete logarithm problem. It consists of both encryption and signature algorithms. The encryption algorithm is similar in nature to the Diffie-Hellman key agreement protocol (see Question 24).

The system parameters consist of a prime p and an integer g, whose powers modulo p generate a large number of elements, as in Diffie-Hellman. Alice has a private key a and a public key y, where $y = g^a \pmod p$. Suppose Bob wishes to send a message m to Alice. Bob first generates a random number k less than p. He then computes

$y_1 = g^k \pmod p$ and $y_2 = m \text{ xor } y^k$,

where xor denotes the bit-wise exclusive-or. Bob sends (y_1, y_2) to Alice. Upon receiving the ciphertext, Alice computes

$m = (y_1^a \bmod p) \text{ xor } y_2$.

The ElGamal signature algorithm is similar to the encryption algorithm in that the public key and private key have the same form; however, encryption is not the same as signature verification, nor is decryption the same as signature creation as in RSA (see Question 8). DSA (see Question 26) is based in part on the ElGamal signature algorithm.

Analysis based on the best available algorithms for both factoring and discrete logarithms shows that RSA and ElGamal have similar security for equivalent key lengths. The main disadvantage of ElGamal is the need for randomness, and its slower speed (especially for signing). Another potential disadvantage of the ElGamal system is that message expansion by a factor of two takes place during encryption. However, such message expansion is negligible if the cryptosystem is used only for exchange of secret keys.

| ***Question 30*** |

015. Examples of known word processing methods and systems such as MS Word 2000.

User's Guide

Microsoft.

WORD

The World's Most Popular Word Processor

Document No. WB51157-1093
Printed in Ireland :09

For example, suppose you installed Word in the WINWORD directory of a file server—where X designates the file server—and distribute a silent script that uses the MYSCRIPT.STF table file to a user named Paul Tanner. The command line to run the script would be:

```
x:\winword\setup.exe /t myscript.stf /n "Paul Tanner" /q
```

Distributing a Script with Microsoft Mail

If you use Microsoft Mail to distribute a script, create a new message and then choose Insert Object from the Edit menu. In the Object Type box, select Package, and then choose the OK button. From the Edit menu in Object Packager, choose Command Line. Type the full path to SETUP.EXE in the WINWORD directory of the file server or the shared directory. (If your network supports UNC pathnames, use that syntax. If not, users will need to make the network connection themselves by using the same drive letter you specified before running Setup.) Type **setup** and the switches and arguments as needed, and then choose the OK button.

To attach the Word Setup icon to the command line, choose the Insert Icon button in Object Packager. Choose the Browse button to locate SETUP.EXE in the WINWORD directory of the network file server, and then choose the OK button. Choose Update from the File menu to add the icon to the Mail message, and then choose Exit from the File menu to close Object Packager. The icon is now ready to distribute. Anyone who receives the message can double-click the icon to run Setup from the network and install Word by using the script you specified with the /t switch.

Network Considerations for Workstation Users

There are two ways to run Word in a network environment:

- You can run Word entirely off the network, without installing it on your own computer.
- You can install Word on your own computer.

Installing Word on a Workstation

If your computer is connected to a network file server or a shared directory, your network administrator may have installed a copy of Word on the network that you can then install on your workstation. The administrator may also have created a process you can use to install Word automatically. Check with your administrator to determine the best way for you to install Word.

The procedure for installing Word on a workstation is discussed in Chapter 1, "Installing and Starting Word," in *Microsoft Word Quick Results*. Once you have installed Word, read the following section for important information about using Word in a network environment. You may also need special network software to manage and synchronize shared files on the file server. For more information, check with your network administrator.

Sharing and Protecting Documents on Networks

Using Word on a network is essentially the same as using Word on a stand-alone computer. On a network, however, you can use the network file server to store documents and exchange them with other users, so you may want to protect some documents from unauthorized access.

Things to Remember About Shared Documents

- If your workgroup uses a standard set of templates to ensure consistency, do not use other templates when you're working on a shared document.
- Be careful when you assign custom shortcut keys, especially when you redefine built-in shortcut keys.
- In order for everyone who works on a shared document to display and print it the same way, the fonts used in the document must be available on the other computers and printers in your workgroup.

  If you use TrueType fonts in shared documents, however, the fonts can be embedded in documents so that others who do not have those fonts installed can still see and print them. For more information, choose Options from the Tools menu, select the Save tab, and then choose the Help button.
- If you assign a file-protection password, you should write it down. Without the password, no one can open the document. Also bear in mind that some kinds of passwords—such as those that prevent any changes except annotations and form field edits—do not prevent other users from setting a file-protection password.

For more information about sharing and protecting documents, see Chapter 25, "Opening, Saving, and Protecting Documents."

Ensuring Compatibility

If you frequently use Word with documents created in other applications, you set options to compensate for limitations in the conversion process. These options don't change the documents, but they can make them easier to work with in Word. To view these options, choose Options from the Tools menu, and then select the Compatibility tab. For more information, see Chapter 26, "Converting File Formats."

Protecting a Form from Changes

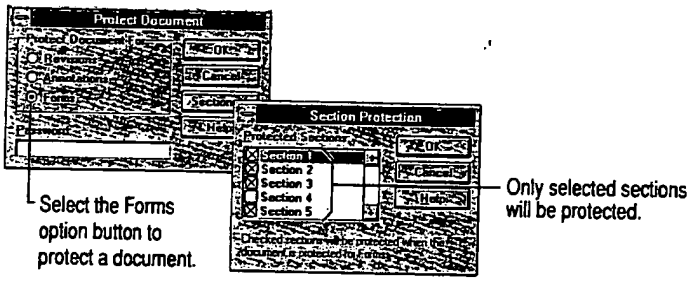
To activate form fields and to ensure that users don't accidentally change a form as they fill it in on line, you must lock the template for the form by choosing the Protect Document command from the Tools menu. When a document is protected, form fields are available for fill-in, and users can type only in form fields or unprotected sections.

Each time a user creates a form based on the protected template, a new, untitled document is created with the same protection as the template.

When you protect a document, Word changes it in the following ways:

- Form fields are activated.
- Field results are displayed instead of field codes.
- The insertion point can move only to form fields and unprotected sections.
- The entire document cannot be selected.
- Table column width is fixed.
- Some commands, such as Find, Replace, and Go To, are usable only in form fields and in sections from which protection has been removed.
- Some menu commands are unavailable.
- Entry macros, exit macros, and form field Help (described in the following sections) are activated.

You can specify a password when you protect a form. Only users who know the password can remove the protection and change the form.



▶ **To protect a form**

1. From the Tools menu, choose Protect Document.
2. Under Protect Document For, select the Forms option button.

   To assign a password to the form, type a password in the Password box. A password can contain up to 15 characters and can include letters, numbers, symbols, and spaces. As you type the password, Word displays an asterisk (*) for each character you type. If you assign a password, you must use the same password to remove protection from the document. Note that passwords are case sensitive. Each time you type the password you must use the same combination of uppercase and lowercase letters.
3. Choose the OK button.

   If you assigned a password, retype the password in the Confirm Password dialog box, and then choose the OK button.

When the active document is protected, the Protect Document command changes to Unprotect Document.


Protect Form button

Tip When you are designing a form, you can quickly turn protection on or off by clicking the Protect Form button on the Forms toolbar or by choosing Protect Document or Unprotect Document from the Tools menu.

▶ **To prevent a section from being protected**

1. From the Tools menu, choose Protect Document.
2. Select the Forms option button, and then choose the Sections button.

   If the Sections button is dimmed. there is only one section.
3. Under Protected Sections, select the check boxes to the left of the sections that you want to protect. Clear the check boxes next to the sections you want to leave unprotected, and then choose the OK button.
4. Choose the OK button to protect the document.

Note Some commands (such as Form Field Options) are unavailable in unprotected sections of a protected document. For full access to all word commands, remove protection from the document.

Protecting Documents from Changes

Word provides several ways to restrict changes to documents. You can assign a password to prevent other users from opening a document or to keep others from saving changes to the document. You can also request or require that other users on a network open a document as read-only.

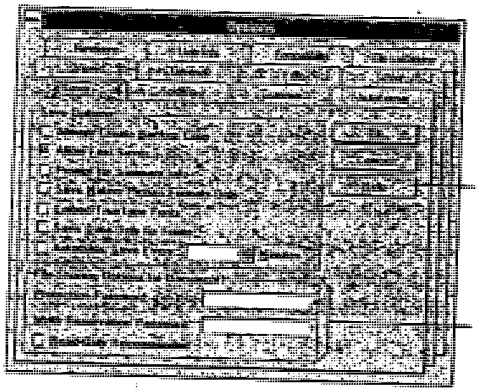
You can also assign a password so that other users can annotate a document and mark revisions. You or someone else who knows the password must open the document normally and review the changes before they become permanent. For more information, see Chapter 25, "Annotating, Revising, and Routing Documents."

If you use form fields to create a form, you can assign a password so that other users can fill in those parts of the form but cannot change anything else in the document. For more information, see Chapter 14, "Forms."

Warning If you assign a password for any of these types of protection, it's a good idea to write it down. Without the password, you cannot open the document.

Setting Passwords and Selecting Save Options

To assign a password to a document and set options that control whether changes can be saved, choose the Options button in the Save As dialog box or choose Options from the Tools menu, and then select the Save tab.



Protection Password To prevent other users from opening a document, type a password in the Protection Password box. Only users who know the password can open the document. Passwords are case-sensitive.

Write Reservation Password To prevent other users from saving changes to a document, type a password in the Write Reservation Password box, and then choose the OK button. Word will prompt you to type the password again to confirm it. Word then requires you to type the password to open the document normally. If you do not know the password, you can still open the document as read-only by choosing the Read Only button in the Password dialog box that appears when you open the document.

Read-Only Recommended To recommend, but not require, that other users open a document as read-only, select the Read-Only Recommended check box. When another user opens a document that's protected by this option, Word indicates that the document should be opened as read-only unless changes need to be saved. The user can then open the document normally or as a read-only document.

▶ **To protect a document with a password**

1. Open the document you want to protect with a password.
2. From the File menu, choose Save As.

   If you have not yet named the document, type a name in the File Name box.
3. Choose the Options button.
4. In the Protection Password box or the Write Reservation Password box, type a password, and then choose the OK button.

   A password can contain up to 15 characters and can include letters, numbers, symbols, and spaces. As you type the password, Word displays an asterisk (*) for each character you type. Note that passwords are case-sensitive.
5. When Word prompts you to confirm the password, retype it and then choose the OK button.
6. To save the document, choose the OK button.

   Make sure that you write down the document password, exactly as you typed it. You will need to type it the next time you open the document.

Tip If you want to allow other users to add only comments to a document, you can protect it by using the Protect Document command on the Tools menu. Other users can then open the document, but they can only make comments by using annotations.

▶ **To change or delete a password**

1. Open the document whose password you want to change or delete.
2. From the File menu, choose Save As.
3. Choose the Options button.
4. In the Protection Password box or the Write Reservation Password box, select the row of asterisks that represents the existing password, and then do one of the following:
   - To change the password, type the new password.
   - To delete the password, press DELETE.
5. Choose the OK button.

   If you changed the password, Word asks you to retype the new password.
6. To save the document with the new password, choose the OK button.

Other Ways of Protecting Documents

Word offers other methods of protecting your documents.

For information about	See
Opening documents as read-only	"To open an existing document," earlier in this chapter
Preventing any changes to documents except for filling in form fields	Chapter 14, "Forms"
Preventing any changes to documents except for annotations and marked revisions	Chapter 25, "Annotating, Revising, and Routing Documents"

Note Protecting a form or locking a document for annotations or revisions does not keep another user from saving that document with a password or from setting other save options. If you want to protect a document from all types of changes, save it with a password by using one of the methods described in this chapter.

Some operating systems and networks also provide ways to protect documents. To find out if your system has these features, check with your network administrator or see the documentation for your operating system or network.

Note You can customize the marks Word uses to show document differences. For more information, see "Customizing Revision Marks," earlier in this chapter.

▶ **To compare two versions of a document**

1. Open the edited version of the document.
2. From the Tools menu, choose Revisions.
3. Choose the Compare Versions button.
4. In the Original File Name box, type or select the name of the original document, and then choose the OK button.

   Word displays the edited document marking inserted, deleted, and revised text with revision marks. The options for displaying revision marks are set on the Revisions tab in the Options dialog box (Tools menu).
5. To accept or reject the revisions, choose Revisions from the Tools menu. For more information, see "Incorporating Revisions," earlier in this chapter.

Protecting a Document for Annotations and Revisions

For more information on ways to protect a document, see Chapter 21, "Opening, Saving, and Protecting Documents."

To allow reviewers to comment on but not make changes to a document, you can protect it for annotations. To allow reviewers to change a document and keep a record of all changes, you can protect it for revisions.

For maximum protection, you should also use a password when you protect a document for annotations or revisions. Otherwise, anyone can remove protection from the document by choosing Unprotect Document from the Tools menu.

▶ **To protect a document for annotations or revision marks**

1. Open the document you want to protect.
2. From the Tools menu, choose Protect Document.
3. Do one of the following:
   - To allow reviewers to insert annotations but not change the contents of the document, select the Annotations option button.
   - To track revisions, select the Revisions option button. The reviewers cannot turn off revision marking, and revisions cannot be accepted or rejected.
4. To ensure that a document is protected against untracked changes, type a password. This prohibits anyone who does not know the password from unprotecting the document.
5. Choose the OK button.

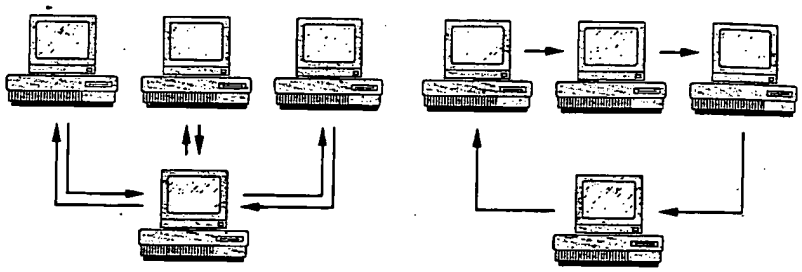
▶ **To unprotect a document for annotations or revision marks**

- From the Tools menu, choose Unprotect Document.

  If the author has protected the document with a password, you must know the password to unprotect the document.

Routing a Document Online

You can use Word and Microsoft Mail or a compatible mail program to route documents online. For example, you might want others to review an important memo before sending it out, or you might want several people to complete an online questionnaire or form.

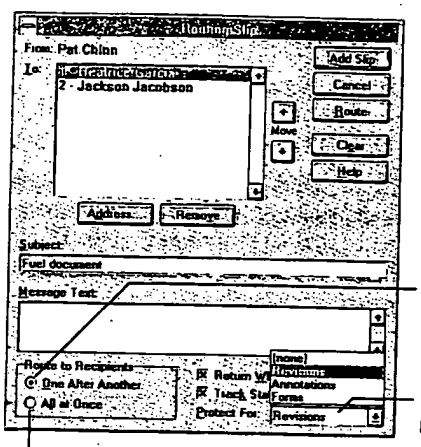


You can route a document to all reviewers at once ...

or you can route it to one reviewer after another.

You can route online copies in two ways. You can send a separate copy to all reviewers at the same time, or you can send a single copy that goes to each person on the list in turn, allowing each reviewer to see the comments of all previous reviewers.

Reviewers return their annotated or revised copies to you or the next person on the distribution list by choosing the Send command from the File menu. When all the copies have been returned, you can merge the annotations and revisions into the original document to simplify review of the comments. For more information on merging comments, see "Merging Annotations and Revisions," later in this chapter.



To route a document to others

1. Open the document you want to route.
2. From the File menu, choose Add Routing Slip.
3. Choose the Address button. Select the names of the people to whom you want to route the document, choose the Add button, and then choose the OK button.

   If you want to route the document to one recipient after another, use the Move up and down arrows to put the names in the correct routing order.
4. In the Subject and Message Text boxes, type the subject and any message or instructions you want to send with the document. Each recipient will receive the same subject and message.

   Word automatically appends instructions to your message telling recipients to choose the Send command when they are finished.
5. Under Route To Recipients, do one of the following:
   - To route one copy of a document to one recipient after another, select the One After Another option button.
   - To route multiple copies of a document to all recipients at the same time, select the All At Once option button.

6. Select any other options you want, and then choose the Route button.

   If you want to continue to edit the document before you route it, choose the Add Slip button, and continue to edit the document. When you are ready to send the document, choose Send from the File menu. Word displays a message asking you to confirm that you want to route the document.

   The document is sent to the distribution list as an attached Word file. The recipients can add annotations or revisions to the document and then return the copy to you by choosing the Send command on the File menu.

   If the document is being routed to one recipient after another, the Send command automatically routes it to the next person on the list before it returns to you. You will receive all the recipients' comments in one document after it has been routed to the last person on the list.

   If you send the document to all recipients at the same time, you will receive multiple copies of the document. You can then merge all changes into one document. For more information, see the following section.

Merging Annotations and Revisions

If you have given individual copies of a document to multiple reviewers, you can combine their annotations and revisions into the original document. When you merge annotations and revisions, any annotations and revisions already in the original document are preserved as additional comments are merged. Word assigns a different color to each reviewer. If there are more than eight reviewers, Word uses the colors again, so some reviewers may share the same color.

Note Annotations and revisions cannot be merged back to the original document unless they are marked. To ensure that revisions to a document are marked, you should protect the document for revisions or annotations before making the revisions. For information on protecting documents, see "Protecting Documents for Annotations and Revisions," earlier in this chapter.

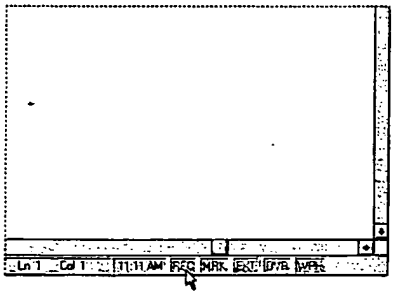
▶ **To merge revision marks and annotations**

1. Open the document that has revisions you want to merge into the original document.
2. From the Tools menu, choose Revisions.
3. Choose the Merge Revisions button.

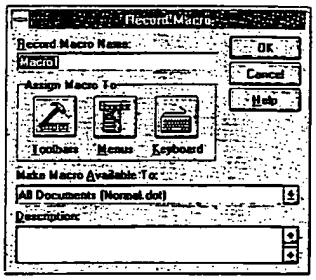
QUICK START

Recording a Macro

You double-click "REC" on the status bar to display the Record Macro dialog box, where you can type a name for your macro or accept the name Word proposes. When you choose the OK button to close the dialog box and begin recording your macro, Word displays the Macro Record toolbar. Word records each command you choose and action you take until you click either the Pause button to temporarily suspend recording or the Stop button to finish your macro. You can also double-click "REC" on the status bar to stop recording a macro.



Double-click "REC" on the status bar to display the Record Macro dialog box.



Accept the name Word proposes or type another name.

To indicate that the recorder is on, Word attaches a recorder graphic to the mouse pointer.

Use the Stop and Pause buttons on the Macro Record toolbar to stop or pause recording.

Assigning a Macro to a Toolbar, a Menu, or Shortcut Keys

Assigning a macro to a toolbar, a menu, or shortcut keys is a good way to make the macro more accessible. You can do this by choosing the appropriate button in the Record Macro dialog box.

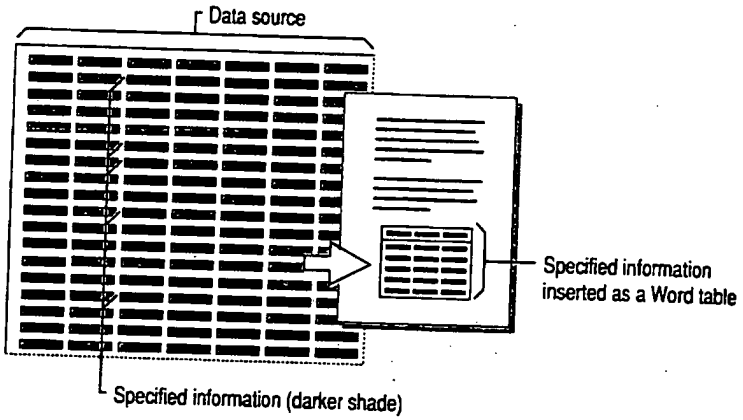
Choose one of these buttons to assign the macro you are recording to a toolbar, a menu, or shortcut keys.

Running a Macro

Once you've assigned a macro to a toolbar, a menu, or shortcut keys, running the macro is as simple as clicking the appropriate toolbar button, choosing the appropriate menu command, or pressing the corresponding key combination. You

Inserting Tables of Information from a Database

Sometimes you may want to include in a Word document information from an existing database, a Microsoft Excel worksheet, or another source of data. By using the Database command on the Insert menu, you can specify the information you want and automatically insert it as a table in a Word document. You can screen, or "filter," the information according to criteria you select. You can also instruct Word to update the information in the Word document if the source file has changed.



Word can retrieve information from the following types of files:

- Files from the following applications that are installed on your system:

  Microsoft Access®
  Microsoft Excel

- Files from single-tier, file-based database applications for which you have an open database connectivity (ODBC) driver installed in the System subdirectory of your Windows directory. ODBC drivers for the following applications are supplied with Word:

  Microsoft Access
  Paradox®
  Microsoft FoxPro® (or other Xbase database application such as dBASE®)

For a list of file converters provided with Word, see Chapter 26, "Converting File Formats."

- Files for which you have a file converter installed. In addition to converters for ASCII text files, Word provides file converters for many applications, including:

Microsoft Word for Windows	WordPerfect 5.*x* for MS-DOS and Windows
Microsoft Word for the Macintosh versions 3.*x*,[1] 4.*x*, and 5.*x*	Microsoft Excel 2.x,[2] 3.0, 4.0,[1] and 5.0[3]
Microsoft Word for MS-DOS 3.0–6.0	Lotus 1-2-3 2.*x*[2] and 3.*x*[1]

[1] Converts only from this format.

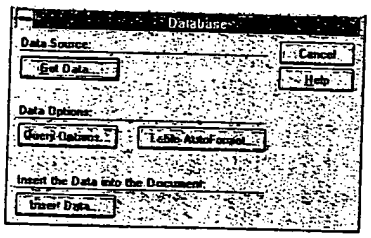
[2] Converter works only with Windows version.

[3] Converter works only with Macintosh version.

You can also insert information from another Word document. For example, you might have set up a membership directory for use as a mail merge data source. Instead of copying and pasting information from various data records, use the Database command to insert just the information you request.

Inserting the Data

When you choose the Database command from the Insert menu, Word displays the Database dialog box. Now you can locate the data source, select the information you want, and format the table in which the information is displayed.



Once you select the data source, the other buttons in the dialog box become available.

By default, Word inserts all of the information from the selected data source. In most cases, however, you'll want to use only some of the available information. For example, from a large personnel file, you might want to list only the names, departments, and hiring dates of all employees who have worked for your company 10 years or longer.

If information in the data source changes frequently and you want to keep your document up to date, you can insert the information as a Word *field*. The field is simply a "placeholder" that represents the table in your document. For more information, see "Keeping the Table Information Up to Date," later in this chapter.

▶ **To insert information from a data source as a table**

1. Position the insertion point where you want the new table of information to be included.
2. From the Insert menu, choose Database.
3. Choose the Get Data button.
4. In the Open Data Source dialog box, type or select the filename of the data source you want to open, and then choose the OK button.

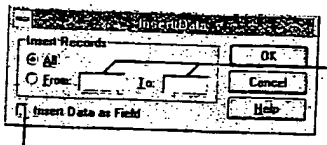
   If the data source is not listed, select the appropriate drive and directory or folder. Then select the appropriate option in the List Files Of Type box.

   If you open a Microsoft Excel worksheet, you can insert the entire worksheet or a range of cells. If you open a Microsoft Access database, you can insert records from a table or a selection of records defined by a query. For more information, see the documentation for the application you are using.
5. To insert specific information from the data source or list the information in a particular order, choose the Query Options button. Do one or more of the following, and then choose the OK button.
   - On the Filter Records tab, specify criteria to select the data records to insert.
   - On the Sort Records tab, select the data fields by which you want to sort the information.
   - On the Select Fields tab, remove any fields you don't want from the Selected Fields list. The order of the fields in the list determines the order in which the fields are inserted left to right.

     If you don't want to insert the field names from the header row with the data records, clear the Include Field Names check box.
6. To format the table, choose the Table AutoFormat button.
7. Choose the Insert Data button.

   In the Insert Data dialog box, you can specify the range of records you want to insert. The range refers only to the records that were selected by the query. If you want to be able to update the information in the table automatically, select the Insert Data As Field check box. Then choose the OK button.

Note If you insert more than 31 data fields, Word inserts tab characters to separate the columns of information.



To specify which of the selected records are inserted, type starting and ending record numbers in the From and To boxes.

Select this check box if you want to keep the table information up to date.

Modifying the table format If you don't select the Insert Data As Field check box, Word inserts the information as an ordinary table. You can resize the table columns and otherwise modify the table by using the commands on the Table menu. If you insert the information as a field, however, you must choose the Database command again to reinsert the table and update the table format by choosing the Table AutoFormat button. Otherwise, the table formatting you've applied is removed the next time you update the DATABASE field. Formatting you've applied to text in the table is also removed. For more information, see "Keeping the Table Information Up to Date," later in this chapter.

Modifying the information in the table You may want to modify the information in the table later. For example, you might want to include another column of information or select a different set of records from the data source. To do this, click in the table and then choose the Database command again to select the information you want in the table. If you insert the information as a field and then edit or format the text in the displayed table, your changes will be deleted the next time you update the DATABASE field. For more information, see "Keeping the Table Information Up to Date," later in this chapter.

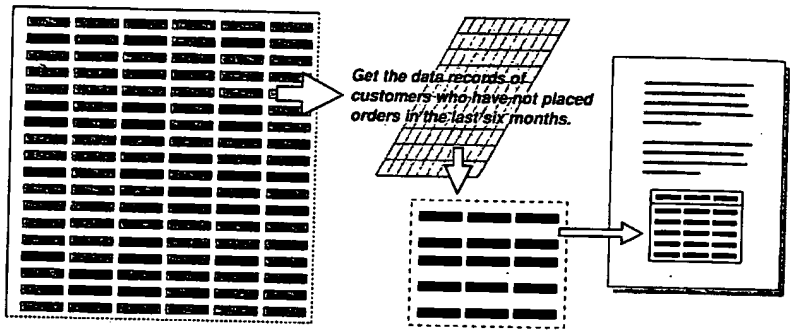
If Word can't recognize the field and record delimiters If Word can't recognize the characters used to separate data fields and data records in text-delimited files (files in which data is separated by commas, tab characters, or other characters), Word asks you to select the separating characters (delimiters). Word recognizes one data field delimiter and one data record delimiter. If a combination of two or more characters is used as a delimiter, then the remaining characters are treated as text in the data fields.

Selecting the Data

To get only the information you want from a data source, you create a *query*. A query is simply a set of instructions, or rules, that describes the information you want from the data source. You can think of the following statement as a query:

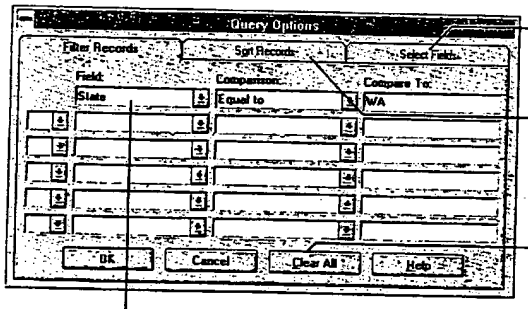
"Give me the names, addresses, and account numbers of all customers who have not placed orders in the last six months."

The first part of the statement identifies the categories of information you want—names, addresses, and account numbers. The second part of the statement indicates that you want information only for certain customers—those who have not ordered anything in the last six months.



A query tells Word which information to select from a data source.

You create queries by selecting options in the Query Options dialog box. You select data fields to specify the categories of information you want. The order in which you select the data fields determines the order of the columns of information in the table, from left to right. To get the information only from certain data records, you specify one or more rules for selecting the records. To list the rows of information in a particular order, you can sort the data records.



Specifying the Record-Selection Rules

On the Filter Records tab, you specify the rules that Word uses to retrieve the information you want, based on the contents of selected data fields. When specifying a rule, you can select any data field in the data source—even a data field you don't want to include in the table.

A record-selection rule is made up of three parts:

- A field name corresponding to a data field in the selected data source
- A comparison phrase, such as "Equal To" or "Is Not Blank"
- Text or numbers you want the contents of the data field to be compared with

If you compare text When comparing a data field that contains text, Word compares the sequence of characters based on the ANSI sorting order. The text "apple" is considered "less than" the word "berry" because, alphabetically, "apple" precedes "berry." Whether the text is uppercase or lowercase isn't significant.

For example, to retrieve data records for members of your organization whose last names begin with "A" through "L," you specify the following rule:

LastName Is Less Than M

Any name beginning with "A" through "L" is considered less than "M," so only the data records that contain those names are selected. (The last name must be contained in a separate data field, or else it must precede the first name in the field—for example, "Bendal, Maria".)

If you compare numbers mixed with text If numbers are mixed with letters, hyphens, plus or minus signs, or other nonnumeric characters, Word compares the numbers as though they were a sequence of text characters. For example, a five-digit U.S. ZIP Code is compared as a number, whereas a nine-digit "ZIP+4" code such as 99999-9099 is compared as text, as are non-U.S. postal codes that contain letters.

Comparing sequences of mixed numbers and nonnumeric characters—code numbers, for example—can have different results if some items contain more sequential numerals than others. For example, the following items are sorted in this order:

0002xy, 002, 011y, 1, 1x, 1yz, 22x, 2x

The following items, however, are sorted in this order:

0001, 0001x, 0001yz, 0002, 0002x, 0002xy, 0011y, 0022x

Specifying Multiple Rules

You can specify as many as six selection rules. Using multiple rules allows you to narrow the range of data records that are selected. When you select multiple rules, you must specify AND or OR to connect each additional rule to the preceding rule, as in the following examples.

Example 1	Example 2
State (Is)-Equal To Oregon	State (Is) Equal To Oregon
AND City (Is) Equal To Portland	OR State (Is) Equal To California

The rules connected by AND select only data records that contain both "Oregon" in the State field and "Portland" in the City field. The rules connected by OR select all data records that contain either "Oregon" or "California" in the State field—a potentially larger number of records. The key difference between AND and OR is as follows:

- When you use AND to connect rules, Word selects only those records that satisfy both (or all) rules. Each rule connected by AND *eliminates* more of the records in the data source.
- When you use OR to connect rules, Word selects any record that satisfies at least one of the connected rules. Each rule connected by OR *selects more* of the records in the data source.

AND has precedence You can use AND and OR separately or in combination. In sets of rules that contain both AND and OR, rules connected by AND have precedence over rules connected by OR. This means that the set of rules connected by AND is used to select records before the set of rules connected by OR. How you connect the rules—by using AND or by using OR—affects which data records are selected.

Suppose you want to select data records of all clients who live in either Portland or Salem, Oregon. In the Query Options dialog box, you would specify the following rules to determine the contents of the data fields "City" and "State":

State (Is) Equal To Oregon
AND City (Is) Equal To Portland
OR State (Is) Equal To Oregon
AND City (Is) Equal To Salem

Using the first set of rules connected by AND, Word compares the data records to identify the clients who live in Portland, Oregon. Next, Word compares the data records with the next set of rules connected by AND. Word then selects only data records of clients in Oregon who live in either Portland or Salem.

Notice that the following set of rules does *not* produce the same result:

State (Is) Equal To Oregon
AND City (Is) Equal To Portland
OR City (Is) Equal To Salem

Because AND takes precedence, the first set of rules connected by AND selects records of clients who live in Portland, Oregon. However, the rule connected by OR also selects records for clients in any city named Salem—including Salem, Massachusetts, for instance

Comparing a range of values You can also use AND to compare a selected field with a range of values rather than a single value. For example, given the following rules, Word selects all data records that have a value of 98001 through 98500 in the PostalCode field.

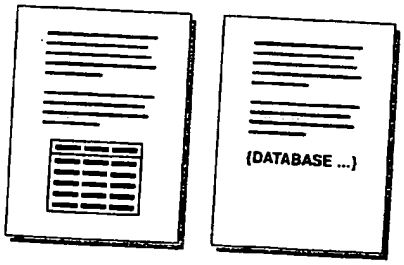
PostalCode (Is) Greater Than Or Equal (To) 98001
AND PostalCode (Is) Less Than Or Equal (To) 98500

Keeping the Table Information Up to Date

If you select the Insert Data As Field check box when you insert the information, Word does not insert an actual table; instead, it inserts a DATABASE field to represent the table.

With field codes hidden, the information is displayed as a table. With field codes displayed, the information is displayed as a DATABASE field. The field contains all information needed to locate and open the selected data source, carry out the query, and insert the information in your document.

To display or hide the field codes, press ALT+F9 (Windows) or OPTION+F9 (Macintosh).



Document with field codes hidden ...

... and with field codes displayed.

Updating information in the table To bring in the latest information from the data source, you update the DATABASE field. To update the field, click anywhere in the table, and then press F9. Word then repeats the query instructions you specified in the Query Options dialog box and inserts the latest information from the data source.

For more information about Word fields, see Chapter 32, "Inserting Information with Fields." For specific information about the DATABASE field, click in the field in your document and then press F1 (Windows) or the HELP key (Macintosh).

Preserving the table format The DATABASE field also describes the table format you selected in the Table AutoFormat dialog box. If you manually resize the table columns, change the text formatting, apply borders, or change other aspects of the table format, the added formatting is lost the next time you update the DATABASE field.

To retain changes to the table format when you update the information, first click anywhere in the table. Then choose Database from the Insert menu, open the data source, and select the data. Choose the Table AutoFormat button to select a new table format, and then reinsert the table.

Inserting data from a Microsoft Access database You can insert information from any table or query you saved in a Microsoft Access database. If you select a saved query and are using Microsoft Access version 1.1 or later, you can specify how Word uses the query instructions. If you clear the Link To Query check box in the Microsoft Access dialog box (Queries tab), Word repeats the original query instructions each time the DATABASE field is updated. If you select the Link To Query check box, Word carries out the latest query instructions saved in the selected query each time the DATABASE field is updated. For more information, see "Linking to a Microsoft Access Query" in Chapter 30, "Mail Merge: Advanced Techniques."

Troubleshooting

The type of object I want to embed does not appear in the Object dialog box.
Object types are listed in the Object dialog box only if the original application was properly installed by using its setup (installation) program. Try reinstalling the source application, making sure that you use its setup program. For example, if you want to embed a Microsoft Excel worksheet and Microsoft Excel is not listed, reinstall Microsoft Excel by using the installation program in the original disk set. You should run the newly installed source application independently at least once before you try to embed an object from that application. On the Macintosh, rebuild the desktop.

Row and column numbers appear in links from Microsoft Excel.

If you copy cells from a Microsoft Excel worksheet and link them to a Word document as a picture or bitmap, the row and column numbers are included in the Word document even if you have not selected them in Excel. If you don't want these numbers in your Word document, do the following before you copy the object in Microsoft Excel:

1. From the Options menu in Microsoft Excel, choose Display.
2. Under Cells, clear the Row & Column Headings check box.
3. Choose the OK button.

I embedded an object and then copied and pasted it, but the copies are not being updated.

Embedding creates a single, independent object. When you copy and paste that object, you create a new, independent object. The copies are not connected, so changes in one are not reflected in the other. If you want multiple copies of an object that reflect changes made to the original object, you should create a link to the source file. First, create a separate source file. Then, in the Word document, create as many links to the source file as you want. Any changes you make to the source file will be reflected in each linked copy of the object.

Word displays a message that it cannot find the application needed to edit an embedded object.

The application may have been either deleted or moved to a different location. If the application is located on a network drive, the drive may no longer be connected. Reinstall the application, or reconnect the network drive.

Word cannot update a particular link.

There are several possible reasons why Word might not be able to update a link. Most commonly, it is because the source document has been moved, renamed, or deleted. If the source document has been deleted, you won't be able to update the link. If the source document has been moved or renamed, follow this procedure:

1. In the Word document, choose Links from the Edit menu.
2. Select the link you want to update, and then choose the Change Source button.
3. Locate the source file, and then choose the OK button.
4. In the Links dialog box, choose the Update Now button to update the link.

I resized an embedded object, but after I edited and updated the object, it resumed its original size.

Display the field codes by choosing Options from the Tools menu, selecting the View tab, and then selecting the Field Codes check box. The \s switch in the field code retains the original scaling and cropping information. Delete the \s from the field code.

For information about editing fields, see Chapter 32, "Inserting Information with Fields."

010. Examples of evidence with regard to network monitoring and control systems

provided using a single 64
rvices can be provided using
low bit rate speech vocoder

sult of video coding, can be
rks. Predicting the overall
h is discussed in Chapter 17.
are discussed in Chapters 18

sidered to be composed of 550
l is equiprobable among 64
aster scanning at a 25 Hz frame
ne minimum bandwidth required
on reception. [4.44 MHz]

he picture frames described in
l, as defined in Figure 5.12, with

y be considered to be composed
ith straightforward 24-bit colour
. Calculate the time required to

te for a television signal for
ines per frame = M, horizont
each pixel comprises one
tem which employs 819 lines
ne period and 100 levels

Part Four

Networks

Part Four is devoted to communication networks which now exist on all scales from geographically small LANs to the global ISDN.

It starts with a discussion, in Chapter 17, of queuing theory which may be used to predict the delay suffered by digital information packets as they propagate through a data network.

Chapter 18 describes the topologies and protocols employed by networks to ensure the reliable, accurate and timely, delivery of information packets between network terminals. Rings, buses and their associated medium access protocols are discussed, and international standards such as ISO OSI, X.25 and FDDI are described. The optical transmission medium, which now forms an integral part of many communications networks, is also examined.

Part Four ends, in Chapter 19, by examining public networks. The current plesiochronous digital hierarchy (PDH) is reviewed before introducing a more detailed discussion of the new synchronous digital heirarchy (SDH) which will gradually come to replace it. The Chapter concludes with a brief discussion of PSTN/PDN data access techniques including the ISDN standard, ATM, and the probable future development of the local loop.

CHAPTER 17

Queuing theory for packet networks

17.1 Introduction

In packet switching, secure bundles of information are assembled, addressed and transmitted through a network without the need for dedicated end-to-end connection paths to be established. The packets are individually transported and delivered by the network to the required destination. The network additionally ensures that packets are output in the correct order at the receiver.

Packet switched networks have existed for many years. Early examples were Euronet which links nine EC countries and Switzerland, running the Direct Information Access Network-Europe (DIANE) service. In 1981, British Telecom opened its first national public network, known as the Packet Switched Service (PSS). This system is controlled by a network management centre, based on duplicated minicomputers. PSS uses the X.25 protocol (see Section 18.6.1). The most well known network today is the Internet which supports the information retrieval service known as the World Wide Web (WWW).

The Joint Academic Network (JANET), Figure 18.2, and its successor SuperJANET are funded by the UK research councils. JANET has a node at every university and runs on leased lines. It provides a service for the communication of data, such as computer files and electronic mail, between sites and onward via the Internet (see section 18.7.6). All these networks introduce queuing, with consequent delays and possible loss of traffic data.

Queuing theory [Nussbaumer] can be used to model these or other networks where customers or data packets arrive, wait their turn for handling or service, are subsequently serviced, and are then transmitted through the network. (Supermarket checkouts, ticket booths, and doctors' waiting rooms are all commonly encountered examples of queuing systems.) Queuing theory was developed originally to model analogue teletraffic but is now widely applied to digital packet traffic [Tanenbaum]. A queuing system can be characterised by the following five attributes:

- The interarrival-time probability density function.
- The service-time probability density function.
- The number of servers, or server processes.
- The queuing discipline.
- The amount of buffer, or waiting, space in the queues.

The interarrival-time probability density function (pdf) describes the interval between consecutive arrivals. After a sufficiently long sampling time, the arrival times can be grouped to obtain the pdf which characterises the arrival process.

Each customer requires a certain amount of the server's time which varies from customer to customer. To analyse a queuing system, the service-time pdf, like the interarrival-time pdf, must be known.

The number of servers speaks for itself. Many banks, for example, have one queue for all customers. Whenever a teller is free, the customer at the front of the queue goes directly to that teller. Such a system is a multiserver queuing system. In other banks, each teller has his, or her, own private queue. This corresponds to a collection of independent single-server queues.

The queuing discipline describes the order in which customers are taken from the queue. Supermarkets use first come, first served. Hospital emergency rooms often use sickest attended to first. In friendly office environments, shortest job first often prevails at the photocopy machine. Not all queuing systems have an infinite amount of buffer space. When too many customers are queued up in a finite number of available slots, some customers can get lost or rejected.

This chapter concentrates predominantly on infinite-buffer, single-server systems using a first come first served queuing discipline. The Kendall notation A/B/m [Kleinrock] is widely used in the queuing literature for these systems. A represents the interarrival-time pdf, B the service-time pdf, and m the number of servers employed. The probability densities A and B are usually chosen from the following set:

M – Markov (implying an exponential pdf);
D – deterministic (all customers have the same constant value implying an impulsive pdf);
G – general (i.e. some arbitrary pdf);
E_k – Erlang distributed.

The state of the art ranges from the M/M/1 system, about which everything is known, to the G/G/m system, for which no exact analytical solution is yet available. We will concentrate on the M/M/1 model. Figure 17.1 shows the queue for such a single server process. We now need to develop a mathematical analysis for queuing systems which can be used to show what limits or restricts the practical performance of these systems. This is achieved by first examining arrivals only, before progressing to model the combined arrival and service processes within the queuing buffer memory.

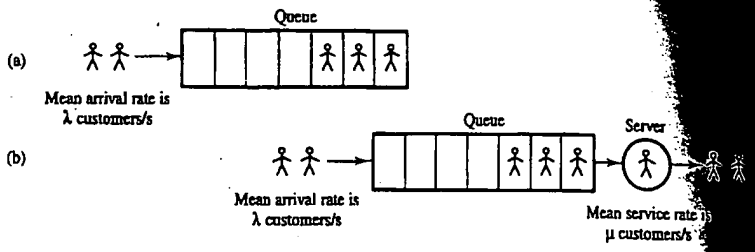


Figure 17.1 *Single server queue model and outcomes for a counting process: (a) arrivals only, (b) arrivals and departures.*

17.2 The arrival process

We make the following assumptions:

- The arrival process is memoryless in the sense that any arrival is statistically independent of all other arrivals;
- The arrival process is statistically stationary. (This implies that the probability of an arrival occuring in any small time interval depends only on the interval's width and not its location in time.)

Queuing systems in which the only transitions are to adjacent states are known as birth–death systems, which is a mathematician's terminology for a counting process with both arrivals and departures. We are considering, at present, *only* arrivals.

We have to model the evolution of the system from one state to another [Gelenbe and Pujolle] where the *state* is synonymous with the number of customers waiting for service. The approach via the Markov probability chain [Chung] is inappropriate, since the probability of transition between any two states at any given point in time, t, is zero. While we cannot characterise the *probability* of transition, we *can* characterise the *rate* of transitions between two states. Suppose that for two particular states the rate of transitions between them is a constant λ. What we mean by this is that in a time δt we can expect an average of $\lambda\ \delta t$ transitions. If δt is very small then $\lambda\ \delta t$ is a number much smaller than unity, and the probability of more than one transition in time δt is vanishingly small. Under these conditions, we can think of $\lambda\ \delta t$ as the probability of one transition (P_1) in time δt, and $(1 - \lambda\ \delta t)$ as the probability of no transition (P_0) in this time.

This leads us to a transition diagram and associated set of differential equations. The transition diagram, Figure 17.2, associates a node with each state. Within node j we denote the probability of being in that state at time t as $P_j(t)$.

17.2.1 pdf for j arrivals in t seconds

Assume arrivals (A) are governed by a randomly distributed pure birth process as shown in Figure 17.3 where the arrival rate does not depend on the state of the system. The

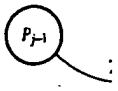


Figure 17.2 *Markov model*

Figure 17.3 *Example of*

only arrivals then the pr
and P_{j-1}:

$$P_j(t + \delta t)$$

hence:

$$\frac{P_j(t + \delta t)}{\delta t}$$

Now let $\delta t \to 0$:

$$\frac{dP_j(t)}{dt} =$$

For $j = 0$, $P_{j-1} = 0$, a

$$\frac{dP_0(t)}{dt} =$$

The solution to equat

$$P_0(t) =$$

which represents the
starting with a pro
exponential time co
derivation of the P
exponential interarr
shown that:

$$P_j(t)$$

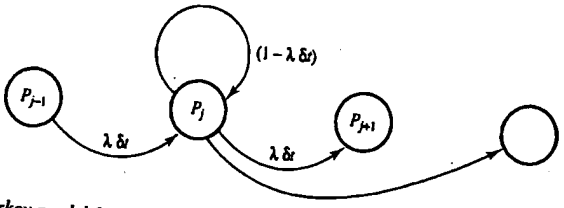


Figure 17.2 *Markov model for queue states corresponding to $j-1$, j, $j+1$ packet arrivals.*

Figure 17.3 *Example of randomly distributed arrivals (A) with interarrival time τ.*

only arrivals then the probability of being in state j after δt seconds is dependent on P_j and P_{j-1}:

$$P_j(t+\delta t) = P_j(t)\,(1-\lambda\,\delta t) + P_{j-1}(t)\,\lambda\,\delta t \tag{17.1}$$

hence:

$$\frac{P_j(t+\delta t) - P_j(t)}{\delta t} = \lambda(P_{j-1}(t) - P_j(t))$$

Now let $\delta t \to 0$:

$$\frac{dP_j(t)}{dt} = \lambda(P_{j-1}(t) - P_j(t)) \tag{17.2}$$

For $j = 0$, $P_{j-1} = 0$, as we cannot have less than zero arrivals. Therefore:

$$\frac{dP_0(t)}{dt} = -\lambda P_0(t) \tag{17.3}$$

The solution to equation (17.3) is:

$$P_0(t) = e^{-\lambda t} \tag{17.4}$$

which represents the probability of no arrivals in time t. This is plotted in Figure 17.4, starting with a probability of 1 at $t = 0$, decaying asymptotically to zero with an exponential time constant of λ. Thus at $t = 1/\lambda$, $P_0(1/\lambda) = 0.37$. This is the start of the derivation of the Poisson distribution for the number of arrivals j in t seconds for an exponential interarrival time distribution, with a mean arrival rate of λ. It can further be shown that:

$$P_j(t) = \frac{(\lambda t)^j}{j!}\,e^{-\lambda t} \tag{17.5(a)}$$

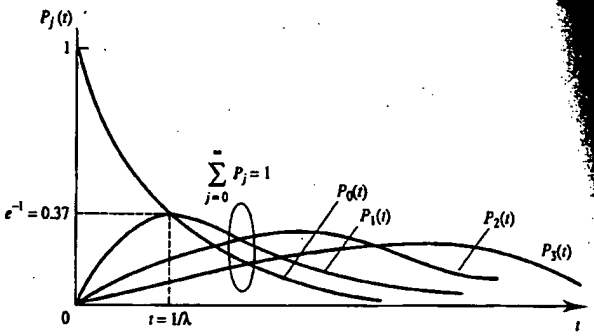


Figure 17.4 *Probability of j arrivals plotted against time for j = 0, 1, 2, 3.*

i.e.:

$$P_1(t) = \lambda t\, e^{-\lambda t} \qquad (17.5(b))$$

$$P_2(t) = \tfrac{1}{2}(\lambda t)^2\, e^{-\lambda t} \quad etc \qquad (17.5(c))$$

Figure 17.4 shows that the probability of only one arrival, $P_1(t)$, peaks at the time interval $t = 1/\lambda$. If Figure 17.4 is plotted with the horizontal axis as offered traffic, λt, then the peak value of $P_1(t)$ occurs at an offered traffic value of unity. For longer time intervals, it becomes increasingly likely that there will be more than one arrival. The probabilities of there being two or three arrival events, $P_2(t)$ and $P_3(t)$, peak at later time intervals and also have progressively smaller probability peaks, so that, for any specified time interval, all the probabilities sum to unity as required, i.e.:

$$\sum_{j=0}^{\infty} P_j(t) = 1 \qquad (17.6)$$

17.2.2 CD and pdf for the time between arrivals

If the time between successive arrivals is τ, as in Figure 17.3, the probability that τ is less than, or equal to, some value of time t, $P(\tau \le t)$, is given by:

$$P(\tau \le t) = 1 - P(\tau > t) \qquad (17.7(a))$$

But $P(\tau > t)$ is the probability of no arrivals in time t. Thus, for the Poisson process:

$$P(\tau > t) = P_0(t) = e^{-\lambda t} \qquad (17.7(b))$$

hence:

$$P(\tau \le t) = 1 - e^{-\lambda t} \qquad (17.7(c))$$

Equation (17.7(c)) is the cumulative distribution (CD) function, equations (3.10) and

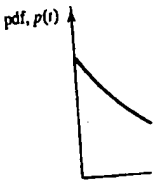


Figure 17.5 *(a) pdf; and (b*

(3.13(b)), for the time
interarrival probability d
obtained by differentiatin

$$P(t \le \tau < t$$

where the mean or avera
is $1/\lambda^2$ (see section 3.2.

17.2.3 Other arrival

These can be specified

- Unpunctual – whi
  random variable;
- Discrete-time arriv
- Non-stationary – v
- Correlated – when

17.3 The servi

The service or trans
which defines the ca
given traffic.

17.3.1 Service ti

As for arrival times
times or alternative
times, the latter aga

$$P(t) =$$

for a service rate

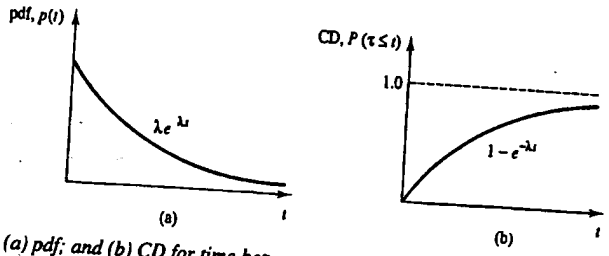


Figure 17.5 *(a) pdf; and (b) CD for time between successive arrivals.*

(3.13(b)), for the time between arrivals, Figure 17.5. This implies an exponential interarrival probability density function (pdf). As shown in section 3.2.4, the pdf is obtained by differentiating the CD, i.e. equation (17.7(c)):

$$P(t \le \tau < t+dt) = p(t) = \frac{dP(\tau \le t)}{dt} = \lambda e^{-\lambda t} \qquad (17.7(d))$$

where the mean or average value of t is $1/\lambda$ and the variance (or second central moment) is $1/\lambda^2$ (see section 3.2.5).

17.2.3 Other arrival patterns

These can be specified to be:

- Unpunctual – which occur at $t = a + E_1, 2a + E_2, \cdots, ma + E_m$, where E_m is a random variable;
- Discrete-time arrivals – these can only occur at a discrete set of allowed instants;
- Non-stationary – where the probabilities vary with time;
- Correlated – where the arrival rate may be affected by the state of the system.

17.3 The service process

The service or transmission mechanism is described by the service time distribution, which defines the capacity, or number of servers, which must be deployed to handle the given traffic.

17.3 1 Service time distributions

As for arrival times there are two extremes. We can have constant (deterministic) service times or alternatively we can have 'completely random' (stastically independent) service times, the latter again leading to an exponential pdf given by:

$$P(t) = \mu\, e^{-\mu t} \qquad (17.8)$$

for a service rate of μ customers/s, Figure 17.1. As before, the mean value, or average

service time, is $1/\mu$ and the variance is $1/\mu^2$. The statistical independence of succ... service times (resulting in an exponential distribution of service time pdf) means th... service time, like arrival interval, has been modelled as a Poisson process. It should be noted, however, that:

- service times may be discrete (word or packet multiples);
- service time may be non-stationary.

The queuing discipline determines how customers are selected from the queue and allocated to servers.

17.3.2 Single server queues

These typically use one of the following queuing disciplines:

- First-in-first-out (FIFO) – the simple queue;
- Last-in-first-out (LIFO or last come first served, LCFS);
- First-in-random-out (FIRO);
- Priority queuing.

17.3.3 Multiserver queues

Here service is allocated according to rules such as:

- Rotation – customers assigned in strict rotation to each queue;
- Random selection – customers themselves decide which queue to join;
- Single queue – customer at the head of queue goes to the next available server.

17.4 The simple single server queue

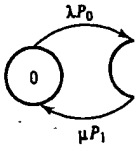
It is instructive to find the distribution of queue lengths and waiting times in an M/M/1 system, where total waiting time equals queuing time plus service time.

17.4.1 Simple queue analysis

We define the *state* of the system as the number of customers waiting (i.e. state m implies m customers waiting or a queue of length m), and denote the probability of being in a state m at time t as $P_m(t)$. Assume, when the system is in state m, customers arrive randomly at an average rate λ_m, and are randomly serviced at rate μ_m (i.e. the average service time is $1/\mu_m$).

What is the probability of such a system, Figure 17.6, being in state m at time $t + \delta t$? This is obtained by extending equation (17.1) to include departures as well as arrivals:

$$\begin{aligned} P_m(t+\delta t) &= P_{m-1}(t)\lambda_{m-1}\delta t + P_{m+1}(t)\mu_{m+1}\delta t + P_m(t)(1-\mu_m\delta t)(1-\lambda_m\delta t) \\ &= P_{m-1}(t)\lambda_{m-1}\delta t + P_{m+1}(t)\mu_{m+1}\delta t + P_m(t)[1-(\mu_m+\lambda_m)\delta t] \end{aligned} \quad (17.9)$$



...igure 17.6 *State transitio*

...for small δt). Therefore

$$\frac{P_m(t+\delta t)-1}{\delta t}$$

...in the limit as $\delta t \to 0$:

$$\frac{dP_m(t)}{dt}$$

...(for $m \geq 0$ where P_{-1}(... ...and Pujolle] for arrival ...with time for state m i... $m-1$ and $m+1$, are ...minus the rate at whic... probability of state m... The process starts in s...

$$P_m(t_0) =$$

...and assuming a statio...

$$0 = \lambda_m$$

where:

$$P_{-1} =$$

$$\lambda_{-1} =$$

$$\mu_0 =$$

and:

$$P_0 + P$$

A typical queue ... staircase occur wi... staircase imply s... displacement meas...

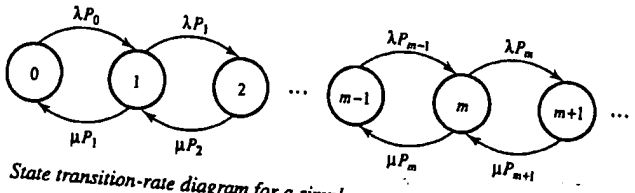


Figure 17.6 *State transition-rate diagram for a simple queue.*

(for small δt). Therefore:

$$\frac{P_m(t+\delta t)-P_m(t)}{\delta t} = \lambda_{m-1}P_{m-1}(t)+\mu_{m+1}P_{m+1}(t)-(\mu_m+\lambda_m)P_m(t) \quad (17.10)$$

In the limit as $\delta t \to 0$:

$$\frac{dP_m(t)}{dt} = \lambda_{m-1}P_{m-1}(t)+\mu_{m+1}P_{m+1}(t)-(\mu_m+\lambda_m)\,P_m(t) \quad (17.11)$$

(for $m \geq 0$ where $P_{-1}(t)=0$). This is the full Chapman-Kolmogorov equation [Gelenbe and Pujolle] for arrivals and departures. It states that the rate of increase of probability with time for state m is equal to the rate at which transitions into that state, from states $m-1$ and $m+1$, are occurring (multiplied by the current probability of those states) minus the rate at which transitions out of state m are occurring (multiplied by the current probability of state m). To solve equation (17.11) we must specify the initial conditions. The process starts in state zero, as there are no arrivals before time t_0, i.e.:

$$P_m(t_0) = \begin{cases} 1, & m=0 \\ 0, & m>0 \end{cases}$$

and assuming a stationary solution so that $dP_m(t)/dt = 0$ then:

$$0 = \lambda_{m-1}P_{m-1}+\mu_{m+1}P_{m+1}-(\mu_m+\lambda_m)\,P_m \quad (17.12(a))$$

where:

$$P_{-1} = P_{-2} = \cdots = 0 \quad (17.12(b))$$

$$\lambda_{-1} = \lambda_{-2} = \cdots = 0 \quad (17.12(c))$$

$$\mu_0 = \mu_{-1} = \cdots = 0 \quad (17.12(d))$$

$$P_0 + P_1 + P_2 + \cdots = 1 \quad (17.13)$$

A typical queue result is shown in Figure 17.7. Here the vertical steps in the solid staircase occur with new customers arriving while the vertical steps in the dashed staircase imply service has been completed for these customers. The horizontal displacement measures the queue plus service time for each customer or unique arrival.

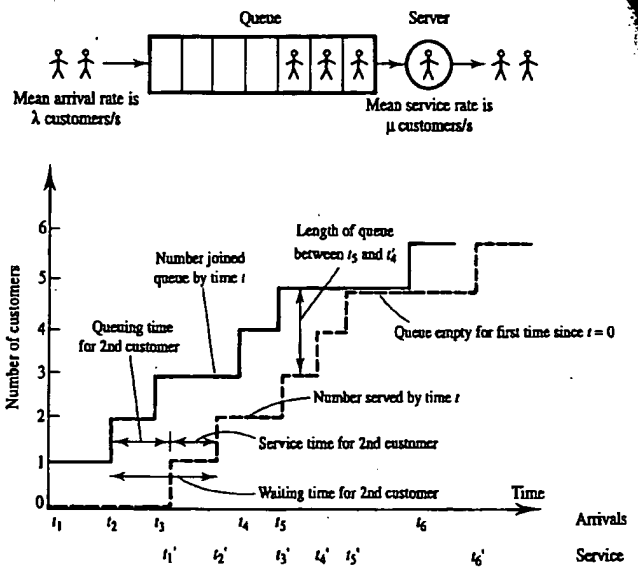


Figure 17.7 *Typical queue performance showing customer (packet) arrivals and departures (after service).*

17.4.2 Queue parameters

The total delay in a time t, $\gamma(t)$, is the sum of the waiting times. If the total number of customers who have arrived in a time t is denoted by $\alpha(t) = \lambda t$ then:

$$\text{average delay,} \qquad T = \frac{\gamma(t)}{\alpha(t)} \tag{17.14(a)}$$

$$\text{average queue length,} \qquad N = \frac{\gamma(t)}{t} \tag{17.14(b)}$$

$$\text{average arrival rate,} \qquad \lambda = \frac{\alpha(t)}{t} \tag{17.14(c)}$$

Now we can rewrite the average queue length as $N = \gamma(t)/t = (\gamma(t)/\alpha(t)) \times (\alpha(t)/t)$, where N is given by the sum, over all m, of the product of queue length, m, and its probability, P_m, to obtain Little's result:

$$N = \sum_{m=0}^{\infty} mP_m = T\lambda \tag{17.15}$$

The performance of a queuing system is controlled by its utilisation factor, defined as:

$$\rho = \frac{\text{demand for service}}{\text{maximum rate of supply}} \tag{17.16}$$

here the demand for service
easure of the traffic intensity
neer of teletraffic theory. (
qual to $\lambda\tau$ where λ is the a
uration. A circuit carrying or
The service rate is often c
ength. For example a 2 Mbi
as a service rate, $\mu = (2 \times$
naximum capacity or rate of
In this case:

$$\rho = \lambda/\mu$$

and $\rho < 1$ is required to pre

17.4.3 Classical queue

Assume a Poisson arrival
arrival and service rates ar
λ and μ_m simplifies to μ.
customers are served on F
From equation (17.5)
detailed balancing, the in
same rate. Thus starting

$$P_1 = \frac{\lambda}{\mu} P_0$$

$$P_2 = \frac{\lambda^2}{\mu^2}$$

$$P_m = \left(\frac{\lambda}{\mu}\right.$$

But $\sum_{m=0}^{\infty} P_m = 1$, as
$\sum_{m=0}^{\infty} P_m = \sum_{m=0}^{\infty} \rho^m P_0$
$\sum_{m=0}^{\infty} \rho^m = 1/(1-\rho)$.

$$P_0 = 1$$

Figure 17.8 (curve
empty queue, P_0, de

$$P_m =$$

implying a geometr
is $1 - P_0 = \rho$. $\rho <$
17.9 shows a typic

where the demand for service = arrival rate × mean service time = λ/μ. This is also a measure of the traffic intensity in erlangs [Dunlop and Smith], named after the Danish pioneer of teletraffic theory. (In telephony systems the total applied traffic in erlangs is equal to $\lambda\tau$ where λ is the arrival rate for call connections and τ is the average call duration. A circuit carrying one call continuously then carries one erlang of traffic.)

The service rate is often controlled directly by the output transmission rate and packet length. For example a 2 Mbit/s link (Chapter 19) with 500 byte packets and 8-bit bytes, has a service rate, $\mu = (2\times10^6)/(500\times8) = 500$ packet/s. For single server queues, maximum capacity or rate of supply is 1 s of service/s, i.e. the maximum rate of supply = 1. In this case:

$$\rho = \lambda/\mu \tag{17.17}$$

and $\rho < 1$ is required to prevent server overload.

17.4.3 Classical queue with single server

Assume a Poisson arrival process and exponentially distributed service times so that the arrival and service rates are independent of the state of the system. Thus λ_m simplifies to λ and μ_m simplifies to μ. Further assume that infinite queuing space is available and that customers are served on FIFO basis.

From equation (17.5) and Figure 17.6, by applying the conditions of equilibrium or detailed balancing, the input and output transitions to and from a state must occur at the same rate. Thus starting with state zero, $\lambda P_0 = \mu P_1$, these balances can be written as:

$$P_1 = \frac{\lambda}{\mu} P_0$$

$$P_2 = \frac{\lambda^2}{\mu^2} P_0$$

$$P_m = \left(\frac{\lambda}{\mu}\right)^m P_0 = \rho^m P_0 \tag{17.18}$$

But $\Sigma_{m=0}^{\infty} P_m = 1$, as shown previously (Figure 17.4 and equation (17.6)). Now $\Sigma_{m=0}^{\infty} P_m = \Sigma_{m=0}^{\infty} \rho^m P_0 = P_0 \Sigma_{m=0}^{\infty} \rho^m = 1$ and the sum of the geometric series $\Sigma_{m=0}^{\infty} \rho^m = 1/(1-\rho)$. Therefore:

$$P_0 = 1 - \rho = 1 - \frac{\lambda}{\mu} \tag{17.19}$$

Figure 17.8 (curve (a)) shows, as ρ increases from 0 to 1, how the probability of an empty queue, P_0, decreases. Also by applying equation (17.18):

$$P_m = \rho^m P_0 = \rho^m(1-\rho) \tag{17.20}$$

implying a geometric distribution for P_m. The probability of the single server being busy is $1 - P_0 = \rho$. $\rho < 1$ ensures that the server has more capacity than is required. Figure 17.9 shows a typical queue length pdf, for $\rho = 0.5$.

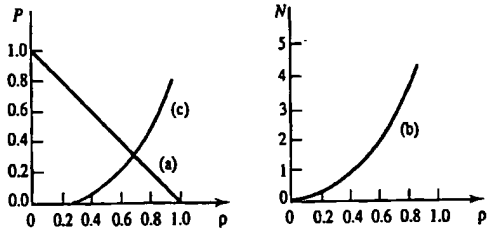


Figure 17.8 *Average queue size or length: (a) probability of an empty queue; (b) mean queue length, N, in packets; (c) probability that queue length exceeds 4.*

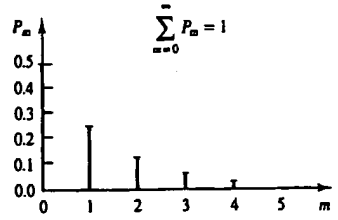


Figure 17.9 *Queue length pdf for ρ = ½.*

EXAMPLE 17.1

For a single server queuing system with Poisson distributed arrivals of average rate 1 message/s and Poisson distributed service of capacity 3 messages/s calculate the probability of receiving no messages in a 5 s period. Also find the probabilities of queue lengths of 0, 1, 2, 3. If the queue length is limited to 4 what percentage of messages will be lost?

From equation (17.5(a)):

$$P_0(t) = e^{-\lambda t}$$

and $\lambda = 1$ and $t = 5$ for a 5 s period. Thus the probability of no arrivals in a 5 s period is given by:

$$P_0(5) = e^{-5} = 0.00674$$

Now from equation (17.18):

$$P_m = \left(\frac{\lambda}{\mu}\right)^m P_0$$

and from equation (17.19):

$$P_0 = 1 - \frac{\lambda}{\mu} = 1 - \frac{1}{3} = \frac{2}{3}$$

Thus the various queue lengt

$$P_1 = \frac{1}{3} \times \frac{2}{3} =$$

$$P_2 = \left(\frac{1}{3}\right)^2 \times \frac{2}{3}$$

$$P_3 = \left(\frac{1}{3}\right)^3 \times \frac{2}{3}$$

This differs slightly from subsequent magnitudes fall of exceeding this restricted

$$P(m > 4) =$$

$$=$$

17.4.4 Queue lengt

The average queue len as the sum of the geon by:

$$N = \sum_{m=}^{\infty}$$

Figure 17.8 (curve (b for $\rho > ½$, N increas queue length becom probability of exceed

$$P(m > 1$$

This is shown in F about 0.8 there is

Thus the various queue lengths can be calculated as:

$$P_1 = \frac{1}{3} \times \frac{2}{3} = \frac{2}{9}$$

$$P_2 = \left(\frac{1}{3}\right)^2 \times \frac{2}{3} = \frac{2}{27}$$

$$P_3 = \left(\frac{1}{3}\right)^3 \times \frac{2}{3} = \frac{2}{81}$$

This differs slightly from Figure 17.9 in that the P_0 value is larger and, in consequence, the subsequent magnitudes fall off more rapidly. Finally for a queue length limited to 4 the probability of exceeding this restricted length is given by:

$$P(m > 4) = 1 - P(m \leq 4) = 1 - (P_0 + P_1 + P_2 + P_3 + P_4)$$

$$= 1 - \left(\frac{2}{3} + \frac{2}{9} + \frac{2}{27} + \frac{2}{81} + \frac{2}{243}\right) = 0.0041 = 0.41\%$$

17.4.4 Queue length and waiting times

The average queue length, equation (17.15), $N = \Sigma_{m=0}^{\infty} m P_m = (1-\rho)\Sigma_{m=0}^{\infty} m\rho^m$. Further, as the sum of the geometric series $\Sigma_{m=0}^{\infty} m\rho^m = \rho/(1-\rho)^2$ the mean queue length is given by:

$$N = \sum_{m=k+1}^{\infty} P_m = \frac{\rho}{1-\rho} \tag{17.21}$$

Figure 17.8 (curve (b)) shows how N increases with increasing ρ. At $\rho = ½$, $N = 1$ and for $\rho > ½$, N increases above unity. As traffic intensity increases and ρ approaches 1 the queue length becomes infinite. If the queue is restricted to some finite value k then the probability of exceeding this value is given by:

$$P(m > k) = \sum_{m=k+1}^{\infty} P_m$$

$$= (1-\rho) \sum_{m=k+1}^{\infty} \rho^m$$

$$= (1-\rho)\left[\sum_{m=0}^{\infty} \rho^m - \sum_{m=0}^{k} \rho^m\right]$$

$$= (1-\rho)\left[\frac{1}{1-\rho} - \frac{1-\rho^{k+1}}{1-\rho}\right]$$

$$= \rho^{k+1} \tag{17.22}$$

This is shown in Figure 17.8 (curve (c)) for various values of ρ. Clearly when ρ exceeds about 0.8 there is a problem. To find average delay or waiting time, T, we use Little's

result, equation (17.15) and equation (17.21):

$$T = \frac{\text{average queue length}}{\text{average arrival rate}} = \frac{N}{\lambda} = \frac{\rho}{\lambda(1-\rho)} \qquad (17.23(a))$$

or, using equation (17.17):

$$T = \frac{1}{\mu(1-\rho)} = \frac{1}{\mu-\lambda} \qquad (17.23(b))$$

Average delay (normalised by μ) is plotted in Figure 17.10, against ρ in the range $0 \leq \rho < 1$, and it is this key result which forms the basis of network delay analysis. When constrained by a finite queue length of k, then $\Sigma_{m=0}^{k} P_m = 1 = P_0 \Sigma_{m=0}^{k} \rho^m$. For this case $P_0 = (1-\rho)/(1-\rho^{k+1})$.

For packets transmitted over a link, at a bit rate of R_b bit/s with packet size K bits, the mean packet delay is, from equation (17.23(b)):

$$T = \frac{1}{R_b/K - \lambda} \qquad (17.24)$$

where R_b/K represents the packet transmission or service rate, μ, in packet/s and λ is the packet arrival rate.

EXAMPLE 17.2

Consider a switch at which packets arrive according to a Poisson distribution. The mean arrival rate is 3 packet/s. The service time is exponentially distributed with a mean value of 100 ms. Assume the packet comprises 70 8-bit bytes and the output transmission rate is 5.6 kbit/s. How long does a packet have to wait in the queue?

The mean service rate is $\mu = 5600/(8 \times 70) = 10$ packet/s. From equation (17.23(b)):

$$T = 1/(\mu(1-\rho)) = 1/(\mu-\lambda)$$

and we find that the mean packet delay is $T = 0.143$ s for queuing plus service. Since the mean service time is 100 ms, the mean queuing time is therefore 143 – 100 = 43 ms.

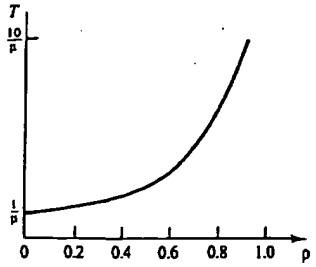


Figure 17.10 *Average time delay (T) for queue against ρ.*

Calculations, such a
links. The mean overa
times their delay divide
analysis is also used fo
Chapter 18 in order to f

17.5 Packet spe

Early packet networks
- store-and-forward,
(with a channel cap
- satellite networks,
kbit/s);
- radio networks, e.g
kbit/s, for local dat

The most general
system (Figure 17.11),
packets are switched
each packet. The inter
a host computer, or a
full range of control an

17.5.1 The compon

Analogue speech mu
waveform processing,
as LPC, section 9.7. T
speech must be compr
(1.7 to 7.4) than delta
kbit/s.

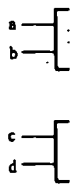


Figure 17.11 *Conventi*
example,

Calculations, such as those shown above, allow us to model the delays on packet data links. The mean overall network delay is given by the sum of the transmitted packets times their delay divided by the total number of transmitted packets. (This type of queue analysis is also used for determining the performance of the Banyan switch networks in Chapter 18 in order to find their blocking performance.)

17.5 Packet speech transmission

Early packet networks for communication (see Chapter 18) were:

- store-and-forward, hard-wired, long-haul networks, e.g. the American ARPANET (with a channel capacity of 50 kbit/s), and the British SuperJANET (Chapter 18);
- satellite networks, e.g. the American Atlantic SATNET (with a channel capacity of 64 kbit/s);
- radio networks, e.g. the American PRNET (with a channel capacity of 100 to 400 kbit/s, for local data distribution).

The most general distinction is that, unlike the circuit switched TDMA telephone system (Figure 17.11), in a packet switched network the individual data (D) and voice (V) packets are switched (Figure 17.12) in accordance with the header information within each packet. The interface between the user and a packet network can be a data terminal, a host computer, or a packet voice terminal which comprises a telephone handset with a full range of control and signalling capabilities.

17.5.1 The components of packet speech

Analogue speech must first be converted into a digital sequence by a coder using waveform processing, e.g. delta modulation, Figure 5.28, or other coding techniques such as LPC, section 9.7. The vocoder is favoured when there is limited channel capacity and speech must be compressed to lower data rates than PCM. LPC produces fewer packet/s (1.7 to 7.4) than delta modulation which generates 9.4 to 38.5 packet/s at a bit rate of 16 kbit/s.

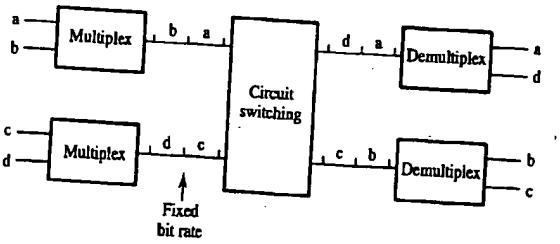


Figure 17.11 *Conventional circuit switched network (e.g. TDMA digital telephony transmission example, as described in section 6.5).*

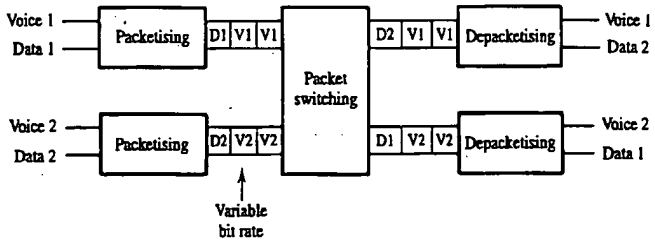


Figure 17.12 *Packet switched network with variable rate traffic where voice rate exceeds the input data rate.*

The digital bit stream is next partitioned into segments and some control (header) information added to each segment to form a packet. The control information consists of a time stamp and a sequence number, Figure 17.13, to assist reconstruction at the receiver. To reduce the bandwidth required for speech transmission, silences between bursts of speech are not packetised or transmitted. The time stamp enables the receiver to generate the appropriate duration of silence before processing the subsequent speech samples. Time stamps also allow reordering of packets which are received out of order. Since the time stamp cannot differentiate silence from packet loss, a sequence number is included to allow detection of lost packets.

It is important that a continuous stream of bits is provided at the receiver in order to produce smooth speech. This is achieved by delaying the arriving packets at the receiver queueing buffer, Figure 17.14. The size of this buffer must be sufficiently long to avoid packet loss due to overflow. Normally the delay will be chosen so that, statistically, a large proportion (e.g. 95%) of packets will be expected to arrive in the time allocated to the buffer delay, which is usually in the range 100 to 170 ms. The delay must not be so long that it dominates the performance of the speech transmission system, however.

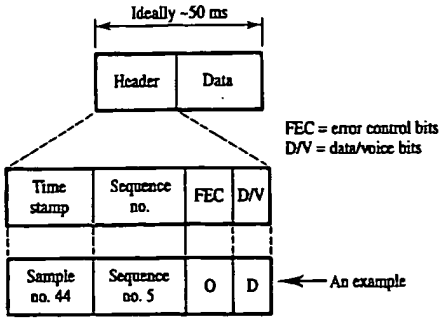


Figure 17.13 *Details of the header information carried within a data packet.*

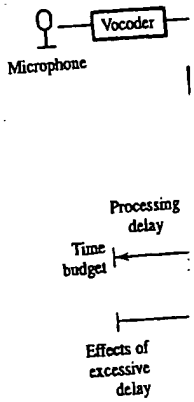


Figure 17.14 *Packet speec*

EXAMPLE 17.3

An X.25 packet switch has
each packet is 960 bytes.
queue, is to be less than 1
(ii) the average length of t
the input is converted into
19.7.2 for an explanation c

In the X.25 switch the ou
service time is equal to pa
(i) Now if $T \leq 15$ ms, fr

$$T = \frac{1}{\mu(1 -$$

Maximum gross inpu

(ii) Average length of qu

$$N = \rho/(1 -$$

(iii) ATM switch
Each packet is 960
The cell input rate i
Now the output rate

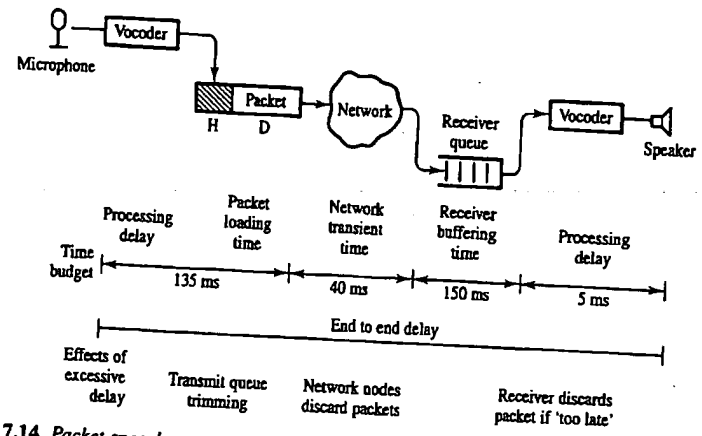


Figure 17.14 *Packet speech transmission system with inherent transmission delays.*

EXAMPLE 17.3

An X.25 packet switch has a single outgoing transmission link at 2 Mbit/s. The average length of each packet is 960 bytes. If the average packet delay through the switch, assuming an M/M/1 queue, is to be less than 15 ms, determine: (i) the maximum gross input packet rate to the switch; (ii) the average length of the queue; (iii) the utilisation factor through the switch if each packet in the input is converted into ATM cells having 48 bytes of data and a 5 byte overhead (see section 19.7.2 for an explanation of ATM).

In the X.25 switch the outgoing link is 2 Mbit/s. An average packet is 960 bytes or 7680 bits. If service time is equal to packet length, then: $\mu = 2 \times 10^6/7680 = 260.4$ packet/s

(i) Now if $T \leq 15$ ms, from equation (17.23(b)):

$$T = \frac{1}{\mu(1-\rho)} = \frac{1}{260.4(1-\rho)} \leq 15 \times 10^{-3} \quad \text{and} \quad \rho_{min} = 0.744$$

Maximum gross input rate is $\lambda = \rho\mu = 0.744 \times 260.4 = 193.7$ packet/s.

(ii) Average length of queue from equation (17.21) is:

$$N = \rho/(1-\rho) = 2.91 \text{ packets}$$

(iii) ATM switch

Each packet is 960 bytes, and corresponds to 960/48 = 20 ATM cells.

The cell input rate is therefore $20 \times 193.7 = 3{,}874$ cell/s.

Now the output rate is $2 \times 10^6/(48+5) \times 8 = 4{,}717$ cell/s. Therefore $\rho = 3874/4717 = 0.821$.

17.5.2 Speech service performance

Three key parameters that describe the performance of a packet speech service are end-to-end delay, throughput and reliability. Delay is the time between speech being presented to the system, to the packet carrying that speech being played at the loudspeaker. Experimental tests show that few people notice any quality degradation if delay is kept below 0.3 s while a delay above 1.5 s is intolerable. Throughput is limited by the processing capability of the nodes. Reliability is defined as the proportion of packets that arrive at the destination in time to be used to reconstruct the speech.

Appropriate choice of packet size and rate can minimise delay and allow high throughput. In particular we must control the number of bits in the message header. Since the header is constant for every packet, regardless of size, to maintain high channel utilisation the number of speech bits per packet should be maximised. Large packets are also more desirable from the point of view of network throughput. However, to minimise the effects of lost packets and delay at the transmitter, packets should be short and, ideally, a packet should contain no more than 50 ms of speech. The trade-off is particularly difficult for narrowband speech, e.g. using LPC, because 50 ms of 2.4 kbit/s speech comprises only 120 data bits. Typical packet size for speech transmission across the Internet is about 300 bits comprising 100 to 170 ms speech segments. Channel loading information should be provided to the voice terminal so that packet rate and size can be varied according to the network load. In cases where the network is lightly loaded, it is capable of supporting a higher packet rate, hence smaller packets with less delay are used while, if the network loading is high, packets are made larger and packet rate is reduced.

17.6 Summary

Packets are groups of data bits to which have been added (as headers and/or trailers) addressing and other control information to facilitate routing through a digital data network. Queuing theory can be used to model packet behaviour at the switching nodes of a network and, in particular, can be used to predict average packet delay, average queue length and probability of packet loss at a node, given the packet interarrival-time pdf, the packet service-time pdf, the number of servers, the queuing discipline and the queuing storage space. For real-time applications, such as packet speech, resources can be saved by not transmitting empty or 'silent' packets. This necessitates packets being time stamped, however, for the receiver to regenerate the appropriate speech gaps.

In current Ethernet based networks data rates are in the range 10 to 100 Mbit/s and propagation delays (typically less than 5 μs) are small compared with the time required to transmit a 1 kbit packet. With the trend towards Gbit/s optical fibre links, operating over long distances, propagation delay becomes much longer than the packet duration, which will significantly alter the analysis of these systems.

17.7 Problems

17.1. The number of mes
network may be assumed
messages per minute, calcul
(a) Probability of receiving
(b) Probability of receiving.
(c) Probability of receiving

17.2. In Problem 17.1: (
messages of greater than 2(
range 20 to 30 s inclusive?

17.3. Assuming a classical
empty queue, (b) a queue
for the service is 0.6. [0.4,

17.4. If the queue length i
the service? [0.36%]

17.5. A packet data netw
data. It is realised with a
London and Southend. Bo
the Northampton and Sou
50 packets/s to Northamp
mean size of 2 kbit? [1.2

17.7 Problems

17.1. The number of messages arriving at a particular node in a message switched computer network may be assumed to be Poisson distributed. Given that the average arrival rate is 5 messages per minute, calculate the following:
(a) Probability of receiving no messages in an interval of 2 minutes. [4.5×10^{-5}]
(b) Probability of receiving just 1 message in the next 30 s. [0.2]
(c) Probability of receiving 10 messages in any 30 s period. [2.16×10^{-4}]

17.2. In Problem 17.1: (a) What is the probability of having a gap between two successive messages of greater than 20 s? (b) What is the probability of having gaps between messages in the range 20 to 30 s inclusive? [0.189, 0.106]

17.3. Assuming a classical queue with a single server, determine the probabilities of having: (a) an empty queue, (b) a queue of 4 or more 'customers'. You should assume that the utilisation factor for the service is 0.6. [0.4, 0.13]

17.4. If the queue length in Problem 17.3 is limited to 10, what percentage of customers are lost to the service? [0.36%]

17.5. A packet data network links London, Northampton and Southend for credit card transaction data. It is realised with a 2-way link between London and Northampton and, separately, between London and Southend. Both links operate with primary rate access at 2 Mbit/s in each direction. If the Northampton and Southend nodes send 200 packets/s to each other and the London node sends 50 packets/s to Northampton what is the mean packet delay on the network when the packets have a mean size of 2 kbit? [1.275 ms]

CHAPTER 18

Network topology and protocols

18.1 Introduction

In 1969, the first major packet-switched communications network, the ARPANET, began operation. The network was originally conceived by the Advanced Research Projects Agency (ARPA) of the US Department of Defense for the interconnection of dissimilar computers, each with a specialised capability. Today systems range from small networks interconnecting microcomputers, hard-disks and laser-printers in a single room (e.g. Appletalk), through terminals and computers within a single building or campus (e.g. Ethernet), to large geographically distributed networks spanning the globe, e.g. the Internet. They are often classified as local, metropolitan or wide area networks (LANs, MANs or WANs). Figure 18.1 shows the relationships between LANs, MANs, WANs, the 'plain old telephone system' (POTS) and other more recent types of network. The major features of LANs, MANs and WANs are summarised in Table 18.1, after [Smythe 1991].

UK examples of WANs are the BT packet switched service (PSS) and the new joint academic network (SuperJANET). The original JANET interconnected all UK university

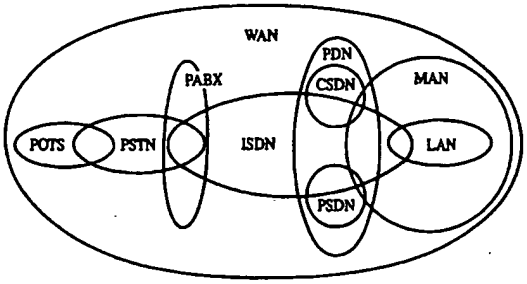


Figure 18.1 *Relationships between network architectures.*

...ANs with a 5000 km l
...ividual university I
...ogressively increased
...Manchester, Edinburgh,
...and 64 kbit/s access rat
...rate to 34 and 140 M
...SuperJANET Phase B
...tandard SDH interfac
...equivalent US system is

Table

Issue
Geographic
No. of node
R_b
P_b
Delays
Routing
Linkages

The general distin
network end-to-end
ratio is close to unity
unity. In the loosely
Broadly speaking

- Circuit-switched:
of communicatin
communications
of the PSTN. C
the circuit must
towards packet s
in section 19.7.2
- Message-switch
stored and forw
pairs are made
and are broken
path need there
- Packet-switche
are then route
forwarded at e
packets at the
circuit packet
network, and

LANs with a 5000 km backbone. With 100 Mbit/s transmission rates available on the individual university LANs the JANET backbone transmission rate has been progressively increased from 9.6 kbit/s to 2 Mbit/s. The main centres (London, Manchester, Edinburgh, etc.) were interconnected at 2 Mbit/s while other sites had 9.6 and 64 kbit/s access rates. The SuperJANET improvement upgraded the backbone bit rate to 34 and 140 Mbit/s using ATM (see section 19.7.2). Figure 18.2 shows the SuperJANET Phase B network, in which access rates are multiples of the 51.8 Mbit/s standard SDH interfaces (see section 19.4), within the PSTN core network. The equivalent US system is NSFnet which operates at 45 Mbit/s.

Table 18.1 *Comparison of LAN/MAN/WAN characteristics.*

Issue	*WANs*	*MANs*	*LANs*
Geographic size	1000s km	1 – 100 km	0 – 5 km
No. of nodes	10,000s	1 – 500	1 – 200
R_b	0.1 – 100 kbit/s	1 – 100 Mbit/s	1 – 100s Mbit/s
P_b	$10^{-3} - 10^{-6}$	$<10^{-9}$	$< 10^{-9}$
Delays	>0.5 s	100 – 100s ms	1 – 100 ms
Routing	sophisticated	simple	none
Linkages	gateways	bridges	bridges

The general distinction between a LAN, MAN and WAN depends on the ratio of the network end-to-end propagation delay to the average packet duration. For a MAN this ratio is close to unity while the more closely coupled LAN has a ratio much smaller than unity. In the loosely coupled WAN, the ratio is much larger than unity [Smythe 1991].

Broadly speaking, networks can be divided into three main categories:

- Circuit-switched: in which a continuous physical link is established between the pair of communicating data terminal equipments (DTEs) for the entire duration of the communications session. Circuit switched networks most commonly utilise portions of the PSTN. Circuit switching is inefficient for variable bit rate transmission since the circuit must always support the highest data rate expected – hence the move towards packet switching and asynchronous transfer mode, ATM. (ATM is discussed in section 19.7.2.)
- Message-switched: in which the complete message (of any reasonable length) is stored and forwarded at each data network node. Physical connections between node pairs are made only for the duration of the message transfer between those node pairs and are broken as soon as the message transfer is complete. No complete physical path need therefore exist between communicating DTEs at any time.
- Packet-switched: in which each message is divided into many standard packets which are then routed individually through the network. Each packet is stored and forwarded at each network node. Messages are reassembled from their constituent packets at the receiving DTE. Two distinct varieties of packet switching exist: virtual circuit packet switching in which all packets follow the same route through the network, and datagram packet switching in which different packets (within the same

Figure 18.2 *SuperJANET wide area network (WAN).*

message) are routed entirely independently. Virtual circuit systems ensure that packets are received in their correct chronological order whilst datagram systems must include packet sequencing information for correct message reassembly.

Hybrids of, and variations on, the above switching philosophies are also sometimes used.

One of the major activities which accelerated the development of packet-switched technology to its present state was the development of the layered communications architecture concept. The proliferation of various architectures, creating possible barriers between different manufacturers' systems, led the International Standards Organisation (ISO) to launch the reference model for Open Systems Interconnection (OSI). This standardised the systems interfaces.

18.2 Network topologies and examples

Any network [Hoiki] must fundamentally be based on some interconnection topology, to link its constituent terminals. The main network topologies are reviewed here.

18.2.1 Point-to-point

This is undoubtedly the si
ransitory and exist only f
may exist permanently, a
sed when a limited numt

18.2.2 Multidrop

When a large number of
are required to be conne
All transmissions from
node A can receive data
one time over the netwo
enforced by employing
permitting only the addr

Multidrop connectio
a single branched circui
this topology is the con
several locations. Mult
replacement of the PS
optical network (PON).

18.2.3 Star

Centralised switched-s
the PSTN and, for this
the star configuration
point links, to a centr
limitations:

Figure 18.3 *Multidr*

18.2.1 Point-to-point

This is undoubtedly the simplest wired network type, and is extensively used. It may be transitory and exist only for the duration of the call, as on the circuit-switched PSTN, or may exist permanently, as on a private (leased) line. This configuration is commonly used when a limited number of physically distinct routes are required.

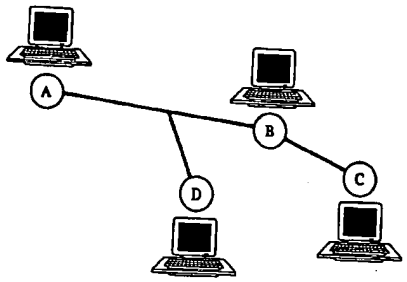
18.2.2 Multidrop

When a large number of locations, which can be partitioned into geographical clusters, are required to be connected the multidrop configuration is often employed, Figure 18.3. All transmissions from node A can be received by nodes B, C and D. However, only node A can receive data from B, C and D, and only one of the latter may transmit at any one time over the network, as there is only one data transmission path. This constraint is enforced by employing a polling protocol at A which addresses B, C and D in turn, permitting only the addressed node to reply.

Multidrop connection provides a way of reducing transmission link costs by utilising a single branched circuit to connect A to B, C and D. The principal current application of this topology is the connection of host computers to terminals – or terminal clusters – at several locations. Multidrop connection is under consideration, however, for optical fibre replacement of the PSTN local loop copper connection (section 19.7.3) with a passive optical network (PON).

18.2.3 Star

Centralised switched-star network configurations have now existed for over a century in the PSTN and, for this reason, represent perhaps the best understood class of network. In the star configuration, the devices comprising the network are connected by point-to-point links, to a central node or computer, Figure 18.4. The star network has two major limitations:



igure 18.3 *Multidrop configuration.*

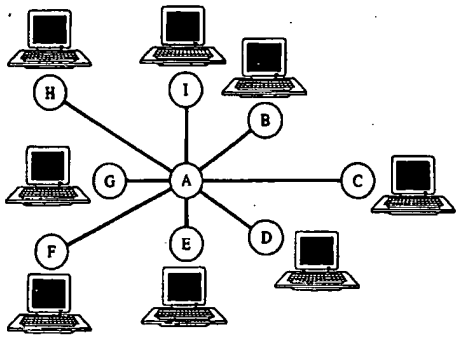


Figure 18.4 *Basic star network.*

- The remote devices are unable to communicate directly and must do so via the central node, which is required to switch these transmissions as well as carrying out its primary processing function.
- Such a network is very vulnerable to failure, either of the central node, causing a complete suspension of operation, or of a transmission link. Therefore, reliability and/or redundancy are particularly important considerations for this topology.

Despite these limitations, the star configuration is important as it has been used extensively for telephone exchange connections and in fibre optic systems. (Until recently it has been difficult to realise cheap, passive, optical couplers for implementation of an optical fibre ring or bus system, hence the attraction of the star network.) This topology is also used for many VSAT networks, section 14.3.9.

18.2.4 Ring

This network consists of a number of devices connected together to form a ring, Figure 18.5. Ring networks employ broadcast transmission, in that messages are passed around the ring from device to device. Each device receives each message, regenerates it, and retransmits it to its neighbour. The message is only retained, however, by the device to which it was addressed. Two variations of the ring network exist. These are:

- Unidirectional: in which messages are passed between the nodes in one direction only. The host, A, controls communication using a mechanism known as 'list polling'. The failure of a single data link will then halt all transmissions.
- Bidirectional: in which the ring is capable of supporting transmission in both directions. In the event of a single data link failing, the host, A, can then maintain contact with the two sectors of the network.

That each network node is involved in the transmission of all data on the network is a potential weakness. The ring topology is simple both in concept and implementation, however, and is popular for fibre optic LANs in which regenerative repeaters are required

Figure 18.5 *Ring networ*

at each node. Access i
follows.

A token bit pattern
data using a technique
'captures' the token b
'possessing' the token
in the ring acting as a
its header which is rec
data and signifies rec
trailer. It then retran
with the response bits
around the ring.

Cambridge ring

The Cambridge ring
data packets – slots
full/empty indicator
transmit a node occ
placing its message
before the sending
variation is to allow

The CR82 Cam
ring using twisted
monitor station che

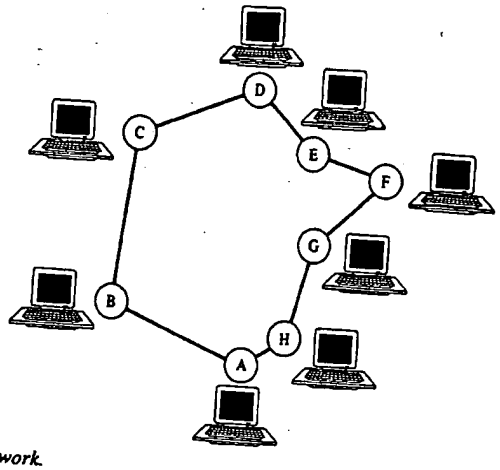


Figure 18.5 *Ring network.*

at each node. Access is via slots or tokens. A simple token ring operates essentially as follows.

A token bit pattern (e.g. 11111111), which is prevented from appearing in genuine data using a technique called bit stuffing, circulates while the ring is idle. A terminal 'captures' the token by removing it, or altering it (e.g. to 11111110). The terminal 'possessing' the token can then transmit one or more packets around the ring, each node in the ring acting as a repeater. Each transmitted packet contains a destination address in its header which is recognised by the destination terminal. The destination node reads the data and signifies receipt by setting response (or acknowledgement) bits in the packet trailer. It then retransmits the packet. When the sending terminal receives the packet with the response bits correctly set it resets the token to the idle pattern and recirculates it around the ring.

Cambridge ring

The Cambridge ring employs the empty slot principle. A constant number of fixed-length data packets – slots – circulate continuously around the ring in one direction only, a full/empty indicator within the slot header being used to signal the state of the slot. To transmit a node occupies the first empty slot by setting the full/empty flag to full and placing its message in the slot. The data packet then completes one revolution of the ring before the sending node 'empties' the slot and resets the indicator to empty. (A minor variation is to allow the receiver to empty the slot.)

The CR82 Cambridge ring, Figure 18.6, is a baseband implementation of a slotted ring using twisted wire pair cable, a bit rate of 10 Mbit/s and, typically, 10 slots. A monitor station checks which stations are active, and fills dummy slots to confirm that the

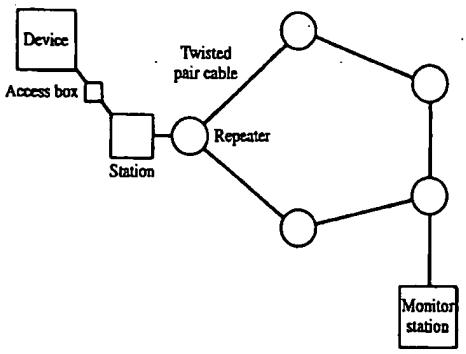


Figure 18.6 *Cambridge backbone ring example.*

ring is unbroken. With optical fibre implementations the data rate can be increased to 600 Mbit/s and beyond.

18.2.5 Mesh

While the star configuration is best suited for host computer/slave terminal connections on a one-to-many basis, mesh networks, Figure 18.7, are primarily used in a many-to-many situation, such as typically exists in WANs. Fully interconnected mesh networks, for more than a small number of nodes, are generally expensive as they require a total of $\frac{1}{2}n(n-1)$ links for n nodes. They are very resilient to failure, however, since alternative routes are available if a link fails. Where link lengths are long or data volumes low, a public packet-switched service may offer a significant cost advantage over a private mesh network. Unlike the ring or star topologies, adding a node to an existing (n node) mesh

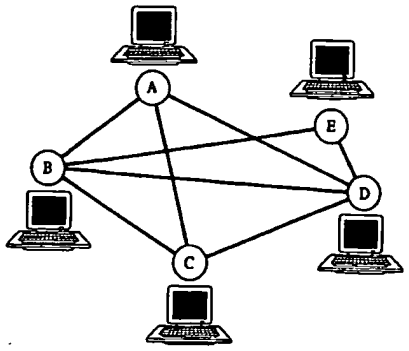


Figure 18.7 *Mesh network.*

...work necessitates a fur

18.2.6 Bus

Bus networks employ a t
to which devices are atta
device are transmitted (b
devices accept only thc
required to retransmit th
associated with the ring
The bus configuratio
network will usually cc
advantage is that bus ne
often expanded to form
buildings. (Another br
traditionally used by sa
is discussed in section
bus used to connect dis

Ethernet bus

Ethernet [Hoiki] is ar
implementation a sin
employed. Of particu
as matched loads to
rates are 1 Mbit/s and
shows, for a 10 Mbit
bus for different level
10.8.1) for error detec

Te

Figure 18.8 *Bus net*

network necessitates a further $n - 1$ new connections.

18.2.6 Bus

Bus networks employ a broadcast philosophy. The bus is formed from a length of cable to which devices are attached by cable interfaces, or taps, Figure 18.8. Messages from a device are transmitted (bidirectionally) to all devices on the bus simultaneously; however, devices accept only those messages addressed to themselves. Since devices are not required to retransmit the messages, there is none of the delay (latency) or complexity associated with the ring topology.

The bus configuration is extremely tolerant of terminal failures, since operation of the network will usually continue if one of the active-component devices fails. A further advantage is that bus networks are easily reconfigured and extended. The bus topology is often expanded to form a tree structure, especially appropriate for use in multistorey buildings. (Another broadcast technique, similar in concept to a physical bus, is that traditionally used by satellite linked networks.) Access to the bus transmission medium is discussed in section 18.5. The small computer system interface (SCSI) is a dedicated bus used to connect discs and tape drives directly to the processor of a computer system.

Ethernet bus

Ethernet [Hoiki] is an example of a proprietary bus network, Figure 18.9(a). In this implementation a simple passive medium and transparent high impedance taps are employed. Of particular importance are the (coaxial cable) line terminators, which serve as matched loads to ensure reflections do not corrupt data transmission. Typical data rates are 1 Mbit/s and 10 Mbit/s, with a maximum bus length of 500 m. Figure 18.9(b) shows, for a 10 Mbit/s system, the expected number of attempts required to access the bus for different levels of network load. Ethernet uses 32-bit polynomial codes (section 10.8.1) for error detection.

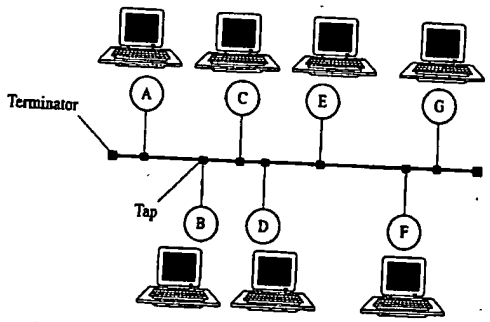


Figure 18.8 *Bus network.*

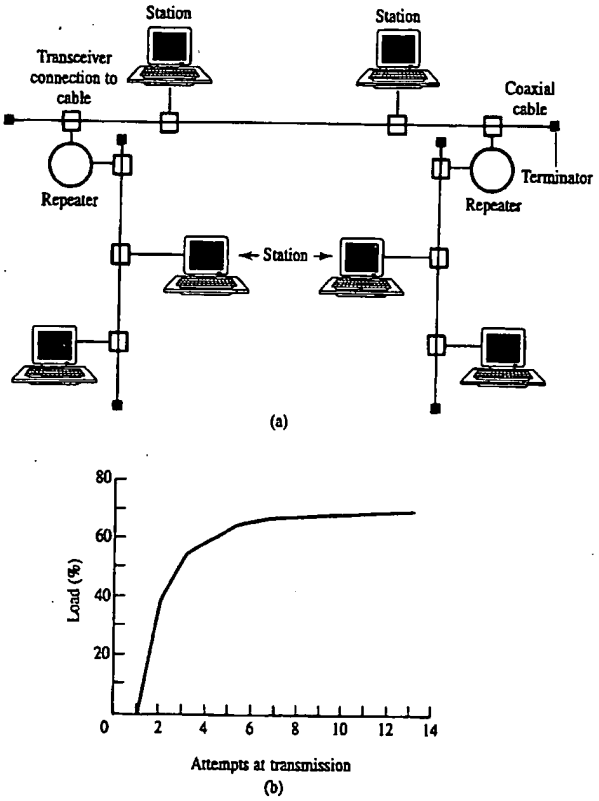


Figure 18.9 *Ethernet: (a) example configuration; (b) typical system response showing how the number of attempts at transmission varies with network load.*

18.2.7 Transmission media

Networks can utilise wire, coaxial cable, fibre optic or wireless links. Table 18.2 compares the different cabled and wireless transmission media. Metallic cable is preferred for many systems as simple, passive, tapped junctions cannot easily be realised for optical fibres. In all systems propagation loss, distortion, delay and noise are potential impairment mechanisms. Many systems use coaxial cable operating with baseband pulse rates up to 10 Mbit/s over 500 m paths. At higher rates 'broadband' data is modulated onto an RF carrier of, typically, 100 MHz, or fibre optic transmission (section 19.5) is used. Recently short runs of simple twisted wire pair have also become attractive for broadband transmission, because optical fibres are 5 to 10 times more expensive than twisted pair cable installations. In the near future broadband wireless LAN technology

sing carrier frequencies o

Table 18.2 *Typical cost*

Transmission medium	*Tw*
Range, m	
Data rate, kbit/s	0
Cost/node, US dollars	

18.3 Network cov

The first computer WAN
With the rise in the numb
responsible for post, te
networks. These WANs
held in a node store b
another node nearer the
the end-to-end delay is c

- Public networks: Fc
international EURO
accessing informatic
- Private networks: F
required. Private t
such as SWIFT emp
- Value added netw
facilities combined
network. The user
include TRANSPA

Local area networ
computer systems and
building or university
the features of a WAN

- Wide bandwidth, (
- Low (1 to 10 μs) d
- Low probabilities
- Simple protocols
- Low cost and eas
- High degree of co
- Geographically b

Metropolitan net
LANs and WANs. A
access to a WAN ar

using carrier frequencies of 5 GHz and 17 GHz will become important.

Table 18.2 *Typical cost per node, data rate and ranges for different transmission media.*

Transmission medium	*Twisted pair*	*Coaxial*	*Fibre optic*	*Radio*	*Infra-red*
Range, m	1 – 1,000	10 – 10,000	10 – 10,000	50 – 10,000	0.5 – 30
Data rate, kbit/s	0.3 – 2,000	300 – 10,000	1 – 100,000	1 – 10	0.05 – 20
Cost/node, US dollars	10 – 30	30 – 50	75 – 200	50 – 100	20 – 75

18.3 Network coverage and access

The first computer WAN, ARPANET, spanned the US and was later extended to Europe. With the rise in the number of potential digital communications users, many of the bodies responsible for post, telegraph and telephone (PTT) services have now built such networks. These WANs generally use store and forward systems in which messages are held in a node store before being switched, via an appropriate transmission link, to another node nearer the message's ultimate destination. With 10 to 50 kbit/s data rates the end-to-end delay is of the order of seconds. WANs can be classified as follows:

- Public networks: For example, the British Telecom Gold electronic mail service, the international EURONET (a packet switched service (PSS), which exists primarily for accessing information databases) and the Internet.
- Private networks: For many users, public data networks do not provide the services required. Private telecommunications networks leased on a semi-permanent basis, such as SWIFT employed by the banking fraternity, are one alternative.
- Value added networks: A value added network (VAN) uses conventional PTT facilities combined with specialised message-processing services to add value to the network. The user is offered flexibility and the economies of shared usage. Examples include TRANSPAC and the specialised banking EFTPOS networks.

Local area networks (LANs) are specifically designed for the interconnection of computer systems and peripherals within a geographically small site, such as a single building or university campus, and are generally privately owned. A LAN has many of the features of a WAN, but it also has its own, distinct, characteristics, i.e.:

- Wide bandwidth, of the order of tens of Mbit/s.
- Low (1 to 10 μs) delay due to resource sharing, and absence of buffering.
- Low probabilities of bit error, typically 10^{-9} to 10^{-11}.
- Simple protocols (compared with those necessary for the longer ranges of WANs).
- Low cost and easy installation.
- High degree of connectability and compatibility of physical connections.
- Geographically bounded, with a maximum range of approximately 5 km.

Metropolitan networks or MANs which may be up to 50 km in diameter lie between LANs and WANs. An access method (or protocol) defines the set of procedures for LAN access to a WAN and vice versa. Since LAN transmission rates are much higher than

those of interconnecting WANs one LAN network node must normally be dedicated to the WAN interface. With high speed MAN interconnection of LANs it is possible to transfer large files electronically, rather than physically using, for example, discs, tapes or CD-ROMs. Two of the most common network interconnection techniques utilise bridges and gateways.

18.3.1 Bridges and gateways

A bridge is a device that interconnects two networks of the same type (using the same protocol). The bridge utilises a store and forward feature to receive, regenerate and retransmit packets while filtering the addresses between connected segments. A gateway on the other hand [Smythe 1995] connects networks using different protocols, typically LANs and WANs. They can therefore provide transparent access to resources on other, remote, networks. It is a similar device to the bridge but also performs the necessary protocol conversion. The JANET network of Figure 18.2 has transparent gateway connections to other X.25 international networks, which link it to the Internet, section 18.7.6.

18.3.2 Network switches

With the increasing power of VLSI technology, a large switch array can now be implemented on a single chip. National Semiconductor developed a 16 × 16 switching matrix in 2 μm CMOS gate array technology in the late 1980s. Figure 18.10 shows an 8 × 8 *Banyan* switch, consisting of twelve 2 × 2 switching elements. There is only one route through the switch from each input to each output. At each stage in this switch the upper or lower output is chosen depending on whether a specific digit in the route control overhead is 1 or 0. This is one of the simplest switch arrays that can be constructed and illustrates the self-routing capability of a packet navigating a network composed of such switches. When routing and sorting is performed locally at each switch in a network

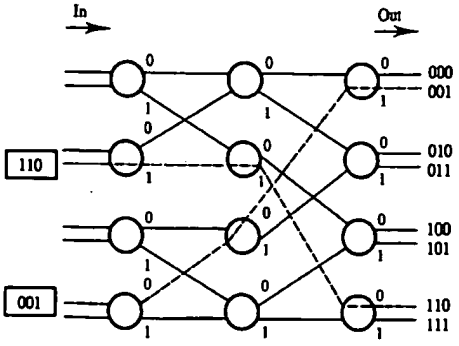


Figure 18.10 *8 × 8 Banyan packet switching network.*

here is no need for recor
herefore greater than th
how two example paths
network.

A problem inherent
locking (i.e. packets a
element). To reduce pa
the inputs or storage buf
To remove packet c
way that their paths ne
is known as Batcher so
descending order acco
crossbar switch which
The Starlight netwc
is shown in Figure 18.
1990s operated at 21(
problem of output bloc
but this can be overc
switching networks,
recirculating the dupli
need for buffered swit
The network is pac
same time. Banyan
packet's path through
presence of other pac
of up to 150 Mbit/s,
associated with a part

Pac
rando
por

Figure 18.11 *Starlig*

there is no need for recourse to a centrally located processing centre and the throughput is therefore greater than that of a circuit switched network. Dashed lines in Figure 18.10 show two example paths that packets with specific route headers would take through the network.

A problem inherent in all packet switched networks is that of packet collision or blocking (i.e. packets arriving simultaneously on the two inputs of a single switching element). To reduce packet collisions, the network can be operated at higher speed than the inputs or storage buffers can be introduced at the switch sites.

To remove packet collisions completely, the input packets must be sorted in such a way that their paths never cross. The technique normally used to perform this operation is known as Batcher sorting, in which the incoming packets are arranged in ascending or descending order according to route header. (Another non-blocking technique is the crossbar switch which was used in the 1970s for PSTN design.)

The Starlight network switch, which uses both Batcher sorting and Banyan switching, is shown in Figure 18.11. A CMOS 32 × 32 Batcher–Banyan switch chip in the early 1990s operated at 210 Mbit/s. Batcher–Banynan switching is not able to resolve the problem of output blocking, when multiple packets are destined for the same output port, but this can be overcome by the insertion of a trap network between the sorting and switching networks, Figure 18.11. The loss of trapped packets is overcome by recirculating the duplicate address packets into the next sorting cycle to eliminate the need for buffered switch elements.

The network is packet synchronous if it requires all packets to enter the network at the same time. Banyan networks can be operated packet asynchronously because each packet's path through the self-routing network is, at least to first order, unaffected by the presence of other packets. Packet switches such as these, which can operate at input rates of up to 150 Mbit/s, have been realised. (Note that the Banyan name is now often associated with a particular commercial system.)

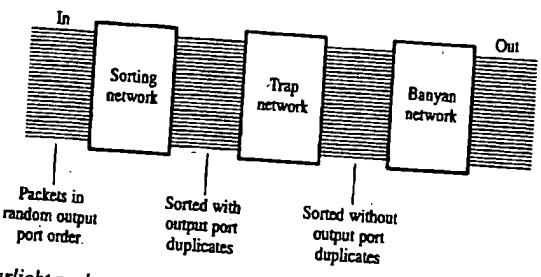


Figure 18.11 *Starlight packet sorting–switching network.*

18.4 Reference model for terminal interfacing

Most network architectures are organised as a series of layers. Previously the number, name and function of each layer differed from network to network. In all cases, however, the purpose of each layer was, and is, to offer certain services to higher layers, while shielding the details of exactly how those services are implemented in the lower layers.

Layer *n* at one node holds a conversation with layer *n* at another node. The rules and conventions used in this conversation are collectively known as the layer *n* protocol. The main functions of such a protocol are:

- Link initiation and termination.
- Synchronisation of data unit boundaries.
- Link control, with reference to polling, contention restriction and/or resolution, time-out, deadlock and restart.
- Error detection and correction.

The messages or blocks of data passed between entities in adjacent layers (i.e. across interfaces) are known as data units. With the exception of layer 1 no data is directly transferred from layer *n* at one node to layer *n* at another. Data and control information is passed by each layer to the layer immediately below, until the lowest layer is reached. It is only here that there is physical communication between nodes. The interfaces between layers must be clearly defined so that:

- The amount of data exchanged between adjacent layers is minimised.
- Replacing the implementation of a layer (and possibly its subordinate layers) with an alternative (which provides exactly the same set of services to its upstairs neighbour) is easily achieved.

18.4.1 The ISO model

This is a set of layers and protocols used to model network architecture, Figure 18.12. The overall purpose of the International Standards Organisation (ISO) model is to define standard procedures for the interconnection of network systems, i.e. to achieve open systems interconnection (OSI). ISO OSI processing is normally performed in software but, with the continuous rise in data rates, hardware protocol processors are, increasingly, being deployed. Several major principles were observed in the design of the ISO OSI model. These principles are that:

- A new layer should be created whenever a different level of abstraction is required.
- Each layer should perform a well defined service related to existing protocol standards.
- The layer boundaries should minimise information flow across layer interfaces.
- The number of layers should be sufficient so that distinct functions are not combined in the same layer, but remain small enough to give a compact architecture.

This has resulted in agreement to use seven layers as the OSI standard. The layers of the model are presented from the viewpoint of connection-mode transmission, starting from the interface to the physical medium. A key feature of the ISO OSI model is that it

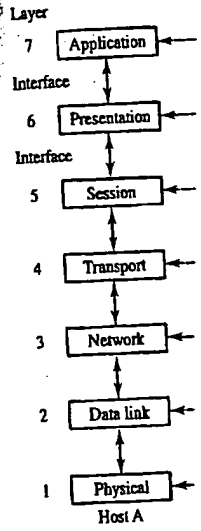


Figure 18.12 *ISO OS*

achieves standardisa
A summary of func
directly interfaces
mechanical aspects,
specification of vol
procedural aspects f

The task of th
transform it into a l
by breaking the in
processing acknow
Figure 18.13. Her
acknowledgement
from the remote t
retransmitted, as sh

The data link l
the protocols for a
10) are widely use
bit-serial data. T
packages, such as
facilities. The firs

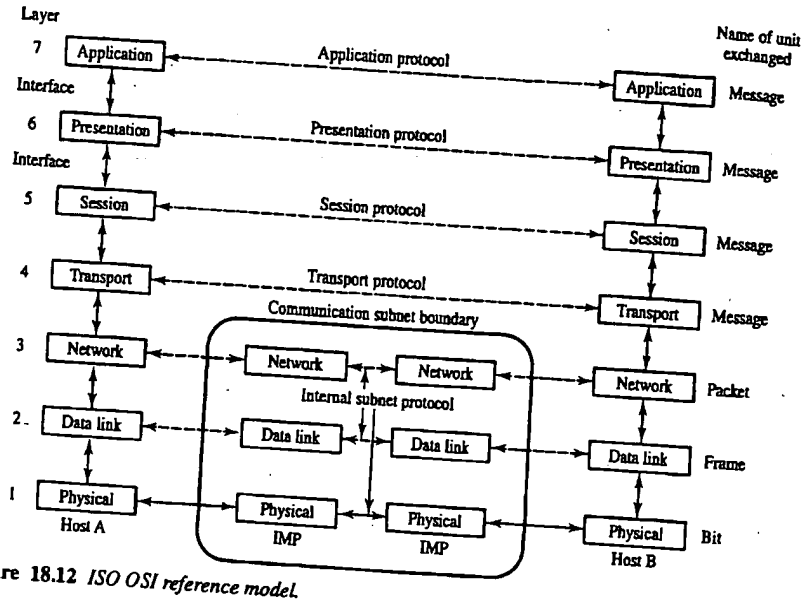


Figure 18.12 *ISO OSI reference model.*

achieves standardisation for data communications combined with error free transmission. A summary of function distribution is provided in Figure 18.12. The physical layer (1) directly interfaces with the transmission medium. This layer therefore includes: mechanical aspects, e.g. specification of cables and connectors; electrical aspects, e.g. specification of voltage levels, current levels and modulation techniques; functions and procedural aspects for the interface to the physical circuit connection.

The task of the data link layer (2) is to take the raw transmission facility and transform it into a link that appears free of transmission errors. It accomplishes this task by breaking the input data into data frames, transmitting the frames sequentially, and processing acknowledgement frames returned by the receiver. This is illustrated in Figure 18.13. Here once the packet is transmitted the transmitter stops and waits for an acknowledgement before the next packet is sent. If the acknowledgement does not arrive from the remote terminal within a prescribed time interval then the original packet is retransmitted, as shown in Figure 18.13 for packet 2.

The data link layer must both create and recognise frame boundaries. It thus defines the protocols for access to the network. Cyclic redundancy or polynomial codes (Chapter 10) are widely used in the data link layer to achieve an error detection capability on the bit-serial data. These processing functions are implemented in hardware. Software packages, such as Kermit, are located here to provide terminal emulation and file transfer facilities. The first two layers together are sometimes called the hardware layer.

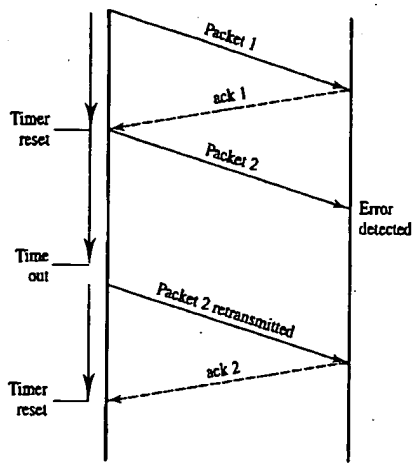


Figure 18.13 *Protocol for acknowledging (ack) successful packet receipt.*

The purpose of the network layer (3) is to provide an end-to-end communications circuit. It has responsibility for tasks such as routing, switching, and interconnection, including the use of multiple transmission resources, to provide a virtual circuit. The bottom three layers, collectively, implement the network function – and can be envisaged as a 'transparent pipe' to the physical medium, Figure 18.12. Data routed between networks, or from node to node within a network, require these functions alone.

The next layer (4) provides a transport service, suitable for the particular terminal equipment, that is independent of the network and the type of service. This includes multiplexing independent message streams over a single connection and segmenting data into appropriately sized units for efficient handling by the lower layers. Modulo $2^n - 1$ checksums (Chapter 10) are frequently computed within the software communications protocols of this layer, and compared with the received value, to determine whether data corruption has occurred. The transport layer can implement up to five different levels of error correction, but in LANs little error correction is required.

The functions of the session layer (5) are to negotiate, initiate, maintain and terminate sessions between application processes. This could for example be a transaction at a bank cash card machine. The layer operates in either half or full duplex, section 15.1.1.

While the session layer (5) selects the type of service, the network layer (3) chooses appropriate facilities, and the data link layer (2) formats the messages. The presentation layer (6) ensures that information is delivered in a form that the receiving system can interpret, understand and use. For example it defines the standard format for date and time information. The services provided by this layer include classification, compression and encryption of the data using codes and ciphers and also conversion, where necessary, of text between different types, e.g. to and from ASCII. The overall aim of layer 6 is to

make communication
The application l
Hence it provides res
emulation, distributed
electronic mail [Wilk
applications, layer 7
when network capaci
These ISO layers
systems and hence t
OSI equivalence. A
come to dominate e
order of magnitude l

18.4.2 Connectio

The ISO OSI model
modes. In the for
connection, which h
This mechanism is t
is employed in the
operate at up to 64 k
In connectionle
to a service, witho
self-contained dat
appropriate where
in electronic mail o

18.4.3 Physical

For many years the
which defines the
and reverse) chan
PCs, printers, etc.,
can accommodate
distances (up to
RS423A (10 kbit
effects of noise pi

18.4.4 Synchrc

A key requireme
synchronisation,
synchronisation
network's termi

make communication machine independent.

The application layer (7) defines the network applications that support file serving. Hence it provides resource management for file transfer, virtual file and virtual terminal emulation, distributed processing and other functions. Conceptually this is where X.400 electronic mail [Wilkinson] and other network utility software resides. In electronic mail applications, layer 7 contains the memories which store the messages for forwarding when network capacity becomes available.

These ISO layers are equivalent to similar functions in other layered communications systems and hence there is commonality between OSI and other standards which have OSI equivalence. As one moves down the layers overhead bits (added by each layer) come to dominate each data packet making the effective data rate at layer 7 at least an order of magnitude less than the actual bit rate at the data link layer.

18.4.2 Connectionless transmission

The ISO OSI model can accommodate both connection and connectionless transmission modes. In the former an association between two or more peer entities, termed a connection, which has a clearly defined lifetime is established for data transfer purposes. This mechanism is the logical equivalent to a (circuit-switched) PSTN telephone call, and is employed in the virtual circuit strategy of X.25 packet-switched networks which operate at up to 64 kbit/s.

In connectionless-mode transmission data transfer can be achieved by a single access to a service, without the requirement for establishment and release phases. Here each self-contained data unit can be routed independently. This mode is particularly appropriate where interaction between pairs of entities is intermittent and infrequent, as in electronic mail or when bursty, variable bit rate, traffic occurs.

18.4.3 Physical layer protocol

For many years the most common, physical layer, connection standard has been RS232C, which defines the signal functions and their electrical characteristics in the two (forward and reverse) channel directions. RS232C is widely used for interconnecting terminals, PCs, printers, etc., and uses a minimum of three wires for the two channel connection. It can accommodate data rates of a little over 1 kbit/s on paths of up to 20 m. For longer distances (up to 1 km) and higher data rates, balanced transmission systems such as RS423A (10 kbit/s) and RS422A (1 Mbit/s) have been developed which reduce the effects of noise pickup on cables.

18.4.4 Synchronisation and line coding

A key requirement of the protocol processor is that it must be able to handle clock synchronisation, line encoding/decoding and frame synchronisation. A special synchronisation problem in LANs and MANs is that of the clock oscillators in the network's terminal equipment and nodes. For synchronised clock operation all

transmissions are synchronised with a common master station clock. For plesiochronous operation (section 19.3) each individual node receives data with a clock signal locked to the clock of the preceding node (or terminal). It then transmits data with its own local clock signal. Due to clock oscillator tolerance, this implies that the number of transmitted bits can differ from the number of information bits, requiring justification.

Line coding usually introduces redundancy into the data stream (see Chapter 6). This redundancy can be utilised for reliable clock recovery and for the coding of special control symbols. Long streams of consecutive identical bits are not desirable if this means that timing information cannot be regenerated easily from the incoming signal. Line codes can be divided into three classes, namely scrambled codes, bit insertion codes, and block codes. The function of a scrambler, in this context, is to generate transitions in the line-encoded data stream when the original data stream contains long sequences of identical bits, as in HDB3, section 6.4.5.

Examples of bit insertion codes are the 5B6B and 8B1C line codes. In the 5B6B-PMSI (periodic mark space insertion) scheme, the encoder alternately inserts a mark (one) and a space (zero) for every five information bits. In the 8B1C (8 binary with 1 complement insertion) scheme, every eight bits are preceded by one extra bit, which is the complement of the first of the eight bits. Block codes allow unique words to be transmitted for synchronisation purposes.

18.5 Medium access control

High-speed channel access schemes are divided into four main classes: random access, demand assignment, fixed assignment, and adaptive assignment protocols [Skov]. For each class, the protocols are further subdivided according to network topology.

18.5.1 Random access and demand assignment protocols

Random access is typified by the ALOHA packet switched systems which simply transmit whenever a message is ready to send and then waits to see if this transmission collides with other data on the system. This is very inefficient giving a maximum channel utilisation (or normalised throughput) of 18% [Kleinrock]. Slotted ALOHA, in which data is constrained to time slots, avoids partial packet overlap and therefore achieves a maximum utilisation of double this.

The best known access scheme for bus systems, because of its low installation cost, is carrier sense multiple access with collision detection (CSMA/CD). This scheme allows nodes to contend for use of the network and, in this sense, is similar to ALOHA. In CSMA/CD, however, any node ready to transmit first listens, sensing the carrier, to check whether another node is transmitting an information frame. If another transmission is already in progress, the node defers operation until the end of the current transmission. Once the network is free, the node transmits its addressed message. However, due to the non-zero propagation delay, two or more nodes can attempt to transmit simultaneously causing contention, which results in a collision. Given this situation, the transmission

attempt is aborted and the
possible step-lock. This is
realise a channel utilisatio
introduce unpredictable net

With demand assignme
serving, or allocating acces
order. A representative t
multimode optical fibre sy
FDDI networks can emp
repeaters. The active o
regenerate data pulses w
18.7.4) has been develope
rate for image transmissi
circuit switching for voic

EXAMPLE 18.1 – Token p
A token passing ring opera
4-bit token. If the cable
access?

The total ring latency is 1
latency plus the token proc

A variation on the
propagate round the r
already filled with dat
which station empties
optic 600 Mbit/s Cam

Unidirectional bu
Access to the bus typi
access is employed,
begins its transmissi
from any upstream s
reservation access.

18.5.2 Fixed assi

In fixed assignme
participating station
always occupies a p
protocols are TDM
section 15.5. Mos

attempt is aborted and the node waits a random period before retransmission to avoid possible step-lock. This is a more sophisticated access scheme than ALOHA and can realise a channel utilisation of 75%. At this utilisation efficiency, however, collisions introduce unpredictable network delay or latency.

With demand assignment access protocols, message transmissions are governed by serving, or allocating access to, the attached stations in a predetermined (typically cyclic) order. A representative token ring protocol (see section 18.2.4) for packet switched multimode optical fibre systems is the 125 Mbit/s fibre distributed data interface (FDDI). FDDI networks can employ ring lengths of 100 km with 2 km spacings between repeaters. The active optical repeaters receive light from the incoming fibre and regenerate data pulses which are coupled into the outgoing fibre. FDDI (see section 18.7.4) has been developed primarily as a dual broadband backbone ring with a high data rate for image transmission applications. Current developments in FDDI II are aimed at circuit switching for voice and video multimedia transmission.

EXAMPLE 18.1 – Token passing

A token passing ring operates at 2 Mbit/s with 10 stations, each with a latency of 2 bits, and uses a 4-bit token. If the cable delay round the ring is 10 μs what is the maximum waiting time for access?

The total ring latency is $10 + (10 \times 2)/2 = 20$ μs. Maximum waiting time is given by the total ring latency plus the token processing time which is $20 + 4/2 = 22$ μs.

A variation on the conventional token ring is to partition time into slots and let these propagate round the ring. Access to a slot is possible, as it passes, provided it is not already filled with data. Such *slotted rings* can be divided into two groups depending on which station empties a full slot, the source station or the destination station. The fibre optic 600 Mbit/s Cambridge fast ring, section 18.2.4, employs source release.

Unidirectional buses integrate traffic with different priorities by means of rounds. Access to the bus typically follows one of three access schemes. When attempt-and-defer access is employed, a station wishing to transmit waits until the media is idle. It then begins its transmission and continues to listen to the media. If it detects a transmission from any upstream station, it aborts its transmission. The other schemes use polled or reservation access.

18.5.2 Fixed assignment protocols

In fixed assignment protocols the total network resource is divided among the participating stations in time, frequency, or code domain, in a fixed way. Thus, a station always occupies a part of the channel capacity, whether or not it has data to transmit. The protocols are TDMA, Figure 5.2, FDMA, Figure 5.12, and contention free CDMA, see section 15.5. Most of the FDMA and CDMA experimental ring and star networks use

optical signalling. CDMA permits uncoordinated accessing but may not offer the capacity of FDMA and TDMA, unless power control is adopted to avoid the near-far problem (see section 15.5.6).

18.5.3 Adaptive assignment protocols

At low loads, a device can usually transmit sucessfully as soon as it senses that the channel is idle. As the load rises, however, packet collisions may occur, thus destroying data. In such cases, adaptive protocols may switch to a more restricted, conflict-free, mode of operation such as token passing, attempt-and-defer, or TDMA access. An example of a high-speed network based on adaptive assignment access is the US 100 Mbit/s fibre optic HYPERchannel-100 network. This is based on a bus topology, and employs an access protocol that starts in CSMA/CD mode and switches to TDMA mode when collisions become too frequent.

18.6 International standards for data transmissions

18.6.1 X.25

This is the PSS standard which effectively replaces the telephone network, using the V-series recommendations (see section 11.6), with a digital system having superior error performance and fast switching. The X.25 recommendation defines the interface between terminal equipments and the public data network for packet-switched communication. A set of associated standards, X.3, X.28, X.29, have been developed to enable simple terminals to access an X.25 network. X.25 is actually a layered network access (interface) protocol that exhibits many of the properties of network architectures. The functionality of the X.25 specification corresponds entirely to the lower three layers in the ISO OSI model, Figure 18.12. In X.25, error checking is conducted at each node in the network. In the new frame relay systems this is only performed at the terminal stations.

X.25 PSS is now available within the UK as a core network for data communications, electronic mail, etc. One use of the network is for credit card verification and connections are available via telephone lines or radio access at 8 kbit/s [Davie and Smith]. Radio coverage in 1991 extended to 75% of the UK population for this data service.

18.6.2 IEEE 802

The IEEE 802.n specifications also map to the bottom three layers of the ISO OSI reference model. The IEEE split the data link layer into sub-layers: logical link control and medium-access control, which are used for bridging between networks. The 802.3 to 802.6 standards describe physical connections and define how access to the physical medium is coordinated for each LAN type. They therefore correspond to layer 1, and a sub-layer of layer 2, in the ISO OSI model.

- 802.2: Defines a lo
- 802.3: Defines a C
  on coaxial, and twis
- 802.4: Defines a to
- 802.5: Defines a to
- 802.6: Is a broadb
- 802.7: Specifies a

These IEEE standa
eventually be replaced

18.7 Network e

18.7.1 Manufactur

The manufacturers a
multidrop bus LAN
modulation. By oper
MHz, simultaneous (
signalling (Chapter
manufacturing envir
interconnection of n
Mbit/s with a maxin
other similar networ
digitising, encoding

18.7.2 Admiral

The Alvey Admiral
1980s, consists of b
interconnected by a
configured as a star
central switch local
switch providing c
time slots within th
required. User sit
commands to the s
first practical comb
ISDN interconnect

18.7.3 Military L

The US MIL-STL
applies to militar

- 802.2: Defines a logical link control layer.
- 802.3: Defines a CSMA/CD protocol, which is the basis of Ethernet, as implemented on coaxial, and twisted pair, cables.
- 802.4: Defines a token-passing bus protocol.
- 802.5: Defines a token-ring protocol (FDDI).
- 802.6: Is a broadband MAN standard (45 Mbit/s) for voice, data and video.
- 802.7: Specifies a broadband FDMA LAN (with 400 MHz of bandwidth).

These IEEE standards are now internationally accepted as ISO equivalent. 802.4 may eventually be replaced by FDDI-II.

18.7 Network examples

18.7.1 Manufacturers application protocol (MAP)

The manufacturers application protocol is implemented on a 10 Mbit/s broadband multidrop bus LAN using an 802.4 token passing bus channel employing RF carrier modulation. By operating on two separate carriers at frequencies between 59.74 and 264 MHz, simultaneous (FDM) data transmission and reception is accomplished using QAM signalling (Chapter 11). MAP is designed to interconnect plant (such as robots) in a manufacturing environment. Another commercial development is the fieldbus for interconnection of measuring instruments [Jordan]. In one network the bit rate is 1 Mbit/s with a maximum access time of 5 ms for a 32-node system. There are various other similar network designs, many of which share the IEEE 488 interface standard for digitising, encoding and transmitting signal samples from remote locations.

18.7.2 Admiral

The Alvey Admiral multimedia prototype network (Figure 18.14), developed in the mid 1980s, consists of baseband CSMA/CD Ethernet LANs at each of the five project sites, interconnected by a high speed network. This early example of a high speed network is configured as a star, with 2 Mbit/s primary rate access circuits from each user site to a central switch located at the star hub. At the centre of the network is a non-blocking switch providing configurable interconnection of any number of consecutive 64 kbit/s time slots within the 2 Mbit/s bearer to provide point-to-point links from site to site as required. User sites are able to alter the configuration of the network by sending commands to the switch controller, via the public X.25 network. This network was the first practical combination of fast local Cambridge rings and Ethernets with primary rate ISDN interconnections, as described in the following chapter.

18.7.3 Military LAN systems

The US MIL-STD-1553B and its UK equivalent DEF STAN 00-18 or STANAG 3838 applies to military ship, submarine and aircraft LANs. These 1970s systems used

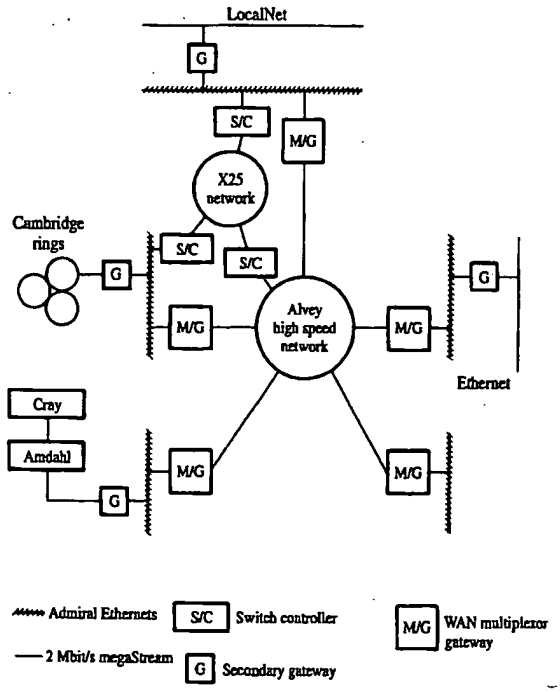


Figure 18.14 *The Admiral high-speed network.*

screened twisted pair cable to achieve 1 Mbit/s transmission rates with up to 31 remote terminals. Further US work has achieved high speed (20 to 40 Mbit/s) fibre optic networks while, in the UK, DEF STAN 00-19 is a 300 m multi-drop bus topology on a screened twin coaxial cable which signals at 3 Mbit/s. The bus for the European fighter aircraft, STANAG 3910, is a dual redundant 20 Mbit/s fibre optic implementation. It defines low speed 1 Mbit/s dual redundant wired channels plus the high speed 20 Mbit/s fibre optic channels. These specialised systems are not generally fully ISO compatible.

In contrast to civilian LANs, military ones require priority accessing schemes with short access delays for threat messages and must also be secure and free from transmission errors. These constraints inevitably result in specialised networks being developed for military applications.

18.7.4 Fibre distributed data interface (FDDI)

FDDI is a backbone broadband ring to which other, slower speed, tree networks and peripherals can be connected, Figure 18.15. (FDDI was conceived as a fast or broadband service to handle multimedia data transfer for applications such as desktop conferencing.)

Figure 18.15 *Networ*

The overall structure plug-in multimode fi techniques described code. Link P_b is 2. length of 200 km, Packet latency for Currently there is i cables, but this int structure of Figure employing ring net

18.7.5 Wireless

There is much cu Ethernet wired co proximity to each for 50 to 300 m employed to comb hundreds of kbit/s spectrum) modula permits the multi for by equalisatio GHz carrier frequ

EXAMPLE 18.2
A wireless networ computers dispers obstructions in the space inverse squa

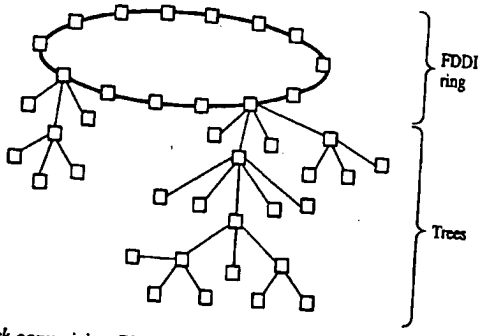


Figure 18.15 *Network comprising FDDI high speed ring with tree connections.*

The overall structure then looks like a ring-of-trees. FDDI operates at 100 Mbit/s as a plug-in multimode fibre ring (section 19.5) with 4B5B coding (which is a variation on the techniques described in section 6.4.7) implying a 125 MHz clock rate for this bit insertion code. Link P_b is 2.5×10^{-10} and packet error probability is 10^{-9}. On the maximum ring length of 200 km, ring transit time is 2.7 ms, corresponding to 60 full length packets. Packet latency for access to a network at 45 Mbit/s load is typically 50 to 200 μs. Currently there is interest in copper distributed data interface (CDDI) using twisted pair cables, but this interest is more intense in North America than Europe. The FDDI structure of Figure 18.15 is not dissimilar to the future SDH inner core proposals employing ring networks (see section 19.4.1).

18.7.5 Wireless networks

There is much current interest in developing wireless systems to obviate the need for Ethernet wired connections between terminals and printers which are located in close proximity to each other. The US NCR WaveLAN is such a system which typically caters for 50 to 300 m link connections. In the indoor environment, if equalisation is not employed to combat multipath responses, then the data rate is typically limited to several hundreds of kbit/s. In WaveLAN systems 2 Mbit/s data is spread by a wideband (spread spectrum) modulator into an 11 MHz bandwidth channel. This wideband transmission permits the multipath signals to be separately resolved and their presence compensated for by equalisation in the receiver. Another system, HYPERLAN, operates using a 5.2 GHz carrier frequency with a 20 Mbit/s BPSK transmission rate [Halls].

EXAMPLE 18.2

A wireless network is to be designed for interfacing several portable, battery powered, laptop computers dispersed around an office environment using the HYPERLAN standard. Due to obstructions in the office path loss follows an inverse cube law with distance (rather than the free space inverse square law of Chapter 12). The portable computer power consumption is 15 W and

the wireless network should not *significantly* degrade the battery operating life. Estimate, using the link budget analysis of Chapter 12, the maximum operating range you might expect to achieve. Assume that the receiver noise figure is 10 dB, the (BPSK) signal requires a receiver CNR of 15 dB, a fade margin of 6 dB is necessary, the total implementation loss is 2 dB and the transmitter efficiency is 40%.

If the transmitter uses 10 to 15% of the battery power then the power available for the radio modem is 1.5 to 2.2 W. At 40% efficiency this gives approximately 0.75 W or 29 dBm of transmitted power.

The thermal noise floor in a 25 MHz bandwidth, from equation (12.8), is −100 dBm. The received carrier level must be larger than this by 10 dB (noise figure), 2 dB (implementation loss), 15 dB (receiver CNR requirement) plus 6 dB (fade margin). Thus, adding a total of 33 dB, the required received carrier level is $C = -67$ dBm.

The maximum tolerable path loss is thus +29 +67 = 96 dB $= 4.0 \times 10^9$. FSPL is defined in Chapter 12. Using an R^3 law, equation (12.71) becomes:

$$\text{FSPL} = \left(\frac{4\pi}{\lambda}\right)^2 R^3$$

For isotropic antennas $G_T = G_R = 0$ dB $\equiv 1$. Thus for an operating frequency f and a free space propagation velocity c, the transmission loss, P_T/C, equals the FSPL, i.e.:

$$\frac{P_T}{C} = \text{FSPL} = \left(\frac{4\pi f}{c}\right)^2 R^3$$

Substituting numerical values and solving to find R:

$$4.0 \times 10^9 = \left(\frac{4\pi \times 5.2 \times 10^9}{300 \times 10^6}\right)^2 R^3 = \frac{4.74}{10^{-4}} R^3$$

$$R^3 = \frac{4.00}{4.74} 10^5 \text{ and } R = 44 \text{ m}$$

This range is typical for an office environment.

18.7.6 Internet

The Internet started life over 25 years ago as the ARPANET, section 18.1, which grew, initially, to link many university and industry laboratories in the US defence research community. It then became linked to Europe and gradually spread beyond the defence community, particularly as people realised that electronic mail could assist groups in many different organisations and countries to communicate effectively. The Internet is a communications medium, and a common set of protocols, which have been unified to form a coherent network.

Two separate developments ensured its rapid and widespread application. Firstly, it was demonstrated how a word in one document, located in one computer connected to the Internet, could be linked electronically to a previously unrelated document (a photograph, for example) stored on another computer, often in a different country. Many millions of

these *hypertext* links
range of different kir
which uses a globa
collection of inform
information, such a
'Hubble' would reac
colour images origi
analysing the data i
the cometary impact

The Internet is a
statistics about it ar
more than 2 millio
around the world.
individual owns or
rapid evolution and
individuals and org
traffic on the Intern

At its current r
Internet by 2003.
commercial use of
and maintenance o
sector. In 1994 th
through which con
and services. The
the Internet. The
provide services fc

18.8 Summa

LANs, MANs a
individual buildir
networks may be
packet switching
former requiring
the latter allowir
network operatio
message switchin

Network top
systems. Transm
Microwave and
network terminal

Bridges are u
LANs). Gatewa

these *hypertext* links have now been put in place to better organise, and access, a wide range of different kinds of information. This has resulted in the world wide web (WWW) which uses a global web of linkages between an already huge, but ever growing, collection of information items and databases. Secondly, individuals started sharing information, such as pictures, on the Internet. For example, selecting the key word 'Hubble' would readily access not just data about the space telescope, but also the latest colour images originating directly from those computers controling the telescope and analysing the data it generates. Thus, many individuals first saw the Hubble images of the cometary impact on Jupiter via the Internet.

The Internet is a network of networks, hence the name – *inter-net*work network, and statistics about it are legion. In early 1995 it comprised around 50,000 networks, linking more than 2 million computers and perhaps 10 to 15 million users in many countries around the world. (It is not accurately known what the numbers are, because no single individual owns or controls the Internet.) It is self-regulating, capable of a high degree of rapid evolution and growth, and has been operated up to now by loose federations of individuals and organisations. What is really startling is the 20% per month growth of traffic on the Internet.

At its current rate of expansion, everyone on the planet would be connected to the Internet by 2003. A factor which is contributing to the growth of traffic is that commercial use of the Internet is now beginning to become important as the operation and maintenance of the underlying backbone networks in the US moves into the private sector. In 1994 there were over 21,000 commercial 'domains' registered on the Internet through which companies offer facilities to browse electronic catalogues and order goods and services. These commercial developments will change the fundamental character of the Internet. The Internet will evolve as this new technology matures and is used to provide services for home shopping, video entertainment, electronic banking, etc.

18.8 Summary

LANs, MANs and WANs refer to communications networks typically spanning individual buildings, individual towns, and individual countries, respectively. Such networks may be circuit switched, message switched or packet switched. Two forms of packet switching can be distinguished, i.e. virtual circuit and datagram switching – the former requiring all packets in a given message to traverse the same network route and the latter allowing packets to traverse independent routes. Historically the trend in network operation is from circuit switching (typified by the traditional PSTN), through message switching and virtual circuit switching, to datagram switching.

Network topologies include point-to-point, multidrop, star, ring, mesh and bus systems. Transmission media include twisted wire pairs, coaxial cable and optical fibres. Microwave and infrared links are also used to provide wireless connections between network terminals and nodes.

Bridges are used to interconnect networks using the same protocols (e.g. two similar LANs). Gateways are devices used to interconnect networks using different protocols

(e.g. a LAN and a WAN). Switches with self-routing properties can be employed at network nodes to improve switching speed over switches which require centralised control. Packet collisions at switch inputs can be avoided using an input sorting network and collisions at switch outputs avoided using a trap network.

The ISO OSI model for network protocol architectures has seven layers. The lowest three layers (physical, data link and network) specify the operation of the communications sub-net. The upper four layers are concerned with terminal equipment/network compatibility, initiation and control of a communication session, data formatting for correct presentation and the particular application being run by the user.

Medium access control (MAC) is one function of the data link layer in the OSI model. MAC protocols govern the allocation of physical medium resources (time, bandwith, orthogonal codes) between network terminal equipments. These resources can be allocated on a fixed, demand or random assignment basis. Fixed assignment protocols are efficient if data terminal equipments (DTEs) have a heavy, and relatively constant, traffic load. Random access protocols are more efficient if traffic loads are highly variable and uncorrelated between terminals. Random access protocols suffer, however, from potential packet collisions if two or more DTEs attempt to access the medium simultaneously. Such contention can be resolved using CSMA/CD protocols. Fixed assignment systems may be adaptive in that the proportions of medium resource allocated to different terminals may track long term variations in terminal traffic loads. Systems may also be adaptive in the sense that an essentially fixed assignment protocol may be substituted for a (normally) random access protocol if collision detection occurs too frequently. Demand assignment systems allow terminals to commandeer medium resources as and when they are required. They may operate using centralised control, in which terminals are polled to ascertain their need for resources, or distributed control, in which token passing is employed. Bus systems typically use CSMA/CD protocols whilst ring systems typically use token passing.

X.25 is the ITU-T standard defining the interface between DTEs and data communication equipment (DCEs) for packet switched networks. The DCE is the interface with a node of the data network and may be located at the same site as the network node or be remote from it (as in the case of a modem used to connect a computer to the packet switched public data network via an analogue local loop of the PSTN). Other X-series standards specify the protocols for connecting low speed character mode terminals to packet networks using packet assemblers and disassemblers (PADs).

FDDI is a 100 Mbit/s, optical fibre, backbone ring network which supports lower speed tree networks at its nodes. Its maximum circumference is 200 km. Wireless networks, using radio transmission between nodes, are susceptible to frequency selective fading due to multipath propagation. Spread spectrum techniques can be used to mitigate the resulting distortion, allowing transmission at data rates many times that which the channel would ordinarily support.

The Internet is, at present, the ultimate WAN in that it gives, potentially, global coverage. In addition to the conventional data communication uses of a WAN, hypertext connections between different documents held on computers connected to the Internet result in the unparalleled information resource called the World Wide Web.

CHAPTER 19

Public I integra networ

19.1 Introduc

This chapter exam applied within the services digital n plesiochronous, a digital hierarchy (payload capacity d

As much of th capabilities of f amplification, are chapter concludes in the local loop basic, rate ISDN a

19.2 The tele

The internationall multiplexing (Fig access level at 1 individual 64 kbi USA the first lev stream.) Multipl for the duration o voice communica as they require terrestial telepho satellite circuits t

CHAPTER 19

Public networks and the integrated services digital network (ISDN)

19.1 Introduction

This chapter examines how many of the techniques discussed earlier in the book are applied within the international public communications network. It covers the integrated services digital network (ISDN) [Griffiths], and includes discussion of the older plesiochronous, and newer synchronous, multiplexing techniques. The synchronous digital hierarchy (SDH) [Omidyar and Aldridge] is described, its frame structure and payload capacity defined, and its advantages outlined.

As much of this network now employs wideband fibre optic links, the principles and capabilities of fibre transmission, and optical pulse generation, reception and amplification, are summarised and a typical optical link budget presented. Finally this chapter concludes with a brief account of accessing schemes and on-going developments in the local loop network for digitised speech and data connections using primary, and basic, rate ISDN access.

19.2 The telephone network

The internationally agreed European ITU-T standard for PCM, TDM digital telephony multiplexing (Figure 5.27), is shown in Figure 19.1. Although this shows the basic access level at 144 kbit/s, the multiplexing hierarchy provides for the combining of 32 individual 64 kbit/s channels (Chapter 6) into a composite 2.048 Mbit/s signal. (In the USA the first level in the multiplex combines only 24 channels into a 1.544 Mbit/s data stream.) Multiplexing allocates a complete communications channel to each active user for the duration of his call or connection. In principle, as the channel utilisation factor for voice communications is low, we could concentrate the traffic by switching between users as they require transmission capacity. This resource saving strategy is not used in terrestial telephony but digital speech interpolation (DSI) is employed on international satellite circuits to achieve a significant increase in capacity, (see section 14.3.7).

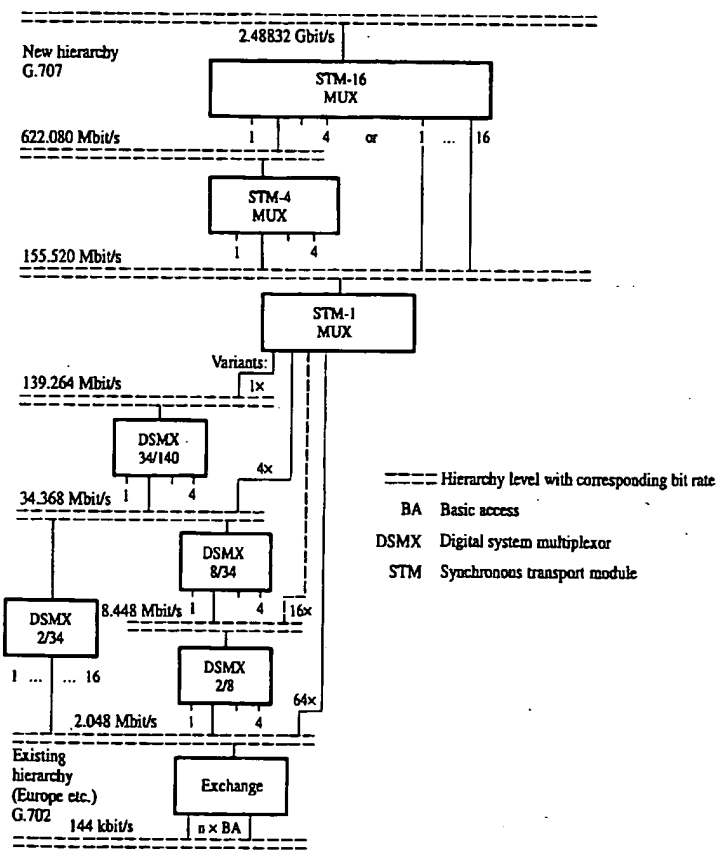


Figure 19.1 *ISDN access to European PCM TDM hierarchy with SDH at the upper levels.*

The ITU-T provides for higher levels of multiplexing, above 2.048 Mbit/s, combining four signals, in the digital system multiplexers (DSMX) 2/8 and 8/34, Figure 19.1, to form the signal at the higher multiplexing level. At each level the bit rate increases by slightly more than a factor of 4 since extra bits are added to provide for frame alignment and to facilitate satisfactory demultiplexing (see section 19.3). The upper levels, beyond 140 Mbit/s, form the synchronous digital hierarchy (SDH), and are only in limited use at present. These will be described later.

The extent of the current digital UK telephone network is shown in Figure 19.2. This is divided into an outer core of main processor exchanges and an inner core of about 55 trunk exchanges (digital main switching centres) which are fully interconnected on 140 Mbit/s, or higher bit rate, transmission links. These are now mainly optical links but also

include coaxial, and
fibre links, describe
on microwave radio
access in Figure 19.
Each subscriber
building, Figure 19.

Figure 19.2 *ISDI*
199

include coaxial, and terrestial microwave relay, paths. On 140 Mbit/s, 1.3 μm, optical fibre links, described later in section 19.5, the repeater spacing is typically 20 km while on microwave radio links, Chapter 14, the spacing is closer to 50 km. International access in Figure 19.2 is provided via satellite links or fibre-optic undersea cables.

Each subscriber telephone has two copper wires that go directly to the local exchange building, Figure 19.3. (The distance is typically 1 to 10 km, being smaller in cities than

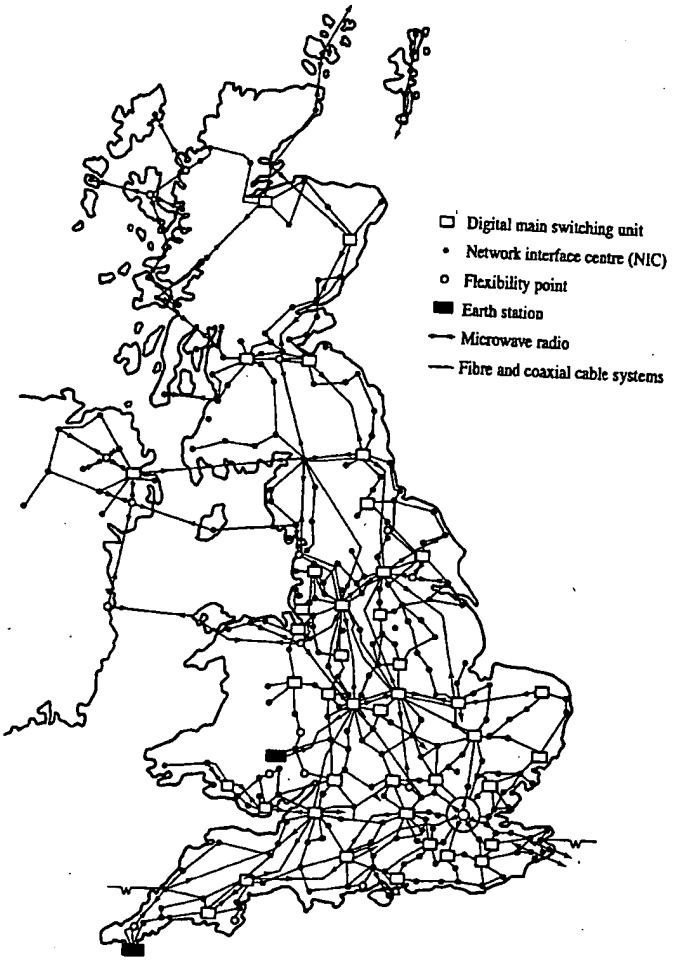


Figure 19.2 *ISDN locations and UK digital trunk telecommunications network (source: Leakey, 1991, reproduced with the permission of British Telecommunications plc.).*

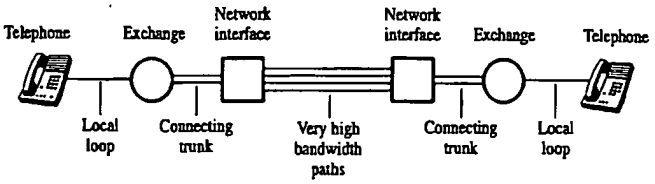


Figure 19.3 *Trunk telecommunications system used for establishing a connection.*

in rural areas where one exchange normally serves 40 km^2 or 5,000 to 50,000 customers.) The two-wire access connection between each subscriber's telephone and the exchange is known as the *local loop*. Each exchange has a number of outgoing lines to one or more nearby switching centres which provide the network interface. Figure 19.3 shows the typical connection for a long distance (trunk) call involving both inner and outer core connections. In this system there are four wires to the microphone and speaker in the handset, two wires in the local loop and four wires again in the national trunk network. (A four wire circuit implies separate transmit and receive channels as in cellular radio systems, Chapter 15.) In the UK there are approximately 6300 local exchange buildings and the local access network comprises 36 million metallic pair cables.

Traditional control signalling in circuit-switched telephone networks for establishing or initiating call connections may either use a separate communications channel or operate on an in-channel basis. With in-channel signalling, the same channel is used to carry control signals as is used to carry message traffic. Such signalling begins at the originating subscriber and follows the same path as the call itself. This has the merit that no additional transmission facilities are needed for signalling. Two forms of in-channel signalling are in use, in-band and out-of-band. In-band signalling uses not only the same physical path as the call it serves, it also uses the same frequency band as the voice signals that are carried. Out-of-band signalling takes advantage of the fact that voice signals do not use the full 4 kHz of available bandwidth. A separate narrow signalling band, within the 4 kHz, is used to send control signals. A drawback of in-channel signalling schemes is the relatively long delay from the time that a subscriber dials a number to the connection being made.

This problem is addressed by common channel signalling in which control signals are carried by an independent signalling network. Since the control signal bandwidth is small, one separate control signal path can carry the signals for a number of subscriber channels. The common channel uses a signalling protocol, and requires the network architecture to support that protocol, which is more complex than the in-channel signalling case. The control signals are messages that are passed between switches and between a switch and the network management centre.

Over the past decade several different general purpose signalling systems have been developed by ITU and other standards organisations. The most important of these, and the one of major relevance to the ISDN, is the set of procedures known as signalling system No. 7, which is structured in accordance with the ISO OSI model of Figure 18.12. The overall objective is to provide an internationally standardised, general purpose,

common channel si
- is optimised for stored program
- is designed to control, remote
- provides a reli sequence witho
- is suitable for u

19.3 Plesioch

The 2.048 Mbit/s Figures 19.1 and Figure 19.5. Time slot 16 is used for channels are used one 8-bit voice sig

The system fo multiplexers (DSI implies separate f the name plesioch than the incoming allows for small e requires some extu speed oscillator. sufficient bits ar required because

Figure 19.4 *DSM*

common channel signalling system which:

- is optimised for use in digital telecommunication networks in conjunction with digital stored program control exchanges utilising 64 kbit/s digital signals;
- is designed to meet present and future information transfer requirements for call control, remote network management, and maintenance;
- provides a reliable means for the transfer of information packets in the correct sequence without loss or duplication;
- is suitable for use on point-to-point terrestrial and satellite links.

19.3 Plesiochronous multiplex

The 2.048 Mbit/s multiplex level (also called the PCM primary multiplex group) in Figures 19.1 and 19.4 comprises frames with 32 8-bit time slots within each frame, Figure 19.5. Time slot zero is reserved for frame alignment and service bits, and time slot 16 is used for multiframe alignment, service bits and signalling. The remaining 30 channels are used for information carrying, or payload capacity, each channel containing one 8-bit voice signal sample.

The system for assembling the TDM telephony data stream assumes that the digital multiplexers (DSMX) in Figure 19.4 are located at physically separate sites which implies separate free running oscillators at each stage in the multiplex hierarchy, hence the name plesiochronous. These oscillators must therefore run at speeds slightly higher than the incoming data, Figure 19.4, to permit local variations to be accommodated. This allows for small errors in the exact data rates in each of the input paths or tributaries but requires some extra bits to be added (i.e. stuffed or justified) to take account of the higher speed oscillator. Elastic stores, Figure 19.6, are used in a typical multiplexer to ensure sufficient bits are always available for transmission or reception. These stores are required because the plesiochronous digital hierarchy (PDH) works by interleaving bytes

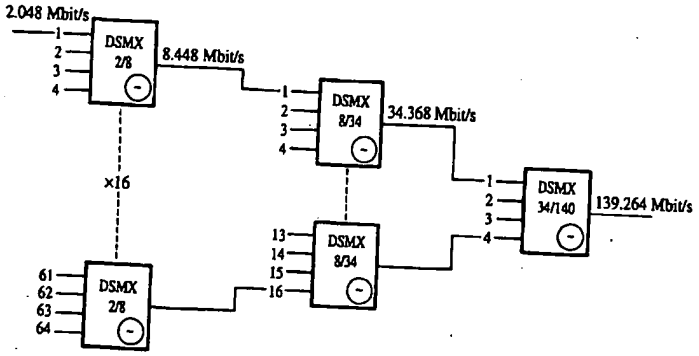


Figure 19.4 *DSMX interconnection to form a plesiochronous multiplex hierarchy.*

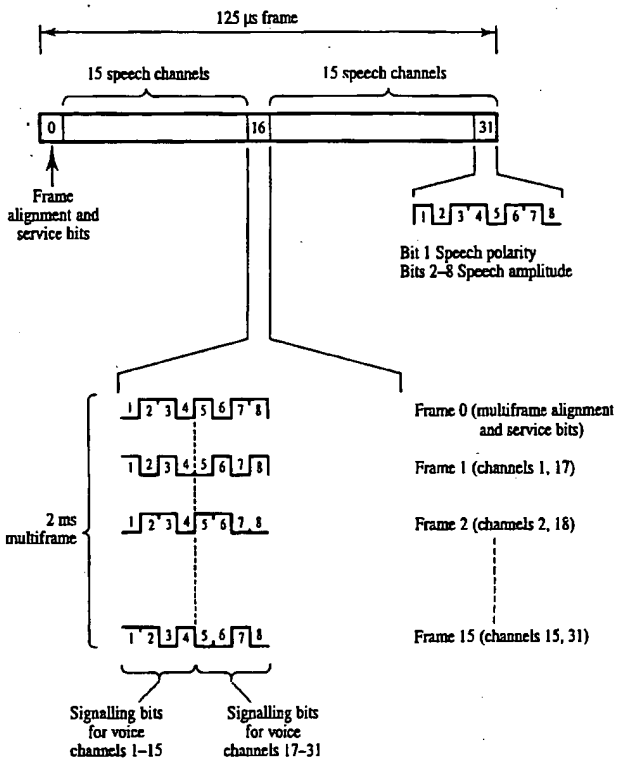


Figure 19.5 *PCM primary multiplex group frame structure.*

or words from each 64 kbit/s tributary, rather than bit interleaving, to form the 2 Mbit/s multiplex. Thus, at the 8 Mbit/s multiplexing level, and above, where bit interleaving is employed, bits must be accumulated for high speed readout.

Figure 19.7 shows more detail of the multiplexer hardware. The code translators in Figure 19.6 convert binary data from, and to, HDB3 (see section 6.4.5). Figure 19.8 shows details of the plesiochronous frame structure at 8 Mbit/s. (Note the bit interleaving, shown explicitly for the justification bits, used at multiplexing levels above 2 Mbit/s.) The ITU-T G series of recommendations (G.702) defines the complete plesiochronous multiplex hierarchy. The frame alignment signal, which is a unique word recognised in the receiver, ensures that the appropriate input tributary is connected to the correct output port. The unique word also permits receiver recovery from loss of synchronisation, if it occurs. The plesiochronous multiplex system was developed at a time when transmission costs were low and switching costs were high. With recent advances in VLSI this premise is now no longer valid, hence the movement to new standards.

Figure 19.6 2/8

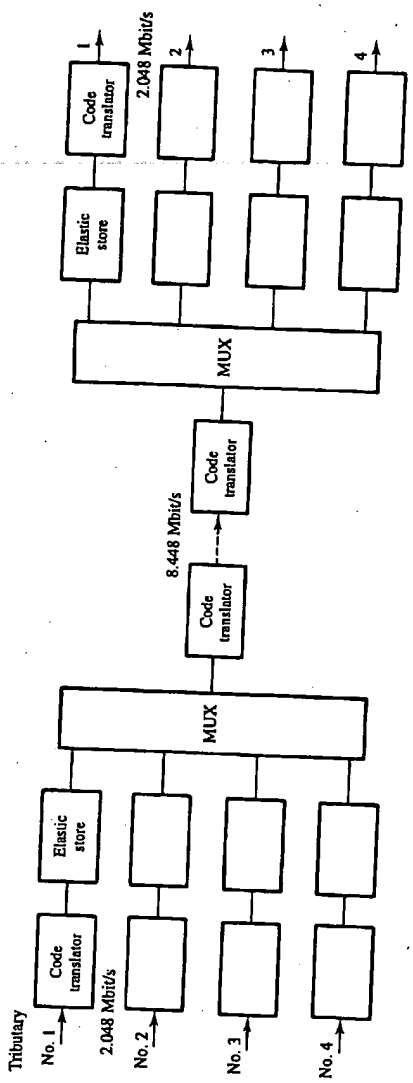


Figure 19.6 *2/8 Multiplexer block diagram.*

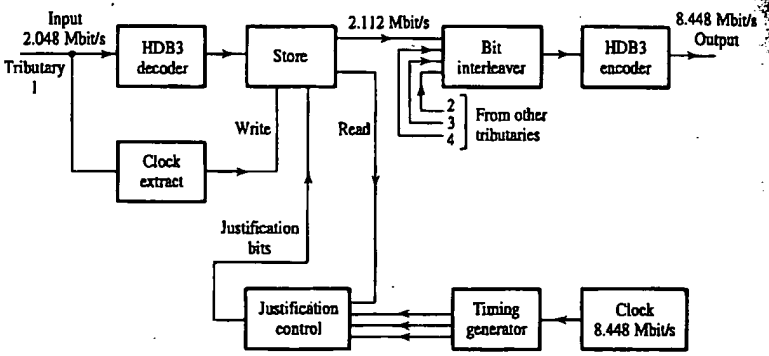


Figure 19.7 *2/8 multiplexer timing details.*

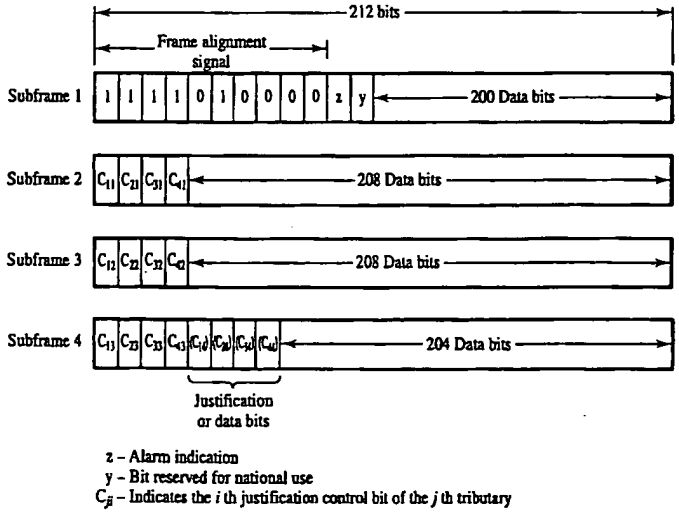


Figure 19.8 *Plesiochronous frame structure.*

A disadvantage of the plesiochronous multiplex is that it was designed for point-to-point transmission applications in which the entire multiplex would be decoded at each end. This is a complicated process since it requires full demultiplexing at each level to recover the bit interleaved data and remove the justification bits. Thus a single 2 Mbit/s channel, for example, cannot be extracted from, or added to, a higher level multiplex signal without demultiplexing down, and then remultiplexing up, through the entire PDH, Figure 19.9. The plesiochronous multiplex does *not* support definitive or clear identification of the various individual channels which are being carried.

Figure 19.9 *i*

Many hi
plesiochrono
limited, how
It does not p
respond to t
since netwo
availability c
plesiochronc
maintenance
of a standar
different par
whose netw
their networ
The PDI
wideband o
SONET an
following s
provide the
to connect
away from
single 64-cl
an input to

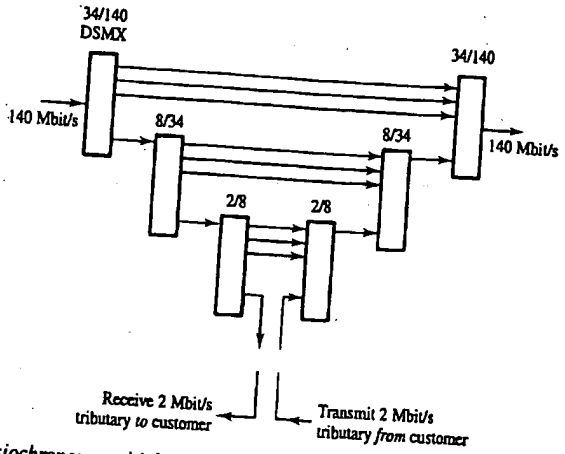


Figure 19.9 *Plesiochronous multiplex add-drop scheme for inserting, and removing, a 2 Mbit/s tributary to, and from, a 140 Mbit/s stream.*

Many high capacity transmission networks are provided by a hierarchy of digital plesiochronous signals. The plesiochronous approach to signal multiplexing is severely limited, however, in its ability to meet the foreseeable requirements of network operators. It does not provide, cost-effectively, the flexible network architecture which is required to respond to the demands of today's evolving telecommunications market. Furthermore, since network management and maintenance strategies were based, historically, on the availability of a manual distribution frame, there was no need to add extra capacity to the plesiochronous frame structure in order to support network operations, administration, maintenance and provisioning (OAM&P) activities. Other PDH drawbacks are the lack of a standard above 140 Mbit/s and the use of different plesiochronous hierarchies in different parts of the world. This leads to problems of interworking between countries whose networks are based on 1.544 Mbit/s (e.g. Japan, North America) and those basing their networks on 2.048 Mbit/s (e.g. Europe, Australia).

The PDH limitations described above, and the desire to move from metallic cable to wideband optical fibres, have been important motivations in the development of the new SONET and synchronous digital hierarchy (SDH) systems which are described in the following sections. These new systems offer improved flexibility and can more readily provide the 2 Mbit/s leased lines which, for example, cellular telephone operators require to connect their base station transmitters to switching centres. (Just prior to the move away from PDH towards SDH systems British Telecom (BT) did develop, in the 1980s, a single 64-channel 2 to 140 Mbit/s multiplexer. This has been included in Figure 19.1 as an input to the new 155.52 Mbit/s standard multiplexer rate.)

EXAMPLE 19.1

Find the number of standard PCM voice signals which can be carried by a PDH level 4 multiplex and estimate the maximum channel utilisation efficiency at this multiplexing level.

PDH Level 1 (primary multiplex group) carries 30 voice signals. Each subsequent level combines four tributaries from previous levels. At level n the number of potential voice signals is therefore given by:

$$\chi = 30 \times 4^{n-1}$$

At level four:

$$\chi = 30 \times 4^{4-1} = 1920 \text{ voice signals}$$

Nominal bit rate at PDH level 4 is 140 Mbit/s. Therefore channel utilisation efficiency is:

$$\eta_{ch} = \frac{\chi}{R_b/R_v} = \frac{1920}{(140 \times 10^6)/(64 \times 10^3)} = 88\%$$

19.4 SONET and SDH

The concept of the digital SONET (synchronous optical network) was initially introduced in the USA in 1986 to establish wideband transmission standards so that international operators could interface using standard frame formats and signalling protocols. The concept also included network flexibility and intelligence, and overhead channels to carry control and performance information between network elements (line systems, multiplexers) and control centres.

In 1988 these concepts were adopted by ITU and ETSI (European Telecommunications Standards Institute) and renamed synchronous digital hierarchy (SDH) with the aim of agreeing worldwide standards for transmission covering optical interfaces, control aspects, equipment, signalling, etc. [Miki and Siller]. Bit transport rates start as low as 52 Mbit/s and go up through 155 Mbit/s with a hierarchy to 622 Mbit/s, 2488 Mbit/s and beyond. The ITU-T G.707/8/9 standards have now reached a mature stage allowing manufacturers to produce common hardware.

19.4.1 Advantages and flexibility

The key advantages of SDH are as follows:

- it is cheaper to add and drop signals to meet customer requirements.
- more bandwidth is available for network management.
- equipment is smaller and cheaper.
- worldwide standards allow a larger manufacturers' marketplace.

- it is easier to int
- it is cheaper t
  between transm

Network flexibi
centre in order to:

- improve capac
  transported in t
- improve availa
  protection sche
- reduce mainten
- provide easier
  upgraded.

Flexibility can
systems or betwee
gradually replace
capacity within the
ability to extract o
high order signal, a

19.4.2 Synchro

The synchronous s
a frame structure s
to a framing (or m
used here is one i
rows and columns
the masterframe st

The signal bits
video signal, start
byte in row 1, an
Then the bits in th
of row 2, and so o

This transmis
duration of each f
the frame is 8 × 8
to one PCM voi
structure may be
The fixed sequen
streams within the

- it is easier to introduce new services.
- it is cheaper to achieve remote digital access to services and cross-connections between transmission systems.

Network flexibility implies the ability to rapidly reconfigure networks from a control centre in order to:

- improve capacity utilisation by maximising the number of 2 Mbit/s channels transported in the higher order system;
- improve availability of digital paths by centrally allocating spare capacity and protection schemes to meet service requirements;
- reduce maintenance costs by diverting traffic away from failed network elements;
- provide easier growth with temporary diversion of traffic around areas being upgraded.

Flexibility can be achieved using automatic cross-connect switches between SDH systems or between SDH and plesiochronous systems. Automatic cross-connects will gradually replace existing manual cross-connects and allow remote reconfiguration of capacity within the network at 2 Mbit/s and above. Add-drop multiplexing refers to the ability to extract or insert individual channels without the need to demultiplex the entire high order signal, as required in the plesiochronous system.

19.4.2 Synchronous signal structure

The synchronous signal comprises a set of 8-bit bytes which are carefully interleaved into a frame structure such that the identity of each byte is preserved and known with respect to a framing (or marker) word. The description of the synchronous signal frame structure used here is one in which the bytes of the signal are represented by boxes appearing in rows and columns on a 2-dimensional map, Figure 19.10 [Hawker]. (This is not unlike the masterframe structure of Figure 14.43.)

The signal bits are transmitted in a raster scanned sequence, similar to the lines on a video signal, starting with those in the top left hand byte, followed by those in the 2nd byte in row 1, and so on, until the bits in the *M*th (last) byte in row 1 are transmitted. Then the bits in the 1st byte of row 2 are transmitted, followed by the bits in the 2nd byte of row 2, and so on, until the bits in the *M*th byte of the *N*th (last) row are transmitted.

This transmission sequence repeats, the repetition rate being 8000 frame/s. The duration of each frame is, therefore, 125 μs and the bit rate associated with each byte in the frame is 8×8 kbit/s = 64 kbit/s. Each byte in the frame is thus equivalent in capacity to one PCM voice channel. One or more 8-bit bytes within the synchronous signal structure may be allocated to provide channel capacity for a lower rate (tributary) signal. The fixed sequence frame alignment word allows the positions of all individual data streams within the frame to be identified.

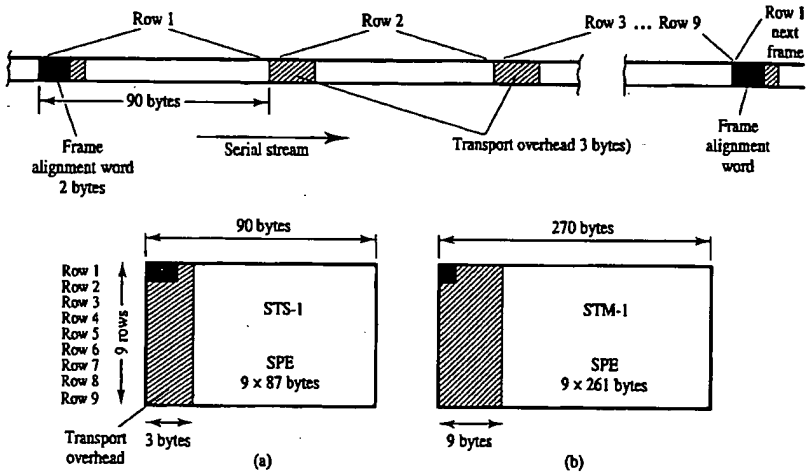


Figure 19.10 *Frame structures showing serial bit stream and the equivalent two dimensional data maps: (a) basic SONET STS-1 frame; and (b) SDH STM-1 frame.*

19.4.3 Frame structure

The lowest level SONET signal is called the synchronous transport signal level 1 (STS-1). It consists of 90 columns and 9 rows of 8-bit bytes giving a total 810 bytes (6480 bits), Figure 19.10(a). With a frame duration of 125 μs, the STS-1 bit rate is 51.84 Mbit/s. The basic SDH frame corresponds to three SONET STS-1 frames and is called a level 1 synchronous transport module (STM-1), Figure 19.10(b). The STM-1 bit rate is therefore 155.52 Mbit/s. Each STS-1 frame can be seen to comprise two parts – an overhead part and a service payload part.

Transport overhead: The first three columns of the STS-1 frame, a total of 27 bytes, Figure 19.10(a), are allocated to overheads that provide operations and maintenance (O&M) facilities. The three bytes in rows 1, 2 and 3 comprise the section overhead while the remaining three bytes in rows 4 to 9 (18 bytes) comprise the line overhead. The section and line terminology is defined in Figure 19.11 for a communications link. (Line terminating equipment might be represented, for example, by an add-drop multiplexer.)

Synchronous payload envelope (SPE): The remaining 87 columns of the STS-1 frame, a total of 783 bytes, provide a channel capacity of 50.112 Mbit/s which supports the transport of the service payload, or traffic data, and also the path overhead, Figure 19.12.

Path overhead: The path overhead supports and maintains the transportation of the SPE between the locations where the SPE is assembled and disassembled. It comprises a total of 9 bytes, Figure 19.12, and is allocated the first column (one byte wide) within the STS-1 SPE.

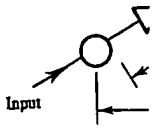


Figure 19.11 *Path, l*

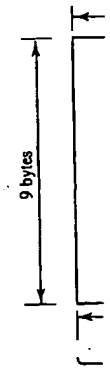


Figure 19.12 *STS-*

Payload capa
columns of 9 byte
with a frame repet
The SONET/S
cross-connect fun

- a modular stru
- extensive over
- byte interleavi
- byte stuffing t

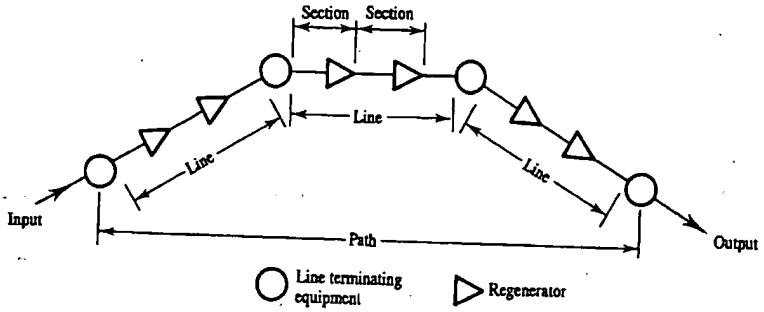


Figure 19.11 *Path, line and section details in a communications link.*

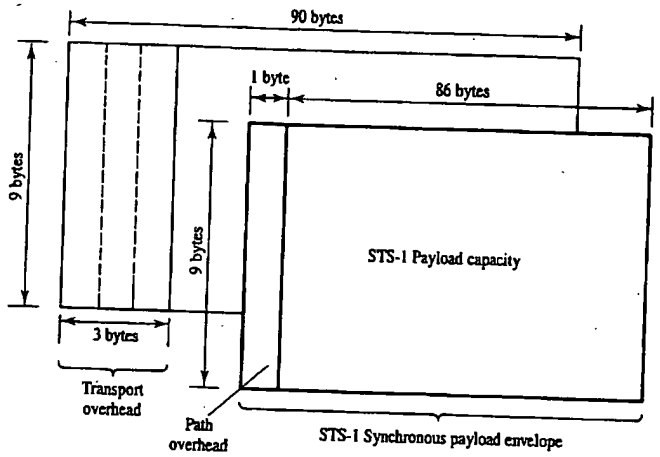


Figure 19.12 *STS-1 synchronous payload envelope (SPE).*

Payload capacity: The STS-1 information capacity comprises the remaining 86 columns of 9 bytes, i.e. a total of 774 bytes providing a 49.536 Mbit/s channel capacity with a frame repetition rate of 8 kHz.

The SONET/SDH frame structure features greatly simplified 'drop and insert' and cross-connect functions by utilising:

- a modular structure made up from SONET/SDH tributaries;
- extensive overheads for monitoring and control;
- byte interleaving with direct visibility of 64 kbit/s channels;
- byte stuffing to improve robustness against loss of synchronisation;

- standard mappings for all common data rates into the SONET/SDH frame format.

The STS-1 SPE represents the unshaded part of Figure 19.10(a). Additional transport capacity is obtained by effectively concatenating SPEs, Figure 19.10(b). Alternatively a reduction in transport capacity may be obtained by partitioning the SPE into smaller segments, called virtual tributaries or VTs (see section 19.4.5).

19.4.4 Payload pointer

To facilitate efficient multiplexing and cross-connection of signals in the synchronous network, the SPE is allowed to 'float' within the payload capacity provided by the STS-1 frames, Figure 19.13. This means that the STS-1 SPE may begin anywhere in the STS-1 frame and is unlikely to be wholly contained in one frame. More likely than not, the STS-1 SPE will begin in one frame and end in the next.

The STS-1 payload pointer, contained in the transport overhead, indicates the location of the first byte of the STS-1 SPE. Byte 1 of the STS-1 SPE is also the first byte of the SPE path overhead. It permits non-synchronous data to be accommodated within the SDH structure, without resorting to justification or bit stuffing.

For synchronous transport, payload pointers provide a means of allowing an SPE to be transferred between network nodes operating plesiochronously. Since the SPE floats freely within the transport frame, with the payload pointer value indicating the location of the first active byte of the SPE, the problem in justified plesiochronous multiplex where the traffic is mixed with stuffing bits is overcome.

Payload pointer processing does, however, introduce a new signal impairment known as 'tributary jitter'. This appears on a received tributary signal after recovery from a synchronous payload envelope which has been subjected to payload pointer movements from frame-to-frame. Excessive tributary jitter will influence the operation of the downstream plesiochronous network equipment processing the tributary signal.

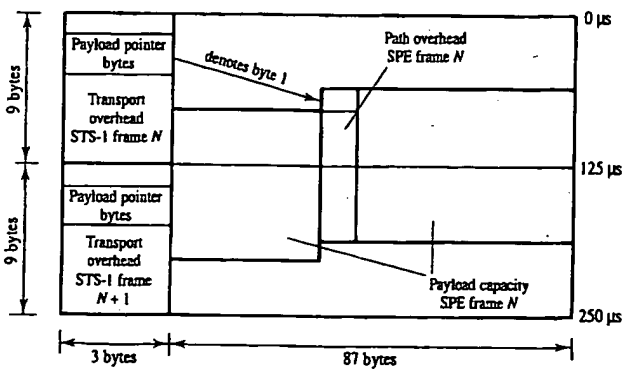


Figure 19.13 *Payload pointer details for locating the start of the SPE.*

19.4.5 Payload c

The virtual tributary
been designed to su
than that provided b
Figure 19.14. These

- VT1.5, consistir
  8 kHz, provides
  1.544 Mbit/s DS
- VT2, consisting
  capacity of 2.30
- VT3, consisting
  capacity of 3.45

These and othe
demand to handle d
plesiochronous mul
be easily written to.

A 155.52 Mbit/
SPEs into one STM
interleaving $N \times 1$
synchronous transp
format allows simu
(e.g. 45/140 Mbit/s

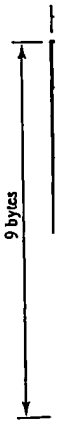


Figure 19.14 *Virtu*

19.4.5 Payload capacity partitioning

The virtual tributary (VT) structure (or tributary unit structure in SDH terminology) has been designed to support the transport and switching of payload capacity which is less than that provided by the full STS-1 SPE. There are three sizes of VTs in common use, Figure 19.14. These are:

- VT1.5, consisting of 27 bytes, structured as 3 columns of 9, which, at a frame rate of 8 kHz, provides a transport capacity of 1.728 Mbit/s and will accommodate a US 1.544 Mbit/s DS1 signal.
- VT2, consisting of 36 bytes, structured as 4 columns of 9, which provides a transport capacity of 2.304 Mbit/s and will accommodate a European 2.048 Mbit/s signal.
- VT3, consisting of 54 bytes, structured as 6 columns of 9, to achieve a transport capacity of 3.456 Mbit/s which will accommodate a US DS1C signal.

These and other VTs allow the SDH structure to be electronically reconfigured on demand to handle different customer requirements. It circumvents the fixed nature of the plesiochronous multiplex and, as the VTs are distributed over the entire payload, they can be easily written to, and read from, the system without requiring large data buffer stores.

A 155.52 Mbit/s SDH transmission capability is obtained by combining three STS-1 SPEs into one STM-1 SPE, Figure 19.10(b). Higher order systems are formed by byte interleaving $N \times 155$ Mbit/s channels (e.g. 16×155.5 Mbit/s = 2488 Mbit/s) to form the synchronous transport module level N (e.g. STM-16) rate, Figure 19.1. The SDH frame format allows simultaneous transport of both narrowband (e.g. 2 Mbit/s) and broadband (e.g. 45/140 Mbit/s) services within the 155 Mbit/s capacity system.

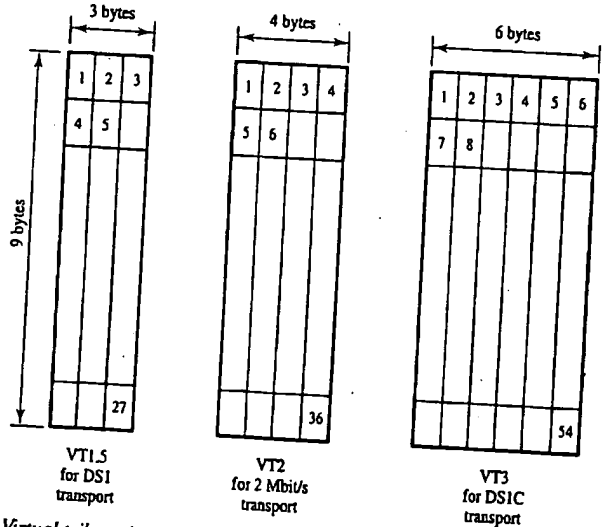


Figure 19.14 *Virtual tributaries of the STS-1 frame.*

EXAMPLE 19.2

Estimate the number of voice channels which can be accommodated by an SDH STM-4 signal assuming that the STM-4 is filled with ITU primary multiplex group signals. Also estimate the channel utilisation efficiency.

Each STS-1 payload envelope has 86 columns (not including the path overhead column). Each primary multiplex group occupies 4 columns. Each STS-1 payload envelope can therefore transport 86/4 = 21 (whole) primary multiplexes.

Each STM-1 payload envelope corresponds to 3 STS-1 payload envelopes and therefore carries $3 \times 21 = 63$ primary multiplexes. STM-4 carries 4 STM-1 signals and therefore carries $4 \times 63 = 252$ primary multiplexes. Each primary multiplex carries 30 voice channels, Figure 19.5. The STM-4 signal can, therefore, carry $30 \times 252 = 7560$ voice channels.

Channel utilisation efficiency, η_{ch}, is thus given by:

$$\eta_{ch} = \frac{7560}{R_b/R_v} = \frac{7560}{(622.08 \times 10^6)/(64 \times 10^3)}$$

$$= \frac{7560}{9720} = 78\%$$

19.5 Fibre optic transmission links

Optical fibres comprise a core, cladding and protective cover and are much lighter than metallic cables [Gowar]. This advantage, coupled with the rapid reduction in propagation loss to its current value of 0.2 dB/km or less, Figure 1.6(e), and the enormous potential bandwidth available, make optical fibre now the only serious contender for the majority of long-haul trunk transmission links. The potential capacity of optical fibres is such that all the radar, navigation and communication signals in the microwave and millimetre wave region, which now exist as free space signals, could be accommodated within 1% of the potential operational bandwidth of a single fibre. Current commercial systems can accommodate 31,000 simultaneous telephone calls in a single fibre and 1M call capacity has been achieved in the laboratory.

19.5.1 Fibre types

In the fibre a circular core of refractive index n_1 is surrounded by a cladding layer of refractive index n_2, where $n_2 < n_1$, Figure 19.15. This results in optical energy being guided along the core with minimal losses.

The size of the core and the nature of the refractive index change from n_1 to n_2 determine the three basic types of optical fibre [Gowar], namely:

- multimode step-index;
- multimode graded-index;

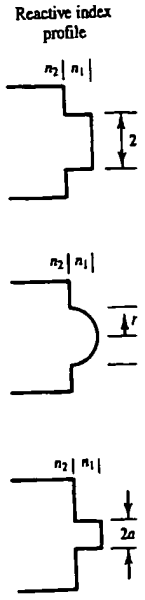


Figure 19.15 *Three*

- monomode ste

The refractive schematic ray dia are shown in Figu the refractive inde originally, a 20% In the more rec typically 0.5%. T had limited bandv of modern multin cladding, pulse br between the diffe 10 Mbit/s. Grade representing diffe propagation velo delay for all the approximately t monomode fibres dominant pulse

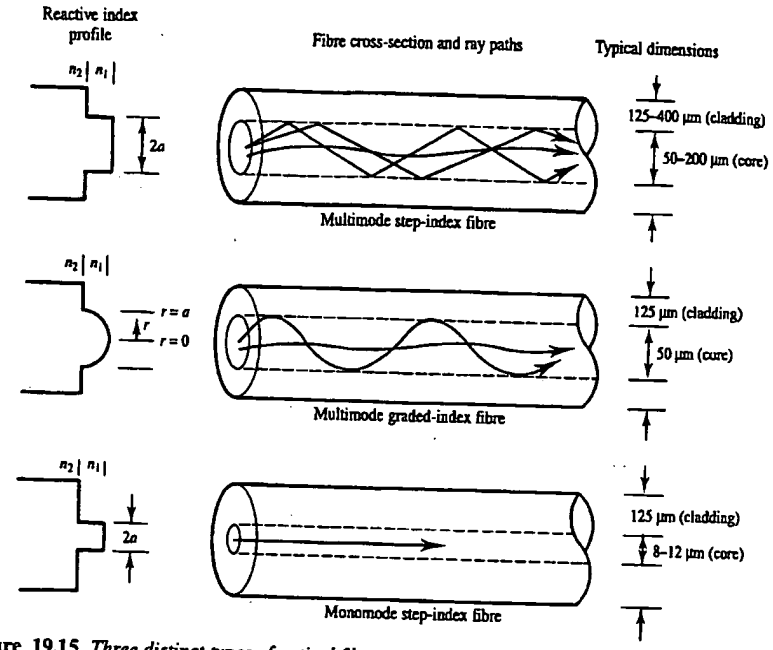


Figure 19.15 *Three distinct types of optical fibre.*

- monomode step-index.

The refractive index profiles, typical core and cladding layer diameters and a schematic ray diagram representing the distinct optical modes in these three fibre types are shown in Figure 19.15. The optical wave propagates down the fibre via reflections at the refractive index boundary or refraction in the core. In multimode fibres there was, originally, a 20% difference between the refractive indices of the core and the cladding. In the more recently developed monomode fibres, the difference is much smaller, typically 0.5%. The early multimode step index fibres were cheap to fabricate but they had limited bandwidth and a limited section length between repeaters. In a 1 km length of modern multimode fibre with a 1% difference in refractive index between core and cladding, pulse broadening or dispersion, caused by the difference in propagation velocity between the different electromagnetic modes, limits the maximum data rate to, typically, 10 Mbit/s. Graded index fibres suffer less from mode dispersion because the ray paths representing differing modes encounter material with differing refractive index. Since propagation velocity is higher in material with lower refractive index, the propagation delay for all the modes can, with careful design of the graded-n profile, be made approximately the same. For obvious reasons *modal* dispersion is absent from monomode fibres altogether and in these fibre types *material* dispersion is normally the dominant pulse spreading mechanism. Material dispersion occurs because refractive

index is, generally, a function of wavelength, and different frequency components therefore propagate with different velocities. (Material dispersion is exacerbated due to the fact that practical sources often emit light with a narrow, but not monochromatic, spectrum.) The rate of change of propagation velocity with frequency (dv/df), and therefore dispersion, in silica fibres changes sign at around 1.3 μm, Figure 19.16, resulting in zero material dispersion at this wavelength. (Fortuitously, this wavelength also corresponds to a local minimum in optical attenuation, see Figure 1.6(e).)

If both modal and material dispersion are zero, or very small, *waveguide* dispersion, which is generally the weakest of the dispersion mechanisms, may become significant. Waveguide dispersion arises because the velocity of a waveguide mode depends on the normalised dimensions (d/λ) of the waveguide supporting it. Since the different frequency components in the transmitted pulse have different wavelengths these components will travel at different velocities even though they exist as the same electromagnetic mode. Because both material and waveguide dispersion relate to changes in propagation velocity with wavelength they are sometimes referred to collectively as chromatic dispersion. The various fibre types are further defined in ITU-T recommendation G.652.

19.5.2 Fibre transmission systems

There have been three generations of optical fibre systems operating at 0.85 μm, 1.3 μm and 1.5 μm wavelengths to progressively exploit lower optical attenuation, Figure 1.6(e), and permit longer distances to be achieved between repeaters. Monomode fibres, with core diameters in the range 8 to 12 μm, have been designed at the two longer wavelengths for second and third generation systems. Since material dispersion is zero at around 1.3

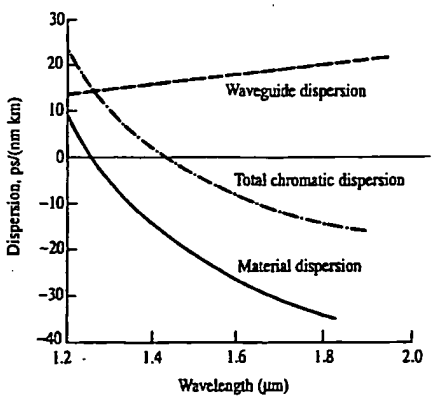


Figure 19.16 *Variation of material dispersion and waveguide dispersion, giving zero total dispersion near λ = 1.5 μm (source, Flood and Cochrane, 1989, reproduced with the permission of Peter Peregrinus).*

μm, where the op
wavelength chosen

Third generatic
Mbit/s to exploit t
resulting increased
Thus, the choice at
wants to maximise

Figure 19.16 sl
chosen correctly, h
at about 1.5 μm re
band.

The impact of
illustrates the evol
coaxial cable used
failures (MTBF) o
plesiochronous mu
still needed a repe
breakthrough cam
1.5 μm, there is no
loss is typically 1
section 12.4.2.) E
80% of the UK t
transmission capa
capacity doubles e

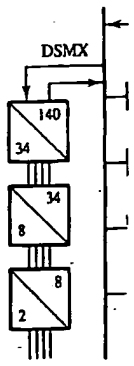


Figure 19.17 *Evo*
199

μm, where the optical attenuation in silica is also a local minimum, this was the wavelength chosen for second generation systems, which typically operate at 280 Mbit/s.

Third generation systems operate at wavelengths around 1.5 μm and bit rates of 622 Mbit/s to exploit the lowest optical attenuation value of 0.15 dB/km, and tolerate the resulting increased chromatic dispersion which in practice may be 15 to 20 ps $nm^{-1}km^{-1}$. Thus, the choice at present between 1.3 and 1.5 μm wavelength depends on whether one wants to maximise link repeater spacing or signalling bandwidth.

Figure 19.16 shows that if core dimensions, and core cladding refractive indices, are chosen correctly, however, material dispersion can be cancelled by waveguide dispersion at about 1.5 μm resulting in very low total chromatic dispersion in this lowest attenuation band.

The impact of fibre developments is clearly seen in Figures 1.7 and 19.17. The latter illustrates the evolution of transmission technology for a 100 km wideband link. The coaxial cable used in the 1970s with its associated 50 repeaters had a mean time between failures (MTBF) of 0.4 years, which was much lower than the 2 year MTBF of the plesiochronous multiplex equipment at each terminal station. Multi-mode fibre (MMF) still needed a repeater every 2 km but was a more reliable transmission medium. The real breakthrough came with single mode fibre (SMF) and, with the low optical attenuation at 1.5 μm, there is now no need for any repeaters on a 100 km link, in which the fibre path loss is typically 10 to 28 dB. (This is *very* much lower than the microwave systems of section 12.4.2.) By 1991 there were 1.5 million km of installed optical fibre carrying 80% of the UK telephone traffic. This UK investment represented 20% of the world transmission capability installed in optical fibre at that time. Optical fibre transmission capacity doubles each year with an exponentially reducing cost.

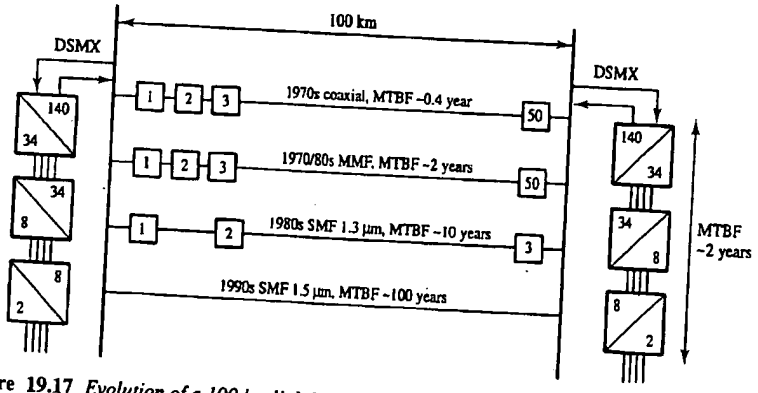


Figure 19.17 *Evolution of a 100 km link from coaxial to optical transmission (source: Cochrane, 1990, reproduced with the permission of British Telecommunications plc.).*

19.5.3 Optical sources

Two devices are commonly used to generate light for fibre optic communications systems: light-emitting diodes (LEDs) and injection laser diodes (ILDs). The edge emitting LED is a PN junction diode made from a semiconductor material such as aluminium–gallium–arsenide (AlGaAs) or gallium–arsenide–phosphide (GaAsP). The wavelength of light emitted is typically 0.94 μm and output power is approximately 3 mW at 100 mA of forward diode current. The primary disadvantage of this type of LED is the non-directionality of its light emission which makes it a poor choice as a light source for fibre optic systems. The planar hetero-junction LED generates a more brilliant light spot which is easier to couple into the fibre. It can also be switched at higher speeds to accommodate wider signal bandwidth.

The injection laser diode (ILD) is similar to the LED but, above the threshold current, an ILD oscillates and lasing occurs. The construction of the ILD is similar to that of an LED, except that the ends are highly polished. The mirror-like ends trap photons in the active region which, as they are reflected back and forth, stimulate free electrons to recombine with holes at a higher-than-normal energy level to achieve the lasing process. ILDs are particularly effective because the optical radiation is easy to couple into the fibre. Also the ILD is powerful, typically giving 10 dB more output power than the equivalent LED, thus permitting operation over longer distances. Finally, ILDs generate close to monochromatic light, which is especially desirable for single mode fibres operating at high bit rates.

19.5.4 Optical detectors

There are two devices that are commonly used to detect light energy in fibre optic systems: PIN (positive–intrinsic–negative) diodes and APDs (avalanche photodiodes). In the PIN diode, the most common device, light falls on the intrinsic material and photons are absorbed by electrons, generating electron-hole pairs which are swept out of the device by the applied electric field. The APD is a positive–intrinsic–positive–negative structure, which operates just below its avalanche breakdown voltage to achieve an internal gain. Consequently, APDs are more sensitive than PIN diodes, each photon typically producing 100 electrons, their outputs therefore requiring less additional amplification.

19.5.5 Optical amplifiers

In many optical systems it is necessary to amplify the light signal to compensate for fibre losses. Light can be detected, converted to an electrical signal and then amplified conventionally before remodulating the semiconductor source for the next stage of the communications link. Optical amplifiers, based on semiconductor or fibre elements employing both linear and non-linear devices, are much more attractive and reliable; they permit a range of optical signals (at different wavelengths) to be amplified simultaneously and are especially significant for sub-marine cable systems.

Basic travellir
the range 10 to
operation and are
with distributed f
common with all
results in an optic
cascaded TWSLA

Recent resear
or undoped fibre
mechanism or th
erbium doped fib
fibre pumped w
coupling of fibre-
30 to 50 nm optic
15 dB. The key
achievable and t
exceeds the 300
Two interesting
wavelength via
effects. They ca
and can operate e

Injecting a h
Introducing a lo
energy, results i
power, coupled
Figure 19.18.

Brillouin sca
realise high gain

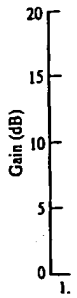


Figure 19.18 *Co*
19

Basic travelling wave semiconductor laser amplifier (TWSLA) gains are typically in the range 10 to 15 dB, Figure 19.18. These Fabry-Perot lasers are multimode in operation and are used in medium distance systems. Single mode operation is possible with distributed feedback (DFB) lasers for longer distance, high bit rate, systems. In common with all optical amplifiers, the TWSLA generates spontaneous emissions which results in an optical (noise) output in the absence of an input signal. For a system with cascaded TWSLAs these noise terms can accumulate.

Recent research has resulted in fibre amplifiers consisting of 10 m to 50 km of doped or undoped fibre. These amplifiers use either a linear, rare earth (erbium), doping mechanism or the non-linear Raman/Brillouin mechanism [Cochrane *et al.* 1990]. The erbium doped fibre amplifier (EDFA) uses a relatively short section (1 to 100 m) of silica fibre pumped with optical, rather than electrical, energy. Because of the efficient coupling of fibre-to-fibre splices, high gains (20 dB) are achievable, Figure 19.18, over a 30 to 50 nm optical bandwidth. Practical amplifier designs generally have gains of 10 to 15 dB. The key attraction of this amplifier is the excellent end-to-end link SNR which is achievable and the enormous 4 to 7 THz (Hz $\times 10^{12}$) of optical bandwidth. (This far exceeds the 300 GHz of the entire radio, microwave and millimeter-wave spectrum.) Two interesting features of these amplifiers are the precise definition of the operating wavelength via the erbium doping and their relative immunity to signal dependent effects. They can therefore be engineered to maintain wide bandwidths when cascaded, and can operate equally well with OOK, FSK or PSK signals.

Injecting a high power laser beam into an optical fibre results in Raman scattering. Introducing a lower intensity signal-bearing beam into the same fibre with the pump energy, results in its amplification with gains of approximately 15 dB per W of pump power, coupled with bandwidths that are slightly smaller than the erbium amplifiers, Figure 19.18.

Brillouin scattering is a very efficient non-linear amplification mechanism that can realise high gains with modest optical pump powers (approximately 1 mW). However,

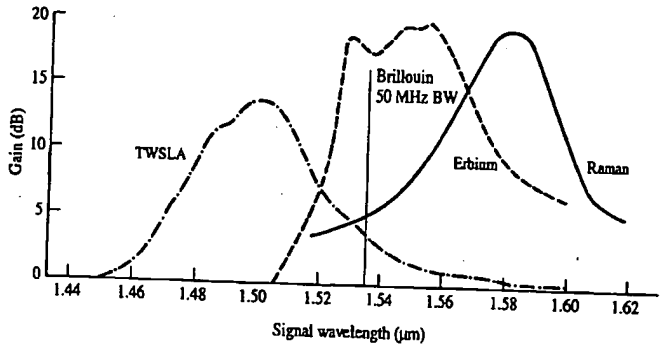


Figure 19.18 *Comparison of the gain of four distinct optical amplifier types (source: Cochrane, 1990, reproduced with the permission of British Telecommunications plc.).*

the bandwidth of only 50 MHz, Figure 19.18, is very limited in pure silica which makes such devices more applicable as narrowband tunable filters. This limited bandwidth fundamentally restricts the brillouin amplifier to relatively low bit rate communication systems.

A comprehensive comparison of the features of optical amplifiers is premature. However, what is clear is that long-haul optical transmission systems (10,000 km) are now feasible, with fibre amplifier based repeaters, and bit rates of 2 to 10 Gbit/s. At the present time TWSLA and erbium amplifiers generally require similar electrical pump power but TWSLAs achieve less gain, Figure 19.18, due to coupling losses. Erbium fibre amplifiers have the lowest noise and WDM channel crosstalk performance of all the amplifier types reported but high splice reflectivities can cause these amplifiers to enter the lasing condition. With the exception of brillouin, these amplifiers can be expected to be used across a broad range of system applications including transmitter power amplifiers, receiver preamplifiers, in-line repeater amplifiers and switches. The key advantage of EDFAs is that the lack of conversion to, and from, electrical signals for amplification gives rise to the 'dark fibre' – a highly reliable data super highway operating at tens of Gbit/s.

19.5.6 Optical repeater and link budgets

The electro-optic repeater is similar to the metallic line regenerator of Figures 6.15 and 6.30. For monomode fibre systems the light emitter is a laser diode and the detector uses an APD. The symbol timing recovery circuit uses zero crossing detection of the equalised received signal followed by pulse regeneration and filtering to generate the necessary sampling signals. Due to the high data rates, the filters often use surface acoustic wave devices [Matthews] exploiting their high frequency operation combined with acceptable Q value. The received SNR is given by:

$$\frac{S}{N} = \frac{I_p^2}{2q_e B(I_p + I_D) + I_n^2} \tag{19.1}$$

where I_p is the photodetector current, I_D is the leakage current, q_e is the charge on an electron, B is bandwidth and I_n is the RMS thermal noise current given by:

$$I_n^2 = \frac{4kTB}{R_L} \tag{19.2}$$

For received power levels of –30 dBm, $I_n^2 >> 2q_e B(I_p + I_D)$ giving, typically, a 15 to 20 dB SNR and hence an OOK BER in the range 10^{-7} to $< 10^{-10}$ [Alexander].

There are many current developments concerned with realising monolithic integrated electro-optic receivers. Integrated receivers can operate with sensitivities of –20 to –30 dBm and a BER of 10^{-10} at 155, 625 and 2488 Mbit/s, with a 20 dB optical overload capability. They are optimised for low crosstalk with other multiplex channels.

The power budget for a typical link in a fibre transmission system might have a transmitted power of 3 dBm, a 60 km path loss of 28 dB, a 1 dB path dispersion allowance and 4 dB system margin to give a –30 dBm received signal level which is

consistent with
with a similar
interfaces on suc
Long haul ex
using up to 12 c
10^{-8} have been
which the indivi

EXAMPLE 19.3
A monomode, 1.3

Optical output of
Connector loss at
Fibre specific atte
Average fibre spli
Fibre lengths, d
Connector loss at
Design margin (in
Required optical c

Find the maximur

Let estimated lo
distributed over th

Total

Allo

Therefore:

8.0 +

$D' =$

The assumption c

ΔL_S

This excess loss
extended by:

consistent with a low cost 155 Mbit/s transmission rate. The higher rate of 2.5 Gbit/s, with a similar receiver sensitivity of –30 dBm, would necessitate superior optical interfaces on such a 60 km link.

Long haul experiments and trials have achieved bit rates from 140 Mbit/s to 10 Gbit/s using up to 12 cascaded amplifiers spanning approximately 1000 km. BERs of 10^{-4} to 10^{-8} have been measured on a 500 km, five amplifier system, operating at 565 Mbit/s in which the individual amplifier gains were 7 to 12 dB [Cochrane *et al.* 1993].

EXAMPLE 19.3

A monomode, 1.3 μm, optical fibre communications system has the following specification:

Optical output of transmitter, P_T	0.0 dBm
Connector loss at transmitter, L_T	2.0 dB
Fibre specific attenuation, γ	0.6 dB/km
Average fibre splice (joint) loss, L_S	0.2 dB
Fibre lengths, d	2.0 km
Connector loss at receiver, L_R	1.0 dB
Design margin (including dispersion allowance), M	5.0 dB
Required optical carrier power at receiver, C	–30 dBm

Find the maximum loss-limited link length which can be operated without repeaters.

Let estimated loss-limited link length be D' km and assume, initially, that the splice loss is distributed over the entire fibre length.

$$\text{Total loss} = L_T + (D' \times \gamma) + \left(\frac{D'}{d} \times L_S\right) + L_R + M$$

$$= 2.0 + 0.6D' + \left(\frac{0.2}{2.0} \times D'\right) + 1.0 + 5.0$$

$$= 8.0 + 0.7D' \text{ dB}$$

$$\text{Allowed loss} = P_T - C = 0.0 - (-30) = 30.0 \text{ dB}$$

Therefore:

$$8.0 + 0.7D' = 30$$

$$D' = \frac{30.0 - 8.0}{0.7} = 31.4 \text{ km}$$

The assumption of distributed splice loss means that this loss has been over estimated by:

$$\Delta L_S = L_s [\, D' - int\,(D'/d)d \,] / 2$$

$$= 0.2 [\, 31.4 - int\,(31.4/2)2 \,] / 2$$

$$= 0.14 \text{ dB}$$

This excess loss can be reallocated to fibre specific attenuation allowing the link length to be extended by:

$$\Delta D = \Delta L_S/\gamma = 0.14/0.6 = 0.2 \text{ km}$$

The maximum link length, D, therefore becomes:

$$D = D' + \Delta D = 31.4 + 0.2 = 31.6 \text{ km}$$

19.5.7 Optical FDM

With the theoretical 50 THz of available bandwidth in an optical fibre transmission system, and with the modest linewidth of modern optical sources, it is now possible to implement optical FDM and transmit multiple optical carriers along a single fibre. The optical carriers might typically be spaced by 1 nm wavelengths. With the aid of optical filters these signals can be separated in the receiver to realise wavelength division multiplex (WDM) communications [Oliphant *et al.*]. It is envisaged that eventually up to 100 separate channels could be accommodated using this technique but the insertion loss of the multiplexers and crosstalk between channels still needs to be assessed. It has been demonstrated that 10 such combined signals, each modulated at 10 Gbit/s, can be transmitted through a practical fibre, and amplified using a single fibre amplifier without having to demultiplex the signals. WDM promises to increase by a hundredfold the information carrying capacity of fibre based systems when the necessary components for modulators and demodulators are fully developed.

Soliton transmission uses pulses that retain their shape for path lengths of thousands of kilometres due to the reciprocal effects of chromatic dispersion and a refractive index which is a function of intensity. Such systems have been constructed for 1,000 km paths with bit rates of 10 to 50 Gbit/s. In the laboratory, 10^6 km recirculating links have been demonstrated, corresponding to many circulations of the earth before the received SNR is unacceptable [Cochrane *et al.* 1993].

19.6 Network advantages of SDH systems

In the current plesiochronous hierarchy, within the transport signal at 140 Mbit/s, we may want to route component signals, for example 2 Mbit/s streams or tributaries, through the network. This requires us to demultiplex the transport signals layer-by-layer through the hierarchy, switch the tributary signals, and then remultiplex them into the next transport signal, Figure 19.9.

In the SDH, individual component signals do not have to be demultiplexed to their original bit rate; instead they are incorporated into a signal called a 'container' which can be handled in a convenient way throughout the network, Figure 19.14. Direct access to these component signals is thus possible, Figure 19.19. The result is a considerable reduction in multiplexer hardware in SDH systems, combined with improved operational flexibility.

With the introduction of SDH the opportunity can now be taken to replace network layers and topologies with those better suited to long haul resilient networks. With the

Figure 19.19 *Ad*
mu

availability of,
reconsider the st
paths between th
line systems, by
(input line, outp
provide ring acc
PDN and is fu
Figure 19.20 rep
independent clo
described in C
information to t
intended destina

Outer-core
exchanges, in cc
central site. Th
much more reli
has less interfa
topologies, but

19.7 Data a

19.7.1 ISDN c

ISDN digital ac
kbit/s and prim
19.1. The cus
communication
channel.

Basic rate
channels at 6
International st
Communicatio
signalling or d

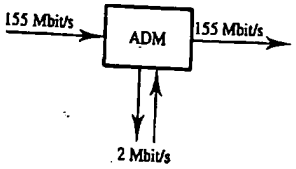


Figure 19.19 *Add-drop multiplexer (ADM) for simplified channel dropping which permits multiplexing into ring networks. (Compare with Figure 19.9.)*

availability of, very high speed, flexible SDH links it now becomes economic to reconsider the structure of the PSTN and replace simple (multiple) two-way transmission paths between the major centres, in which terminal multiplexers are two port or tributary-line systems, by high speed optical rings, as illustrated in Figure 19.20. The three port (input line, output line and tributary) add-drop multiplexers (ADMs), Figure 19.19, provide ring access and egress. This structure (Figure 19.20) will form the heart of the PDN and is fundamentally more reliable and less costly than the previous solution. Figure 19.20 represents an enhanced version of Figure 18.14. When the rings incorporate independent clockwise and anti-clockwise transmission circuits, as in the FDDI example described in Chapter 18, they offer immense flexibility and redundancy allowing information to be transmitted, via the ADM, in either direction around the ring to its intended destination.

Outer-core topologies will be mainly rings of SDH multiplexers linking local exchanges, in contrast to a plesiochronous multiplex where all traffic is routed through a central site. The SDH ring structure with its clockwise and anti-clockwise routing is much more reliable than centre-site routing. Furthermore, the SDH ring based network has less interface, and other, equipment. Access regions will remain, principally, star topologies, but will probably be implemented using optical technology, Figure 19.21.

19.7 Data access

19.7.1 ISDN data access

ISDN digital access was opened in the UK in 1985 and provided basic rate access at 144 kbit/s and primary rate access at 2 Mbit/s to the multiplex hierarchical structure of Figure 19.1. The customer interface for primary rate access provides for up to 30 PCM communications channels (e.g. a PABX) under the control of a common signalling channel.

Basic rate access provides the customer with two independent communications channels at 64 kbit/s together with a common signalling channel at 16 kbit/s. International standards for ISDN access (I.420) have now been agreed within ITU. Communications or bearer channels (B-channels) operate at 64 kbit/s, whilst the signalling or data channel (D-channel) operates at 16 kbit/s giving the basic rate total of

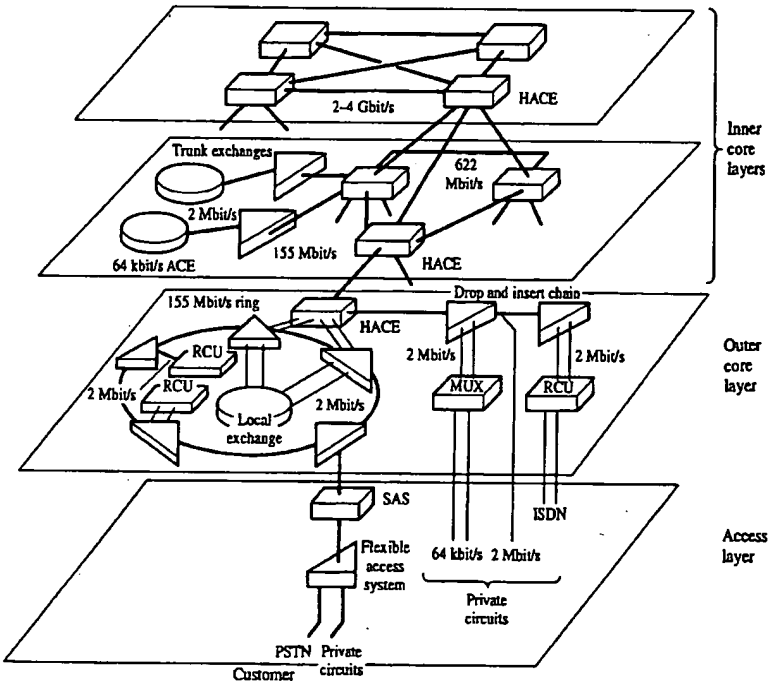


Figure 19.20 *Probable future SDH transmission network hierarchy (source: Leakey, 1991, reproduced with the permission of British Telecommunications plc.).*

144 kbit/s. The basic rate voice/data terminal transmits, full duplex, over a two-wire link with a reach of up to 2 km using standard telephone local loop copper cables. The transceiver integrated circuits employ a 256 kbaud, modified DPSK, burst modulation technique, Chapter 11, to minimise RFI/EMI and crosstalk. The D channel is used for signalling to establish (initiate) and disestablish (terminate) calls via standard protocols. During a call there is no signalling information and hence the D channel is available for packet switched data transmission.

Data access at 64 kbit/s is used in low bit rate image coders for videophone applications, Chapter 16. For high quality two way confravision services with full TV (512 × 512 pixel) resolution reduced bit rate coders have been designed to use primary access at 2 Mbit/s. Access at 2 Mbit/s is also required to implement wide area networks, Chapter 18, or to carry cellular telephone traffic between cell sites, Chapter 15.

TDM/WDM fibre trunk

Figure 19.21 *Pr*

The layer 1 the physical c (network termi are basically e equipment (TE

TE2

Figure 19.22 *E*

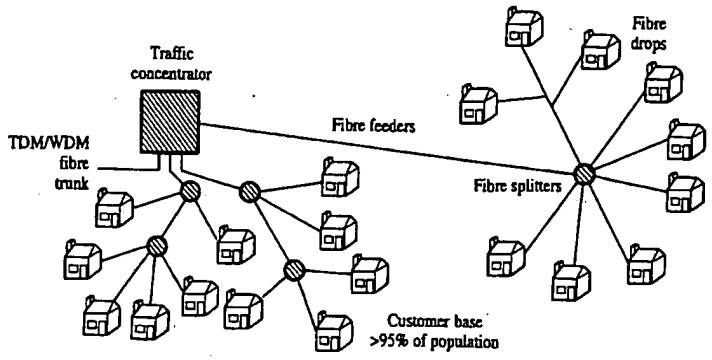


Figure 19.21 *Passive optical network for future local loop implementation.*

The layer 1 specification based on ITU-T I series recommendations [Fogarty] defines the physical characteristics of the user-network interface, Figure 19.22. The NT1 (network termination 1) terminates the transmission system and includes functions which are basically equivalent to layer 1 of the OSI architecture, Figure 18.12. The terminal equipment (TE) includes the ITU-T NT2 (network termination 2) function which

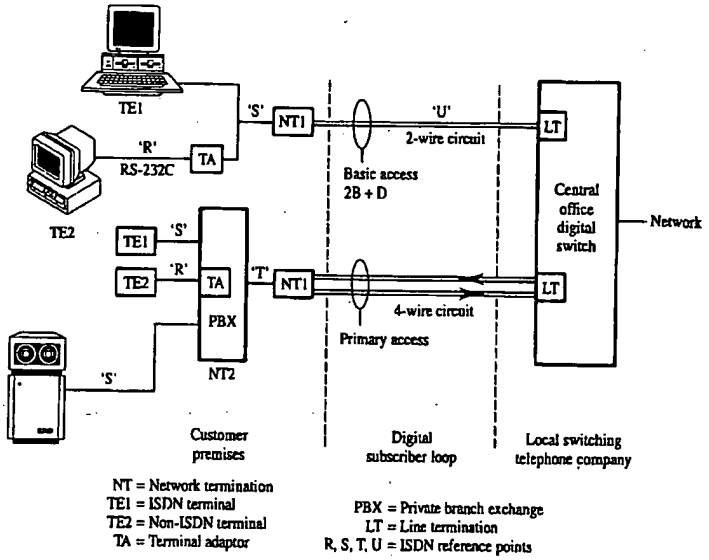


Figure 19.22 *Basic rate (144 kbit/s) and primary rate (2 Mbit/s) access to the ISDN.*

terminates ISO layers 1 to 3 of the interface. Figure 19.22 illustrates how digitised speech and data have access to the ISDN and includes the R, S, T and U ISDN reference points, which can be interconnected by standard interface integrated circuits.

Examples of TE1 equipments are ISDN telephone or fax machines which use the S interface. A TE2 might be a V.24 (RS232) data terminal or computer which requires the terminal adaptor (TA) to interface with the ISDN. The TA performs the processing to establish and disestablish calls over the ISDN and handles the higher level OSI protocol processing. For computer connection the TA is usually incorporated in the PC.

19.7.2 STM and ATM

Synchronous transfer mode (STM) and asynchronous transfer mode (ATM) both refer to techniques which deal with the allocation of usable bandwidth to user services. STM as used here is not to be confused with the synchronous transport module defined in section 19.4.3. In a digital voice network, STM allocates information blocks, at regular intervals (bytes/125 μs). Each STM channel is identified by the position of its time slots within the frame, Figure 19.5, which could easily be extended to a 30 channel, 2 Mbit/s system. STM works best when the network is handling a single service, such as the continuous bit-rate (bandwidth) requirements of voice, or a limited heterogeneous mix of services at fixed channel rates. Now, however, a dynamically changing mix of services requires a much broader range of bandwidths, and a switching capability adequate for both continuous traffic (such as voice and video conferencing) and non-continuous traffic (such as high-speed data and coded video traffic) in which bandwidth may change with time depending on the information rate.

While STM can provide data transfer services, the network operator must supply, and charge for, facilities with the full bandwidth needed by each service for 100% of the time – even if users require the peak bandwidth for only a small fraction of the time. The network therefore operates at low efficiency and the cost to users is prohibitive. This is not an attractive option for variable bit rate (VBR) traffic, such as coded video transmission, in which the data rate is dependent on how fast the image is changing.

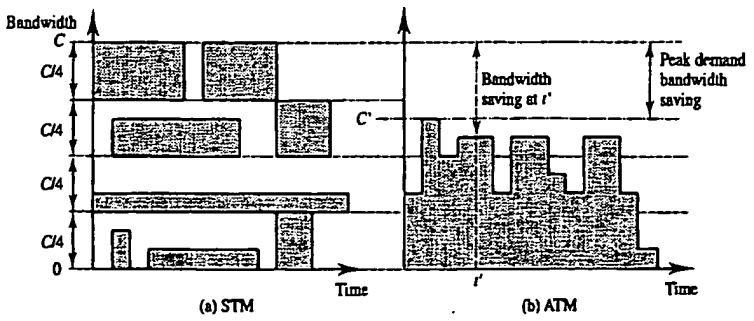


Figure 19.23 *Example of (a) STM and (b) ATM with mix of fixed and variable bit rate traffic.*

Figure 16.19 s
much smaller t
transmission ra

The structu
sharing both ba
fixed, predeter
Figure 19.23(a)
service require
system, Chapte
traffic. When
users, there is
overhead is sm
representing a
audio, and 12
generally, a co
traffic applicati
for data applic
10.8.1, is imple

M

to achieve a
comparison to
(149.76 Mbit/s
hardware VLSI
network nodes.

One of the
LANs and PCs
ATM layer all
switched netw
efficiency, end
carried over lo
the power of
much enhanced

The term a
connection may
actual demand
transmission te
services in a gr
ISDN, for pac
Chapter 17, all
switches descr
seamlessly into
directly into the

Figure 16.19 shows an example of such traffic for which the average data rate is very much smaller than the peak rate. When many VBR sources are averaged then the peak transmission rate requirement comes much closer to the average rate.

The structure of the ATM protocol improves on the limited flexibility of STM by sharing both bandwidth and time [de Prycker]. Instead of breaking down bandwidth into fixed, predetermined, channels to carry information, as shown schematically for STM in Figure 19.23(a), ATM transfers fixed-size blocks of information, called cells, whenever a service requires transmission bandwidth, in a manner analogous to a packet switched system, Chapter 17. Figure 19.23(b) illustrates the (statistical) multiplexing of the ATM traffic. When the transmission bandwidth is dynamically allocated to variable bit rate users, there is a consequent peak bandwidth saving ($C' < C$), provided that the packet overhead is small. ATM uses a 5 byte header combined with a 48 byte data packet representing a compromise between the optimum data packet lengths of 16 bytes, for audio, and 128 bytes for video traffic. (The 53 bit ATM cell also represents, more generally, a compromise between short cell lengths required for real time, low delay, traffic applications such as packet speech, and long cell lengths which are more efficient for data applications due to their reduced proportion of overhead bits.) CRC, section 10.8.1, is implemented on the header information only using the polynomial:

$$M(x) = x^8 + x^2 + x^1 + 1 \tag{19.3}$$

to achieve a single error correction, but multiple error detection, capability. In comparison to other packet data networks (e.g. packet speech) ATM achieves high speed (149.76 Mbit/s corresponding to the payload bit rate of the SDH STM-1) by using hardware VLSI chips, rather than software, for the protocol processing and switching at network nodes.

One of the first services to use an ATM network for efficient data transport between LANs and PCs was a high-speed switched data service which carried VBR traffic. The ATM layer allows much higher data transfer rates than is possible with X.25 packet-switched networks (Chapter 18). As a result of ATM's remarkable flexibility and efficiency, end users can enjoy very-high-bandwidth services. These services can be carried over long distances by the PSTN with, potentially, very attractive tariffs. When the power of ATM is joined with the bandwidth and transmission quality of ISDN, a much enhanced service is achieved.

The term asynchronous in ATM refers to the fact that cells allocated to the same connection may exhibit an irregular occurrence pattern, as cells are filled according to the actual demand. This is an unfortunate term as it implies that ATM is an asynchronous transmission technique which is not the case. ATM facilitates the introduction of new services in a gradual and flexible manner, and is now the preferred access technique to the ISDN, for packet speech, video and other variable bit rate traffic. Queuing theory, Chapter 17, allows the analysis of ATM cell throughput rates and losses. The routing switches described in Chapter 18 are used for the ATM interfaces. ATM can fit seamlessly into the SDH frames of Figures 19.12 and 19.13 by accommodating the cells directly into the SDH payload envelope.

19.7.3 The local loop

Half of the investment of a telephone company is in the connections between subscriber handsets and their local exchange. Furthermore, this part of the network generates the least revenue since local calls are often cheap or, as in the USA, free. The length of these connections is 2 km on average and they seldom exceed 7 km. In rural areas the expense of installing copper connections now favours radio access for the local loop implementation. ISDN access demands 144 kbit/s duplex operation over 4 to 5 km which requires sophisticated signal processing if copper pair cables are used.

Assuming that the local loop is to be used for speech telephony and low speed data connections only, its replacement by fibre systems will be very gradual. If, however, we were to combine this requirement with cable TV and many other broadband services such as videophone, Chapter 16, then there would be an immediate demand for wideband local loop connections.

There will thus be a progressive move from copper based conductors to a fibre based passive optical network (PON) for the local connection. This will not be based on the current structure of one dedicated wire pair, or fibre, per household because of the large fibre bandwidth, section 19.5. (Furthermore, with 750 million telephones worldwide it would take more than 300 years for manufacturers to produce all the required cable at current production rates!) The future local network is therefore likely to comprise wideband fibre feeders with splitters and subsequent single fibre drops to each household, Figure 19.21. (One problem with the PON configuration is that there is no longer a copper connection to carry the power required for standby telephone operation.)

19.8 Summary

Multiplexing of PCM–TDM telephone traffic has been traditionally provided using the plesiochronous digital hierarchy. The PDH frame rate is 8000 frame/s and its multiplexing levels, bit rates and constituent signals are as follows:

PDH-1	2.048 Mbit/s	30+2, byte interleaved, 64 kbit/s voice channels – the PCM primary multiplex group
PDH-2	8.448 Mbit/s	4 bit-interleaved PDH-1 signals
PDH-3	34.368 Mbit/s	4 bit-interleaved PDH-2 signals
PDH-4	139.264 Mbit/s	4 bit-interleaved PDH-3 signals

Bit rates increase by a little more than a factor of four at each successive PDH level to allow for small differences in multiplexer clock speeds. Empty slots in a multiplexer output are filled with justification bits as necessary. A serious disadvantage of PDH multiplexing is the multiplex mountain which must be scaled each time a lower level signal is added to, or dropped from, a higher order signal. This is necessary because bit interleaving combined with the presence of justification bits means that complete demultiplexing is required in order to identify the bytes belonging to a given set of voice

channels.

The synch
equivalent, SO
that low level
lower level mu
higher order m
cross-connecti
8000 frame/s
(STMs). The
synchronous tr
rates, and cons

SON
SD
SD
SD

SONET si
The capacity o
the operation c

SONET a
transmission s
Laser diodes
performance,
dispersion and
contain PIN d
more sensitive
powers are –3
detector, a con
TWSLAs) ma
and consist of
signal propaga

Fibres cur
1.3 and 1.5 μ
refractive ind
dimensions a
Multimode gr
refractive inde
in larger avai
dispersion wh
(material disp
electromagnet
contributions
which also co

channels.

The synchronous digital hierarchy (SDH) and its originating North American equivalent, SONET, will eventually replace the PDH. The principal advantage of SDH is that low level signals remain visible in the multiplexing frame structure. This allows lower level multiplexes (down to individual voice channels) to be added or dropped from higher order multiplex signals without demultiplexing the entire frame. This simplifies cross-connection of traffic from one signal multiplex to another. The SDH frame rate is 8000 frame/s and each frame contains one or more synchronous transport modules (STMs). The SONET frame rate is also 8000 frame/s and each contains one or more synchronous transport signals (STSs). The standard SONET/STM payload capacities, bit rates, and constituent signals are as follows:

SONET	STS-1	9 × 90 bytes	51.84 Mbit/s	
SDH	STM-1	9 × 270 bytes	155.52 Mbit/s	3 STS-1s
SDH	STM-4	9 × 1080 bytes	622.08 Mbit/s	4 STM-1s
SDH	STM-16	9 × 4320 bytes	2.488 Gbit/s	4 STM-4s

SONET signals with capacities based on other multiples of STS-1 are also possible. The capacity of an STM-1 is such that it can carry one PDH-4 signal which will facilitate the operation of PSTNs during the period in which both multiplexing schemes are in use.

SONET and SDH have been designed, primarily, to operate with optical fibre transmission systems. Optical sources may be coherent [Hooijmans] or incoherent. Laser diodes have narrow spectra and are therefore usually the choice for high performance, high bit rate links. LEDs are less spectrally pure, leading to greater dispersion and smaller useful bandwidth, but are cheaper. Optical detectors typically contain PIN diodes or avalanche photodiodes (APDs). APDs are more expensive but more sensitive. Typical optical transmit powers are 0 dBm and typical optical receiver powers are −30 dBm. An optical fibre repeater may be implemented using an optical detector, a conventional electronic repeater and an optical source. Optical amplifiers (e.g. TWSLAs) may also be used. The most recent types of optical amplifier are distributed and consist of doped fibre sections which are optically pumped and lase as the optical signal propagates through them.

Fibres currently operate, in decreasing order of attenuation, at wavelengths of 0.85, 1.3 and 1.5 μm. They can be divided into three types depending on the profile of their refractive index variations. Multimode step-index fibres have relatively large core dimensions and suffer from modal dispersion which limits their useful bandwidth. Multimode graded-index fibres also have large core dimensions but use their variation in refractive index to offset the difference in propagation velocity between modes resulting in larger available bandwidth. Monomode (step-index) fibres suffer only chromatic dispersion which arises partly from the frequency dependence of refractive index (material dispersion) and partly from the frequency dependence of a propagating electromagnetic mode velocity in a waveguide of fixed dimensions. These two contributions can be made to cancel at an operating wavelength around 1.5 μm, however, which also corresponds to a minimum in optical attenuation of about 0.15 dB/km. This

wavelength is therefore an excellent choice for long, low dispersion, high bit rate, links. Repeaterless links, hundreds of kilometres long, operating at Gbit/s data rates are now possible. Wavelength division multiplexing (WDM) promises to increase the communications capacity of a single fibre still further.

SDH, combined with optical fibre transmission, has allowed a re-evaluation of the PDN network topology. Future access is likely to be via a passive optical network (PON) at ISDN basic (144 kbit/s) or primary (2 Mbit/s) rates. 155 Mbit/s rings will connect to a network of higher-order automatic cross-connect equipment (HACE) which will themselves be interconnected at bit rates of 155 and 622 Mbit/s and higher. The highest layer of cross-connect equipment will be fully interconnected with 2.4 Gbit/s links.

Asynchronous transfer mode (ATM) will provide efficient use of time and bandwidth resources in the access layers of the network when variable bit rate services are provided. ATM frames consist of 53 byte cells, 48 of which carry traffic and 5 of which carry overhead. At the higher network layers ATM cells will be carried within the payload envelopes of SDH frames.

19.9 Problems

19.1. How does a plesiochronous multiplex function?

19.2. Explain the notion of section, line and path as entities in an SDH transmission network. Taking an STS-1 frame as an example, where is the information concerning these entities carried in the SDH signal? What is the nature of the information?

19.3. What mechanism is used to allow an SDH SPE to pass between SDH networks which are not synchronised? How does this differ from the mechanism used to allow tributaries from the old plesiochronous system (e.g. 2.048 Mbit/s) to be taken in and out of an SDH system?

19.4. Explain the term add-drop in the context of multiplexers. Draw block diagrams to show why this function is simpler to perform in the SDH system than in the older PDH system.

APPE

Tabulated val

e

x	erf x
0.00	0.000000
0.01	0.011283
0.02	0.022565
0.03	0.033841
0.04	0.045111
0.05	0.056372
0.06	0.067622
0.07	0.078858
0.08	0.090078
0.09	0.101282
0.10	0.112463
0.11	0.123623
0.12	0.134758
0.13	0.145867
0.14	0.156947
0.15	0.167996
0.16	0.179012
0.17	0.189992
0.18	0.200936
0.19	0.21184(
0.20	0.222703
0.21	0.233522
0.22	0.24429(
0.23	0.255023
0.24	0.26570(
0.25	0.27632(
0.26	0.28690(
0.27	0.29741(
0.28	0.30788(
0.29	0.31828:
0.30	0.32862:
0.31	0.33890(
0.32	0.34912(
0.33	0.35927:
0.34	0.36936:
0.35	0.37938:
0.36	0.38933(
0.37	0.39920(
0.38	0.40900:
0.39	0.41873:

Table 10.4 Terminology

CCITT	USA	UK
Primary centre	Toll centre	Group switching centre
Secondary centre	Primary centre	District switching centre
Tertiary centre	Sectional centre	Main switching centre
Trunk exchange	Toll office	Trunk exchange
Trunk network	Toll network	Trunk network
Trunk circuit	Trunk	Trunk (circuit)
Local exchange	End (central) office	Local exchange
Junction circuit	Inter-office trunk	Junction

be used. The words used in North America and the UK differ significantly in their meaning. Here we will refer to the various levels of switching by using the CCITT recommended terms. Table 10.4 relates them to those commonly used in North America and the UK.

From Fig. 10.35 we can see that there are several levels of switching that combine to form the complete network. It is usual to think of the systems in two parts. The first is the junction network, serving the subscriber and consisting of the link from subscriber to local exchange, from local exchange to primary switching centre, and back to the called subscriber via another local exchange. The second part of the system is the trunk network, which is concerned only with calls passing at primary centre level and above. Thus the primary centre is associated with both parts of the network.

The structure shown by the solid lines represents the basic hierarchical network of the system, and telephone calls routed along these links are said to be travelling on a backbone or final route. When there is much traffic between exchanges, either belonging to the same cluster, or at different levels, direct routes may be installed to save a stage of switching. Such routes are shown as dotted lines in Fig. 10.35. These direct routes are dimensioned to a different grade of service from the final routes. Because a call finding all the direct-route circuits busy can be placed on the backbone route, the direct route can be dimensioned to a higher grade of service than the final route, and thus work more efficiently.

The number of exchanges at the various levels depends on several factors: the physical extent of the network, the number of subscribers, the amount of traffic, the forecast growth and the transmission methods used. Beyond the top level of the national system is a layer that gives access to the international network. This layer may consist of one or more international (usually called gateway) exchanges.

10.32 Numbering Schemes

In modern systems, the numbering scheme used by a telephone administration to allocate subscribers' numbers has an underlying plan, and there are a few constraints on the development of the plan that must be taken into account:

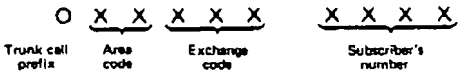


Fig. 10.36 National telephone number.

(i) It must provide each subscriber with a unique number within the national network.
(ii) The allocation to areas must be able to meet forecast growth for several decades.
(iii) The number of digits should not exceed that recommended by CCITT.

In principal, (i) is easy to satisfy if (ii) and (iii) have been met. The length of the number recommended by CCITT is $11 - n$ where n is the country code (see below). If, for example, n is 2, then the national number should not exceed 9 digits in length. These digits are used to denote the subscriber's number on the local exchange, the exchange within a given area, and the area within the national numbering scheme. In many local exchanges there is a maximum capacity of 10 000 lines, thus the last four digits of the national number are allocated to the subscriber's number in the exchange. Of the remaining five digits in our example, the first two would denote the area and the remaining three the exchange within the area. Thus the number has the form shown in Fig. 10.36.

The division of digits between area exchange and subscribers code may not be the same as that shown – but all three components exist in every number.

For automatic long-distance dialling a prefix is necessary to indicate to the exchange equipment that a trunk call is being made. In many countries a "0" is used, but any other digit would do. In calculating the length of the national number the prefix is not included.

International Number

In the introduction to this chapter we imagined a future when person-to-person calls could be made very easily via individual instruments and satellite links. For such availability of connections to be possible each subscriber in the world must have a unique number. To achieve that, some agreement between countries is essential; each country must be identified by a number different from that of all other countries. That is achieved by agreement through CCITT on the way in which these codes, called country codes, are allocated.

The first digit of the country code is the zone code; the world is divided up into nine zones and each country belongs to a zone. The relationship between zone number and geographical area is shown in Table 10.5.

In all but two of the zones, one or two digits are added to the zone number to produce the country code. For example, Brazil has a zone number 5 and an additional country code digit 5 to give its country code 55. Brunei is in zone 6 and two further digits are added to give a country code 673.

The two exceptions are zones 1 (North America and the Caribbean) and 7

Table 10.5 Zone numbers

1	North America	6	Australasia
2	Africa	7	USSR
3	Europe	8	Eastern Asia
4	Europe	9	Far East and Middle East
5	South America	0	Spare

(USSR). Throughout each of these zones there is a linked numbering scheme that means, for example, that no subscriber in Canada has the same national number as a subscriber in the USA. Consequently, to connect to anyone in zone 1 the digit 1 is followed by the national number. A similar situation exists in the USSR. Europe is at the other extreme; there are many countries with large national networks that have nine digits in their national numbers. For these, a two-digit country code is required, and that can only be achieved by having two zone numbers allocated to Europe.

The division of the world into the zones shown in Table 10.5 is intended to be satisfactory until early in the next century, but clearly, as some large countries develop their telephone networks, some adjustment will be necessary at some future time.

The national and international numbering schemes we have discussed above are the simplest. However, in several parts of the world there are small exceptions, particularly in regard to local calls. In the scheme where the national number is used for all calls within a country, it can lead to irritation on the part of the subscriber and long set-up times for the exchange equipment. Consequently, in many countries local calls use a shorter code. For calls within the same area, the area code is omitted, and for small single-exchange areas no exchange code is used for own-exchange calls. Coupled with this last arrangement will be a very short code for calls to adjacent exchanges; these arrangements are particularly well suited to rural areas. The disadvantage of short codes is that they change with the location of the calling subscriber, and therefore a short code directory must be available in each exchange area.

10.33 Routeing Calls

The early type of switching equipment, called step-by-step or Strowger, operated by using the dialled pulses to move the selectors to the position corresponding to the digit dialled. In many ways this was an excellent system, but one major disadvantage was that it allowed no flexibility in the way calls were routed – the route was predetermined by the dialled digits. Although some systems were modified to overcome this problem it was not until common-control equipment became widely used that the path a call took between calling and called subscribers could be chosen to allow the most efficient use of the available capacity in the system. The function of the common control in the routing process was to store the dialled digits in a register and then translate them into routing digits which would indicate to the switching system the path to take through the network. This register – translator combination is essential to automatic trunk and international dialling schemes; it allows the telephone administration to manage the system efficiently by changing routes as circumstances alter without having to change subscribers' numbers. This therefore separates the subscriber from the system. The subscriber dials the national number from any location and the register–translator automatically selects an appropriate route.

10.34 Digital Systems

Several factors have acted to push telephony from analogue to digital working. To make such a major change, telephone administrations and equipment manufacturers have been persuaded that it is in digital operation that the future lies, and the decade from the mid-seventies has been characterized in all equipment producing countries by huge investments of manpower and plant in a race to manufacture an efficient digital telecommunications system. There are more manufacturers involved than at any other time and great financial commitments have been made in the hope that a market will be available for the many products being developed.

Traditionally the two basic elements of a telephone system were transmission and

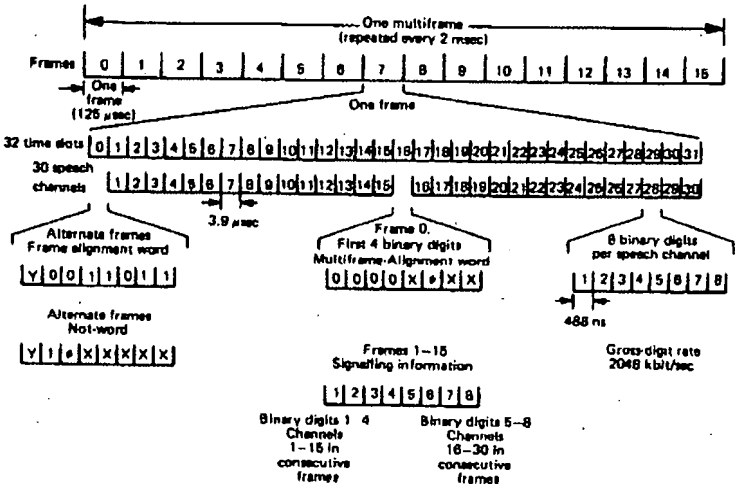


Fig. 10.37 32-frame PCM multiplex frame arrangement: *x*, digits not allocated to any particular function and set to state *one*; *y*, reserved for international use (normally set to state *one*); 0, digits normally *zero* but changed to *one* when loss of frame alignment occurs and/or system-fail alarm occurs (TS0 only) or when loss of multiframe alignment occurs (TS16 only).

switching. However, with digital operation the whole system is considered as an entity.

Following the development of PCM several countries installed PCM links between analogue exchanges. These worked in a basic 24- or 32-channel format with an encoder and decoder. We can see how the 32-channel system is formed with reference to Fig. 10.37. Strictly speaking, it should be referred to as a 30-channel, 32-time-slot system, because two of the time slots contain signalling and synchronizing information, not speech samples. The sampling rate of each speech channel determines the length of each time slot. For telephony the sampling rate used is 8 kHz, which means that the time between adjacent samples of the same channel is 125 μsec. One frame, consisting of 32 time-slots lasts for this time. A time slot is therefore approximately 3.9 μsec long. For reasons that we shall see in a moment, 16 frames are put together to form a multi-frame which has a time span of 2 msec. This is the basic unit of the PCM system applied to telephony.

Within a time slot there is the encoded information about the speech sample for that channel. In most systems it is coded into 256 ($= 2^8$) levels and there are therefore 8 binary digits within each time slot; each digit must be less than 0.48 μsec. long. The technique of PCM is given in detail in Section 3.5.

The channel digits bearing speech information must be sent to the right destination and monitored for release and clear-down. Signals must therefore be associated with each channel and in PCM telephony that is done by allocating a signalling word to each channel once per multi-frame. Sufficient information for signalling can be contained in a 4-digit word so that an 8 digit word can contain two signals. Hence a signalling time slot can contain enough information for two signals.

Figure 10.37 indicates that time slot sixteen (TS16) is used to carry the signalling. In the second frame it carries the signals related to speech channels 0 and 15, in the third frame those for channels 1 and 16 and so on until frame 16 when TS16 has signals for speech channels 14 and 29. The next frame is the first of the following multi-frame and the sequence is repeated.

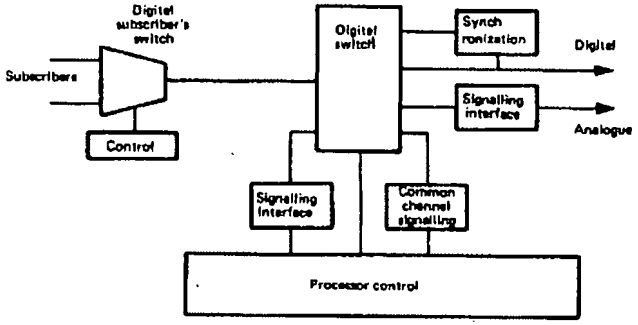


Fig. 10.38 Digital telephone exchange.

TS0 in each frame and TS16 in the first frame, carry synchronization and alignment words to ensure that the transmission and reception of the system is maintained in step.

Modern digital electronics, with its fast logic and complex integrated circuitry, has provided the techniques for producing digital switching and control systems at a cost comparable with analogue, coupled with an ability to provide better facilities, cheaper maintenance and more flexibility. The increasingly widespread use of data in various forms, and the need to send such information over large distances, has also encouraged the development of digital telecommunications and it has led to the intention to integrate together speech and data links.

Digital exchanges consist of the basic components showing in Fig. 10.38. Most of the control of the system is handled by microprocessor devices, singly or in clusters, which are driven by software. Here we are not able to discuss the huge new field of telecommunications software engineering, but it provides the most important challenge in modern system design. The software must be efficient, reliable, secure, understandable and well documented. In theory it affords a degree of flexibility in the operation of the system which is much higher than is possible in hard-wire control. However, such is the complexity of large telephone networks that software development presents the most difficult problems to designers, for it must last for tens of years and although made up of very long programs, it must be able to cope with dramatic changes in hardware technology.

In analogue exchanges the usual figure of merit used is the grade of service, or probability of blocking, but in digital switches the blocking is virtually zero. In these systems the major problems concern delay in the processing of calls caused by the processor units becoming overloaded. The analysis of such problems is very difficult and if simple queueing theory models do not apply resort must be made to computer simulation. One of the difficult tasks of the software engineer is to produce a satisfactory compromise between short efficient programs and those that are longer, more complex and more reliable and secure.

Referring again to Fig. 10.38, look first at the digital switch. It can have many structural forms, depending on the size of the system and the technology used, but as an example we will consider the common time–space–time (TST) configuration. TST is a shorthand for the time-switch, space-switch, time-switch arrangement shown in Fig. 10.39(a).

The time switch is split into several units, each having M PCM links of L channels, as Fig. 10.39(b). Consequently, if the time switch is non-blocking it will have an outlet highway of $N = ML$ time slots. The space switch is square with R inlet highways and R outlet highways. The purpose of the TST unit is to allow a particular call, which occupies a specific channel into one of the time switches, to be connected to a particular outlet channel. Basically the switching is between highways on either side of the space switch. Each highway has N time slots and in order for a particular call to be connected say from $H1$ to $H3$ it must find a time slot which is free in both highways. This slot may not be the same as the required incoming and outgoing slots for the call, and so some time delay, provided by the time switches, is necessary. The method used to produce the delay again depends on

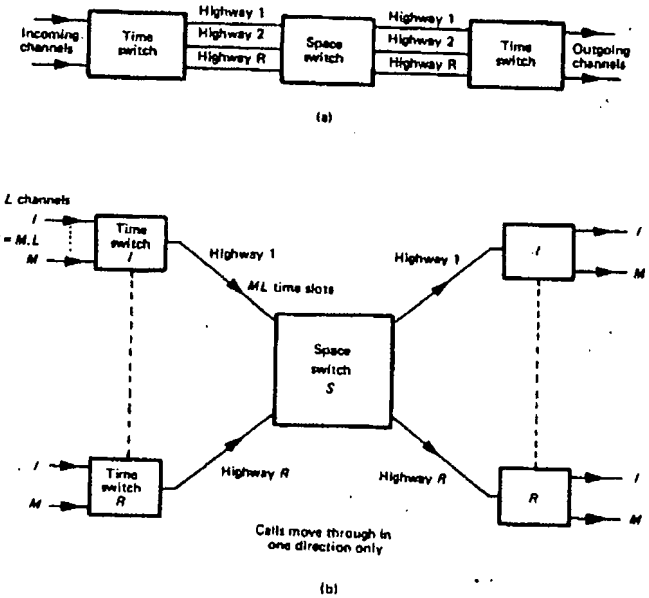


Fig. 10.39 Digital switch: (a) TST block diagram; (b) network representation.

the technology employed, but the delay may be from 1 to 31 time slots depending on the relative positions of the time slots in and out of the unit, and the chosen free time slot in the space switch.

To understand the behaviour of the TST switch in terms of the link systems considered earlier it is important to appreciate that for each time slot the interconnections in the space switch will be different; at each time slot there will be a different set of calls in progress and the connections between the highways will only last for one time slot period then new connections will be established. This can be represented by having N space switches (Fig. 10.40), one for each time slot.

Whether or not blocking occurs in the TST unit depends entirely on the dimensions of the space switch, and since in modern systems switches are comparatively inexpensive they are usually large enough to make blocking negligible. For total non-blocking there must be at least as many outlets as inlets on the time switches, and the space switch highways must have $2N - 1$ time slots, where N is the number of time slots in a link to a time switch.

Digital switches are uni-directional, and that implies that two paths are required to connect two channels X and Y, one for conversation from X to Y and the other

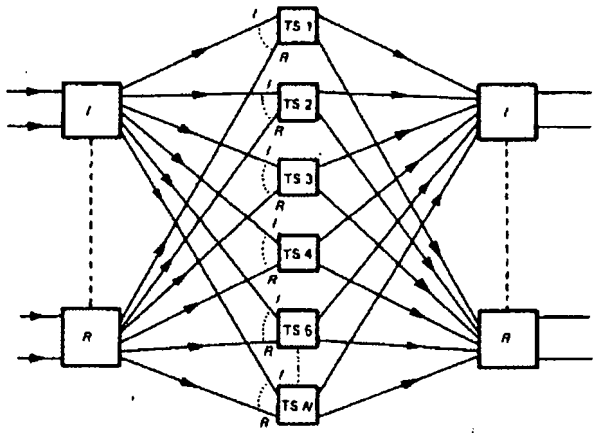


Fig. 10.40 Analogue equivalent of TST switch.

for conversation from Y to X. To reduce the control process, the X to Y slot is chosen according to whatever rules are used by the designer, and the Y to X interconnection is allocated a fixed number of time-slots from it, e.g. one, or half a frame. By this method, if the X to Y connection is available, the Y to X must also be free.

The digital switch just described is situated in a main exchange, forming part of the trunk network. Interconnections between exchanges are made, for the information paths, via PCM links. However, signalling is carried over a common channel, using signalling system CCITT No. 7 as described in Section 10.3. The inter-exchange signals will not only be concerned with setting up calls between exchanges, but with accounting, administration fault diagnosis and maintenance. As described earlier, the CCITT No. 7 system transmits signals as messages that can have variable length. Each message is preceded by labels that identify its origin and destination exchange, the type of message (call handling, fault, etc.) and includes error-detection and acknowledgement bits. If an error is detected in a message, that and all subsequent messages are retransmitted to ensure that the sequence received at the far end is in the correct order.

The error rate has an important bearing on the capacity of the signalling channel; re-transmission of incorrectly received messages obviously takes up time that could be used for new signals and consequently slows down the overall process. The capacity is specified in terms of number of messages per busy hour given that the delay from end to end is not greater than some predetermined value.

On the subscriber side of the digital switch there will be a local unit of some description. For large areas a local digital exchange would be used with mf signalling from subscriber to exchange where conversion to a PCM format would

take place before concentration through a digital switch. Alternatively for very small units no exchange facility would be available, but a simple digital concentrator would be used to take in the analogue channels, convert them to PCM and multiplex them onto a single highway to the nearest local exchange. Calls between subscribers on the same concentrator would then have to pass through the local exchange. Signalling in these PCM links would be on TS16 and the control, software and firmware at the main exchange would convert it to common channel if a trunk call was required.

The introduction of digital systems is rarely a starting point for the telephone system. Usually a system exists and the digital equipment has to be grafted on to it. Whatever method is used, interworking between the old and new systems is required, and one area of difficulty is the interfacing of various analogue signalling systems with the new equipment designed to operate on TS16 or common channel. This interfacing can be a severe problem if there are many existing signalling schemes in a particular network, and the development of satisfactory units can add considerably to system costs.

10.35 Conclusion

In some respects, the whole concept of telecommunications is changing, and with it the services and functions provided by telecommunications operating companies. The very rapid growth of information technology, with its demand for high-speed, large-capacity, data links, with video output, and the introduction of new facilities on telephone exchanges, are both causing manufacturers to rethink the whole philosophy of how communication systems should best be provided. In the next few years the main point at issue will be to decide whether future systems should be fully integrated, or not. If so, should they be based on a digital telephone system, with its centralized switching units, or on one of the many data networks, with distributed control? Conversely, economy and efficiency may dictate an extension of the present hybrid scheme in which both methods co-exist, with perhaps an element of inter-working between them. It is too early in the development of distributed systems to predict what changes will take place in the next decade, but that major redesign will occur is beyond doubt.

References

1. Brockmeyer, B., Halstron, H.L. and Jensen, A., *The Life and Works of A.K. Erlang*, Copenhagen Telephone Company, 1943.
2. Jacobaeus, C., "A study on congestion in link systems", *Ericsson Technics*, No. 48, 1950.
3. Cooper, R.B., *Introduction to Queueing Theory*, Edward Arnold, London, 1981, Chapter 5.
4. Fry, T.C., *Probability and its Engineering Uses*, Van Nostrand Reinhold, Wokingham, 1965.

5. Fishman, G.S., *Principles of Discrete Event Simulation*, Wiley, Chichester, 1978.
6. Flood, J.E., *Telecommunication Networks*, Peter Perigrinus, London, 1975.
7. Bear, D., *Telecommunication Traffic Engineering*, Peter Perigrinus, London, 1976.
8. Hills, M.T., *Telecommunication Switching Principles*, George Allen & Unwin, London, 1979.

Problems

10.1 A traffic-recording machine takes measurements of the number of busy devices in a group every three minutes during the busy hour. If the sum of the devices busy over that period is 600, what is the value of the traffic carried?

Answer: 30 erlangs.

10.2 A loss-system full availability group consists of five devices. If the mean call holding time is 180 sec, and the call intensity is 80 calls/hour, what is the mean load per device?

Answer: 0.8 erlangs.

10.3 In a particular system, it was found that during the busy hour, the average number of calls in progress simultaneously in a certain full availability group of circuits was 15. All circuits were busy for a total of 30 sec during the busy hour. Calculate the traffic offered to the group.

Answer: 15.13 erlangs.

10.4 A group of 8 circuits is offered 6 erlangs of traffic. Find the time congestion of the group, and calculate how much traffic is lost.

Answer: 0.122 erlangs; 0.73 erlangs.

10.5 A ninth circuit is added to the group in Question 10.1. What traffic will it carry?

Answer: 5.55 erlangs.

10.6 A system of six telephones has full availability access to six devices. Find the probability that $1, 2, \ldots, 6$ devices are busy. What is (a) the call, and (b) the time congestion of the system if the carried traffic is 2.4 erlangs?

Answer: (a) 0; (b) 0.004.

10.7 (a) Two erlangs of traffic are fed to three devices. What is the congestion, and how much traffic is lost?
(b) Two erlangs of traffic are fed to one device. The overflow is fed to a

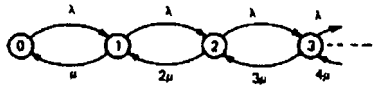
second device, and the overflow from that to a third. What is the overall congestion, and is the value of the traffic lost the same as in (a)?

Answer: (a) 0.2105, 0.421 erlangs; (b) 0.432, No.

10.8 Show that, if the assumption of statistical equilibrium is valid, the probability of a system being in state i is given in terms of the probability that it is in state 0 by

$$[i] = \frac{\prod_{j=0}^{i-1} \lambda_j}{\prod_{j=1}^{i} \mu_j} [0]$$

10.9 The state transition diagram below represents a system with an infinite number of devices subjected to calls arriving at random with fixed mean arrival rate, λ



If $\lambda/\mu = A$, the mean offered traffic, use the birth and death equations, and assume statistical equilibrium, to show that

$$[i] = \frac{A^i \exp(-A)}{i!}$$

10.10 An infinite number of sources feed 5 erlangs of traffic into 8 devices. Find the probability of the network being in each of its possible states, i, and check that $\sum_i [i] = 1$. Plot $[i]$ against i.

Answer: 0.0072, 0.0362, 0.0904, 0.1504, 0.1883, 0.1883, 0.1564, 0.1121, 0.0702.

10.11 Repeat question 10 for (i) 2 erlangs, and (ii) 7 erlangs of traffic, noting the variation in both the congestion and the distribution of $[i]$, with increasing traffic.

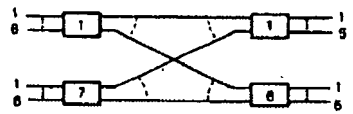
Answer: (i) 0.1354, 0.2707, 0.2707, 0.1905, 0.0902, 0.0361, 0.0120, 0.0035, 0.0009; (ii) 0.0013, 0.0088, 0.0306, 0.0715, 0.1251, 0.1751, 0.2044, 0.2044, 0.1788.

10.12 A telephone route of n circuits has to carry a normal load of 3 erlangs. If the grade of service must not exceed 0.03, what is the smallest value that n

can have? In an emergency, there is a 20% increase in offered traffic. What will be the grade of service in this overload condition?

Answer: $n = 7$, 0.045.

10.13 Draw the chicken diagram equivalent of the two-stage link network shown below, and calculate the internal blocking.

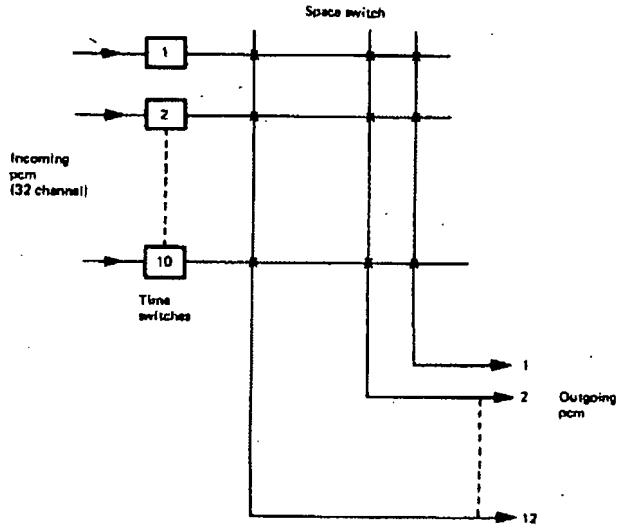


Corresponding outlets from second-stage switches form a route.

Traffic carried per route = 2.5 erlangs
Traffic carried per inlet switch = 1.8 erlangs

Answer: 0.0285.

10.14 A time–space digital switching configuration is shown. What are the functions of the time and space elements in relation to the speech channels on the PCM inlets and outlets.? Draw the analogue equivalent of this switch.



Bruce Schneier

Crypto Bibliography

Citations by First Author - B

A. Back, U. Möller, and A. Stiglic, Traffic Analysis Attacks and Trade-Offs in Anonymity Providing Systems, Proceedings of the 4th Information Hiding Workshop (IHW2001), Springer-Verlag, LNCS v. 2137, pp. 243-254. [.pdf]

S. Bakhtiari, R. Safavi-Naini, and J. Pieprzyk, A Message Authentication Code based on Latin Squares, Australian Conference on Information Security and Privacy (ACISP '97), Springer-Verlag, LNCS 1270, pp. 194-203, 1997. [.ps.Z]

S. Bakhtiari, R. Safavi-Naini, and J. Pieprzyk, On Password-Based Authenticated Key Exchange using Collisionful Hash Functions. In Australian Conference on Information Security and Privacy (ACISP '96), Springer-Verlag, LNCS 1172, pp. 299-310, 1996. [.ps.Z]

S. Bakhtiari, R. Safavi-Naini, and J. Piprzyk, On Selectable Collisionful Hash Functions, Australian Conference on Information Security and Privacy (ACISP '96), Springer-Verlag, LNCS 1172, pages 287-298, 1996. [.ps.Z]

T. Baldin, G. Bleumer, and R. Kanne, CryptoManager - Eine intuitive Programmierschnittstelle für kryptographische Systeme; Sicherheitsschnittstellen - Konzepte, Anwendungen und Einsatzbeispiele, Proc. Workshop Security Application Programming Interfaces 94, Deutscher Universitäts Verlag, München 1994, 79-94. [.ps.gz]

T. Baldin and G. Bleumer, CryptoManager++ -- An object oriented software library for cryptographic mechanisms; 12th IFIP International Conference on Information Security (IFIP/Sec '96), Chapman & Hall, London 1996, 489-491. [.ps.gz]

D. Balfanz and L. Gong, Experience with Secure Multi-Processing in Java, Proceedings of the 18th IEEE International Conference on Distributed Computing Systems (ICDCS), Amsterdam, Netherlands, May 1998. [.ps.gz]

J. Bar-Ilan and D. Beaver, Non-Cryptographic Fault-Tolerant Computing in a Constant Expected Number of Rounds of Interaction (extended abstract); Proceedings of **PODC**, ACM, 1989, 201-209. [.pdf]

R. Bar-Yehuda, B. Chor, E. Kushilevitz, and A. Orlitsky, Privacy, Additional Information, and Communication, IEEE IT 39(6), 1993, pp. 1930-1943. [.ps.Z]

N. Baric and B. Pfitzmann, Collision-Free Accumulators and Fail-Stop Signature Schemes Without Trees; Eurocrypt '97, LNCS 1233, Springer-Verlag, Berlin 1997, 480-494. [.ps.gz]

E. Basturk, M. Bellare, C. S. Chow, and R. Guerin, Secure transport protocols for high-speed networks, IBM Research Report 19981, March, 1994.

O. Baudron, H. Gilbert, L. Granboulan, H. Handschuh, A. Joux, P. Nguyen, F. Nolihan, D. Pointcheval, T. Pornin, G. Poupard, J. Stern, and S. Vaudenay, Report on the AES Candidates, Proceedings of the Second AES Candidate Conference, Rome, Italy, 1999. [.pdf]

B. Baum-Waidner, B. Pfitzmann, and M. Waidner, Unconditional Byzantine Agreement with Good Majority; STACS'91, LNCS 480, Springer-Verlag, Heidelberg 1991, 285-295. [.ps.gz]

D. Bayer, S. Haber, and W. Stornetta, Improving the Efficiency and Reliability of Digital Time-Stamping, Sequences II: Methods in Communication, Security, and Computer Science, eds. R. Capocelli, A. DeSantis, and U. Vaccaro, Springer-Verlag, 1993, pp. 329-334. [.pdf]

P. Beauchemin, G. Brassard, C. Crépeau, C. Goutier, and C. Pomerance, Two observations on probabilistic primality testing; In Advances in Cryptology: Proceedings of Crypto '86, volume 263 of Lecture Notes in Computer Science, pages 443-450. Springer-Verlag, 1987. [.ps.gz]

P. Beauchemin, G. Brassard, C. Crépeau, C. Goutier, and C. Pomerance, The generation of random

numbers that are probably prime, Journal of Cryptology, 1(1):53-64, 1988. [.ps]

D. Beaver, S. Micali, and P. Rogaway, The Round Complexity of Secure Protocols (extended abstract); Proceedings of the 22nd **STOC**, ACM, 1990, 503-513. [.ps] [.ps.gz]

D. Beaver, J. Feigenbaum, J. Kilian, and P. Rogaway, Security with Low Communication Overhead (extended abstract), Advances in Cryptology - Crypto '90 Proceedings, Springer-Verlag, 1991, 62-76. [.pdf]

D. Beaver, J. Feigenbaum, J. Kilian, and P. Rogaway, Locally Random Reductions: Improvements and Applications, Journal of Cryptology, 10 (1997), pp. 17-36. [.pdf] [.ps]

D. Beaver, Commodity-Based Cryptography (extended abstract); Proceedings of the 29th **STOC**, ACM, 1997, 446-455. [.pdf]

D. Beaver and S. Haber, Cryptographic Protocols Provably Secure Against Dynamic Adversaries (extended abstract); Advances in Cryptology - Eurocrypt '92, Springer-Verlag, 1993, 307-323. [.pdf]

D. Beaver, J. Feigenbaum, and V. Shoup, Hiding Instances in Zero-Knowledge Proof Systems (extended abstract), in Advances in Cryptology - Crypto '90, Lecture Notes in Computer Science, vol. 537, Springer, Berlin, 1991, pp. 326-338. [.pdf]

D. Beaver, S. Micali, and P. Rogaway, The round complexity of secure protocols; Proceedings of the 22nd Annual ACM Symposium on the Theory of Computing, (STOC 90), 1990, 503-513. [.ps] [.ps.gz]

D. Beaver, Foundations of Secure Interactive Computing (extended abstract); Advances in Cryptology - Crypto '91 Proceedings, Springer-Verlag, 1992, 377-391. [.pdf]

D. Beaver and S. Goldwasser, Multiparty Computation with Faulty Majority, Advances in Cryptology: Crypto '89, ed. Gilles Brassard. [.pdf]

D. Beaver and N. So, Global, Unpredictable Bit Generation Without Broadcast (extended abstract); Advances in Cryptology - Eurocrypt '93, Springer-Verlag, 1994, 424-434. [.pdf]

D. Beaver, Efficient Multiparty Protocols Using Circuit Randomization (extended abstract); Advances in Cryptology - Crypto '91 Proceedings, Springer-Verlag, 1992, 420-432. [.pdf]

D. Beaver, How to Break a "Secure" Oblivious Transfer Protocol (extended abstract); Advances in Cryptology - Eurocrypt '92, Springer-Verlag, 1993, 285-296. [.pdf]

D. Beaver, J. Feigenbaum, R. Ostrovsky, and V. Shoup, Instance-Hiding Proof Systems; submitted for journal publication. Available as DIMACS Technical Report 93-65, Rutgers University, Piscataway, 1993. [.ps.Z]

R. Beigel and J. Feigenbaum, On Being Incoherent Without Being Very Hard, Computational Complexity, 2 (1992), pp. 1-17.

A. Beimel, Y. Isahi, T. Malkin, and E. Kushilevitz, One-way functions are essential for single-server private information retrieval, Proc. of the 31st Annu. ACM Symp. on the Theory of Computing (STOC), pp. 89-98, 1999. [.ps]

A. Beimel and B. Chor, Secret Sharing with Public Reconstruction, IEEE Trans. on Info. Theory, 44 (5):1887-1896, 1998. Extended abstract in Crypto '95. [.ps]

A. Beimel and M. Franklin, Reliable communication over partially authenticated networks, Theoretical Computer Science, (220)1:185--210, 1999. Preliminary version in WDAG '97, volume 1320 of LNCS, pages 245-259, Springer, 1997. [.ps]

A. Beimel, Secure Schemes for Secret Sharing and Key Distribution, Ph.D. Thesis, Dept. of Computer Science, Technion, 1996. [.ps]

A. Beimel, T. Malkin, and S. Micali, The All-or-Nothing Nature of Two-Party Secure Computation, CRYPTO '99., vol. 1666 of LNCS, pages 80 - 97, 1999. [.ps]

A. Beimel and B. Chor, Universally ideal secret sharing schemes. IEEE Trans. on Info. Theory, 40 (3):786-794, 1994. Extended abstract in Crypto '92. [.ps]

Ius mentis
Law and technology explained

Category: Top > Technology > Encryption 01 October 2005

The ElGamal public key system

(Nederlandse versie)

The ElGamal cryptographic algorithm is a public key system like the Diffie-Hellman system. It is mainly used to establish common keys and not to encrypt messages.

In this document

Introduction

The ElGamal cryptographic algorithm is comparable to the Diffie-Hellman system. Although the inventor, Taher Elgamal, did not apply for a patent on his invention, the owners of the Diffie-Hellman patent (US patent 4,200,770) felt this system was covered by their patent. For no apparent reason everyone calls this the "ElGamal" system although Mr. Elgamal's last name does not have a capital letter 'G'.

A disadvantage of the ElGamal system is that the encrypted message becomes very big, about twice the size of the original message m. For this reason it is only used for small messages such as secret keys.

Generating the ElGamal public key

As with Diffie-Hellman, Alice and Bob have a (publicly known) prime number p and a generator g. Alice chooses a random number a and computes $A = g^a$. Bob does the same and computes $B = g^b$.

Alice's public key is A and her private key is a. Similarly, Bob's public key is B and his private key is b.

Encrypting and decrypting messages

If Bob now wants to send a message m to Alice, he randomly picks a number k which is smaller than p. He then computes:

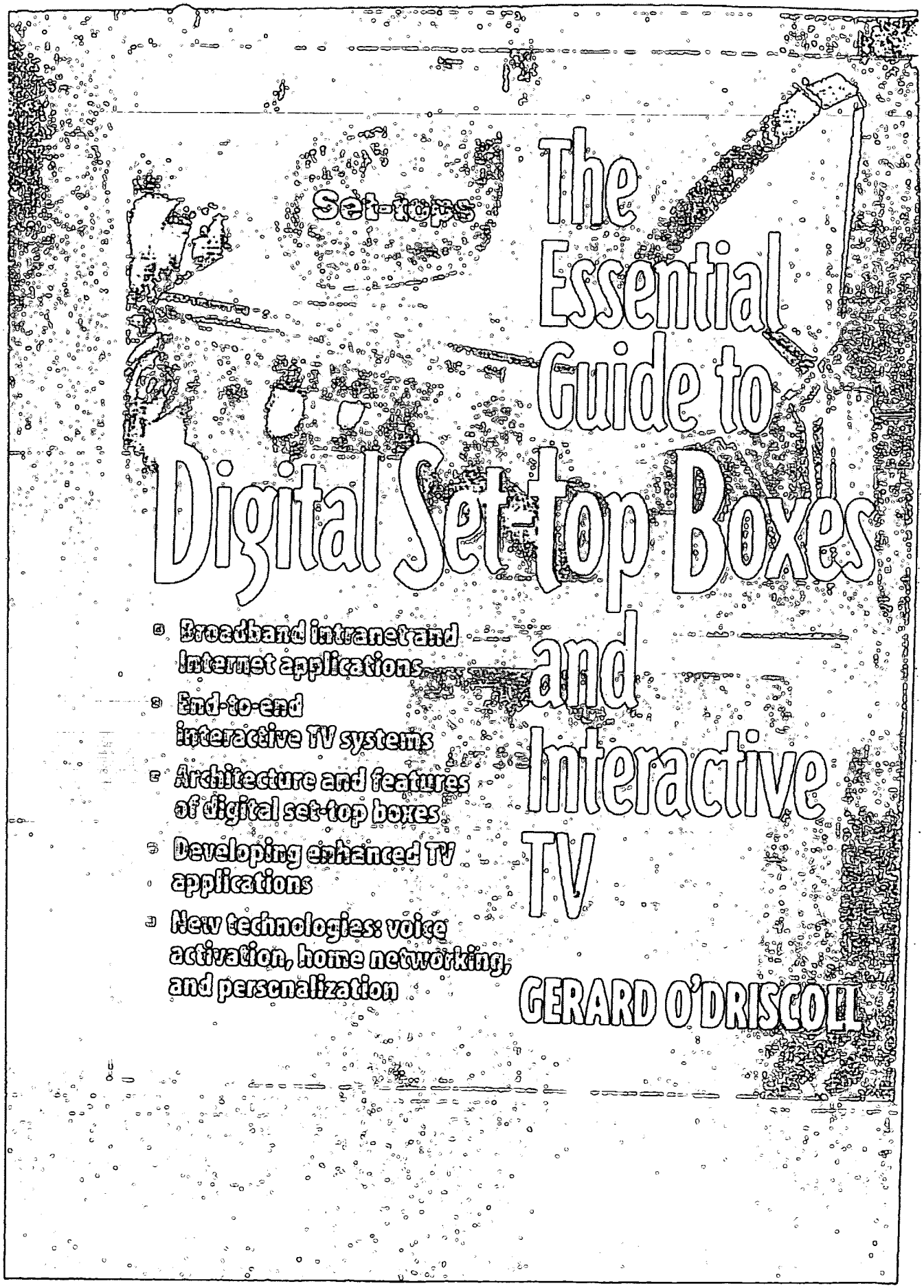
$$c_1 = g^k \bmod p$$
$$c_2 = A^k * m \bmod p$$

and sends c_1 and c_2 to Alice. Alice can use this to reconstruct the message m by computing

$$c_1^{-a} * c_2 \bmod p = m$$

because

$$c_1^{-a} * c_2 \bmod p = (g^k)^{-a} * A^k * m = g^{a*k} * A^k * m = (g^a)^{-k} * A^k * m = A^{-k} * A^k * m = 1 * m = m$$


Set-tops
The Essential Guide to Digital Set-top Boxes and Interactive TV
Broadband intranet and Internet applications
End-to-end interactive TV systems
Architecture and features of digital set-top boxes
Developing enhanced TV applications
New technologies: voice activation, home networking, and personalization
GERARD O'DRISCOLL

passes ETSI projects, technical committees, and special committees. More than 3,500 experts are at present working for ETSI in over 200 groups. (Additional information about ETSI is available from their web site at http://www.etsi.org/).

Digital Video Broadcasting (DVB)

The DVB project was conceived in 1991 and was formally inaugurated in 1993 with approximately 80 members. Today, the DVB project has made huge advancements and boasts a membership of over 230 organizations in more than 30 countries worldwide.

Members of the group include electronic manufacturers, network operators, broadcasters, software companies, and various regulatory bodies.

The DVB project has been a big success and has generated various standards for delivering digital TV to people throughout Europe, Asia, Australia, and North America.

The work of the DVB project has resulted in a comprehensive list of technical and nontechnical documents that describe solutions for implementing digital television in a variety of different environments.

The international standards and solutions developed by DVB over the past few years can be classified and summarized as follows:

1. DVB-S—An international standard for transmitting digital television using satellites.
2. DVB-C—An international standard for transmitting digital television using digital cable systems.
3. DVB-T—An international standard for transmitting digital television in a terrestrial environment.
4. DVB-MC/S—An international standard for transmitting digital television using microwave multipoint video distribution systems.
5. DVB-SI—An international standard that defines the data structures that accompany a digital television signal.
6. DVB-CA—An international standard that defines digital television security standards.
7. DVB-CI—An international standard that defines a common interface to the digital TV security system.
8. DVB-I—An international standard for deploying interactive TV.
9. DVB-Data—An international standard designed to allow operators to deliver software downloads and high speed data services to their customers.
10. Interfaces—An international standard that defines digital TV interfaces to high speed backbone networks.

Figure 1.1
DVB Logo

Copies of these standards are available for download on ETSI's web site.

DVB-compliant digital equipment is widely available and is easily identified by the DVB logo illustrated in Figure 1.1. The DVB has had its greatest success in Europe, however the standard has implementations in North and South America, Africa, Asia, and Australia. For additional information about DVB, visit their web site at http://www.dvb.org/.

Advanced Television Systems Committee (ATSC)

The ATSC committee was formed to establish a set of technical standards for broadcasting standard and High Definition Television (HDTV). Pictures based on this standard can have 3 to 5 times the sharpness of today's analog broadcasts.

The committee is composed of 136 member organizations, standard bodies, IT corporations, educational institutions, and electronic manufacturers. It has been formally adopted in the United States, where an aggressive implementation of digital TV has already begun. In addition to the U.S., Canada, South Korea, Taiwan, and Argentina have also adopted the ATSC digital TV standard for terrestrial broadcasts. A sample of the ATSC standards are outlined in Table 1.1.

Table 1.1 ATSC Standard Documents

Document Number	Standard Description	Brief Overview	Web Address of Detailed Document
A/52	ATSC Digital Audio Compression	Specifies coded representation of audio information and the decoding process, as well as information on the encoding process	www.atsc.org/Standards/A52/
A/53	ATSC Digital Television Standard	Specifications and characteristics for an advanced TV (ATV) system	www.atsc.org/Standards/A53/
A/54	ATSC Guide	Description of ATV system	www.atsc.org/Standards/A54/
A/64	Transmission measurement and compliance for digital television	Description of measurement and ATSC compliance system	www.atsc.org/Standards/A64/

This table only displays a snapshot of the ATSC standards. To review the complete listings of ATSC standards, we recommend you visit the ATSC Web page at http://www.atsc.org/Standards/stan_rps.html for a more detailed listing.

For the latest information and updates about ATSC, visit their web site at http://www.atsc.org/.

Digital Audio Visual Council (DAVIC)

The organization was formed in 1994 with the aim of defining standards for the end-to-end transfer of digital audio, video, and Internet-based content.

DAVIC is a nonprofit standards organization currently located in Switzerland. The organization currently has a membership of over 180 companies from 25 countries around the globe, representing companies and individuals from all sectors of the audio-visual industry. DAVIC members meet on a regular basis to define specifications and use their web site (www.davic.org) to collaborate and implement various international projects.

European Cable Communications Association (ECCA)

ECCA is the European Association of cable operators. The main goal of the Association is to foster cooperation between operators, and to promote their interests

at a European level. ECCA gathers European cable operators, consisting of more than 40 million subscribers. The first informal cooperation between European cable operators started in 1949. As these informal meetings became more frequent, a formal structure for European cooperation was required and on September 2, 1955, the Alliance Internationale de la Distribution par câble (AID) was set up by representatives of Switzerland, Belgium, and The Netherlands. In 1993, AID was renamed the European Cable Communications Association, thus stressing the communication role of its members as well as its European goals.

ECCA now has 29 members in 17 countries. It also has 5 associate members in central and eastern Europe. ECCA has considerably contributed to European policies related to cable on the regulatory as well as on the technical standards field.

On the regulatory, ECCA has done a lot of work on areas such as digital TV, copyright, must-carry, and open-access issues. In addition to these projects, ECCA members have also compiled the following technical specifications.

Eurobox

On initiative of the ECCA organization, a common specification for cable set-top boxes following DVB standards was agreed upon by a large number of cable operators in Europe (the Eurobox platform).

The Eurobox platform was set up in 1997, and has more than 5.5 million subscribers. A more detailed description of the Eurobox is available in Chapter 5 of this book.

Euromodem

A collective resolution to develop a global standard for high speed cable modems was signed at the ECCA Cable Forum in November 1998. The standard fully complies with European standards and with several DVB specifications. The ECCA group has considered two different types of modems: class A and class B. Class A modems are capable of transmitting data at very high speeds in a downstream direction (maximum of 50.8 Mbits/sec) and 3 Mbits/sec in the upstream direction. They are capable of accessing the Internet at high speeds and support a number of security technologies. Class B is the second type of modem considered by the group. It extends the functionality of class A devices through the support of time critical services such as video conferencing and telephony. At the time of going to press, a number of electronic manufacturing companies were invited to submit plans to manufacture modems compliant with the Euromodem standard.

Cable telephony

On the basis of the full liberalization of the telecommunications sector in Europe, cable companies, satellite providers, and terrestrial broadcasters in different countries are planning to become competitors to the local telephony companies. Therefore, their networks are being or have been upgraded to broadband telecommunications networks, which are able to provide all kinds of services from telephony and local Internet access to high speed broadband connections. ECCA is also actively working in this area. For additional information about ECCA, visit their web site at http://www.ecca.be/.

CableLabs

Cable Television Laboratories, Incorporated (CableLabs), was originally established in May 1988 as a research and development consortium of cable television system operators. To qualify as a member of CableLabs, a company needs to be a cable television system operator. CableLabs currently represents more than 85 percent of the cable subscribers in the United States, 70 percent of the subscribers in Canada, and 10 percent of the subscribers in Mexico. CableLabs plans, funds, and implements a number of research and projects that help cable companies take advantage of future opportunities in the areas of digital TV, telephony, and high speed Internet. For additional information about CableLabs, visit their web site at http://www.cablelabs.com/.

W3 Consortium (W3C)

The W3 Consortium (W3C) was originally founded in 1994 to lead the World Wide Web to its full potential by developing common protocols that promote its evolution and ensure its interoperability. The organization is an international consortium, jointly hosted by the Massachusetts Institute of Technology in the U.S; an organization in Europe called the Institut National de Recherche en Informatique et en Automatique, and Keio University in Japan.

The consortium provides a range of services, including: a repository of information about the World Wide Web for developers and users; reference code implementations to embody and promote standards; and various prototype and sample applications to demonstrate use of new technology. For detailed information about the W3C, visit their web site at http://www.w3c.org/.

Federal Communications Commission (FCC)

The Federal Communications Commission (FCC) is an independent United States government agency, directly responsible to Congress. The FCC was established by the

Communications Act of 1934 and is charged with regulating interstate and international communications by radio, television, wire, satellite, and cable. The FCC's jurisdiction covers the 50 states, the District of Columbia, and U.S. possessions. There are six operating bureaus. The bureaus are: Mass Media, Cable Services, Common Carrier, Compliance and Information, Wireless Telecommunications, and International. These bureaus are responsible for developing and implementing regulatory programs, processing applications for licenses or other filings, analyzing complaints, conducting investigations, and taking part in FCC hearings.

The Cable Services Bureau was established in 1993 to administer the cable Television Consumer Protection and Competition Act of 1992. The Bureau enforces regulations designed to ensure that cable rates are reasonable under the law. It is also responsible for regulations concerning "must carry," retransmission consent, customer services, technical standards, home wiring, consumer electronics, equipment compatibility, indecency, leased access, and program access provisions. The Bureau also analyzes trends and developments in the industry to assess the effectiveness of the cable regulations. For additional information about the FCC, visit their web site at http://www.fcc.gov/.

BUILDING BLOCKS OF A DIGITAL TV SYSTEM

A TV operator normally receives content from a variety of sources, including local video, cable, and satellite channels. The content needs to be prepared for transmission to the customer's home by passing the signal through a digital broadcasting system. The diagram in Figure 1.2 depicts the basic building blocks of a digital broadcasting system.

Note that the components shown in this diagram are logical units and do not necessarily correspond to the number of physical devices that are deployed in a total end-to-end digital solution. The role of each component shown in Figure 1.2 is briefly outlined in the following categories.

Compression and Encoding

Central to a digital video-broadcasting network is the compression system, whose job is to deliver high quality video and audio to consumers using a small amount of network bandwidth. The main goal of any compression system is to minimize the storage capacity of information. This is particularly useful for service providers who want to "squeeze" many digital channels into a digital stream.

A compression system consists of *encoders* and *multiplexers*. Encoders are devices used to digitize, compress, and scramble a range of audio, video, and data channels. Digital encoders allow TV operators to broadcast several high quality video

Figure 1.2
Simplified block diagram depicting the basic building blocks of a digital broadcasting system

programs over the same bandwidth that was formerly used to broadcast just one analog video program.

Once the signal is encoded and compressed, an MPEG-2 stream is transmitted to the multiplexer (MPEG-2 is an acronym for Moving Pictures Experts Group). This group has defined a range of compression standards and file formats, including the MPEG-2 video animation system. MPEG-2 is generally accepted in 190 countries worldwide as the standard for digital video compression. There are two major MPEG standards available on the market today: MPEG-1 and MPEG-2.

The MPEG-1 file format is normally used by interactive TV developers to create TV "stills" and has a quality level slightly less than conventional video cassette recorders. The MPEG-2 file format is used in a digital broadcasting environment and features CD-quality audio complemented with a high screen resolution. Once the signal has been compressed into MPEG-2 format, the multiplexer combines the outputs from the various encoders together with the security and program information and data into a single digital stream.

Modulation

Once the digital signal has been processed by the multiplexer, it is now time to amalgamate the video, audio, and data with the carrier signal in a process called *modulation*. The unmodulated digital signal outputted from the multiplexer has only two possible states, either a "zero" or a "one." By passing the signal through a modulation process, a number of states are added, which increases the data transfer rate. The modulation technique used by TV operators will depend on the geography of the franchise area and the overall network architecture.

The three major types of digital modulation are Quadrature Amplitude Modulation, Quadrature Phase Shift Keying, and Coded Orthogonal Frequency Division Multiplexing.

Quadrature Amplitude Modulation (QAM)

QAM is a relatively simple technique for carrying digital information from the TV operator's broadcast center to the customer. This form of modulation modifies the amplitude and phase of a signal to transmit the MPEG-2 transport stream. QAM is the preferred modulation scheme for cable companies because it can achieve transfer rates up to 40 Mbits/sec.

Simplified block diagram depicting the basic building blocks of a digital broadcasting system

Quadrature Phase Shift Keying (QPSK)

QPSK is more immune then QAM to electromagnetic noise and is normally used in a satellite environment or on the return path for a cable television network. QPSK works on the principle of shifting the digital signal so that it is out of phase with the incoming signal. QPSK will improve the robustness of a network, however, this modulation scheme is only capable of transmitting data at 10 Mbits/sec.

Coded Orthogonal Frequency Division Multiplexing (COFDM)

COFDM operates extremely well in heavily built-up areas where digital transmissions become distorted by obstacles such as buildings, bridges, and hills. COFDM is different to QAM because it uses multiple signal carriers to transfer information from one node on the network to another. At the moment, COFDM may be implemented with either 2,000 (2K) or 8,000 (8K) carrier signals. European terrestrial and MMDS operators mainly use the COFDM modulation scheme. In contrast, COFDM has not been deployed in the United States because the ATSC (Advanced Television Systems Committee) has defined a digital terrestrial system that meets the needs of a less-rugged geographical terrain.

Conditional Access System

Broadcast and TV operators are now interacting with their viewers on many levels, offering them a greater program choice than ever before. Additionally, the deployment of a security system or conditional access (CA), as it is commonly called, provides them with unprecedented control over what they watch and when. A CA system is best described as a virtual gateway that allows viewers to access a new world of digital services.

The main goal of any CA system is to control subscribers' access to digital TV pay services and secure the operators revenue streams. Consequently, only customers that have a valid contract with the network operator can access a particular service. Using today's CA systems, network operators are able to directly target programming, advertisements, and promotions to subscribers by geographical area, market segment, or according to personal preferences. The CA system is therefore a vital aspect of the digital TV business. In technical terms, the key elements of the CA system are illustrated in Figure 1.3.

Restricting access to a particular service is accomplished by using a technique called cryptography. It protects the digital service by transforming the signal into an unreadable format. The transformation process is known as "encryption" in a digital environment and "scrambling" in an analog domain. Once the signal is encrypted, it can only be decrypted by means of a digital set-top box. Decryption is the process

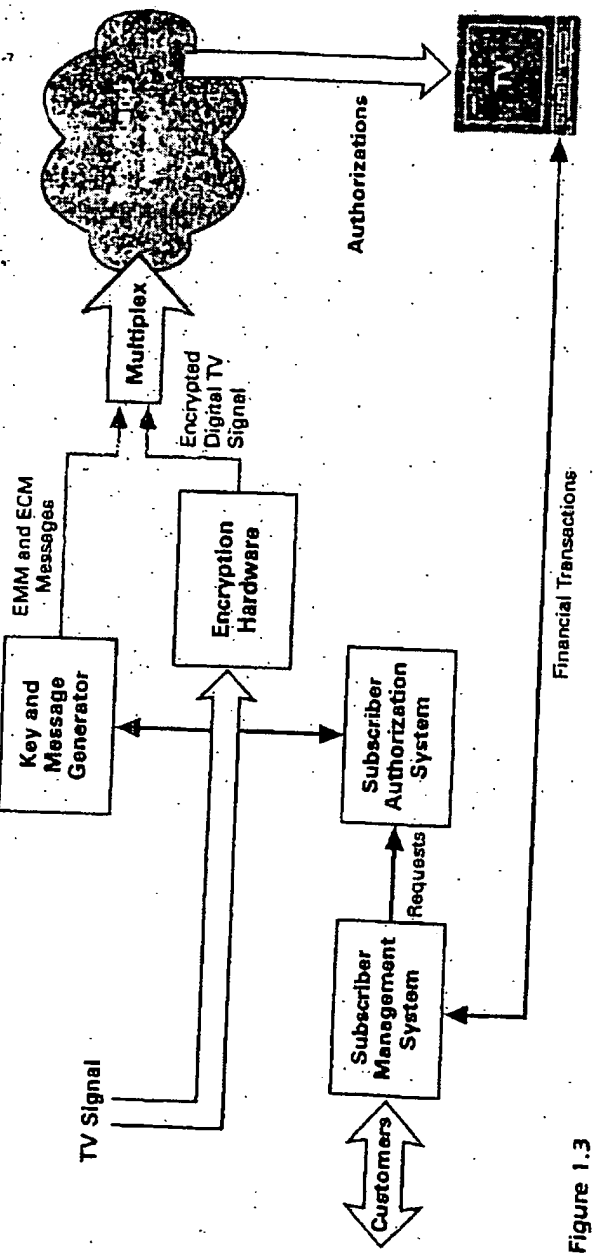


Figure 1.3
Basic principle of an end-to-end conditional access system

used to convert the message back to its original format. This is carried out using a decryption key. A key is best described as a secret value, consisting of a random string of bits, which is used by a computer in conjunction with mathematical formulas called algorithms to encrypt and decrypt information.

The box incorporates the necessary hardware and software subsystems to receive and decrypt the signal. These components are comprised of a de-encryption chip, a secure processor, and some appropriate hardware drivers. The de-encryption chip is responsible for holding the algorithm section of the CA. The secure processor can either be soldered onto the set-top box's printed circuit board or else attached to a smart card. Smart cards are plastic cards that look like credit cards. This processor contains the necessary keys needed to decrypt the various services. Chapter 11 discusses the cryptography aspects of smart card security in more detail.

A given subscriber may decrypt and access the digital signal only if the subscriber has purchased the relevant entitlement. As an example, the entitlement may be provided in the form of an electronic smart card that is plugged into the set-top box. Alternatively, in a pay-per-view scenario, the entitlement may be delivered electronically by entitlement management messages (EMMs) and entitlement control messages (ECMs) within the broadcast stream. An EMM is used to carry authorization details and are subscriber-specific. Consequently, the number of EMMs that need to be sent over the broadband network is proportional to the number of set-tops on the network. In addition to sending EMMs to specific customers, operators can also broadcast EMMs to groups of subscribers in different geographical areas. ECMs, on the other hand, carry program- and service-specific information, including control words that are used by the smart card to decrypt the relevant program. However, if a subscriber is not entitled to watch the program, then a signal is sent to the set-top box to indicate that this program has not been authorized for de-encryption. ECMs and EMMs are generated and broadcasted at the TV operations center using specialized hardware devices. They are then transmitted to the viewer's smart card. The card will check access rights and descramble the requested digital services. It is possible to change the value of an ECM every 10 seconds in order to maximize security on a digital network. A typical smart card is capable of storing up to a hundred entitlement messages, which means that each subscriber on the network is capable of ordering 100 pay-TV events at any one time.

In addition to encrypting digital services, the CA also interfaces with the following subsystems:

Subscriber Management System (SMS)

To exploit the commercial potential of digital broadcasting, TV operators need to interface their technical systems with a subscriber management system (SMS). The SMS provides the support required to accurately manage the digital TV business model. It handles the customer database and sends requests to the subscriber autho-

rization system (SAS)—the technical management part of the CA system. Functions typically provided by an SMS software application system include:

- register, modify, and cancel subscriber records;
- targeted marketing campaigns;
- inventory management of set-tops and smart cards;
- customer experience tracking;
- cross-selling of services;
- interfacing with banks and credit card companies;
- fault management;
- multilingual and multicurrency capability;
- bill preparation and formatting;
- presentation of bills in electronic formats; and
- accounting and auditing facilities.

Many of the software solutions currently available in the marketplace are capable of supporting the increasing variety of interactive services offered to subscribers. The main goal of any SMS system is to ensure that subscribers view exactly what they pay for.

Subscriber Authorization System (SAS)

The main task of the SAS is to translate the requests coming from the SMS into EMMs. These authorization messages are then sent via the digital multiplex to the smart card, which is located in the set-top box. They are sent to customers on a regular interval (for example, every month) to renew subscription rights on the smart card. In the case of Pay Per View (PPV) applications, the SAS sends a certain amount of electronic tokens to the smart card that will allow customers to purchase a variety of PPV events. The SAS contains database(s) that are capable of storing the following items of information:

- pay TV product information,
- data to support the electronic TV guide,
- identification numbers of smart cards,
- customer profiles, and
- scheduling data.

Additionally, SAS security can be enhanced by periodically changing the authorization keys broadcasted to the subscriber base. Some well-known CA systems include:

- CryptoWorks from Philips,
- Viaccess from France Telecom,
- Nagra from NagraVision,
- MediaGuard from Canal+ Technologies,
- VideoGuard from NDS,
- DigiCipher from General Instruments, and
- Iredeto from MindPort.

Network Transmission Technologies

Several different technologies have been deployed to bring broadband entertainment services from a central point to customers on a digital TV network. The different distribution systems (or mix of systems) adopted to broadcast digital TV services in countries around the world has largely been a function of each market's unique characteristics, including elements such as topography, population density, existing broadcast infrastructure, as well as social and cultural factors.

The most popular of these technologies are detailed in the following subsections.

Digital Via Hybrid Fiber-Coax (HFC)

Hybrid fiber-coax (HFC) technology refers to any network configuration of fiber-optic and coaxial cable that may be used to redistribute a variety of broadband entertainment services. These broadband services include telephony, interactive multimedia, high speed Internet access, video-on-demand, and distance learning. The types of services provided to consumers will vary between cable companies.

Many of the major cable television companies in the United States, Europe, Latin America, and Southeast Asia are already using it. Networks built using HFC technology have many characteristics that make it ideal for handling the next generation of communication services. First and foremost, HFC networks can simultaneously transmit broadband analog and digital services. This is extremely important for network operators who are rolling out digital TV to their subscribers on a phased basis. Additionally, HFC meets the expandable capacity and reliability requirements of a new digital TV system. HFC's expandable capacity allows network operators to add services incrementally without major changes to the overall plant infrastructure. HFC is essentially a "pay as you go" architecture that matches infrastructure investment with new revenue streams, operational savings, and reliability enhancements. The HFC network architecture is comprised of fiber transmitters, optical nodes, fiber and coaxial cables, and distribution hubs. An end-to-end HFC network is illustrated in Figure 1.4.

band entertainmen
. The different dig
tal TV services in
rket's unique char
ity, existing broad

owing subsections

ation of fiber-optic
band entertainment
multimedia, high
e types of services

ed States, Europe
s built using HFC
ng the next genera
can simultaneous
important for net
on a phased basis
requirements of a
k operators to add
nfrastructure. HFC
ructure investment
nhancements. The
al nodes, fiber and
rk is illustrated in

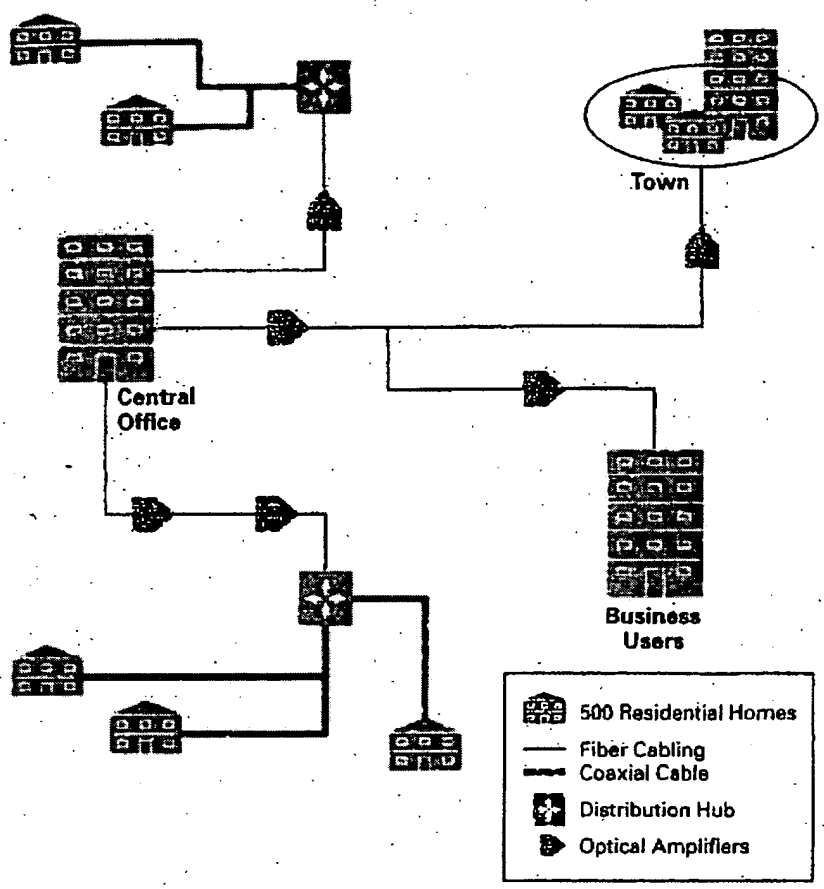


Figure 1.4
End-to-end HFC Network

From the diagram we can see that the signal is transmitted from the central office in a star-like fashion to the fiber nodes using fiber-optic feeders. The fiber node, in turn, distributes the signals over coaxial cable, RF amplifiers, and taps throughout the customer serving area. In conclusion, HFC is the lowest-cost alternative available in terms of cost-per-home-passed. This fact, combined with the other advantages already discussed, ensures that HFC will remain the primary technology for distributing advanced broadband services in a cabled environment.

Digital via Wireless Cable

Wireless cable is a relatively new service used to broadcast TV signals at microwave frequencies from a central point or head-end to small antennas located on the subscriber's roof. It is enabled through the use of two distribution technologies: multichannel multipoint distribution system (MMDS) and local multipoint distribution system (LMDS).

MMDS

Analog-based MMDS began in the mid-1970s with the allocation of two channels for sending business data. The service, however, became very popular for TV subscriber programming and applications were made to allocate part of the ITFS (Instructional Television Fixed Service) band to wireless cable TV. Once the regulations had been amended, it became possible for a wireless cable system to offer up to thirty-one 6 MHz channels in the 2.5 to 2.7 GHz band. During this timeframe, the system was used by nonprofit organizations to broadcast educational and religious programs. In 1983, the FCC allocated frequencies in both of these spectrums, providing 200 MHz bandwidth for licensed network providers. The basic components of an end-to-end digital MMDS system is shown in Figure 1.5.

An MMDS system consists of a head-end that receives signals from satellites, fiber optic cable, off-the-air TV stations, and local programming. At the head-end, the signals are mixed with commercials and other inserts, scrambled, converted to the 2.1 and 2.7 GHz frequency range, and sent to microwave towers. The signals are then rebroadcast from low-powered base stations within a 35-mile diameter of the subscriber's home. Signals are received with home rooftop antennas, which are 18 to 36 inches wide. The receiving antenna should have a clear line of site to the transmitting antenna. A down converter, usually a part of the antenna, converts the microwave signals into standard cable channel frequencies. From the antenna, the signal travels to a set-top box where it is decrypted and from there the signal passes into the television. If the subscriber requires interactivity, then the digital set-top box is also connected to the public telephone network.

Today, there are systems in use all around the U.S. and in many other countries, including Australia, South Africa, South America, Ireland, and Canada. Currently, MMDS is an analog service providing about 20 channels of programming to subscribers. Digital MMDS increases the number of channels to between 130 and 180. Digital MMDS also reduces the line-of-sight restrictions by providing a more efficient signal that will require less signal strength at the set-top box. Digital signals will need about 100 times less signal strength than analog signals, which translates to a substantial increase in the range of service area. Where an analog signal degrades with distance, the digital signal will remain constant and perfect as long as it can be received. In addition to more channels, digital MMDS customers will also be able to receive a variety of Internet, telephony, and interactive TV-based services. MMDS is presently using a standard phone line for the return path, but trials are under way to utilize a portion of the wireless bandwidth for return capabilities.

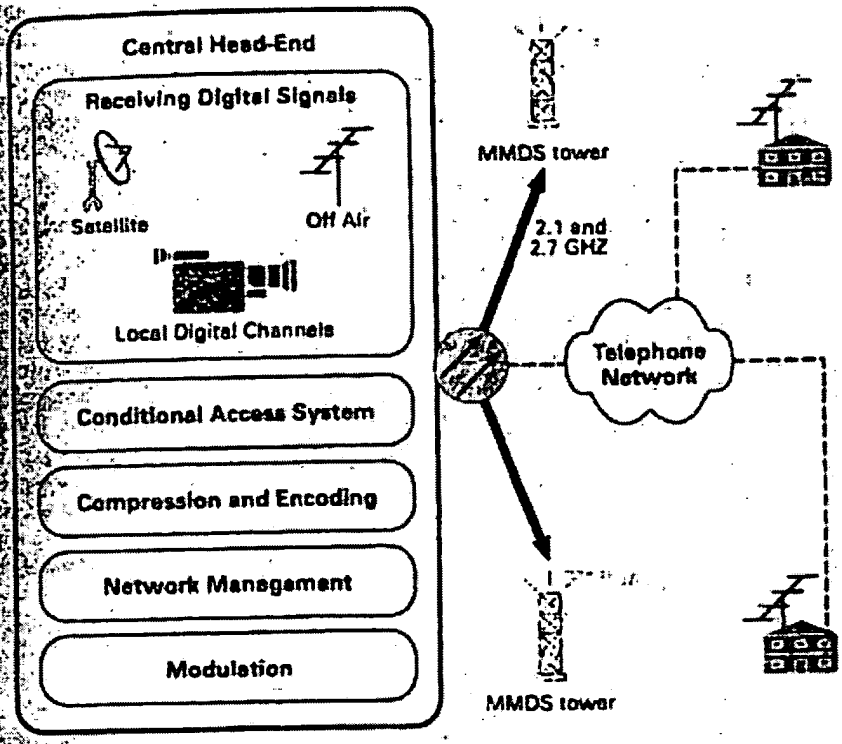


Figure 1.5
End-to-end digital MMDS solution

In Ireland for example, MMDS operators are currently very active in testing and delivering a diversity of advanced digital TV and Internet services using MMDS network transmission techniques to customers across the island. The services on offer to customers include:

- high speed access to the Internet;
- private data networks for companies on the island;
- broadcast video and Pay Per View television;
- Plain Old Telephone Service (POTS); and
- fractional and full leased lines

Future services discussed by Irish operators include video conferencing and delivering multimedia training courses to remote parts of the country using advanced MMDS digital technologies. MMDS operators across the world are adopting similar approaches to their Irish counterparts and are poised to take advantage of the exciting new digital MMDS broadcasting revolution, allowing the delivery of a variety of services to their customer bases.

LMDS

LMDS uses microwave frequencies in the 28 GHz frequency range to send and receive broadband signals, which are suitable for the transmission of video, voice, and multimedia data. Digital LMDS has been commercially deployed and is used to deliver video programming from local and cable channels. Additionally, it is also capable of delivering a plethora of Internet- and telephony-based services to consumers. The system architecture for LMDS is very similar to the MMDS system. The reception and processing of programming and other head-end functions are the same. The signals are then rebroadcasted from low-powered base stations in a 4–6 mile radius of the subscriber's home. Signals are then received using six square-inch antennas, which can be mounted either inside or outside the home. As with the MMDS, the signal travels to the set-top box, decrypted, and formatted for display on the customer's television. In addition to a high video and audio quality, other benefits of LMDS include its bandwidth range of 1 GHz and the availability of a return channel for interactive TV services.

Digital via Terrestrial

Commercially launched in the U.K. in November 1998, terrestrial communications, or DTT as it is commonly called, can also be used to broadcast a range of digital services.

Elements of a terrestrial communications network include:

1. Transmission medium—Services are normally provided via the ultra high frequency band (UHF). The frequencies in this band range from 300 MHz up to 3 GHz. Standard 8 MHz channels are used and shared with analog transmissions.
2. Modulation scheme—DTT uses the COFDM modulation scheme. The main purpose of COFDM is to make the terrestrial signal immune to multipath reflections. In other words, the signal needs to be robust enough to traverse geographical areas that include mountains, trees, and large buildings.
3. Transmission infrastructure—Uses an existing network of broadcast stations and transmitters.
4. Customer's premises equipment—With a modern aerial, there should be no need to replace it to receive the DTT service. If the aerial is a very old one, the viewer would certainly benefit from updating. Additionally, DTT

necessitates the purchase of a new digital set-top box to receive and decode the digital signal.

Digital via Direct Broadcast Satellite (DBS)

Digital television is also available through direct broadcast satellite (DBS), which can provide higher bandwidth than terrestrial, MMDS, or cable transmission.

Direct Broadcast Satellite (DBS) is a service whereby you receive subscription television from a single high-powered satellite. This satellite is typically located about 22,000 miles above the surface of the earth. At the moment, when you subscribe to an analog service you receive a state-of-the art mini-dish that is maintained and owned by the local distributor, along with a decoder for your television set that unscrambles the signals received from the satellite. This year, consumers will be able to receive digital satellite service by installing a new and smaller digital satellite dish and buying a new digital satellite set-top box. Digital via DBS brings consumers more channels to choose from, new features, and new services.

Network Management

As you can see the broadcasting center is made up of many complex components. As these components handle more and more services, network problems must be quickly detected and resolved. To maximize system uptime and monitor the services delivered to customers, a network monitoring and control system is installed at the broadcasting center. The main goal of such a system is to minimize service interruptions to digital TV customers. Features of a typical head-end control system include:

- monitoring the availability of devices,
- gathering statistics,
- reporting alarms and problems to support personnel, and
- remote diagnostics.

The systems available at present are vendor-specific and will run on either Windows NT or UNIX platforms.

SUMMARY

Digital television brings about many challenges, but with those challenges come a lot of opportunities. Advances in technology over the past few years have meant that the

possibility of delivering digital television services to billions of people around the globe has moved from the realms of fantasy into reality.

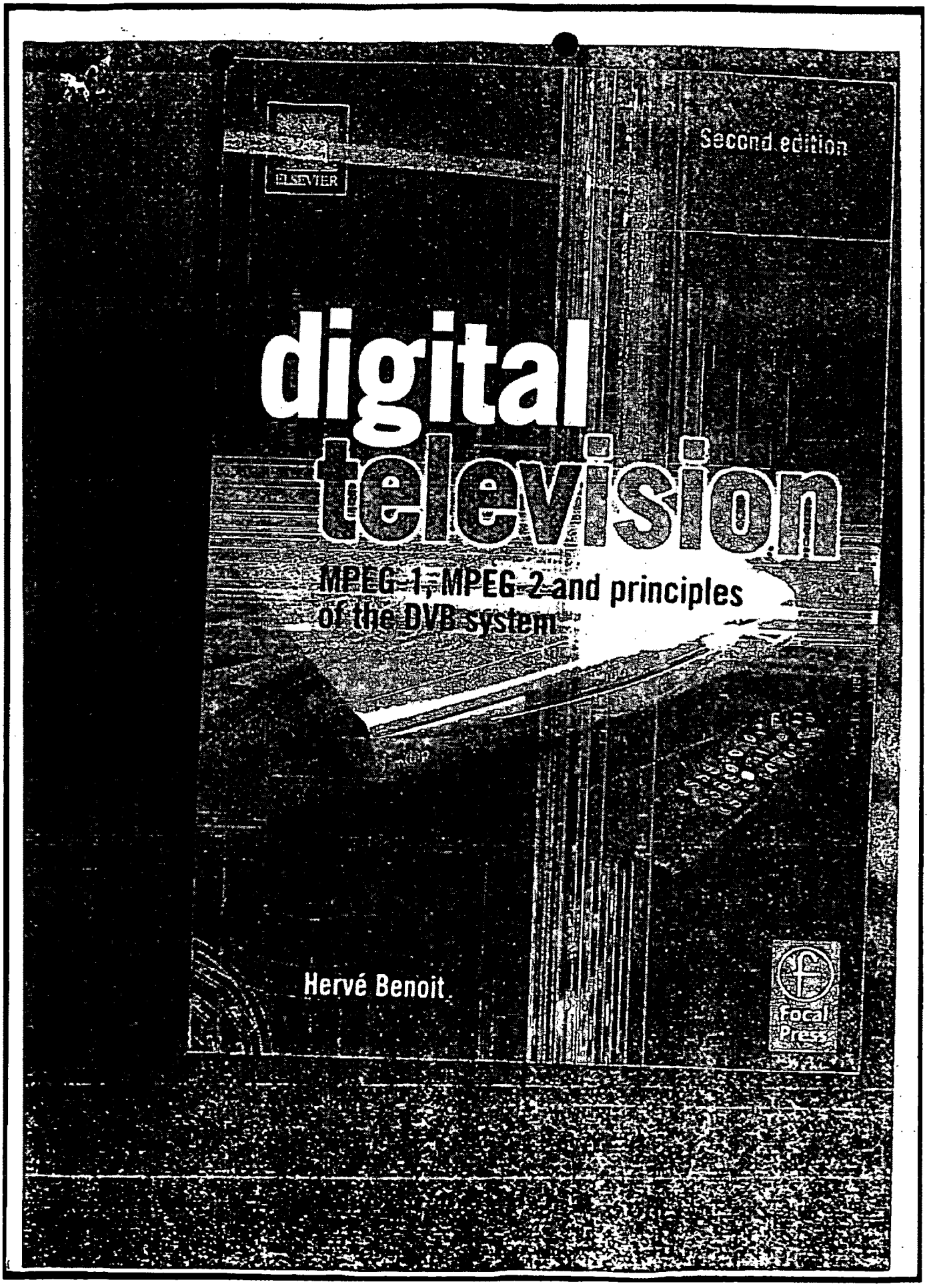
Digital TV offers a potential mechanism through which every home, school, business, and community center in the world could be included in the information society.

It opens up a new world of opportunity for companies to develop and utilize their existing network infrastructures. This includes broadcasters; cable and satellite companies; the creative community in television and film; Internet content providers; web site producers; and new, innovative companies that will form around the future of digital TV. The broadcast of digital TV and multimedia data works well because of the agreements and partnerships forged by a number of organizations around the world. A complete digital broadcasting system is comprised of a number of building blocks including the compression, encoding, and modulation-system, a CA system for security purposes; network transmission media to deliver the digital services; and, finally, a network management system to detect and resolve problems.

2

In t

- Evolu
- Set-tc
- Basic
- How a
- Under
- Install
- Troubl
- Summ


Second edition
digital
television
MPEG-1, MPEG-2 and principles
of the DVB system
Hervé Benoit
Focal Press

5 Scrambling and conditional access

The proportion of free access programmes among analogue TV transmissions by cable or satellite is decreasing continuously, at the same time as their number increases; hence, it is almost certain that the vast majority of digital TV programmes will be pay-TV services, in order to recover as quickly as possible the high investments required to launch these services. Billing forms will be much more diversified (conventional subscription, pay per view, near video on demand) than what we know today, made easier by the high available bit-rate of the system and a 'return channel' (to the broadcaster or a bank) provided by a modem.

The DVB standard, as explained in the previous chapter, envisages the transmission of access control data carried by the conditional access table (CAT) and other private data packets indicated by the program map table (PMT). The standard also defines a common scrambling algorithm (CSA) for which the trade-off between cost and complexity has been chosen in order that piracy can be resisted for an appropriate length of time (of the same order as the expected lifetime of the system).

The conditional access (CA) itself is not defined by the standard, as most operators did not want a common system, everyone guarding jealously their own system for both commercial (management of the subscribers' data base) and security reasons (the more open the system, the more likely it is to be 'cracked' quickly). However, in order to avoid the problem of the subscriber who wishes to access networks using different conditional access systems having a stack of boxes (one set-top box per network), the DVB standard envisages the following two options:

1. **Simulcrypt**. This technique, which requires an agreement between networks using different conditional access systems but the same scrambling algorithm (for instance the CSA of the DVB), allows access to a given service or programme by any of the conditional access systems which are part of the agreement. In this case, the transport multiplex will have to carry the conditional access packets for each of the systems that can be used to access this programme.

2. **Multicrypt**. In this case, all the functions required for conditional access and descrambling are contained in a *detachable module* in a PCMCIA form factor which is inserted into the transport stream data path. This is done by means of a standardized interface (common interface, DVB-CI) which also includes the processor bus for information exchange between the module and the set-top box. The set-top box can have more than one DVB-CI slot, to allow connection of many conditional access modules. For each different conditional access and/or scrambling system required, the user can connect a module generally containing a smart card interface and a suitable descrambler.

The multicrypt approach has the advantage that it does not require agreements between networks, but it is more expensive to implement (cost of the connectors, housing of the modules, etc.). The DVB-CI connector may also be used for other purposes (data transfers for instance).

Only the future will tell us which of these options will be used in practice, and how it will be used.

5.1 Principles of the scrambling system in the DVB standard

Given the very delicate nature of this part of the standard, it is understandable that only its very general principles are available, implementation details only being accessible to network operators and equipment manufacturers under non-disclosure agreements.

The scrambling algorithm envisaged to resist attacks from hackers for as long as possible consists of a cipher with two layers, each palliating the weaknesses of the other:

- a *block layer* using blocks of 8 bytes (reverse cipher block chaining mode);
- a *stream layer* (pseudo-random byte generator).

Table 5.1 Meaning of transport_scrambling_flag bits

Transport_scrambling_flags	Meaning
00	No scrambling
01	Scrambling with the DEFAULT control word
10	Scrambling with the EVEN control word
11	Scrambling with the ODD control word

The scrambling algorithm uses two control words (even and odd) alternated with a frequency of the order of 2 s in order to make the pirate's task more difficult. One of the two encrypted control words is transmitted in the entitlement control messages (ECM) during the period that the other one is in use, so that the control words have to be stored temporarily in the registers of the descrambling device. There is also a *default* control word (which could be used for free access scrambled transmission) but it is of little interest.

The DVB standard foresees the possibility of scrambling at two different levels (transport level and PES level) which cannot be used simultaneously.

Scrambling at the transport level

We have seen in the preceding chapter (Fig. 4.6) that the transport packet header includes a 2-bit field called 'transport_scrambling_flags'. These bits are used to indicate whether the transport packet is scrambled and with which control word, according to Table 5.1 above.

Scrambling at transport level is performed after multiplexing the whole payload of the transport packet, the PES at the input of the multiplexer being 'in the clear'. As a transport packet may only contain data coming from one PES, it is therefore possible to scramble at transport level all or only a part of the PES forming part of a programme of the multiplex.

Scrambling at the PES level

In this case, scrambling generally takes place at the source, before multiplexing, and its presence and control word are indicated by the 2-bit PES_scrambling_control in the PES packet header, the format of which is indicated in Fig. 4.4. Table 5.2 indicates the possible options.

Table 5.2 Meaning of PES_scrambling_control bits

PES_scrambling_control	Meaning
00	No scrambling
01	No scrambling
10	Scrambling with the EVEN control word
11	Scrambling with the ODD control word

The following limitations apply to scrambling at the PES level:

- the header itself is, of course, not scrambled; the descrambling device knows where to start descrambling due to information contained in the PES_header length field, and where to stop due to the packet_length field;
- scrambling should be applied to 184-byte portions, and only the last transport packet may include an adaptation field;
- the PES packet header should not exceed 184 bytes, so that it will fit into one transport packet;
- the default scrambling word is not allowed in scrambling at the PES level.

5.2 Conditional access mechanisms

The information required for descrambling is transmitted in specific conditional access messages (CAM), which are of two types: entitlement control messages (ECM) and entitlement management messages (EMM). These messages are generated from three different types of input data:

- a *control_word*, which is used to initialize the descrambling sequence;
- a *service_key*, used to scramble the control word for a group of one or more users;
- a *user_key*, used for scrambling the service key.

ECM are a function of the control_word and the service_key, and are transmitted approximately every 2 s. EMM are a function of the service_key and the user_key, and are transmitted approximately every 10 s. The process for generating ECM and EMM is illustrated in Fig. 5.1

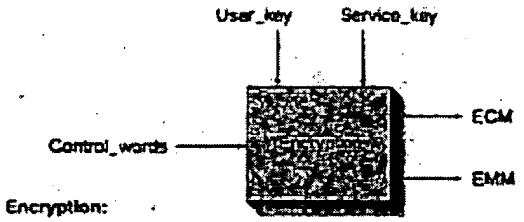


Fig. 5.1 Schematic illustration of the ECM and EMM generation process

In the set-top box, the principle of decryption consists of recovering the service_key from the EMM and the user_key, contained for instance in a smart card. The service_key is then used to decrypt the ECM in order to recover the control_word allowing initialization of the descrambling device. Figure 5.2 illustrates schematically the process for recovering control_words from the ECM and the EMM.

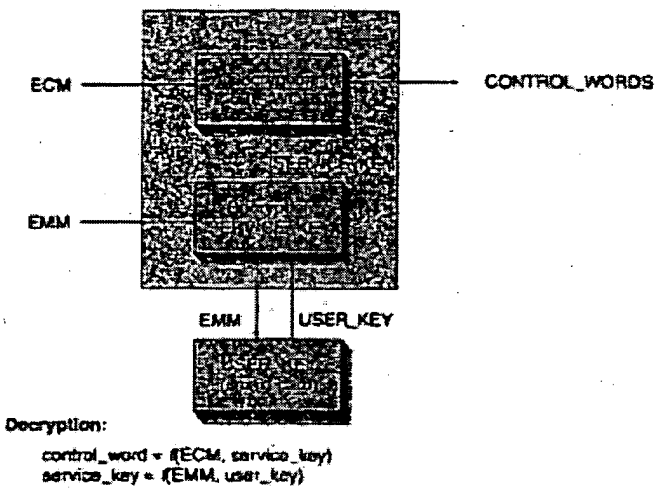


Fig. 5.2 Principle of decryption of the control words from the ECM and the EMM

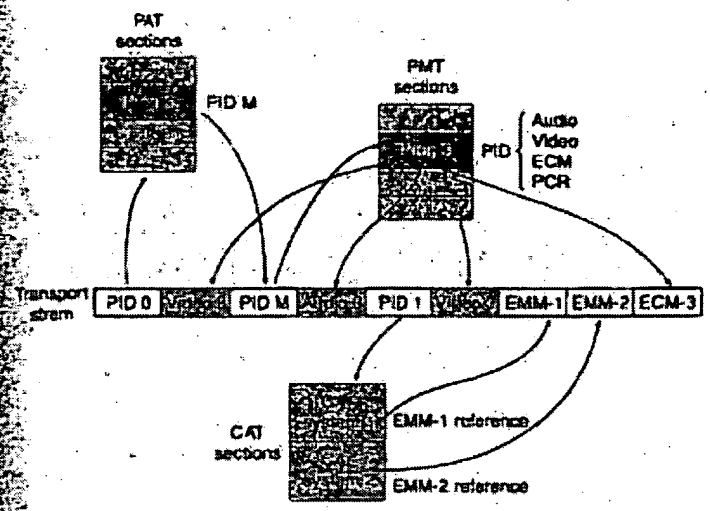


Fig. 5.3 Process by which the EMM and the ECM are found in the transport stream

Figure 5.3 illustrates the process followed to find the ECM and EMM required to descramble a given programme (here programme no. 3):

1. the program allocation table (PAT), rebuilt from sections in packets with PID = 0 x 0000, indicates the PID (M) of the packets carrying the program map table (PMT) sections;
2. the PMT indicates, in addition to the PID of the packets carrying the video and audio PESs and the PCR, the PID of packets carrying the ECM;
3. the conditional access table (CAT), rebuilt from sections in packets with PID = 0 x 0001, indicates which packets carry the EMM for one (or more) access control system(s);
4. from this information and the user_key contained in the smart card, the descrambling system can calculate the control_word required to descramble the next series of packets (PES or transport depending on the scrambling mode).

The above-described process is indeed very schematic; the support containing the user_key and the real implementation of the system can vary from one operator to another. The details of these systems are, of course, not in the public domain, but their principles are similar.

5.3 Main conditional access systems

Table 5.3 indicates the main conditional access systems used by European digital pay TV service providers.

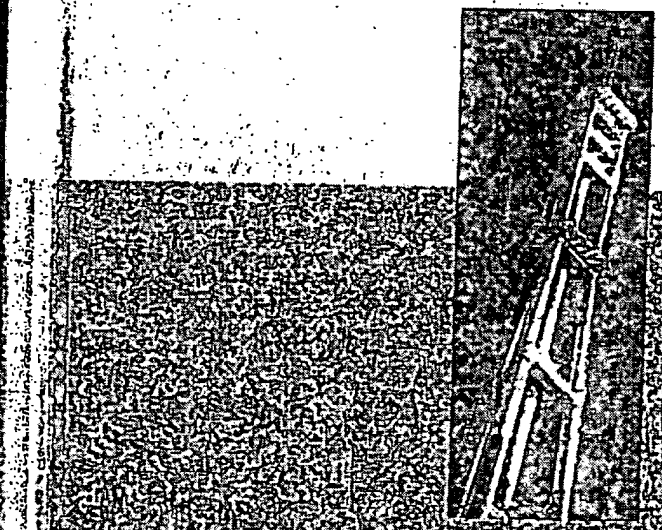
Most of these systems use the DVB-CSA scrambling standard specified by the DVB. The receiver has an internal descrambler controlled by an embedded conditional access software which calculates the descrambler control words from the ECM messages and keys contained in a subscriber smart card with valid access rights updated by the EMM messages.

Systems allowing pay per view often have a second card reader slot for a banking card as well as a modem to order the programmes as well as charging the bank account.

Table 5.3 Main conditional access systems

System	Origin	Service providers
Betacrypt	Betaresearch/IrDETO	Premiere World, German cable
CryptoWorks	Philips	Viacom, MTV Networks
IrDETO	IrDETO Access	Telepiu, Stream, Multichoice
Mediaguard	SECA (Canal+)	Canal+, Canal Satellite, ITV Digital
Nagravision	Kudelski SA	Via Digital, Quiero, Dish Network
Viaccess	France Telecom	TPS, AB-Sat, Arabesque, SSR/SRG
Videoguard	News Datacom (NDS)	BSkyB, Stream

UNDERSTANDING DIGITAL TERRESTRIAL BROADCASTING



AH DIGITAL AUDIO AND VIDEO SERIES

measure such a low bit error rate at point 3. This long measurement time is not practical, and this is one of the reasons why point 2 (instead of point 3) is used as a measurement point for transmission system quality.

However measuring the BER alone could mask underlying transmission problems if the carrier-to-noise ratio is not also measured. This is because digital systems employing Reed Solomon error correction can be close to total system failure and yet display very little residual errors. This phenomenon is due to the very steep nature of the characteristic BER versus C/N ratio curve for systems measured at point 2. It is often termed the cliff-effect as the BER is seen to fall from very high levels to extremely low levels within a few dB variation of C/N ratio. The huge variation of BER for a relatively small change in carrier-to-noise ratio means that a system, which is not showing significant degradations (low BER), could be on the verge of failure (on the cliff edge). If then, for any reason the carrier-to-noise ratio were degraded by a small amount the overall transmission channel could fail.

9.4.1.3 Measurement Point 3

Measurements made at point 3 are at the output of the demodulator and are the least useful for technical evaluation of the transmission system and channel. This is because these measurements are made after using all of the powerful error correction techniques that DTV can employ to remove errors. There is a risk that the error correction techniques could mask any channel impairments or transmission system degradations. Point 3 should not be used for transmission quality evaluation as it will inevitably lead to problems when link budgets become reduced due to climatic effects or transmission system degradations [4].

9.5 Set Top Boxes (STB) and Integrated Receiver Decoders (IRD)

A set top box (STB) or set top unit (STU) is a device which can translate digital television signals to a format that can be interpreted and displayed by existing analog television receivers. STBs are sometimes referred to as digital set top boxes (DSTB). It includes all of the hardware, middleware, and access entitlement software to allow the consumer to decode all the available digital video, audio, and data. It allows access to all of the digital

based content and services using existing analog television receivers as the display unit. An integrated receiver decoder (IRD) contains also a display unit. Most IRDs available to date can only demodulate the terrestrial COFDM modulation, as traditionally receivers were manufactured to decode off-air or terrestrial signals. Also they are very new additions to the consumer electronics market and at present most can only decode unencrypted DTV signals. However they should become more popular as the DTV market consolidates.

9.5.1 Set Top Box Functionality

Set top boxes are widely available for DTV and in the case of the COFDM based DVB-T system, both the 2k and the 8k systems are now supported. A typical STB will connect to an existing analog television receiver using standard interconnections such as a SCART connector and/or an RF loop through connection, and in a similar manner connect to a video cassette recorder. (The predominant interconnection standard for domestic equipment is the SCART connector, which derives its name from the French committee that has promoted its use.) Many STBs use telephone modems as the return channel, however other terrestrial based systems will be deployed in the future for the return channel. Stereo audio in either digital or analog format can be output to, for instance, a HiFi system. The internal MPEG-2 decoder typically supports Main Profile@Main Level with data rates per service of up to 15 Mbps (see Chapter 4). Current STB units generally support widescreen video format. The hardware can be manufactured to support various application programming interfaces, and conditional access systems. At the moment the network operator financially supports most STB deployment and this is leading to the emergence of proprietary STB systems for different transmission media. In Europe the DVB common interface (CI) is not mandatory for STB. This interface allows for the interconnection of DVB equipment and subassemblies.

9.5.2 Integrated Receiver Decoder (IRD) Functionality

At present, a few manufacturers have begun offering complete DTV receivers, a configuration that is called an integrated receiver decoder. These devices allow for the reception and decoding of DTV without the need for a STB. Some offer flat screen displays and all can decode the unencrypted signals at the moment. They incorporate primarily the demodulation of the terrestrial based transmission standards.

9.6 Middleware and Application Programming Interfaces (API)

There is no doubt that one of the most important topics for broadcasters and service providers at the present time is the topic of set top box middleware and application programming interface (API). In order to provide data services the set top box requires a layer of software that resides between the STB hardware and the application which is to be consumed. This software is called "middleware" and is often referred to as an API when used in connection with DTV. The term API is commonly used in software engineering but in DTV engineering some middleware solutions incorporate the API, but some do not. Some commentators refer to the API as the equivalent to "Windows" for the STB. It can be a helpful analogy for those new to this terminology.

It should be noted that STBs having different middleware might not be able to access the same data or interactive services. Each service must be designed or authored for each particular middleware system that it is intended to run on. This is a major problem for traditional analog broadcasting organizations entering the DTV market [5]. Clearly a fragmented STB market with different STB manufacturers using different middleware will mean that the same material will need to be authored many times to run on different STBs. This is recognized as a potential disaster for DTV and as a result DVB wish to adopt a standard for middleware known as the DVB multimedia home platform (DVB-MHP). This API will meet the need for the next generation of STB, including interactive applications, and Internet access from the STB. Other middleware solutions are proposed in the meantime until the DVB-MHP is developed and standardized. It is beyond the scope of this book to explore the issues relating to middleware and APIs.

9.6.1 DVB and Java

The DVB group has decided to use the Java programming language as the core specification for the software of the DVB-MHP (multimedia home platform). Java is a powerful programming language that should allow the implementation of new applications in a platform independent manner. Providing a so-called virtual machine environment, that is, a software entity that processes applications in the same way on any microprocessor in which it is implemented, does this. This will allow many different hardware realizations of the same applications. With Java, the virtual machine is commonly called a Java virtual machine (JVM).

9.6.2 The MHEG API

MHEG is a sister organization of MPEG within ISO/IEC JTC1 (see Section 3.2). The multimedia and hypermedia information expert coding group (MHEG) develops standards primarily for the transmission and representation of data and applications to allow for interactive services on set top boxes. The MHEG-5 standard has been chosen in the United Kingdom implementation of DTV and is now in use. It is an open and nonproprietary standard for the set top box API.

9.6.3 The EuroMHEG System

The EuroMHEG is a development of the MHEG-5 API to include extensions to the standard. The purpose of these extensions is to provide greater functionality to the consumer, within an open standard API. These functions include provision for a return path in various ways, and a financial toolkit to allow for home shopping and e-commerce. Downloading of different text fonts and text input from a remote control or keyboard are also supported. The EuroMHEG API has other functions which are a development of the original open standard MHEG-5 API, and as such is regarded as a migratory step from MHEG-5 toward the DVB-MHP. This is useful for broadcasters and network operators who wish to launch a DTV service prior to the standardization of the DVB-MHP.

9.6.4 Data Carousels

Data carousels are of interest to DTV service providers and network operators. A carousel is defined as a rotating magazine, for example slides in a projector, or luggage on a rotating conveyor belt. In broadcasting carousels have been used for a long period of time in conjunction with analog services such as teletext, where the contents of the carousel are cyclically repeated and the repetition rate is quite low. The data is broadcast to some defined playout, and the simplest playout is a carousel or rotation of information. If a receiver wants to access a particular module it simply awaits the next time the data from that module is broadcast. These simple data broadcasting techniques are popular with the public, despite the basic nature of the service provided.

It is expected that DTV carousels will be able to improve on these carousels and approach near multimedia presentation, through the better delivery of text, support for audio-visual clips and bitmaps, easier navigation, better graphics, and better scheduling.

Data broadcast according to the data carousel specification of the MPEG-2 DSM-CC is transmitted in a data storage media command and control (DSMCC) data carousel. The DVB data broadcast specification for data carousels supports data broadcast services, which require the periodic transmission of data modules through DVB compliant networks. Again, the application decoder must await the transmission of a particular module to access the data contained within that module.

For the support of interactive services DVB has adopted another part of the DSM-CC specification, known as the object carousel, which provides the facility to transmit structured groups of objects from a broadcast server to specific receivers. This provides enhanced capability as compared to the above mentioned data carousel [6].

9.6.5 Resident and Interactive Applications

A part of the middleware is often referred to as an application, and consists of software code which is used to allow a user to interact with various services and products. If the application is resident, that is, resides within the set top box, then it can be loaded into memory when the set top box is initially switched on. It may allow for tuning of the STB to various channels, set-up configuring, and also manage any interactions between the user and the service provider. Alternatively interactive applications require the user to authenticate the requested service or product from the provider before downloading, and hence control of access to the application remains with the service provider.

9.6.5.1 Electronic Program Guide (EPG)

The electronic program guide has been available in limited form with analog television, however it is expected to change dramatically with DTV. With DTV it allows the consumer to navigate through the various services and programs offered by the network operator or service provider from a graphical user interface (GUI). The EPG is regarded as a very powerful tool in guiding the consumer through the vast amount of services and programs offered in a DTV platform. The EPG will be able to present choices to the consumer based not only on a channel by channel basis (as is the case with analog television), but also by categorization of program content into different interest groups. It enables the consumer to create a viewing schedule based on content. It contains some of the information contained within a traditional paper based television listing service.

The EPG will use service information (SI) as the basis of most of the information that it will display (see Chapter 5). However, it will need to be collated and presented to the consumer in a responsible and coherent manner. Competition issues may arise on a platform between different service providers if undue prominence of a particular service is shown over others. Therefore some degree of regulation in the provision of the EPG is expected.

Some of the features that will be available from an EPG include program guides for a number of days in advance. Program browsers will display the current and next programs on all available services and allow for immediate switching of channels. These browsers will be available for both audio and video services. It will also allow for notification of events on other channels while the consumer is viewing a completely separate channel. Interactive services will be accessed through the EPG, for instance access to the Internet, email, e-commerce, and games, to mention but a few. The power of the EPG is realized when the complete platform is available through a single EPG.

References

[1] ETR 290, "Digital Video Broadcasting (DVB); Measurement Guidelines for DVB Systems," May 1997.

[2] DVB Document TM1748, Draft MG 119 Rev. 1, "DVB-T Measurement Guidelines," September, 1996.

[3] ITU-T Recommendation 0.151: "Error Performance Measuring Equipment Operating at the Primary Rate and Above."

[4] Nokes, C., "Bit Error Rates for DVB-T Signal," *Digital News*, Digital TV Group, No. 7, February 1999, p. 18.

[5] AGITS/WG7 Document, "Digital Television—A Preliminary Guide," 1999, http://www.teltec.dcu.ie/agitswg7/dtv/

[6] Horst, H., "DVB Data Broadcasting: Building the Info Highway," *World Broadcast News*, Special Supplement, November 1998, pp. 16–21.

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : H04N 7 /24 — A2

(11) International Publication Number: **WO 00/04727**

(43) International Publication Date: 27 January 2000 (27.01.00)

(21) International Application Number: PCT/EP99/04704

(22) International Filing Date: 2 July 1999 (02.07.99)

(30) Priority Data:
60/092,726 14 July 1998 (14.07.98) US
09/276,437 25 March 1999 (25.03.99) US

(71) Applicant: KONINKLIJKE PHILIPS ELECTRONICS N.V. [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

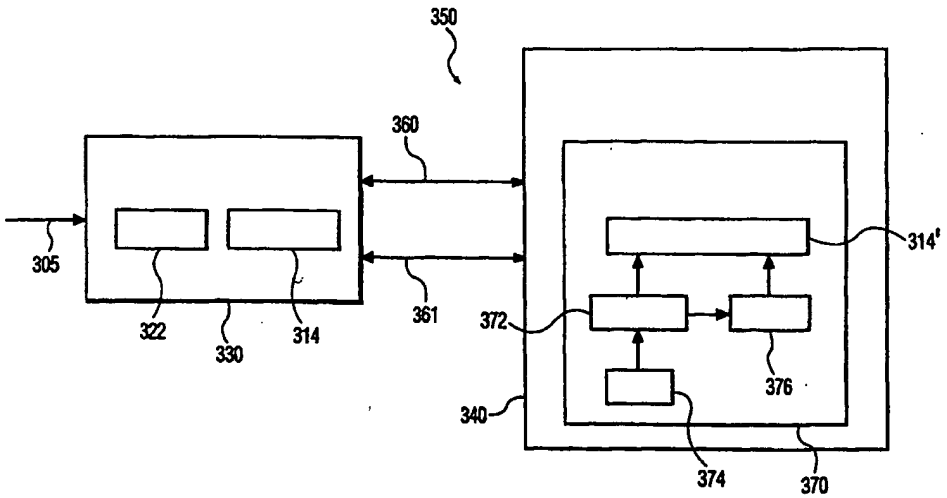
(72) Inventor: EPSTEIN, Michael, A.; Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(74) Agent: GROENENDAAL, Antonius, W., M.; Internationaal Octrooibureau B.V., Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(81) Designated States: BR, CN, JP, KR, MX, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: USE OF A WATERMARK FOR THE PURPOSE OF COPY PROTECTION



(57) Abstract

A copyright protection system for protecting content wherein a time dependent ticket is calculated (314) at a source device (330) by combining a checkpoint with a ticket. The checkpoint is transmitted (361) from a display device (340) to the source device prior to the source device transmitting (360) watermarked content to the display device. The checkpoint is also stored (376) at the display device. Thereafter, the source device transmits, to the display device, watermarked content, the ticket, and the time dependent ticket. At the display device, the stored checkpoint is compared (314) to a current count of a local clock (374) that was utilized for producing the checkpoint. If the stored checkpoint is within a window of time of the local clock, then the stored checkpoint is combined (314') with the ticket in the same way that the checkpoint is combined with the ticket at the source device. A result of the combination is compared to the time dependent ticket and if the result equals the time dependent ticket, then the watermark and ticket may be compared in the usual way to determine the copy protection status of the copy protected content (314').

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

Use of a Watermark for the Purpose of Copy Protection.

Field of the Invention

This invention generally relates to a system for protecting copyrighted content. Specifically, the present invention pertains to utilizing a ticket and a watermark to protect content.

Background of the Invention

The ability to transmit digital information securely is increasingly important. Owners of content want to be able to provide the content to authorized users without having the content utilized by unauthorized users. However, one problem with digital content is that an exact copy can be made without any degradation in the quality of the copy. Therefore, the copying of digital content is very attractive to pirating operations or attackers.

Both small-scale and commercial pirates are interested in defeating copy-protected content in order to produce and sell illegal copies of the content. By avoiding payments to the rightful owner of the copy-protected content, the pirates may reap large profits. Typically, the pirate may take advantage of the difference in release windows in order access high value content and distribute it.

For instance, in the movie industry, release windows are utilized to maximize profit from content. The essence of these release windows is to first release the content to a premium service such as a pay-per-view service or a video on demand service. Thereafter, the content may be released on a lower price service such as a home-box-office service. At this time, the content may also be available to a consumer through a purchased storage medium such as a Digital Video Disc (DVD).

Pirates however, frustrate the use of these release windows by pirating the content that is available through the premium service and then releasing pirated versions of the content to the public. This may cause substantial financial losses to the rightful owners of the content. Accordingly, a successful copy protection scheme should at least frustrate a pirates attempt for a sufficient period of time till the legitimate owner of the content may reap their rightful profits.

Beyond some level of attacker, the expense of defeating the attacker exceeds a reasonable limit whereby the device must be priced beyond what consumer is willing to pay. Thus, a copy protection solution must be cost effective but secure against a large number of attackers.

A cost-effective method of copy protection is discussed in detail by Jean-Paul Linnartz et al., in Philips Electronics Response to Call for Proposals Issued by the Data Hiding Subgroup Copy Protection Technical Working Group, July 1997 ("Linnartz"). Within a digital transmission, such as an MPEG transport stream, additional data may be embedded within the transport stream to set the copy protection status of content contained within the digital transmission. For instance, the desired copy protection status may be "copy-once", "no-more-copy", "copy-never", and "copy-freely". Content that has a status of copy-once may be played and copied. During copying, the copy-once content is altered such that the content is in the no-more-copy state. Copy-never content is content that may only be played and may not be copied. Copy-freely content may be played and copied without restriction.

The additional data may take the form of a digital watermark. The watermark may be embedded directly into the content so that removal of the watermark will degrade the quality of the content. The watermark may be utilized as part of the copy protection scheme. As an example, the copy-freely state may be designated by the lack of a watermark within the content.

In operation, a transmission, such as a digital transmission, is sent from a source device and received by a receiving device. A source device is a device that is writing content onto a data bus, initiating a broadcast transmission, initiating a terrestrial transmission, etc. A sink device is a device that reads content from the data bus, etc.

Fig. 1 shows a typical system for the transmission of content. In Fig. 1, the source device is a broadcast initiator 101 that utilizes a transmitting antenna 102 to transmit content. The sink device is a broadcast receiver, such as a set-top-box (STB) 104 that utilizes a receiving antenna 103 for receiving the transmitted content. The STB 104 is shown connected to a display device 105, a player 106, and a player/recorder 107, through a bus 108. The term bus is utilized herein to refer to any system for connecting one device to another device. The bus may be a hard wired system such as a coaxial wire, an IEEE 1553 bus, etc., or the bus may be a wireless system such as an infra-red (IR) or radio frequency (RF) broadcast system. Several of the devices shown in Fig. 1 may at one time act as a source device and at another time act as a sink device. The STB 104 may be a sink for the broadcast transmission and be a

source for a transmission on the bus 108. The player/recorder 107 may be a source/sink of a transmission to/from, respectively, the bus 108.

In the copy protection scheme discussed by Linnartz, a watermark (W) is embedded within transmitted content. A ticket is transmitted along with the transmitted content. The embedded watermark and the ticket together are utilized to determine the copy protection status of the transmitted content. The watermark may be embedded into the content by at least two known methods. One method embeds the watermark (W) in the MPEG coding of the content. Another method embeds the watermark (W) in the pixel data of the content. The ticket (T) is mathematically related to the watermark (W) as discussed in more detail below.

Performing one or more one-way functions on the ticket (T) derives the watermark (W). By use of the term one-way function, what is meant is that it is computationally unfeasible to compute the inverse of the function. An example of a publicly known mathematical one-way function is a hashing function, such as secure hash algorithm one (SHA-1) or RACE Integrity Primitives Evaluation Message Digest (RIPEMD). Computing an inverse means finding which particular x_0 leads to a given y_0 with $y_0=F(x_0)$. The term unfeasible is intended to mean that the best method will take too long to be useful for a pirate. For instance, the time that is required for a pirate to compute the inverse of a hashing function is too long for the pirate to frustrate the intended release window for protected content. The most efficient method known to find such an x_0 may be to exhaustively search all possible bit combinations of x_0 and to compute and verify $F(x_0)$ for each attempt. In other cases, there may be a more efficient method than an exhaustive search to compute an inverse of a one-way function, yet these methods are still too time consuming to be feasible for the pirate.

The bit content of the ticket (T) is generated from a seed (U). The content owner provides the seed (U). From the seed (U), a physical mark (P) is created. The physical mark (P) may be embedded on a storage medium such as a Read-Only Memory (ROM) disk. Performing one or more one-way functions on the physical mark (P), produces the ticket (T). The number of functions performed on the physical mark (P) to create the ticket (T) depends on the copy protection intended for the content.

In accordance with the system, the ticket (T) changes state during every passage of a playback device (e.g., a source device) and a recording device (e.g., a sink device). As discussed above, the state modifications are mathematically irreversible and reduce the remaining copy and play rights of the content that are granted by the ticket (T). In this way,

the ticket (T) indicates the number of sequential playback and recordings that may still be performed and acts as a cryptographic counter that can be decremented but not incremented.

It should be noted that the copy protection scheme only protects content on compliant systems. A compliant system is any system that obeys the copy protection rules described above and hereinafter. A non-compliant system may be able to play and copy material irrespective of the copy protection rules. However, a compliant system should refuse to play copies of content illegally made on a non-compliant system.

In accordance with the copy protection scheme, a physical mark (P) (e.g., data) is embedded on a storage medium and is not accessible by other user equipment. The physical mark (P) data is generated at the time of manufacturing of the storage medium as described above and is attached to the storage medium in a way in which it is difficult to remove the physical mark (P) data without destroying the storage medium. The application of a one-way mathematical function, such as a hashing function, to the physical mark (P) data four times results in a watermark. Much like watermarks embedded in paper, the watermark is embedded in the medium (e.g., containing video, audio, or data) in such a way that it is infeasible to remove the watermark without destroying the material. At the same time the watermark should be imperceptible when the medium is used in the usual manner, such as when content from the medium is displayed.

A watermark by itself may indicate whether or not content stored on the storage medium is copy-once or copy-never. For instance, the absence of a watermark may indicate that the content may be copied freely. The presence of the watermark without a ticket on a storage medium may indicate copy-never content.

When the content is transmitted over a bus or other transmission medium, the physical mark (P) data is hashed twice to generate a ticket. When a compliant player receives the content, the ticket is hashed twice and matched to the watermark. In the case where the twice-hashed ticket and the watermark match, the content is played. In this way, a party may not substitute a false ticket along with the content to frustrate the copy protection scheme. In the case were there is a watermark but no ticket in the content, a compliant system will refuse to record the content.

When a compliant recorder reads the content, the watermark is checked to see if the material is copy-freely, copy-once, or copy-never. When there is no watermark, the content is copy-freely and may be copied freely as discussed above. When the content contains a watermark but no ticket, the content is copy-never and a compliant recorder will refuse to copy the content. However, a compliant player will play the content as long as the ticket

hashed two times matches the watermark. When the content is copy-once, the content contains both a watermark and a ticket, a compliant recorder will hash the ticket twice and compare the twice-hashed ticket to the watermark. In the case where the watermark matches the twice-hashed ticket, the content may be recorded along with a once-hashed ticket and the watermark, thereby creating copy-no-more content (e.g., content with a once-hashed ticket and a watermark). The physical mark will be different on a writable disc and thus, even if an illegal copy is made of copy-never content via a non-compliant recording device, a compliant player will refuse to play the content recorded on the writable disc.

It should be noted that in a broadcast system, such as a pay-per-view system, a copy-never state may be indicated by the presence of a once-hashed ticket and a watermark. Both copy-no-more stored content and copy-never broadcast content are treated by a compliant system similarly. The content containing the once-hashed ticket may be played but may not be recorded in a compliant system. In the event that a party tries to record the content with the once-hashed ticket, a compliant recorder will first twice-hash the once-hashed ticket and compare the result (e.g., a thrice-hashed ticket) with the watermark. Since the thrice-hashed ticket will not match the watermark, the compliant recorder will refuse to record the content.

A compliant player that receives the once-hashed ticket will hash the once-hashed ticket and compare the result (e.g., a twice-hashed ticket) to the watermark. Since the twice-hashed ticket matches the watermark, the compliant player will play the content.

However, a problem exists wherein a non-compliant recorder receives content containing a ticket (a twice-hashed physical mark) and a watermark. In the event that a non-compliant recorder does not alter the ticket upon receipt or recording (e.g., the non-compliant recorder makes a bit-for-bit copy), the non-compliant recorder may make multiple copies of the ticket and the watermark that will play on a compliant player and that may be recorded on a compliant recorder. The same problem can exist where a non-compliant recorder receives content containing a once-hashed ticket (a thrice-hashed physical mark) and a watermark indicating copy-no-more content. In this case, the non-compliant recorder may make multiple copies of the once-hashed ticket and the watermark that will play on the compliant player.

In a case wherein the player receives the content directly from a read only medium, such as a Compact Disc ROM (CD-ROM), a physical mark can be embedded in the physical medium of the CD-ROM that is produced by an authorized manufacturer. The player may then check the physical mark to ensure that the content is being received from an authorized medium. In this way, if a pirate makes an unauthorized copy, the physical mark

will not be present on the unauthorized copy and a compliant player will refuse to play the content. However, in the case of broadcast data for instance, wherein a player does not read content directly from the read-only medium, this method of copy protection is unavailable. Thus, for instance, a non-compliant player may deceive a compliant display device.

Accordingly, it is an object of the present invention to overcome the disadvantages of the prior art.

Summary of the Invention

This object of the present invention is achieved by a copy protection system for protecting content, such as content containing a watermark embedded therein (e.g., watermarked content). To this end, the invention provides a content protecting method, a copy protection system, a source device, and a display device as defined in the independent claims. The dependent claims define advantageous embodiments. In accordance with the present invention, a relative time dependent ticket is created at a source device preferably utilizing a display device dependent time reference (a checkpoint). In accordance with one embodiment of the present invention, the checkpoint is combined with a ticket utilizing a concatenation function and a one-way function (e.g., a hashing function). The checkpoint is transmitted from the display device to the source device prior to the source device transmitting watermarked content to the display device. The checkpoint is also stored at the display device. Thereafter, the source device transmits to the display device watermarked content, the ticket, and the relative time dependent ticket.

At the display device, the stored checkpoint is compared to a current relative time reference. If the difference between the stored checkpoint and the current relative time reference is acceptable, then further steps, as discussed below, may proceed. What is an acceptable difference between the stored checkpoint and the current relative time reference will depend on the nature of the desired content protection. For example, in one embodiment or for one particular type of content, the difference may be short to ensure that the content is being transmitted and received in real time. In another embodiment or for another type of content, the difference may be longer to allow for storage of the content for later playback.

When the difference between the stored checkpoint and the current relative time reference is acceptable, the ticket is next hashed twice and compared to the watermark in the usual way. In the event that the ticket compares to the watermark ($W = H(H(T))$), the stored checkpoint is combined with the ticket in the same way that the checkpoint was combined with the ticket at the source device. A result of the combination is compared to the relative

time dependent ticket. If the result equals the relative time dependent ticket, then the display device is provided with access (e.g., enabled to display) to the watermarked content.

Preferably, the checkpoint is derived from a counter that purposely is inaccurate such that the count can be said to be unique as compared to the count from other display devices. The counter is constructed with a sufficient number of bits such that the counter will not roll over to zero in the lifetime of the display device. The counter is constructed to only count up, such that the count may not be reversed and thereby, allow expired content to be displayed.

In yet another embodiment, a certificate containing the public key of the source device is sent to the display device prior to the above described process. A public key known to the display device may be used to verify the certificate. Preferably, the public key used to verify the certificate is built into the display device by the manufacturer of the display device. In this embodiment, the relative time dependent ticket (the checkpoint concatenated with the ticket) may be encrypted utilizing a private key of the source device. The encrypted relative time dependent ticket is then transmitted from the source device to the display device along with the watermarked content and the ticket. Thereafter, prior to the display device verifying the checkpoint, the display device decrypts the relative time dependent ticket utilizing a public key of the source device. In still yet another embodiment, the relative time dependent ticket may be signed (as is know in the art, by hashing the relative time dependent ticket and encrypting that hashed result) utilizing a private key of the source device. The resulting signature is sent along with the watermarked content, the relative time dependent ticket, and ticket to the display device. Thereafter, prior to the display device verifying the checkpoint, the display device verifies the signature on the relative time dependent ticket utilizing a public key of the source device.

Brief Description of the Drawings

The following are descriptions of embodiments of the present invention that when taken in conjunction with the following drawings will demonstrate the above noted features and advantages, as well as further ones. It should be expressly understood that the drawings are included for illustrative purposes and do not represent the scope of a present invention. The invention is best understood in conjunction with the accompanying drawings in which:

Fig. 1 shows a conventional system for the transmission of content;

Fig. 2 shows an illustrative communication network in accordance with an embodiment of the present invention; and

Fig. 3 shows details of an illustrative communication network in accordance with embodiment of the present invention wherein a source device provides content to a sink device.

Detailed Description of the Invention

Fig. 2 depicts an illustrative communication network 250 in accordance with an embodiment of the present invention. A source device 230, such as Set Top Box (STB), a Digital Video Disc (DVD), a Digital Video Cassette Recorder (DVCR), or another source of content, utilizes a transmission channel 260 to transmit content to a sink device 240. The transmission channel 260 may be a telephone network, a cable television network, a computer data network, a terrestrial broadcast system, a direct broadcast satellite network, some combination thereof, or some other suitable transmission system that is know in the art. As such, the transmission channel 260 may include RF transmitters, satellite transponders, optical fibers, coaxial cables, unshielded twisted pairs of wire, switches, in-line amplifiers, etc. The transmission channel 260 may also operate as a bi-directional transmission channel wherein signals may be transmitted from/to the source device 230, respectively, to/from the sink device 240. An additional transmission channel 261 may also be utilized between the source device 230 and the sink device 240. Typically, the transmission channel 260 is a wide-bandwidth channel that in addition to transmitting copy protection content (e.g., copy protection related messages), transmits copy protected content. The transmission channel 261 typically is a low-bandwidth channel that is utilized to transmit copy protection content.

The sink device 240 contains a memory 276 that is utilized for storing a checkpoint. The sink device 240 also contains a counter, such as a counter 272, that is utilized for generating the checkpoint. Preferably, the counter 272 should increment on a microsecond or better resolution as suitable for the application. The counter 272 should be free running. For instance, the counter 272 should count at all times that the sink device 240 is on. The bits of the counter 272 should employ non-volatile memory such as an electrically erasable programmable read-only memory (EEPROM) for the storage of the count. The counter 272 preferably is constructed to only count in one direction (e.g., up) and not in another direction (e.g., down). In a preferred embodiment, the counter 272 is driven by an inaccurate time source (e.g., inaccurate in terms of keeping time over hours, not necessarily over seconds), such as clock 274. The clock 274 is preferably unreliable so that drift with respect to time and

temperature is also non-negligible. Over time, this has the effect of randomizing the count of a counter for each sink device of a population of sink devices. In addition, the counter 272 may be driven fast for a random period of time to initialize the counter 272 to a random number at the time of manufacture. All of the above, has an effect of further randomizing the counter 272. The counter 272 is also configured such that it is inaccessible to a user. Accordingly, the user may not reset the counter 272.

The checkpoint, in accordance with the present invention, is transmitted to the source device 230 utilizing at least one of the transmission channels 260, 261. The source device 230 utilizes the checkpoint to change the ticket such that the watermarked content may only be utilized (e.g., played) by a corresponding sink device as described in more detail below. In the event that the corresponding sink device, such as the sink device 240, receives the watermarked content, then the content may be provided to a device, such as a display device 265, for display thereon. Preferably, the display device 265 is integral to the sink device 240 such that the display device 265 is the final arbiter in determining whether the copy protected content may be utilized. It should be obvious that although the device is illustratively shown as the display device 265, in fact the device may be any known device that may be suitably utilized for the copy protected content. For instance, in a case wherein the copy protected content is audio content, the device may be the device that outputs the audio signal.

In one embodiment of the present invention, the content may be provided from the source device 230 in the form of a Moving Picture Experts Group (MPEG) compliant transport stream, such as an MPEG-2 compliant transport stream. However, the present invention is not limited to the protection of an MPEP-2 compliant transport stream. As a person skilled in the art would readily appreciate, the present invention may be suitably employed with any other data stream that is known in the art for transmitting content.

In another embodiment, the source device 230 may be a conditional access (CA) device. In this embodiment, the transmission channel 260 is a conditional access module bus.

Fig. 3 depicts details of an illustrative communication network 350 in accordance with an embodiment of the present invention. In the communication network 350, a source device 330 provides content including copy protected content to a sink device 340 over a transmission channel 360. As discussed above with regard to the transmission channel 260, the transmission channel 360 may be a wide bandwidth transmission channel that may also have a bi-directional capability, such as a CA module bus.

The sink device 340 contains a copy protection status determination circuit 370 for creating/storing a checkpoint (C) and for determining the copy protection status of received content. The copy protection status determination circuit 370 contains a counter 372 and a clock 374 for creating the checkpoint (C). The counter 372 preferably contains a large number of bits (e.g., 64 bits for a clock 374 that increments on a millisecond basis). Preferably, the counter 372 should have a total count cycle time (the time required for the counter 372 to reach a top count from a bottom count) longer than a useful life of the sink device 340 (e.g., ten years). The clock 374 is preferably randomized (e.g., unreliable such that drift with respect to time and temperature is non-negligible) as discussed above with regard to the clock 274 shown in Fig. 2. The counter 372 is configured such that it is inaccessible and has no reset function even in the event of a removal of power. As such, the counter 372 may contain non-volatile storage, such as programmable read-only memory (PROM), electrically erasable PROM (EEPROM), static random access memory (static-RAM), etc. Further, the copy protection status determination circuit 370 contains a memory device 376 for storing the checkpoint (C).

In operation, the source device 330 may request the checkpoint (C) from the sink device 340 prior to transmitting copy protected content. In alternate embodiments, the sink device 340 may transmit the checkpoint (C) to the source device 330 as a portion of a request for the source device 330 to begin transmission of copy protected content to the sink device 340. The sink device 340 may utilize either of the transmission channels 360, 361 for transmission of the request for copy protected content and/or for transmission of the checkpoint (C). However, in some embodiments of the present invention, the transmission channel 360 may be unidirectional and may only be utilized for the transmission of content to the sink device 340 from the source device 330. In these embodiments, the transmission channel 361 is utilized for the transmission of the checkpoint (C) from the sink device 340 to the source device 330. The transmission channel 361 may also be utilized for transmitting a request for copy protected content from the sink device 340 to the source device 330.

In an alternate embodiment, the transmission channel 360 has bi-directional capability and may be utilized for transmissions both to and from the source device 330, and to and from the sink device 340. In this embodiment, the transmission channel 361 may not be present or it may be utilized solely for the transmission of content requiring low bandwidth. For instance, the source device 330 may utilize the transmission channel 361 to transmit to the sink device 340 a request for the transmission of the checkpoint (C).

In one particular embodiment, the source device 330 is a conditional access (CA) device 330, the transmission channel 360 is a CA module bus 360, and the sink device 340 is a display device 340. Prior to the transmission of copy protected content, the CA device 330 transmits a request for a checkpoint (C) (e.g., the current count from the free running counter 372) from the display device 340. In response to the request, the display device 340 transmits the checkpoint (C) to the CA device 330 over the CA module bus 360. In addition to sending the checkpoint (C) to the CA device 330, the display device 340 saves the checkpoint (C) in the memory 376.

The CA device 330 contains a processor 314. The processor 314 utilizes a ticket and the checkpoint (C), received from the display device 340, to create a relative time dependent ticket (TDT) as discussed in more detail below. In one embodiment, the processor 314 may simply be a fixed hardware device that is configured for performing functions, such as mathematical functions, including a concatenation function, a one-way function, such as a hashing function, etc. In alternate embodiments, the processor 314 may be a microprocessor or a reconfigurable hardware device. What is intended by the term "relative time dependent ticket (TDT)" is that due to the randomization of the counter 372 as discussed above, the checkpoint (C) is not directly related to an absolute time amongst all sink devices. The checkpoint (C) is only related to a relative time of a given sink device such as the display device 340.

In one embodiment, the copy protected content is received via an input 305 as an audio/video (A/V) signal. Preferably, in this embodiment, the A/V signal contains a watermark (W) and a ticket (T). The watermark (W) and the ticket (T) are related as discussed with regard to the prior art (e.g., $W = H(H(T))$). Preferably, the watermark (W) is embedded into the copy protected content. In this way, removal of the watermark (W) from the copy protected content will result in the copy protected content becoming largely degraded. The ticket accompanies the content and is not embedded in it.

In an alternate embodiment, the copy protected content is read from a physical medium, such as a digital video disc (DVD). In this embodiment, the DVD may contain a physical mark (P) as described above. Further, content contained on the DVD (e.g., A/V content) has a watermark (W) embedded therein (e.g., watermarked content) such that removal of the watermark (W) from the A/V content results in the A/V content becoming largely degraded. In this embodiment, the physical mark (P), the ticket (T), and the watermark (W) are related as follows:

$$T = H(H(P)) \quad (1)$$

$$W = H(H(T)) \quad (2)$$

In any event, at the CA device 330, the checkpoint (C) is combined with the ticket (T), utilizing for instance concatenation and hashing functions. Thereby, a time dependent ticket (TDT) is created as follows:

$$TDT = H(T.C). \quad (3)$$

The watermarked content, containing a watermark (W) embedded therein, the time dependent ticket (TDT), and the ticket (T), are then transmitted via the CA module bus 360 to the display device 340.

At the receiver 340, the copy protection status determination circuit 370 extracts the watermark (W) from the watermarked content. The copy protection status determination circuit 370 compares the watermark (W) and the ticket (T) in the usual way, as is known in the art (e.g., $W = H(H(T))$?).

In the event that the comparison does not pass (e.g., $W \neq H(H(T))$), then the content is discarded and any selected operation at the display device 340 (e.g., play, record, etc.) regarding the content is disabled. However, if the comparison does pass (e.g., $W = H(H(T))$), then the copy protection determination circuit 370 retrieves the stored checkpoint (C) from the memory 376 and combines the ticket (T) with the stored checkpoint (C), utilizing the same operation that was utilized at the source device 330 for creating the time dependent ticket (TDT). To this end, the receiver 340 comprises a processor 314' that is comparable to the processor 314 in the source device 330. For instance, concatenation and hashing functions may be utilized at the display device 340 for combining the ticket (T) with the stored checkpoint (C). A result of the combination is then compared to the time dependent ticket (TDT):

$$TDT = H(T.C)? \quad (4)$$

In the event that the result does not equal the time dependent ticket (TDT), then the content is discarded and any selected operation at the display device (e.g., play, record, etc.) regarding the content is disabled. This may happen, for instance, in a case wherein an improper display device (e.g., a display device other than the display device that requested the content) has received the content. If the result does equal the time dependent ticket (TDT), then access to the content is enabled in accordance with the access granted by the ticket.

In a preferred embodiment, a further step is performed prior to the display device 340 having access to the copy protected content. Specifically, the checkpoint (C) stored in the memory 376 is compared to a current count of the (running) counter 372. In the event that the stored checkpoint (C) is within an allowable window of the current count from the counter 372 (e.g., within 24 hours of the count for some applications), then the display device 340 is provided with access to the copy protected content. What is an allowable window between the stored checkpoint (C) and the current count will depend on the nature of the desired content protection. For example, in one embodiment or for one particular type of content, the allowed window (the difference between the stored checkpoint (C) and the current count) may be short to ensure that the content is being transmitted and received in real time. In another embodiment or for another type of content, the allowed window may be longer (e.g., months or years) to allow for storage of the content for later playback.

If the checkpoint (C) has expired (e.g., not within the allowed window), then the checkpoint (C) is erased and the display device 340 is not provided with access to the copy protected content. As is readily ascertained by a person of ordinary skill in the art, the comparison of the checkpoint (C) to the current count may be performed any time prior to the display device having access to the copy protected content. In a preferred embodiment, the checkpoint (C) is compared to the current count prior to the comparison of the watermark (W) to the ticket (T).

It should be clear that a trusted source should be utilized to create the recorded content or the real time transmitted content (e.g., received over the input 305). A CA device, such as the CA device 330, which is inherently designed to be tamper resistant is an example of a trusted real time source. In this case, it may be assumed that the CA device 330 decrypts the watermarked content so that prior to the watermarked contents arrival at the CA device 330, the watermarked content cannot be recorded.

In a case wherein the ticket (T) does not properly compare to the watermark (W), or some other portion of the copy protection status determination process fails, the copy protected content is discarded. In addition, when the copy protection status determination process fails, no operation regarding the copy protected content is enabled at the display device 340.

In accordance with the present invention, a checkpoint (C) from a counter of a given display device is in effect unique. Accordingly, the copy protected content transmitted by the CA device 330 may not be distributed to a display device other than the display device that sent the checkpoint (C). In addition, by comparing the checkpoint (C) to the count of the

counter 372, the copy protected content may be restricted to being played within a time, as determined by the window of time as discussed above.

In yet another embodiment, a private/public key system, as is known by a person of ordinary skill in the art, is utilized to further secure the copy protected content in accordance with the present invention. In accordance with this embodiment, the display device 340 has a public key that is trusted e.g., secure for example by being installed in part of the display device hardware, such as stored in the memory 376. The public key corresponds to a private key of the manufacturer of the display device 340 and is stored, for instance, in a memory 322 at the CA device 330. The private key is utilized to sign certificates of each CA device manufacturer, as is known in the art.

In operation, when the CA device 330 is connected to the display device 340 via the CA module bus 360, a certificate containing the CA device 330 public key is sent to the display device 340. Once the certificate containing the public key of the CA device 330 is verified by the display device 340, as is known in the art, the public key of this CA device 330 is stored at the display device 340. Thereafter, the CA device 330 may digitally sign the time dependent ticket (TDT). For instance, the time dependent ticket (TDT) may be hashed and the result may be encrypted by the private key of the CA device 330 to form a signature. The signature is sent from the CA device 330 to the display device 340 together with the watermarked content, the ticket, and the time dependent ticket (TDT). At the display device 340, the signature is verified utilizing the public key of the CA device 330 and thereafter, the time dependent ticket (TDT) and checkpoint (C) are utilized as described above.

In yet another embodiment, the time dependent ticket (TDT) may be encrypted utilizing the private key of the CA device 330. The encrypted time dependent ticket (TDT) is then transmitted from the CA device 330 to the display device 340 along with the watermarked content and the ticket (T). Thereafter, prior to the display device 340 verifying the checkpoint (C), the display device 340 decrypts the time dependent ticket (TDT) utilizing the public key of the CA device 330. Thereafter, the time dependent ticket (TDT) may be utilized as discussed above.

An illustrative protocol for use of a checkpoint and a private/public key system in accordance with an embodiment of the present invention is described below. In accordance with the present invention, after a CA device is connected to a display device, the CA device sends a certificate containing the CA device public key to the display device. The display device verifies the certificate utilizing the embedded public key of the manufacturer and stores the verified public key of the CA device. In response to a request for copy protected content

from the display device, the CA device requests a checkpoint (C) from the display device. The display device sends the checkpoint (C) to the CA device and also stores a copy of the checkpoint (C) locally (e.g., at the display device). The CA device combines the checkpoint (C) with the ticket (T) utilizing concatenation and hashing functions to produce a time dependent ticket (TDT). The CA device encrypts the time dependent ticket (TDT) utilizing the CA device private key. The encrypted time dependent ticket (TDT) is then sent to the display device along with the watermarked content and the ticket (T). The display device compares the stored checkpoint (C) with the current state of a counter to determine if the checkpoint (C) is within an allowable window of time of the current state of the counter. If the stored checkpoint (C) is not within the allowable window of time of the current state of the counter, then access to the content is disabled. If the stored checkpoint (C) is within the allowable window, then the display device utilizes the public key of the CA device to decrypt the time dependent ticket (TDT). The display device combines the ticket (T) with the stored checkpoint (C) utilizing concatenation and hashing functions and compares a result to the time dependent ticket (TDT). If the result is not equal to the time dependent ticket (TDT), then access to the content is disabled. If the result is equal to the time dependent ticket (TDT), the ticket and watermark are compared in the usual way. If step 480 fails (e.g., $W \neq H(H(T))$), then in step 485, access to the content is disabled. If the ticket and the watermark do no correspond, (e.g., $W = H(H(T))$), access to the content is enabled (e.g., the content may be displayed).

The following embodiments of the invention overcome the disadvantages of the prior art. A display device is provided that is the final arbiter in deciding whether to display the protected content. In this way, the display device is the gatekeeper that disallows recordings that are made and played back on non-compliant players/recorders. A further embodiment provides a method of transmitting copy protected copy-never content that will prevent a pirate from making copies that will display on a compliant display device. A ticket is created that is unique to a particular display device so that copy protected content will only play on the particular display device. A still further embodiment creates a ticket that is inspected by the display device to decide whether the content is being transmitted in real time. A time dependent ticket is created that is checked by a display device to determine if content has expired or aged beyond an allowable window of time from a checkpoint. Another embodiment of the invention uses a relative time reference configured such that each display device has a different relative time reference.

It should be noted that the above-mentioned embodiments illustrate rather than limit the invention, and that those skilled in the art will be able to design many alternative

embodiments without departing from the scope of the appended claims. In the claims, any reference signs placed between parentheses shall not be construed as limiting the claim. The word "comprising" does not exclude the presence of other elements or steps than those listed in a claim. Another embodiment of the invention can be implemented by means of hardware comprising several distinct elements, and by means of a suitably programmed computer. In a device claim enumerating several means, several of these means can be embodied by one and the same item of hardware.

CLAIMS:

1. A method of protecting content transmitted as a stream of data, the method comprising the steps of:

determining a checkpoint at a receiving device (240);

calculating, at a source device (230), a time dependent ticket utilizing the checkpoint, wherein a watermark, a ticket, and the checkpoint together indicate a copy protection status of the content;

transmitting said stream of data, said watermark, said ticket, and said time dependent ticket to said receiving device (240); and

comparing said time dependent ticket to a stored checkpoint at said receiving device (240).

2. The method of claim 1, wherein said step of calculating said time dependent identifier comprises the steps of:

combining said checkpoint with said ticket, and

calculating a one-way operation on said combined checkpoint and ticket.

3. The method of claim 2, further comprising the step of selecting said one-way function to be a hashing function.

4. The method of claim 1, further comprising the step of comparing, at said receiving device (240), said ticket and said watermark to determine the copy protection status of the content if said time dependent ticket compares to said stored checkpoint.

5. The method of claim 1, wherein said checkpoint is a checkpoint from a receiver counter (272).

6. The method of claim 5, wherein said receiver counter (272) is randomized.

7. The method of claim 5, wherein the step of comparing said time dependent ticket further comprises the step of comparing said stored checkpoint to a current count from said receiver counter (272).

8. The method of claim 1, wherein said step of calculating said time dependent ticket further comprises the step of signing said time dependent ticket with a private key of said source device (230), and wherein said step of comparing said time dependent ticket further comprises the step of verifying the signature using a public key of said source device (230).

9. A copy protection system for protecting content wherein a ticket and a watermark indicates a copy protection status of said content, the system comprising:

a source device (330) configured to calculate a time dependent ticket using a checkpoint and a one-way function, and to provide a data stream containing said content, said ticket, a watermark, and said time dependent ticket; and

a display device (340) configured to produce said checkpoint, configured to receive said data stream, and configured to compare said time dependent ticket to said checkpoint using said ticket and said one-way function.

10. The system of claim 9, wherein said display device (340) is further configured to compare said ticket to said watermark and to display said content if said time dependent ticket compares to said checkpoint.

11. The system of claim 9, wherein said display device (340) comprises a counter (372) and wherein said checkpoint is a checkpoint from said counter (372).

12. The system of claim 11, wherein said display device (340) is further configured to randomize said counter (372).

13. The system of claim 11, wherein said display device (340) is further configured to compare said checkpoint to a current count from said counter (372) prior to displaying said content.

14. A source device (330) for protecting content wherein a ticket and a watermark indicate a copy protection status of the content, said source device (330) comprising:

a reader device configured to read watermarked content from a physical medium and configured to read a physical mark from said physical medium; and

a processor (314) configured to receive a checkpoint, configured to calculate said ticket using said physical mark and a one-way function, configured to calculate a time dependent ticket using said ticket, said checkpoint, and said one-way function, and configured to provide to a receiver (340) a data stream containing said watermarked content, said ticket, and said time dependent ticket.

15. A display device (340) for receiving data containing watermarked content and a ticket, wherein said ticket and watermark together indicate a copy protection status of the content, said display device comprising:

a counter (372) configured to provide a checkpoint and a current time reference; and

a processor (314'), wherein if said checkpoint is contained within a time window determined by said current time reference, said (314') processor is configured to:

receive a time dependent ticket and said data,

combine said ticket with said checkpoint to produce a first result,

perform a one-way function on said first result to produce a second result, and compare said second result to said time dependent ticket, wherein said display device (340) is further configured to display said data if said second results compares to said time dependent ticket.

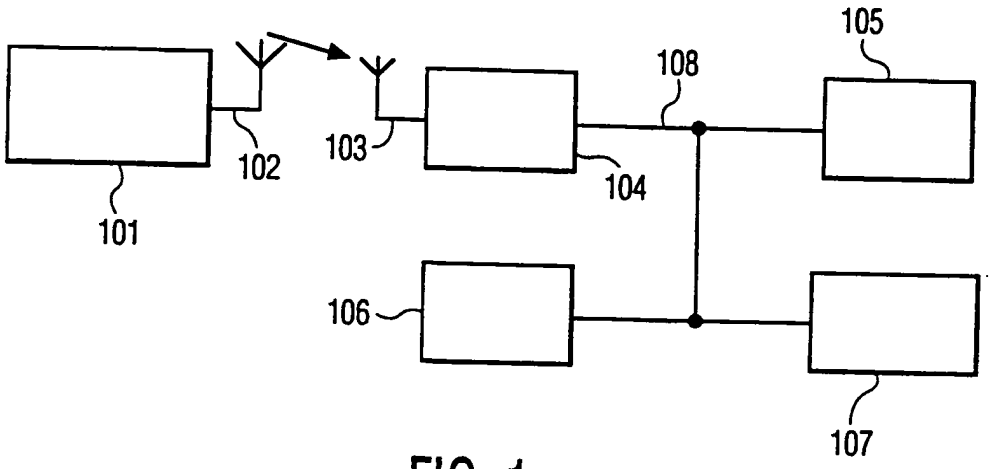

105
108
102
103
104
101
106
107

FIG. 1

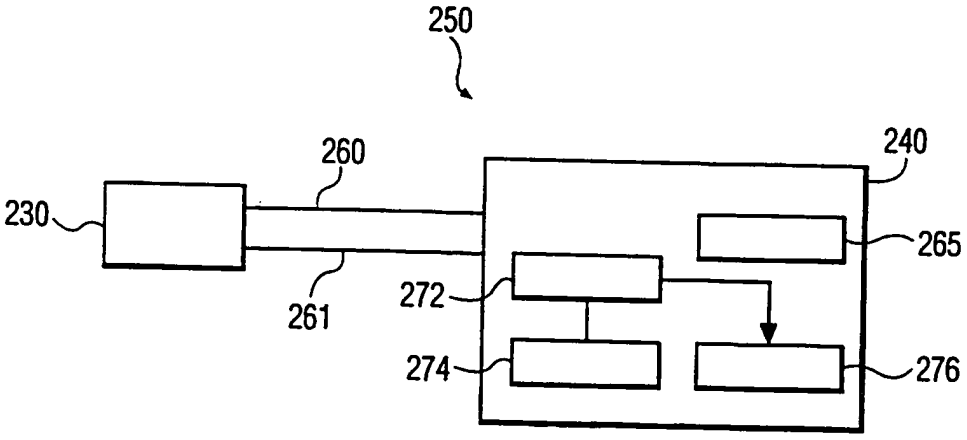

250
260
240
230
265
272
261
274
276

FIG. 2

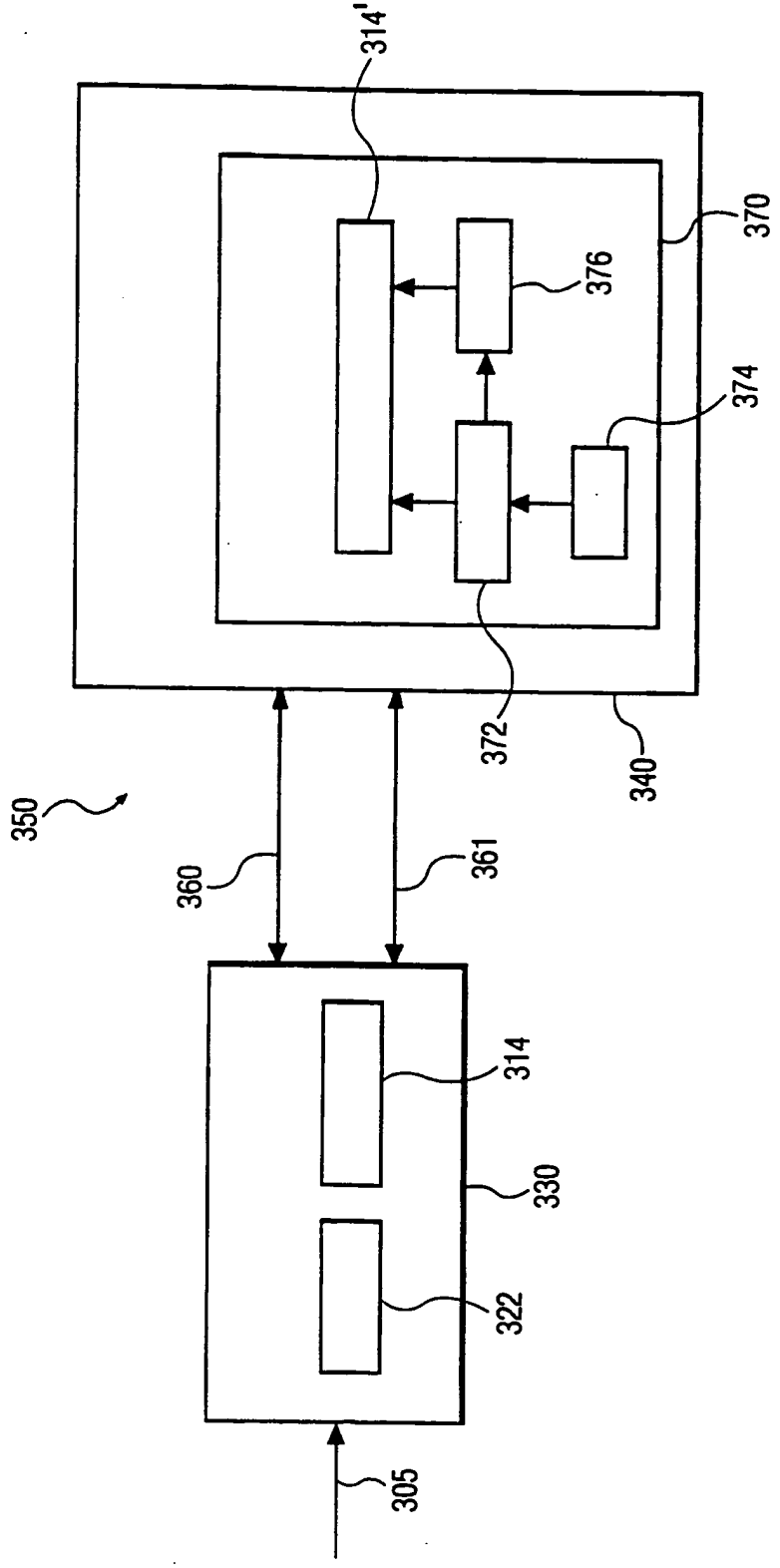


FIG. 3

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04L 9/30 | A1

(11) International Publication Number: WO 99/35782

(43) International Publication Date: 15 July 1999 (15.07.99)

(21) International Application Number: PCT/US98/27896

(22) International Filing Date: 31 December 1998 (31.12.98)

(30) Priority Data:

60/070,344	2 January 1998 (02.01.98)	US
60/089,529	15 June 1998 (15.06.98)	US

(71) Applicant: CRYPTOGRAPHY RESEARCH, INC. [US/US]; Suite 1088, 870 Market Street, San Francisco, CA 94102 (US).

(72) Inventors: KOCHER, Paul, C.; 143 Fillmore Street, San Francisco, CA 94117 (US). JAFFE, Joshua, M.; 21B Bird Street, San Francisco, CA 94110 (US).

(74) Agents: LAURIE, Ronald, S. et al.; Skadden, Arps, Slate, Meagher & Flom LLP, 525 University Avenue, Palo Alto, CA 94301 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published

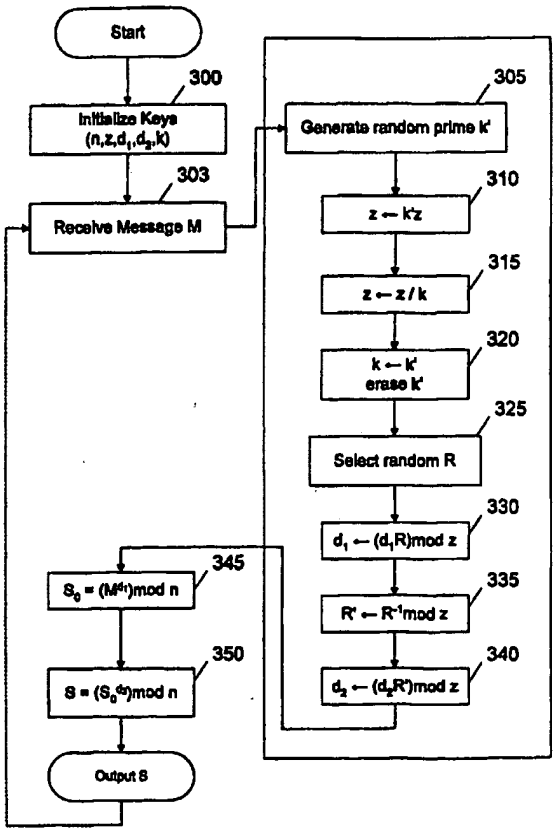
With international search report.

Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: LEAK-RESISTANT CRYPTOGRAPHIC METHOD AND APPARATUS

(57) Abstract

The present invention provides a method and apparatus for securing cryptographic devices against attacks involving external monitoring and analysis. A "self-healing" property is introduced, enabling security to be continually re-established following partial compromises. In addition to producing useful cryptographic results, a typical leak-resistant cryptographic operation modifies or updates (330) secret key material in a manner designed to render useless any information about the secrets that may have previously leaked from the system. Exemplary leak-proof and leak-resistant implementations of the invention are shown for symmetric authentication (350), certified Diffie-Hellman (when either one or both users have certificates), RSA, ElGamal public key decryption (303).



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

LEAK-RESISTANT CRYPTOGRAPHIC METHOD AND APPARATUS

This application claims the benefit of US Provisional Application No. 60/070,344 filed January 2, 1998, and US Provisional Application No. 60/089,529 filed June 15, 1998.

TECHNICAL FIELD

The method and apparatus of the present invention relate generally to cryptographic systems and, more specifically, to securing cryptographic tokens that must maintain the security of secret information in hostile environments.

BACKGROUND OF THE INVENTION

Most cryptosystems require secure key management. In public-key based security systems, private keys must be protected so that attackers cannot use the keys to forge digital signatures, modify data, or decrypt sensitive information. Systems employing symmetric cryptography similarly require that keys be kept secret. Well-designed cryptographic algorithms and protocols should prevent attackers who eavesdrop on communications from breaking systems. However, cryptographic algorithms and protocols traditionally require that tamper-resistant hardware or other implementation-specific measures prevent attackers from accessing or finding the keys.

If the cryptosystem designer can safely assume that the key management system is completely tamper-proof and will not reveal any information relating to the keys except via the messages and operations defined in the protocol, then previously known cryptographic techniques are often sufficient for good security. It is currently extremely difficult, however, to make hardware key management systems that provide good security, particularly in low-cost unshielded cryptographic devices for use in applications where attackers will have physical control over the device. For example, cryptographic tokens (such as smartcards used in electronic cash and copy protection schemes) must protect their keys even in potentially hostile environments. (A token is a device that contains or manipulates cryptographic keys that need to be protected from attackers. Forms in which tokens may be manufactured include, without limitation, smartcards, specialized encryption and key management devices, secure telephones, secure picture phones, secure web servers, consumer electronics devices using cryptography, secure microprocessors, and other tamper-resistant cryptographic systems.)

A variety of physical techniques for protecting cryptographic devices are known, including enclosing key management systems in physically durable enclosures, coating integrated circuits with special coatings that destroy the chip when removed, and wrapping devices with fine wires that detect tampering. However, these approaches are expensive, difficult to use in single-chip solutions (such as smartcards), and difficult to evaluate since there is no mathematical basis for their security. Physical tamper resistance techniques are also ineffective against some attacks. For example, recent work by Cryptography Research has shown that attackers can non-invasively extract secret keys using careful measurement and analysis of many devices' power consumption. Analysis of timing measurements or electromagnetic radiation can also be used to find secret keys.

Some techniques for hindering external monitoring of cryptographic secrets are known, such as using power supplies with large capacitors to mask fluctuations in power consumption, enclosing devices in well-shielded cases to prevent electromagnetic radiation, message blinding to prevent timing attacks, and buffering of inputs/outputs to prevent signals from leaking out on I/O lines. Shielding, introduction of noise, and other such countermeasures are often, however, of limited value, since skilled attackers can still find keys by amplifying signals and filtering out noise by averaging data collected from many operations. Further, in smartcards and other tamper-resistant chips, these countermeasures are often inapplicable or insufficient due to reliance on external power sources, impracticality of shielding, and other physical constraints. The use of blinding and constant-time mathematical algorithms to prevent timing attacks is also known, but does not prevent more complex attacks such as power consumption analysis (particularly if the system designer cannot perfectly predict what information will be available to an attacker, as is often the case before a device has been physically manufactured and characterized).

The present invention makes use of previously-known cryptographic primitives and operations. For example: U.S. patent 5,136,646 to Haber et al. and the pseudorandom number generator used in the RSAREF cryptographic library use repeated application of hash functions; anonymous digital cash schemes use blinding techniques; zero knowledge protocols use hash functions to mask information; and key splitting and threshold schemes store secrets in multiple parts.

SUMMARY OF THE INVENTION

The present invention introduces leak-proof and leak-resistant cryptography, mathematical approaches to tamper resistance that support many existing cryptographic primitives, are inexpensive, can be implemented on existing hardware (whether by itself or via software capable of running on such hardware), and can solve problems involving secrets leaking out of cryptographic devices. Rather than assuming that physical devices will provide perfect security, leak-proof and leak-resistant cryptographic systems may be designed to remain secure even if attackers are able to gather some information about the system and its secrets. This invention describes leak-proof and leak-resistant systems that implement symmetric authentication, Diffie-Hellman exponential key agreement, ElGamal public key encryption, ElGamal signatures, the Digital Signature Standard, RSA, and other algorithms.

One of the characteristic attributes of a typical leak-proof or leak-resistant cryptosystem is that it is "self-healing" such that the value of information leaked to an attacker decreases or vanishes with time. Leak-proof cryptosystems are able to withstand leaks of up to L_{MAX} bits of information per transaction, where L_{MAX} is a security factor chosen by the system designer to exceed to the maximum anticipated leak rate. The more general class of leak-resistant cryptosystems includes leak-proof cryptosystems, and others that can withstand leaks but are not necessarily defined to withstand any defined maximum information leakage rate. Therefore, any leak-proof system shall also be understood to be leak-resistant. The leak-resistant systems of the present invention can survive a variety of monitoring and eavesdropping attacks that would break traditional (non-leak-resistant) cryptosystems.

A typical leak-resistant cryptosystem of the present invention consists of three general parts. The initialization or key generation step produces secure keying material appropriate for the scheme. The update process cryptographically modifies the secret key material in a manner designed to render useless any information about the secrets that may have previously leaked from the system, thus providing security advantages over systems of the background art. The final process performs cryptographic operations, such as producing digital signatures or decrypting messages.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 shows an exemplary leak-resistant symmetric authentication method.

Figure 2 shows an exemplary leak-resistant Diffie-Hellman exponential key exchange operation.

Figure 3 shows an exemplary leak-resistant RSA private key operation.

Figure 4 shows an exemplary leak-resistant ElGamal signing operation.

DETAILED DESCRIPTION OF THE INVENTION

The sections following will describe an introduction to leak-proof/leak-resistant cryptography, followed by various embodiments of the general techniques of the invention as applied to improve the security of common cryptographic protocols.

I. Introduction and Terminology

The leakage rate L is defined as the number of bits of useful information about a cryptosystem's secrets that are revealed per operation, where an operation is a cryptographic transaction. Although an attacker may be able to collect more than L bits worth of measurement data, by definition this data yields no more than L bits of useful information about the system's secrets.

The implementer of a leak-proof system chooses a design parameter L_{MAX}, the maximum amount of leakage per operation the system may allow if it is to remain uncompromised. L_{MAX} should be chosen conservatively, and normally should significantly exceed the amount of useful information known to be leaked to attackers about the system's secrets during each transaction. Designers do not necessarily need to know accurately or completely the quantity and type of information that may leak from their systems; the choice of L_{MAX} may be made using estimates and models for the system's behavior. General factors affecting the choice of L_{MAX} include the types of monitoring potentially available to attackers, the amount of error in attackers' measurements, and engineering constraints that limit L_{MAX}. (Larger values of L_{MAX} increase memory and performance requirements of the device, and in some cases may increase L.) To estimate the amount of useful information an attacker could collect by monitoring a device's power consumption, for example, a designer might consider the amount of noise in the device's power usage, the power line capacitance, the useful time resolution for power consumption measurements, as well as the strength of the signals being monitored. Similarly, the designer knows that timing measurements can rarely yield more than a few bits of information per operation, since timing information is normally quantized to an integral number of clock cycles. In choosing L_{MAX}, the designer should assume that

attackers will be able to combine information gleaned from multiple types of attacks. If the leakage rate is too large (as in the extreme case where L equals the key size because the entire key can be extracted during a single transaction), additional design features should be added to reduce L and reduce the value needed for L_{MAX}. Such additional measures can include known methods, such as filtering the device's power inputs, adding shielding, introducing noise into the timing or power consumption, implementing constant-time and constant execution path algorithms, and changing the device layout. Again, note that the designer of a leak-resistant system does not actually need to know what information is being revealed or how it is leaked; all he or she need do is choose an upper bound for the rate at which attackers might learn information about the keys. In contrast, the designer of a traditional system faces the much harder task of ensuring that no information about the secrets will leak out.

There are many ways information about secrets can leak from cryptosystems. For example, an attacker can use a high-speed analog-to-digital converter to record a smartcard's power consumption during a cryptographic operation. The amount of useful information that can be gained from such a measurement varies, but it would be fairly typical to gain enough information to guess each of 128 key bits correctly with a probability of 0.7. This information can reduce the amount of effort required for a brute force attack. For example, a brute force attack with one message against a key containing k bits where each bit's value is known with probability p can be completed in

$$\mathrm{E}(k,p) = \sum_{i=0}^{k}\left[\binom{k}{i}(1-p)^i p^{k-i}\left[\left(\sum_{j=0}^{i}\binom{k}{j}\right)-\frac{1}{2}\binom{k}{i}\right]+\frac{1}{2}\right]$$

operations. The reduction in the effort for a brute force attack is equivalent to shortening the key by $L = \log_2(\mathrm{E}(k,½) / \mathrm{E}(k,p)) = \log_2(k - \mathrm{E}(k,p) - 1)$ bits. (For example, in the case of $k =$ 128 and $p = 0.7$, L is estimated to be about 11 bits for the first measurement. With a multiple message attack, the attacker's effort can fall to as low as $\mathrm{E}(k,p) = \frac{1}{p^k}$.) Attackers can gain additional information about the keys by measuring additional operations; unless leak-resistance is used, finding the key becomes easy after just a few dozen operations.

When choosing L_{MAX}, a system designer should consider the signal-to-noise ratio of an attacker's measurements. For example, if the signal and noise are of roughly equivalent magnitude, the designer knows that an attacker's measurements should be incorrect about 25 percent of the time (e.g., $p = 0.75$ if only one observation per key bit is possible). Many

measurement techniques, such as those involving timing, may have signal-to-noise ratios of 1:100 or worse. With such systems, L is generally quite small, but attackers who can make a large number of measurements can use averaging or other statistical techniques to recover the entire key. In extreme cases, attackers may be able to obtain all key bits with virtually perfect accuracy from a single transaction (i.e., $L = k$), necessitating the addition of shielding, noise in the power consumption (or elsewhere), and other measures to reduce p and L. Of course, L_{MAX} should be chosen conservatively; in the example above where less than 4 useful bits are obtained per operation for the given attack, the designer might select $L_{MAX} = 64$ for a leak-proof design.

Leak-proof (and, more generally, leak-resistant) cryptosystems provide system designers with important advantages. When designing a traditional (i.e., non-leak-resistant and non-leak-proof) cryptosystem, a careful cryptosystem designer should study all possible information available to attackers if he or she is to ensure that no analytical techniques could be used to compromise the keys. In practice, many insecure systems are developed and deployed because such analysis is incomplete, too difficult even to attempt, or because the cryptographers working on the system do not understand or cannot completely control the physical characteristics of the device they are designing. Unexpected manufacturing defects or process changes, alterations made to the product by attackers, or modifications made to the product in the field can also introduce problems. Even a system designed and analyzed with great care can be broken if new or improved data collection and analysis techniques are found later. In contrast, with leak-proof cryptography, the system designer only needs to define an upper bound on the maximum rate at which attackers can extract information about the keys. A detailed understanding of the information available to attackers is not required, since leak-proof (and leak-resistant) cryptosystem designs allow for secret information in the device to leak out in (virtually) any way, yet remain secure despite this because leaked information is only of momentary value.

In a typical leak-proof design, with each new cryptographic operation i, the attacker is assumed to be able to choose any function F_i and determine the L_{MAX}-bit result of computing F_i on the device's secrets, inputs, intermediates, and outputs over the course of the operation. The attacker is even allowed to choose a new function F_i with each new operation. The system may be considered leak-proof with a security factor n and leak rate L_{MAX} if, after observing a large number of operations, an attacker cannot forge signatures, decrypt data, or

perform other sensitive operations without performing an exhaustive search to find an n-bit key or performing a comparable $O(2^n)$ operation. In addition to choosing L_{MAX}, designers also choose n, and should select a value large enough to make exhaustive search infeasible. In the sections that follow, various embodiments of the invention, as applied to improve the security of common cryptographic operations and protocols, will be described in more detail.

II. Symmetric Cryptographic Protocols

A. Symmetric Authentication

An exemplary cryptographic protocol that can be secured using the techniques of the present invention is symmetric authentication.

1. Conventional Symmetric Authentication

Assume a user wishes to authenticate herself to a server using an n-bit secret key, K, known to both the server and the user's cryptographic token, but not known to attackers. The cryptographic token should be able to resist tampering to prevent, for example, attackers from being able to extract secrets from a stolen token. If the user's token has perfect tamper resistance (i.e., L=0), authentication protocols of the background art can be used. Typically the server sends a unique, unpredictable challenge value R to the user's token, which computes the value $A = H(R \| K)$, where "||" denotes concatenation and H is a one-way cryptographic hash function such as SHA. The user sends A to the server, which independently computes A (using its copy of K) and compares its result with the received value. The user authentication succeeds only if the comparison operation indicates a match.

If the function H is secure and if K is sufficiently large to prevent brute force attacks, attackers should not be able to obtain any useful information from the (R, A) values of old authentication sessions. To ensure that attackers cannot impersonate users by replaying old values of A, the server generates values of R that are effectively (with sufficiently high probability) unique. In most cases, the server should also make R unpredictable to ensure that an attacker with temporary possession of a token cannot compute future values of A. For example, R might be a 128-bit number produced using a secure random number generator (or pseudorandom number generator) in the server. The properties of cryptographic hash functions such as H have been the subject of considerable discussion in the literature, and need not be described in detail here. Hash functions typically provide functionality modeled after a random oracle, deterministically producing a particular output from any input. Ideally, such functions should be collision-resistant, non-invertable, should not leak partial

information about the input from the output, and should not leak information about the output unless the entire input is known. Hash functions can have any output size. For example, MD5 produces 128-bit outputs and SHA produces 160-bit outputs. Hash functions may be constructed from other cryptographic primitives or other hash functions.

While the cryptographic security of the protocol using technology of the background art may be good, it is not leak-proof; even a one-bit leak function (with L=1) can reveal the key. For example, if the leak function F equals bit (R mod n) of K, an attacker can break the system quickly since a new key bit is revealed with every transaction where (R mod n) has a new value. Therefore, there is a need for a leak-proof/leak-resistant symmetric authentication protocol.

2. Leak-Resistant Symmetric Authentication

The following is one embodiment of a leak-resistant (and, in fact, also leak-proof) symmetric authentication protocol, described in the context of a maximum leakage rate of L_{MAX} bits per transaction from the token and a security factor n, meaning that attacks of complexity $O(2^n)$, such as brute-force attacks against an n-bit key, are acceptable, but there should not be significantly easier attacks. The user's token maintains a counter t, which is initialized to zero, and an $(n+2L_{MAX})$-bit shared secret K_t, which is initialized with a secret K_0. Note that against adversaries performing precomputation attacks based on Hellman's time/memory trade-off, larger values of n may be in order. Note also that some useful protocol security features, such as user and/or server identifiers in the hash operation inputs, have been omitted for simplicity in the protocol description. It is also assumed that no leaking will occur from the server. For simplicity in the protocol description, some possible security features (such as user and/or server identifiers in the hash operation inputs) have been omitted, and it is assumed that the server is in a physically secure environment. However, those skilled in the art will appreciate that the invention is not limited to such assumptions, which have been made as a matter of convenience rather than necessity.

As in the traditional protocol, the server begins the authentication process by generating a unique and unpredictable value R at step 105. For example, R might be a 128-bit output from a secure random number generator. At step 110, the server sends R to the user's token. At step 112, the token receives R. At step 115, the token increments its counter t by computing $t \leftarrow t + 1$. At step 120, the token updates K_t by computing $K_t \leftarrow H_K(t \| K_t)$, where H_K is a cryptographic hash function that produces an $(n+2L_{MAX})$ bit output from the old value

of K_t and the (newly incremented) value of t. Note that in the replacement operations (denoted "←"), the token deletes the old values of t and K_t, replacing them with the new values. By deleting the old K_t, the token ensures that future leak functions cannot reveal information about the old (deleted) value. At step 122, the token uses the new values of t and K_t to compute an authenticator $A = H_A(K_t \| t \| R)$. At step 125, the token sends both t and the authenticator A to the server, which receives them at step 130. At step 135, the server verifies that t is acceptable (e.g., not too large but larger than the value received in the last successful authentication). If t is invalid, the server proceeds to step 175. Otherwise, at step 140, the server initializes its loop counter i to zero and its key register K_t' to K_0. At step 145, the server compares i with the received value of t, proceeding to step 160 if they are equal. Otherwise, at step 150, the server increments i by computing $i \leftarrow i + 1$. At step 155, the server computes $K_t' \leftarrow H_K(i \| K_t')$, then proceeds back to step 145. At step 160, the server computes $A' = H_A(K_t' \| t \| R)$. Finally, at step 165, the server compares A and A', where the authentication succeeds at step 170 if they match, or fails at 175 if they do not match.

This design assumes that at the beginning of any transaction the attacker may have L_{MAX} bits of useful information about the state of the token (e.g., K_t) that were obtained using the leak function F in a previous operation. During the transaction, the attacker can gain an additional L_{MAX} bits of useful information from the token. If, at any time, any $2L_{MAX}$ (or fewer) bits of useful information about the secret are known to the attacker, there are still $(n+2L_{MAX})-2L_{MAX} = n$ or more unknown bits. These n bits of unknown information ensure that attacks will require $O(2^n)$ effort, corresponding to the desired security factor. However, the attacker should have no more than L_{MAX} bits of useful information about K_t at the end of the transaction. The property that attackers lose useful information during normal operation of the system is a characteristic of the leak-proof or leak-resistant cryptosystem. In general, this information loss is achieved when the cryptosystem performs operations that convert attackers' useful partial information about the secret into useless information. (Information is considered useless if it gives an attacker nothing better than the ability to test candidate values in an $O(2^n)$ exhaustive search or other "hard" operation. For example, if exhaustive search of X is hard and H is a good hash function, $H(X)$ is useless information to an attacker trying to find X.)

Thus, the attacker is assumed to begin with L_{MAX} bits of useful information about K_t before the token's $K_t \leftarrow H_K(t \| K_t)$ computation. (Initial information about anything other

than K_t is of no value to an attacker because K_t is the only secret value in the token. The function H_K and the value of t are not assumed to be secret.) The attacker's information can be any function of K_t produced from the previous operation's leaks.

3. Security Characteristics of Leak-Proof Systems

The following section provides a technical discussion of the security characteristics of the exemplary leak-proof system described above. The following analysis is provided as an example of how the design can be analyzed, and how a system may be designed using general assumptions about attackers' capabilities. The discussion and assumptions do not necessarily apply to other embodiments of the invention and should not be construed as limiting the scope or applicability of the invention in any way.

During the course of a transaction, the leak function F might reveal up to L_{MAX} information about the system and its secrets. The design assumes that any information contained in the system may be leaked by F, provided that F does not reveal useful new information about values of K_t that were deleted before the operation started, and F does not reveal useful information about values of K_t that will be computed in future operations. These constraints are completely reasonable, since real-world leaks would not reveal information about deleted or not-yet-existent data. (The only way information about future K_t values could be leaked would be the bizarre case where the leak function itself included, or was somehow derived from, the function H_K.) In practice, these constraints on F are academic and of little concern, but they are relevant when constructing proofs to demonstrate the security of a leak-proof system.

If the leak occurs at the beginning of the H_K computation, it could give the attacker up to $2L_{MAX}$ bits of useful information about the input value of K_t. Because K_t contains $(2L_{MAX}+n)$ bits of secret information and the attacker may have up to $2L_{MAX}$ bits of useful information about the initial value of K_t, there remain at least $(2L_{MAX}+n)-2L_{MAX} = n$ bits of information in K_t that are secret. The hash function H_K effectively mixes up these n bits to produce a secure new K_t during each transaction such that the attacker's information about the old K_t is no longer useful.

If the leak occurs at the end of the H_K computation, it could give an attacker up to L_{MAX} bits of information about the final value of H_K, yielding L_{MAX} bits of information about

the input to the subsequent transaction. This is not a problem, since the design assumes that attackers have up to L_{MAX} bits of information about K_t at the beginning of each transaction.

A third possibility is that the attacker's L_{MAX} bits of information might describe intermediates computed during the operation H_K. However, even if the attacker could obtain L_{MAX} new bits of information about the input to H_K and also L_{MAX} bits of information about the output from H_K, the system would be secure, since the attacker would never have more than $2L_{MAX}$ bits of information about the input K_t or more than L_{MAX} bits of information about the output K_t. Provided that L_{MAX} bits of information from within H_K cannot reveal more than L_{MAX} bits of information about the input, or more than L_{MAX} bits of information about the output, the system will be secure. This will be true unless H_K somehow compresses the input to form a short intermediate which is expanded to form the output. While hash functions whose internal states are smaller than their outputs should not be used, most cryptographic hash functions are fine.

A fourth possibility is that part or all of the leak could occur during the $A = H_A(K_t \| t \| R)$ calculation. The attacker's total "budget" for observations is L_{MAX} bits. If L_1 bits of leak occur during the H_K computation, an additional L_2 bits of information can leak during the $A = H_A(K_t \| t \| R)$ operation, where $L_2 \leq L_{MAX} - L_1$. If the second leak provides information about K_t, this is no different from leaking information about the result of the H_K computation; the attacker will still conclude the transaction with no more than L_{MAX} bits of information about K_t because $L_1 + L_2 \leq L_{MAX}$. However, the second leak could reveal information about A. To keep A secure against leaks (to prevent, for example, an attacker from using a leak to capture A and using A before the legitimate user can), the size of A should include an extra L_{MAX} bits (to provide security even if $L_2 = L_{MAX}$). Like H_K, H_A should not leak information about deleted or future values of K_t that are not used in or produced by the given operation. As with the similar assumptions on leaks from H_K, this limitation is primarily academic and of little practical concern, since real-world leak functions do not reveal information about deleted or not-yet-computed data. However, designers might be cautious when using unusual designs for H_A that are based on or derived from H_K, particularly if the operation $H_A(K_t \| t \| R)$ could reveal useful information about the result of computing $H_K(t \| K_t)$.

B. Other Leak-Resistant Symmetric Schemes

The same basic technique of updating a key (K) with each transaction, such that leakage about a key during one transaction does not reveal useful information about a key in a

subsequent (or past) transaction, can be easily extended to other applications besides authentication.

1. Symmetric Data Verification

For example and without limitation, leak-resistant symmetric data verification is often useful where a device needs to support symmetrically-signed code, data, content, or parameter updates (all of which will, as a matter of convenience, be denoted as "data" herein). In existing systems, a hash or MAC of the data is typically computed using a secret key and the data is rejected if computed hash or MAC does not match a value received with the data. For example, a MAC may be computed as HMAC(K, data), where HMAC is defined in "RFC 2104, HMAC: Keyed-Hashing for Message Authentication" by H. Krawczyk, M. Bellare, and R. Canetti, 1997. Traditional (non-leak-resistant) designs are often vulnerable to attacks including power consumption analysis of MAC functions and timing analysis of comparison operations.

In an exemplary leak-resistant verification protocol, a verifying device (the "verifier") maintains a counter t and a key K_t, which are initialized (for example at the factory) with $t \leftarrow 0$ and $K_t \leftarrow K_0$. Before the transaction, the verifier provides t to the device providing the signed data (the "signer"), which also knows K_0. The signer uses t to compute K_{t+1}' (the prime indicating a quantity derived by the signer, rather than at the verifier) from K_0 (or K_t' or any other available value of K_i'). using the relation $K_i' = H_K(i \| K_{i-1}')$, computes signature $S' = \text{HMAC}(K_{t+1}', \text{data})$, and sends S' plus any other needed information (such as data or t) to the verifier. The verifier confirms that the received value of t (if any) matches its value of t, and rejects the signature if it does not. If t matches, the verifier increments t and updates K_t in its nonvolatile memory by computing $t \leftarrow t + 1$ and $K_t \leftarrow H_K(t \| K_t)$. In an alternative embodiment, if the received value of t is larger than the internal value but the difference is not unreasonably large, it may be more appropriate to accept the signature and perform multiple updates to K_t (to catch up with the signer) instead of rejecting the signature outright. Finally, the verifier computes $S = \text{HMAC}(K_t, \text{data})$ and verifies that $S = S'$, rejecting the signature if S does not equal the value of S' received with the data.

2. Symmetric Encryption

Besides authentication and verification, leak-resistant symmetric cryptography can also be tailored to a wide variety of applications and environments. For example, if data

encryption is desired instead of authentication, the same techniques as were disclosed above may be used to generate a key K_t used for encryption rather than verification.

3. Variations in Computational Implementation

In the foregoing, various applications were disclosed for the basic technique of updating a key K_t in accordance with a counter and deleting old key values to ensure that future leakage cannot reveal information about the now-deleted key. Those skilled in the art will realize, however, that the exemplary techniques described above may be modified in various ways without departing from the spirit and scope of the invention. For example, if communications between the device and the server are unreliable (for example if the server uses voice recognition or manual input to receive t and A), then small errors in the signature may be ignored. (One skilled in the art will appreciate that many functions may be used to determine whether a signature corresponds – sufficiently closely -- to its expected value.) In another variation of the basic technique, the order of operations and of data values may be adjusted, or additional steps and parameters may be added, without significantly changing the invention. In another variation, to save on communication bandwidth or memory, the high order bits or digits of t may not need to be communicated or remembered. In another variation, as a performance optimization, devices need not recompute K_t from K_0 with each new transaction. For example, when a transaction succeeds, the server can discard K_0 and maintain the validated version of K_t. In another variation, if bi-directional authentication is required, the protocol can include a step whereby the server can authenticates itself to the user (or user's token) after the user's authentication is complete. In another variation, if the server needs to be secured against leaks as well (as in the case where the role of "server" is played by an ordinary user), it can maintain its own counter t. In each transaction, the parties agree to use the larger of their two t values, where the device with the smaller t value performs extra updates to K_t to synchronize t. In an alternate embodiment for devices that contain a clock and a reliable power source (e.g., battery), the update operation may be performed periodically, for example by computing $K_t \leftarrow H_K(t \| K_t)$ once per second. The token uses the current K_t to compute $A = H_A(K_t \| t \| R)$ or, if the token does not have any means for receiving R, it can output $A = H_A(K_t)$. The server can use its clock and local copy of the secret to maintain its own version of K_t, which it can use to determine whether received values of A are recent and correct. All of the foregoing show that the method and apparatus of the present invention can be implemented using numerous variations and modifications to

the exemplary embodiments described herein, as would be understood by one skilled in the art.

III. Asymmetric Cryptographic Protocols

The foregoing illustrates various embodiments of the invention that may be used with symmetric cryptographic protocols. As will be seen below, still other techniques of the present invention may be used in connection with asymmetric cryptographic operations and protocols. While symmetric cryptosystems are sufficient for some applications, asymmetric cryptography is required for many applications. There are several ways leak resistance can be incorporated into public key cryptosystems, but it is often preferable to have as little impact as possible on the overall system architecture. Most of the exemplary designs have thus been chosen to incorporate leak resistance into widely used cryptosystems in a way that only alters the key management device, and does not affect the certification process, certificate format, public key format, or processes for using the public key.

A. Certified Diffie-Hellman

Diffie-Hellman exponential key exchange is a widely used asymmetric protocol whereby two parties who do not share a secret key can negotiate a shared secret key. Implementations of Diffie-Hellman can leak information about the secret exponents, enabling attackers to determine the secret keys produced by those implementations. Consequently, a leak-resistant implementation of Diffie-Hellman would be useful. To understand such a leak-resistant implementation, it will be useful to first review a conventional Diffie-Hellman implementation.

1. Conventional Certified Diffie-Hellman

Typical protocols in the background art for performing certified Diffie-Hellman exponential key agreement involve two communicating users (or devices) and a certifying authority (CA). The CA uses an asymmetric signature algorithm (such as DSA) to sign certificates that specify a user's public Diffie-Hellman parameters (the prime p and generator g), public key (p^x mod g, where x is the user's secret exponent), and auxiliary information (such as the user's identity, a description of privileges granted to the certificate holder, a serial number, expiration date, etc.). Certificates may be verified by anyone with the CA's public signature verification key. To obtain a certificate, user U typically generates a secret exponent (x_U), computes his or her own public key $y_u = g^{x_u} \bmod p$, presents y_U along with any required auxiliary identifying or authenticating information (e.g., a passport) to the CA,

who issues the user a certificate C_U. Depending on the system, p and g may be unique for each user, or they may be system-wide constants (as will be assumed in the following description of Diffie-Hellman using the background art).

Using techniques of the background art, Alice and Bob can use their certificates to establish a secure communication channel. They first exchange certificates (C_{Alice} and C_{Bob}). Each verifies that the other's certificate is acceptable (e.g., properly formatted, properly signed by a trusted CA, not expired, not revoked, etc.). Because this protocol will assume that p and g are constants, they also check that the certificate's p and g match the expected values. Alice extracts Bob's public key (y_{Bob}) from C_{Bob} and uses her secret exponent (x_{Alice}) to compute $z_{Alice} = (y_{Bob})^{x_{Alice}} \bmod p$. Bob uses his secret exponent and Alice's public key to compute $z_{Bob} = (y_{Alice})^{x_{Bob}} \bmod p$. If everything works correctly, $z_{Alice} = z_{Bob}$, since:

$$\begin{aligned} z_{Alice} &= (y_{Bob})^{x_{Alice}} \bmod p \\ &= (g^{x_{Bob}})^{x_{Alice}} \bmod p \\ &= (g^{x_{Alice}})^{x_{Bob}} \bmod p \\ &= (y_{Alice})^{x_{Bob}} \bmod p \\ &= z_{Bob}. \end{aligned}$$

Thus, Alice and Bob have a shared key $z = z_{Alice} = z_{Bob}$. An attacker who pretends to be Alice but does not know her secret exponent (x_{Alice}) will not be able to compute $z_{Alice} = (y_{Bob})^{x_{Alice}} \bmod p$ correctly. Alice and Bob can positively identify themselves by showing that they correctly found z. For example, each can compute and send the other the hash of z concatenated with their own certificate. Once Alice and Bob have verified each other, they can use a symmetric key derived from z to secure their communications. (For an example of a protocol in the background art that uses authenticated Diffie-Hellman, see "The SSL Protocol Version 3.0" by A. Freier, P. Karlton, and P. Kocher, March 1996.)

2. Leak-Resistant Certified Diffie-Hellman

A satisfactory leak-resistant public key cryptographic scheme should overcome the problem that, while certification requires the public key be constant, information about the corresponding private key should not leak out of the token that contains it. In the symmetric protocol described above, the design assumes that the leak function reveals no useful information about old deleted values of K_t or about future values of K_t that have not yet been

computed. Existing public key schemes, however, require that implementations repeatedly perform a consistent, usually deterministic, operation using the private key. For example, in the case of Diffie-Hellman, a leak-resistant token that is compatible with existing protocols and implementations should be able to perform the secret key operation $y^x \bmod p$, while ensuring that the exponent x remains secret. The radical reshuffling of the secret provided by the hash function H_K in the symmetric approach cannot be used because the device should be able to perform the same operation consistently.

The operations used by the token to perform the private key operation are modified to add leak resistance using the following variables:

Register	Comment
x_1	First part of the secret key (in nonvolatile updateable memory)
x_2	Second part of the secret key (in nonvolatile updateable memory)
g	The generator (not secret).
p	The public prime, preferably a strong prime (not secret).

The prime p and generator g may be global parameters, or may be specific to individual users or groups of users (or tokens). In either case, the certificate recipient should be able to obtain p and g securely, usually as built-in constants or by extracting them from the certificate.

To generate a new secret key, the key generation device (often but not always the cryptographic token that will contain the key) first obtains or generates p and g, where p is the prime and g is a generator mod p. If p and g are not system-wide parameters, algorithms known in the background art for selecting large prime numbers and generators may be used. It is recommended that p be chosen with $\frac{p-1}{2}$ also prime, or at least that $\phi(p)$ not be smooth. (When $\frac{p-1}{2}$ is not prime, information about x_1 and x_2 modulo small factors of $\phi(p)$ may be leaked, which is why it is preferable that $\phi(p)$ not be smooth. Note that ϕ denotes Euler's totient function.) Once p and g have been chosen, the device generates two random exponents x_1 and x_2. The lowest-order bit of x_1 and of x_2 is not considered secret, and may be set to 1. Using p, g, x_1, and x_2, the device can then compute its public key as $g^{x_1 x_2} \bmod p$ and submit it, along with any required identifying information or parameters needed (e.g., p and g), to the CA for certification.

Figure 2 illustrates the process followed by the token to perform private key operations. At step 205, the token obtains the input message y, its own (non-secret) prime p, and its own secret key halves (x_1 and x_2). If x_1, x_2, and p are stored in encrypted and/or

authenticated form, they would be decrypted or verified at this point. At this step, the token should verify that $1 < y < p-1$. At step 210, the token uses a random number generator (or pseudorandom number generator) to select a random integer b_0, where $0 < b_0 < p$. At step 215, the token computes $b_1 = b_0^{-1} \bmod p$. The inverse computation mod p may be performed using the extended Euclidean algorithm or the formula $b_1 = b_0^{\phi(p)-1} \bmod p$. At step 220, the token computes $b_2 = b_1^{x_1} \bmod p$. At this point, b_1 is no longer needed; its storage space may be used to store b_2. Efficient algorithms for computing modular exponentiation, widely known in the art, may be used to complete step 220. Alternatively, when a fast modular exponentiator is available, the computation b_2 may be performed using the relationship $b_2 = b_0^{\phi(p)-x_1} \bmod p$. At step 225, the token computes $b_3 = b_2^{x_2} \bmod p$. At this point, b_2 is no longer needed; its storage space may be used to store b_3. At step 230, the token computes $z_0 = b_0 y \bmod p$. At this point, y and b_0 are no longer needed; their space may be used to store r_1 (computed at step 235) and z_0. At step 235, the token uses a random number generator to select a random integer r_1, where $0 < r_1 < \phi(p)$ and $\gcd(r_1, \phi(p)) = 1$. (If $\frac{p-1}{2}$ is known to be prime, it is sufficient to verify that r_1 is odd.) At step 240, the token updates x_1 by computing $x_1 \leftarrow x_1 r_1 \bmod \phi(p)$. The old value of x_1 is deleted and replaced with the updated value. At step 245, the token computes $r_2 = (r_1^{-1}) \bmod \phi(p)$. If $\frac{p-1}{2}$ is prime, then r_2 can be found using a modular exponentiator and the Chinese Remainder Theorem. Note that r_1 is not needed after this step, so its space may be used to store r_2. At step 250, the token updates x_2 by computing $x_2 \leftarrow x_2 r_2 \bmod \phi(p)$. The old value of x_2 should be deleted and replaced with the updated value. At step 255, the token computes $z_1 = (z_0)^{r_1} \bmod p$. Note that z_0 is not needed after this step, so its space may be used to store z_1. At step 260, the token computes $z_2 = (z_1)^{r_2} \bmod p$. Note that z_1 is not needed after this step, so its space may be used to store z_2. At step 265, the token finds the exponential key exchange result by computing $z = z_2 b_3 \bmod p$. Finally, at step 270, the token erases and frees any remaining temporary variables.

The process shown in Figure 2 correctly computes $z = y^x \bmod p$, where $x = x_1 x_2 \bmod \phi(p)$, since:

$$
\begin{aligned}
z &= z_2 b_3 \bmod p \\
&= \left(z_1^{x_2} \bmod p\right)\left(b_2^{x_2} \bmod p\right) \bmod p \\
&= \left(\left(z_0^{x_1} \bmod p\right)^{x_2}\right)\left(\left(b_1^{x_1} \bmod p\right)^{x_2}\right) \bmod p \\
&= \left(b_0 y \bmod p\right)^{x_1 x_2}\left(b_0^{-1} \bmod p\right)^{x_1 x_2} \bmod p \\
&= y^{x_1 x_2} \bmod p \\
&= y^{x} \bmod p.
\end{aligned}
$$

The invention is useful for private key owners communicating with other users (or devices) who have certificates, and also when communicating with users who do not.

If Alice has a certificate and wishes to communicate with Bob who does not have a certificate, the protocol proceeds as follows. Alice sends her certificate (C_{Alice}) to Bob, who receives it and verifies that it is acceptable. Bob extracts y_{Alice} (along with p_{Alice} and g_{Alice}, unless they are system-wide parameters) from C_{Alice}. Next, Bob generates a random exponent x_{BA}, where $0 < x_{\text{AB}} < \phi(p_{\text{Alice}})$. Bob then uses his exponent x_{AB} and Alice's parameters to calculate $y_{\text{BA}} = \left(g_{\text{Alice}}^{x_{\text{BA}}}\right) \bmod p_{\text{Alice}}$ and the session key $z = \left(y_{\text{Alice}}^{x_{\text{BA}}}\right) \bmod p_{\text{Alice}}$. Bob sends y_{BA} to Alice, who performs the operation illustrated in Figure 2 to update her internal parameters and derive z from y_{BA}. Alice then proves that she computed z correctly, for example by sending Bob $H(z \parallel C_{\text{Alice}})$. (Alice cannot authenticate Bob because he does not have a certificate. Consequently, she does not necessarily need to verify that he computed z successfully.) Finally, Alice and Bob can use z (or, more commonly, a key derived from z) to secure their communications.

If both Alice and Bob have certificates, the protocol works as follows. First, Alice and Bob exchange certificates (C_{Alice} and C_{Bob}), and each verifies that other's certificate is valid. Alice then extracts the parameters p_{Bob}, g_{Bob}, and y_{Bob} from C_{Bob}, and Bob extracts p_{Alice}, g_{Alice}, and y_{Alice} from C_{Alice}. Alice then generates a random exponent x_{AB} where $0 < x_{\text{AB}} < \phi(p_{\text{Bob}})$, computes $y_{\text{AB}} = \left(g_{\text{Bob}}\right)^{x_{\text{AB}}} \bmod p_{\text{Bob}}$, and computes $z_{\text{AB}} - \left(y_{\text{Bob}}\right)^{x_{\text{AB}}} \bmod p_{\text{Bob}}$. Bob generates a random x_{BA} where $0 < x_{\text{BA}} < \phi(p_{\text{Alice}})$, computes $y_{\text{BA}} = \left(g_{\text{Alice}}\right)^{x_{\text{BA}}} \bmod p_{\text{Alice}}$, and computes $z_{\text{BA}} - \left(y_{\text{Alice}}\right)^{x_{\text{BA}}} \bmod p_{\text{Alice}}$. Bob sends y_{BA} to Alice, and Alice sends y_{AB} to Bob. Alice and Bob each perform the operation shown in Figure 2, where each uses the prime p from their own certificate and their own secret exponent halves (x_1 and x_2). For the message y in Figure 2, Alice uses y_{BA} (received from Bob), and Bob uses y_{AB} (received from Alice). Using the process shown in Figure 2, Alice computes z. Using z and z_{AB} (computed

previously), she can find a session key K. This may be done, for example, by using a hash function H to compute $K = H(z \| z_{AB})$. The value of z Bob obtains using the process shown in Figure 2 should equal Alice's z_{AB}, and Bob's z_{BA} (computed previously) should equal Alice's z. If there were no errors or attacks, Bob should thus be able to find K, e.g., by computing $K = H(z_{BA} \| z)$. Alice and Bob now share K. Alice can prove her identity by showing that she computed K correctly, for example by sending Bob $H(K \| C_{Alice})$. Bob can prove his identity by sending Alice $H(K \| C_{Bob})$. Alice and Bob can then secure their communications by encrypting and authenticating using K or a key derived from K.

Note that this protocol, like the others, is provided as an example only; many variations and enhancements of the present invention are possible and will be evident to one skilled in the art. For example, certificates may come from a directory, more than two parties can participate in the key agreement, key escrow functionality may be added, the prime modulus p may be replaced with a composite number, etc. Note also that Alice and Bob as they are called in the protocol are not necessarily people; they would normally be computers, cryptographic devices, etc.

For leak resistance to be effective, attackers should not be able to gain new useful information about the secret variables with each additional operation unless a comparable amount of old useful information is made useless. While the symmetric design is based on the assumption that leaked information will not survive the hash operation H_K, this design uses multiplication operations mod $\phi(p)$ to update x_1 and x_2. The most common variety of leaked information, statistical information about exponent bits, is not of use to attackers in this design, as the exponent update process ($x_1 \leftarrow x_1 r_1 \bmod \phi(p)$ and $x_2 \leftarrow x_2 r_2 \bmod \phi(p)$) destroys the utility of this information. The only relevant characteristic that survives the update process is that $x_1 x_2 \bmod \phi(p)$ remains constant, so the system designer should be careful to ensure that the leak function does not reveal information allowing the attacker to find new useful information about $x_1 x_2 \bmod \phi(p)$.

There is a modest performance penalty, approximately a factor of four, for the leak-resistant design as described. One way to improve performance is to remove the blinding and unblinding operations, which are often unnecessary. (The blinding operations prevent attackers from correlating input values of y with the numbers processed by the modular exponentiation operation.) Alternatively or additionally, it is possible to update and reuse

values of b_0, b_3, r_1, and r_2 by computing $b_0 \leftarrow (b_0)^v \bmod p$, $b_3 \leftarrow (b_3)^v \bmod p$, $r_1 \leftarrow (r_1)^w \bmod \phi(p)$, and $r_2 \leftarrow (r_2)^w \bmod \phi(p)$, where v and w are fairly short random exponents. Note that the relationship $b_3 \leftarrow b_0^{-x_1 x_2} \bmod p$ remains true when b_0 and b_3 are both raised to the power v (mod p). The relationship $r_2 = (r_1^{-1}) \bmod \phi(p)$ also remains true when r_1 and r_2 are exponentiated (mod $\phi(p)$). Other parameter update operations may also be used, such as exponentiation with fixed exponents (e.g., $v = w = 3$), or multiplication with random values and their inverses, mod p and $\phi(p)$. The time per transaction with this update process is about half that of the unoptimized leak-resistant implementation, but additional storage is required and care should be taken to ensure that b_0, b_3, r_1, and r_2 will not be leaked or otherwise compromised.

It should also be noted that with this particular type of certified Diffie-Hellman, the negotiated key is the same every time any given pair of users communicate. Consequently, though the blinding operation performed using b_0 and b_3 does serve to protect the exponents, the result K can be leaked in the final step or by the system after the process is complete. If storage is available, parties could keep track of the values of y they have received (or their hashes) and reject duplicates. Alternatively, to ensure that a different result is obtained from each negotiation, Alice and Bob can generate and exchange additional exponents, w_{Alice} and w_{Bob}, for example with $0 < w < 2^{128}$ (where $2^{128} << p$). Alice sets $y = (y_{\text{BA}})^{w_{\text{Alice}} w_{\text{Bob}}} \bmod p$ instead of just $y = y_{\text{BA}}$, and Bob sets $y = (y_{\text{AB}})^{w_{\text{Bob}} w_{\text{Alice}}} \bmod p$ instead of $y = y_{\text{AB}}$ before performing the operation shown in Figure 2.

B. Leak-Resistant RSA

Another asymmetric cryptographic protocol is RSA, which is widely used for digital signatures and public key encryption. RSA private key operations rely on secret exponents. If information about these secret exponents leaks from an implementation, its security can be compromised. Consequently, a leak-resistant implementation of RSA would be useful.

To give RSA private key operations resistance to leaks, it is possible to divide the secret exponent into two halves such that information about either half is destroyed with each operation. These are two kinds of RSA private key operations. The first, private key signing, involves signing a message with one's own private key to produce a digital signature verifiable by anyone with one's corresponding public key. RSA signing operations involve computing $S = M^d \bmod n$, where M is the message, S is the signature (verifiable using $M = S^e$

mod n), d is the secret exponent and equals e^{-1} mod $\phi(n)$, and n is the modulus and equals pq, where n and e are public and p and q are secret primes, and ϕ is Euler's phi function. An RSA public key consists of e and n, while an RSA private key consists of d and n (or other representations of them). For RSA to be secure, d, $\phi(n)$, p, and q should all be secret.

The other RSA operation is decryption, which is used to recover messages encrypted using one's public key. RSA decryption is virtually identical to signing, since the decrypted message M is recovered from the ciphertext C by computing $M = C^d$ mod n, where the ciphertext C was produced by computing $C=M^e$ mod n. Although the following discussion uses variable names from the RSA signing operation, the same techniques may be applied similarly to decryption.

An exemplary leak-resistant scheme for RSA implementations may be constructed as illustrated in Figure 3. At step 300, prior to the commencement of any signing or decryption operations, the device is initialized with (or creates) the public and private keys. The device contains the public modulus n and the secret key components d_1, d_2, and z, and k, where k is a prime number of medium-size (e.g., $0 < k < 2^{128}$) chosen at random, $z = k\phi(n)$, d_1 is a random number such that $0 < d_1 < z$ and $\gcd(d_1, z) = 1$, and $d_2 = (e^{-1} \bmod \phi(n))(d_1^{-1} \bmod z) \bmod z$. In this invention, d_1 and d_2 replace the usual RSA secret exponent d. Techniques for generating the initial RSA primes (e.g., p and q) and modulus (n) are well known in the background art. At step 305, the device computes a random prime k' of medium size (e.g., $0 < k' < 2^{128}$). (Algorithms for efficiently generating prime numbers are known in the art.)

At step 303, the device (token) receives a message M to sign (or to decrypt). At step 310, the device updates z by computing $z \leftarrow k'z$. At step 315, the device updates z again by computing $z \leftarrow z / k$. (There should be no remainder from this operation, since k divides z.) At step 320, k is replaced with k' by performing $k \leftarrow k'$. Because k' will not be used in subsequent operations, its storage space may be used to hold R (produced at step 325). At step 325, the device selects a random R where $0 < R < z$ and $\gcd(R, z) = 1$. At step 330, the device updates d_1 by computing $d_1 \leftarrow d_1 R \bmod z$. At step 335, the device finds the inverse of R by computing $R' \leftarrow R^{-1} \bmod z$ using, for example, the extended Euclidean algorithm. Note that R is no longer needed after this step, so its storage space may be erased and used to hold R'. At step 340, the device updates d_2 by computing $d_2 \leftarrow d_2 R' \bmod z$. At step 345, the device computes $S_0 = M^{d_1} \bmod n$, where M is the input message to be signed (or the message

to be decrypted). Note that M is no longer needed after this step, so its storage space may be used for S_0. At step 350, the device computes $S = S_0^{d_2} \bmod n$, yielding the final signature (or plaintext if decrypting a message). Leak-resistant RSA has similar security characteristics as normal RSA; standard message padding, post-processing, and key sizes may be used. Public key operations are also performed normally (e.g., $M = S^e \bmod n$).

A simpler RSA leak resistance scheme may be implemented by splitting the exponent d into two halves d_1 and d_2 such that $d_1 + d_2 = d$. This can be achieved during key generation by choosing d_1 to be a random integer where $0 \le d_1 \le d$, and choosing $d_2 \leftarrow d - d_1$. To perform private key operations, the device needs d_1 and d_2, but it does not need to contain d. Prior to each private key operation, the cryptographic device identifies which of d_1 and d_2 is larger. If $d_1 > d_2$, then the device computes a random integer r where $0 \le r \le d_1$, adds r to d_2 (i.e., $d_2 \leftarrow d_2 + r$), and subtracts r from d_1 (i.e., $d_1 \leftarrow d_1 - r$). Otherwise, if $d_1 \le d_2$, then the device chooses a random integer r where $0 \le r \le d_2$, adds r to d_1 (i.e., $d_1 \leftarrow d_1 + r$), and subtracts r from d_2 (i.e., $d_2 \leftarrow d_2 - r$). Then, to perform the private key operation on a message M, the device computes $s_1 = M^{d_1} \bmod n$, $s_2 = M^{d_2} \bmod n$, and computes the signature $S = s_1 s_2 \bmod n$. While this approach of splitting the exponent into two halves whose sum equals the exponent can also be used with Diffie-Hellman and other cryptosystems, dividing the exponent into the product of two numbers mod ϕ(p) is usually preferable since the assumption that information about d_1+d_2 will not leak is less conservative than the assumption that information about $x_1 x_2 \bmod \phi(p)$ will not leak. In the case of RSA, updates mod $\phi(n)$ cannot be done safely, since $\phi(n)$ must be kept secret.

When the Chinese Remainder Theorem is required for performance, it is possible to use similar techniques to add leak resistance by maintaining multiples of the secret primes (p and q) that are updated every time (e.g., multiplying by the new multiple then dividing by the old multiple). These techniques also protect the exponents (d_p and d_q) as multiples of their normal values. At the end of the operation, the result S is corrected to compensate for the adjustments to d_p, d_q, p, and q.

An exemplary embodiment maintains state information consisting of the values n, B_i, B_p, k, p_k, q_k, d_{pk}, d_{qk}, p_{Inv}, and f. To convert a traditional RSA CRT private key (consisting of p, q, d_p, and d_q with $p < q$) into the new representation, a random value for k is chosen, where $0 < k < 2^{64}$. The value B_i is chosen at random where $0 < B_i < n$, and R_1 and R_2 are chosen at

random where $0 < R_1 < 2^{64}$ and $0 < R_2 < 2^{64}$. (Of course, constants such as 2^{64} are chosen as example values. It is possible, but not necessary, to place constraints on random numbers, such as requiring that they be prime.) The leak-resistant private key state is then initialized by setting $n \leftarrow pq$, $B_f \leftarrow B_i^{-d} \bmod n$, $p_k \leftarrow (k)(p)$, $q_k \leftarrow (k)(q)$, $d_{pk} \leftarrow d_p + (R_1)(p) - R_1$, $d_{qk} \leftarrow d_q + (R_2)(q) - R_2$, $p_{Inv} \leftarrow k(p^{-1} \bmod q)$, and $f \leftarrow 0$.

To update the system state, first a random value α may be produced where $0 < \alpha < 2^{64}$. Then compute $p_k \leftarrow ((\alpha)(p_k)) / k$, $q_k \leftarrow ((\alpha)(q_k)) / k$, $p_{Inv} \leftarrow ((\alpha)(p_{Inv})) / k$, $k \leftarrow \alpha$. The exponents d_{pk} and d_{qk} may be updated by computing $d_{pk} \leftarrow d_{pk} \pm (R_3 p_k - R_3 k)$ and $d_{qk} \leftarrow d_{qk} \pm (R_4 q_k - R_4 k)$, where R_3 and R_4 can be random or constant values (even 1). The blinding factors B_i and B_f may be updated by computing $B_i = B_i^2 \bmod n$ and $B_f = B_f^2 \bmod n$, by computing new blinding factors, by exponentiating with a value other than 2, etc. Update processes should be performed as often as practical, for example before or after each modular exponentiation process. Before the update begins, a failure counter f is incremented, and when the update completes f is set to zero. If f ever exceeds a threshold value indicating too many consecutive failures, the device should temporarily or permanently disable itself. Note that if the update process is interrupted, memory values should not be left in intermediate states. This can be done by using complete reliable memory updates. If the total set of variable changes is too large for a single complete update, it is possible to store α first then do each variable update reliably which keeping track of how many have been completed.

To perform a private key operation (such as decryption or signing), the input message C is received by the modular exponentiator. Next, the value is blinded by computing $C' \leftarrow (C)(B_i) \bmod n$. The blinded input message is then used to compute modified CRT intermediates by computing $m_{pk} \leftarrow (C')^{d_{pk}} \bmod p_k$ and $m_{qk} \leftarrow (C')^{d_{qk}} \bmod q_k$. Next in the exemplary embodiment, the CRT intermediates are multiplied by k, e.g. $m_{pk} \leftarrow (k)(m_{pk}) \bmod p_k$ and $m_{qk} \leftarrow (k)(m_{qk}) \bmod q_k$. The CRT difference is then computed as $m_{pqk} = (m_{pk} [+ q_k] - m_{qk}) [\bmod q_k]$, where the addition of q_k and/or reduction mod q_k are optional. (The addition of q_k ensures that the result is non-negative.) The blinded result can be computed as

$$M' = \frac{(m_{pk})k + p_k\left[\left(\frac{(p_{Inv})(m_{pqk})}{k}\right) \bmod q_k\right]}{k^2}$$

, then the final result M is computed as $M = (M')B_f \bmod n$.

As one of ordinary skill in the art will appreciate, variant forms of the invention are possible. For example, the computational processes can be re-ordered or modified without significantly changing the invention. Some portions (such as the initial and blinding steps) can be skipped. In another example, it is also possible to use multiple blinding factors (for example, instead of or in addition to the value k).

In some cases, other techniques may also be appropriate. For example, exponent vector codings may be rechosen frequently using, for example, a random number generator. Also, Montgomery arithmetic may be performed mod j where j is a value that is changed with each operation (as opposed to traditional Montgomery implementations where j is constant with $j = 2^k$). The foregoing shows that the method and apparatus of the present invention can be implemented using numerous variations and modifications to the exemplary embodiments described herein, as would be known by one skilled in the art.

C. Leak-Resistant ElGamal Public Key Encryption and Digital Signatures

Still other asymmetric cryptographic protocols that may be improved using the techniques of the invention. For example, ElGamal and related cryptosystems are widely used for digital signatures and public key encryption. If information about the secret exponents and parameters leaks from an ElGamal implementation, security can be compromised. Consequently, leak-resistant implementations of ElGamal would be useful.

The private key in the ElGamal public key encryption scheme is a randomly selected secret a where $1 \le a \le p-2$. The non-secret parameters are a prime p, a generator α, and $\alpha^a \bmod p$. To encrypt a message m, one selects a random k (where $1 \le k \le p-2$) and computes the ciphertext (γ, δ) where $\gamma = \alpha^k \bmod p$ and $\delta = m(\alpha^a \bmod p)^k \bmod p$. Decryption is performed by computing $m = \delta(\gamma^{p-1-a}) \bmod p$. (See the Handbook of Applied Cryptography by A. Menezes, P. van Oorschot, and S. Vanstone, 1997, pages 294-298, for a description of ElGamal public-key encryption).

To make the ElGamal public-key decryption process leak-resistant, the secret exponent $(p - 1 - a)$ is stored in two halves a_1 and a_2, such that $a_1 a_2 = (\phi(p)-a) \bmod \phi(p)$. When generating ElGamal parameters for this leak-resistant implementation, it is recommended, but not required, that p be chosen with $\frac{p-1}{2}$ prime so that $\phi(p)/2$ is prime. The variables a_1 and a_2 are normally chosen initially as random integers between 0 and $\phi(p)$.

Alternatively, it is possible to generate a first, then choose a_1 and a_2, as by selecting a_1 relatively prime to $\phi(p)$ and computing $a_2 = (a^{-1} \bmod \phi(p))(a_1^{-1} \bmod \phi(p)) \bmod \phi(p)$.

Figure 4 illustrates an exemplary leak-resistant ElGamal decryption process. At step 405, the decryption device receives an encrypted message pair (γ, δ). At step 410, the device selects a random r_1 where $1 \le r_1 < \phi(p)$ and $\gcd(r_1, \phi(p)) = 1$. At step 415, the device updates a_1 by computing $a_1 \leftarrow a_1 r_1 \bmod \phi(p)$, over-writing the old value of a_1 with the new value. At step 420, the device computes the inverse of r_1 by computing $r_2 = (r_1)^{-1} \bmod \phi(p)$. Because r_1 is not used after this step, its storage space may be used to hold r_2. Note that if $\frac{p-1}{2}$ is prime, then r_2 may also be found by finding $r_2' = r_1^{(p-1)/2-2} \bmod \frac{p-1}{2}$, and using the CRT to find $r_2 \pmod{p-1}$. At step 425, the device updates a_2 by computing $a_2 \leftarrow a_2 r_2 \bmod \phi(p)$. At step 430, the device begins the private key (decryption) process by computing $m' = \gamma^{a_1} \bmod p$. At step 435, the device computes $m = \delta(m')^{a_2} \bmod p$ and returns the message m. If verification is successful, the result equals the original message because:

$$\begin{aligned}(\delta)(m')^{a_2} \bmod p &= \left(m(\alpha^a)^k\right)\left(\gamma^{a_1} \bmod p\right)^{a_2} \bmod p \\ &= \left(m\alpha^{ak}\right)\left(\gamma^{a_1 a_2 \bmod \phi(p)}\right) \bmod p \\ &= \left(m\alpha^{ak}\right)\left((\alpha^k \bmod p)^{-a \bmod \phi(p)}\right) \bmod p \\ &= \left(m\alpha^{ak}\right)\left(\alpha^{-ak}\right) \bmod p \\ &= m\end{aligned}$$

As with the ElGamal public key encryption scheme, the private key for the ElGamal digital signature scheme is a randomly-selected secret a, where $1 \le a \le p-2$. The public key is also similar, consisting of a prime p, a generator α, and public parameter y where $y = \alpha^a \bmod p$. To sign a message m, the private key holder chooses or precomputes a random secret integer k (where $1 \le k \le p-2$ and k is relatively prime to $p-1$) and its inverse, $k^{-1} \bmod \phi(p)$. Next, the signer computes the signature (r, s), where $r = \alpha^k \bmod p$, $s = \left((k^{-1} \bmod \phi(p))[\mathrm{H}(m) - ar]\right) \bmod \phi(p)$, and $\mathrm{H}(m)$ is the hash of the message. Signature verification is performed using the public key (p, α, y) by verifying that $1 \le r < p$ and by verifying that $y^r r^s \bmod p = \alpha^{\mathrm{H}(m)} \bmod p$.

To make the ElGamal digital signing process leak-resistant, the token containing the private key maintains three persistent variables, a_k, w, and r. Initially, $a_k = a$ (the private exponent), $w = 1$, and $r = \alpha$. When a message m is to be signed (or during the

precomputation before signing), the token generates a random number b and its inverse b^{-1} mod $\phi(p)$, where b is relatively prime to $\phi(p)$ and $0 < b < \phi(p)$. The token then updates a_k, w, and r by computing $a_k \leftarrow (a_k)(b^{-1}) \bmod \phi(p)$, $w \leftarrow (w)(b^{-1}) \bmod \phi(p)$, and $r \leftarrow (r^b) \bmod p$. The signature (r, s) is formed from the updated value of r and s, where $s = (w(\mathrm{H}(m) - a_k r)) \bmod \phi(p)$. Note that a_k, w, and r are not randomized prior to the first operation, but should be randomized before exposure to possible attack, since otherwise the first operation may leak more information than subsequent ones. It is thus recommended that a dummy signature or parameter update with $a_k \leftarrow (a_k)(b^{-1}) \bmod \phi(p)$, $w \leftarrow (w)(b^{-1}) \bmod \phi(p)$, and $r \leftarrow (r^b) \bmod p$ be performed immediately after key generation. Valid signatures produced using the exemplary tamper-resistant ElGamal process may be checked using the normal ElGamal signature verification procedure.

It is also possible to split all or some the ElGamal variables into two halves as part of the leak resistance scheme. In such a variant, a is replaced with a_1 and a_2, w with w_1 and w_2, and r with r_1 and r_2. It is also possible to reorder the operations by performing, for example, the parameter updates as a precomputation step prior to receipt of the enciphered message. Other variations and modifications to the exemplary embodiments described herein will be evident to one skilled in the art.

D. Leak-Resistant DSA

Another commonly used asymmetric cryptographic protocol is the Digital Signature Algorithm (DSA, also known as the Digital Signature Standard, or DSS), which is defined in "Digital Signature Standard (DSS)," Federal Information Processing Standards Publication 186, National Institute of Standards and Technology, May 19, 1994 and described in detail in the Handbook of Applied Cryptography, pages 452 to 454. DSA is widely used for digital signatures. If information about the secret key leaks from a DSA implementation, security can be compromised. Consequently, leak-resistant implementations of DSA would be useful.

In non-leak-proof systems, the private key consists of a secret parameter a, and the public key consists of (p, q, α, y), where p is a large (usually 512 to 1024 bit) prime, q is a 160-bit prime, α is a generator of the cyclic group of order q mod p, and $y = \alpha^a \bmod p$. To sign a message whose hash is $\mathrm{H}(m)$, the signer first generates (or precomputes) a random integer k and its inverse k^{-1} mod q, where $0 < k < q$. The signer then computes the signature (r, s), where $r = (\alpha^k \bmod p) \bmod q$, and $s = (k^{-1} \bmod q)(\mathrm{H}(m) + ar) \bmod q$.

In an exemplary embodiment of a leak-resistant DSA signing process, the token containing the private key maintains two variables in nonvolatile memory, a_k and k, which are initialized with $a_k = a$ and $k = 1$. When a message m is to be signed (or during the precomputation before signing), the token generates a random integer b and its inverse b^{-1} mod q, where $0 < b < q$. The token then updates a_k and k by computing $a_k \leftarrow (a_k b^{-1} \bmod q)(k) \bmod q$, followed by $k \leftarrow b$. The signature (r, s) is formed from the updated values of a_k and k by computing $r = \alpha^k \bmod p$ (which may be reduced mod q), and $s = [(b^{-1}H(m) \bmod q) + (a_k r) \bmod q] \bmod q$. As indicated, when computing s, $b^{-1}H(m) \bmod q$ and $(a_k r) \bmod q$ are computed first, then combined mod q. Note that a_k and k should be randomized prior to the first operation, since the first update may leak more information than subsequent updates. It is thus recommended that a dummy signature (or parameter update) be performed immediately after key generation. Valid signatures produced using the leak-resistant DSA process may be checked using the normal DSA signature verification procedure.

IV. Other Algorithms and Applications

Still other cryptographic processes can be made leak-proof or leak-resistant, or may be incorporated into leak-resistant cryptosystems. For example, cryptosystems such as those based on elliptic curves (including elliptic curve analogs of other cryptosystems), secret sharing schemes, anonymous electronic cash protocols, threshold signatures schemes, etc. be made leak resistant using the techniques of the present invention.

Implementation details of the schemes described may be adjusted without materially changing the invention, for example by re-ordering operations, inserting steps, substituting equivalent or similar operations, etc. Also, while new keys are normally generated when a new system is produced, it is often possible to add leak resistance retroactively while maintaining or converting existing private keys.

Leak-resistant designs avoid performing repeated mathematical operations using non-changing (static) secret values, since they are likely to leak out. However, in environments where it is possible to implement a simple function (such as an exclusive OR) that does not leak information, it is possible use this function to implement more complex cryptographic operations.

While the exemplary implementations assume that the leak functions can reveal any information present in the system, designers may often safely use the (weaker) assumption

that information not used in a given operation will not be leaked by that operation. Schemes using this weaker assumption may contain a large table of precomputed subkey values, from which a unique or random subset are selected and/or updated for each operation. For example, DES implementations may use indexed permutation lookup tables in which a few table elements are exchanged with each operation.

While leak resistance provides many advantages, the use of leak resistance by itself cannot guarantee good security. For example, leak-resistant cryptosystems are not inherently secure against error attacks, so operations should be verified. (Changes can even be made to the cryptosystem and/or leak resistance operations to detect errors.) Similarly, leak resistance by itself does not prevent attacks that extract the entire state out of a device (e.g., $L=L_{MAX}$). For example, traditional tamper resistance techniques may be required to prevent attackers from staining ROM or EEPROM memory cells and reading the contents under a microscope. Implementers should also be aware of interruption attacks, such as those that involve disconnecting the power or resetting a device during an operation, to ensure that secrets will not be compromised or that a single leaky operation will not be performed repeatedly. (As a countermeasure, devices can increment a counter in nonvolatile memory prior to each operation, and reset or reduce the counter value when the operation completes successfully. If the number of interrupted operations since the last successful update exceeds a threshold value, the device can disable itself.) Other tamper resistance mechanisms and techniques, such as the use of fixed-time and fixed-execution path code or implementations for critical operations, may need to be used in conjunction with leak resistance, particularly for systems with a relatively low self-healing rate (e.g., L_{MAX} is small).

Leak-resistant algorithms, protocols, and devices may be used in virtually any application requiring cryptographic security and secure key management, including without limitation: smartcards, electronic cash, electronic payments, funds transfer, remote access, timestamping, certification, certificate validation, secure e-mail, secure facsimile, telecommunications security (voice and data), computer networks, radio and satellite communications, infrared communications, access control, door locks, wireless keys, biometric devices, automobile ignition locks, copy protection devices, payment systems, systems for controlling the use and payment of copyrighted information, and point of sale terminals.

The foregoing shows that the method and apparatus of the present invention can be implemented using numerous variations and modifications to the exemplary embodiments described herein, as would be known by one skilled in the art. Thus, it is intended that the scope of the present invention be limited only with regard to the claims below.

WHAT IS CLAIMED IS:

1. A method for implementing RSA with the Chinese Remainder Theorem for use in a cryptographic system, with resistance to leakage attacks against said cryptographic system, comprising the steps of:

(a) obtaining a representation of an RSA private key corresponding to an RSA public key, said private key characterized by secret factors p and q;

(b) storing said representation of said private key in a memory;

(c) obtaining a message for use in an RSA cryptographic operation;

(d) computing a first modulus, corresponding to a multiple of p, where the value of said multiple of p and the value of said multiple of p divided by p are both unknown to an attacker of said cryptographic system;

(e) reducing said message modulo said first modulus;

(f) performing modular exponentiation on the result of step (e);

(g) computing a second modulus, corresponding to a multiple of q, where the value of said multiple of q and the value of said multiple of q divided by q are both unknown to an attacker of said cryptographic system;

(h) reducing said message modulo said second modulus;

(i) performing modular exponentiation on the result of step (h);

(j) combining the results of said steps (e) and (h) to produce a result which, if operated on with an RSA public key operation using said RSA public key, yields said message; and

(k) repeating steps (c) through (j) a plurality of times using different values for said multiple of p and for said multiple of q.

2. The method of claim 1 where:

(i) said step (b) includes storing an exponent d_p of said RSA private key in said memory as a plurality of parameters;

(ii) an arithmetic function of at least one of said plurality of parameters is congruent to d_p, modulo (p-1);

(iii) none of said parameters comprising said stored d_p is equal to d_p;

(iv) an exponent used in said step (f) is at least one of said parameters;

(v) at least one of said parameters in said memory changes with said repetitions of said steps (c) through (j).

3. The method of claim 2 where said plurality of parameters includes a first parameter equal to said d_p plus a multiple of phi(p), and also includes a second parameter equal to a multiple of phi(p), where phi denotes Euler's totient function.

4. The method of claim 1 where the value of said multiple of p divided by p is equal to the value of said multiple of q divided by q.

5. The method of claim 1 where said multiple of p and said multiple of q used in said steps (c) through (j) are updated and modified in said memory after said step (b).

6. The method of claim 1 performed in a smart card.

7. The method of claim 1 where at least two of said steps are performed in an order other than (a) through (k)

8. A method for implementing RSA for use in a cryptographic system, with resistance to leakage attacks against said cryptographic system, comprising the steps of:

(a) obtaining an RSA private key corresponding to an RSA public key, said RSA public key having an RSA modulus n;

(b) storing said private key in a memory in a form whereby a secret parameter of said key is stored as an arithmetic combination of phi(x) and a first at least one key masking parameter, where

(i) an operand x in said phi(x) is an exact multiple of at least one factor of said modulus n of said RSA public key; and

(ii) said first key masking parameter is unknown to an attacker of said cryptosystem;

(iii) a representation of said first key masking parameter is stored in said memory;

(iv) phi denotes Euler's totient function;

(c) receiving a message;

(d) deriving an RSA input from said message;

(e) performing modular exponentiation to raise said RSA input to a power dependent on said secret parameter, modulo an RSA modulus stored in said memory, to produce an RSA result such that said RSA result raised to the

power of the public exponent of said RSA public key, modulo the modulus of said RSA public key, equals said RSA input;

(f) updating said secret parameter in said memory by:

(i) modifying said first key masking parameter to produce a new key masking parameter, where said modification is performed in a manner such that an attacker with partial useful information about said first key masking parameter has less useful information about said new key masking parameter; and

(ii) using said new key masking parameter to update said secret parameter in said memory;

(g) repeating steps (d) through (f) a plurality of times, where the power used for each of said modular exponentiation steps (e) is different.

9. The method of claim 8 where said operand x in said phi(x) corresponds to said RSA modulus n of said RSA public key.

10. The method of claim 8 where said operand x in said phi(x) corresponds to a prime factor of said RSA modulus n of said RSA public key, and where said modular exponentiation of said step (e) is performed using the Chinese Remainder Theorem.

11. A method for implementing exponential key exchange for use in a cryptographic system, with resistance to leakage attacks against said cryptographic system, comprising the steps of:

(a) obtaining, and storing in a memory, exponential key exchange parameters g and p, and a plurality of secret exponent parameters on which an arithmetic relationship may be computed to produce an exponent x;

(b) using a key update transformation to produce a plurality of updated secret exponent parameters while maintaining said arithmetic relationship thereamong;

(c) receiving a public value y from a party with whom said key exchange is desired;

(d) using said updated secret exponent parameters to perform a cryptographic computation yielding an exponential key exchange result z = y^x mod p;

(e) using said result z to secure an electronic communication with said party; and

(f) performing said steps (b), (c), (d), and (e) in a plurality of transactions.

12. The method of claim 11 where each of said transactions involves a different said party.

13. The method of claim 11 where said arithmetic relationship is such that said exponential key exchange result is a product of certain of said secret exponent parameters, both before and after said step (b).

14. The method of claim 11 where said key update transformation includes choosing a random key update value r; and where said step (b) includes multiplying one of said secret exponent parameters by r and another of said secret exponent parameters by an inverse of r, said multiplication being performed modulo phi(p), where phi is Euler's totient function.

15. The method of claim 11 where said key update transformation includes choosing a random key update value r; and where said step (b) includes adding r to one of said secret exponent parameters and subtracting r from another of said secret exponent parameters.

16. The method of claim 15 where said secret exponent parameters include two values x_1 and x_2 such that x_1+x_2 is congruent to x, modulo phi(p), where phi is Euler's totient function, and where said step of performing said cryptographic computation yielding said exponential key exchange result includes computing $z_1 = y\hat{}x_1 \bmod p$, $z_2 = y\hat{}x_2 \bmod p$, and $z=z_1z_2 \bmod p$.

17. A cryptographic token configured to perform cryptographic operations using a secret key in a secure manner, comprising:

(a) an interface configured to receive power from a source external to said token;

(b) a memory containing said secret key;

(c) a processor:

(i) configured to receive said power delivered via said interface;

(ii) configured to perform said processing using said secret key from said memory;

(d) said token having a power consumption characteristic:

(i) that is externally measurable; and

(ii) that varies over time in a manner measurably correlated with said cryptographic operations; and

(e) a source of unpredictable information usable in said cryptographic operations to make determination of said secret key infeasible from external measurements of said power consumption characteristic.

18. The cryptographic token of claim 17, in the form of a secure microprocessor.

19. The cryptographic token of claim 17, in the form of a smart card.

20. The cryptographic token of claim 19, wherein said cryptographic operations performed by said smart card enable a holder thereof to decrypt an encrypted communication received via a computer network.

21. The cryptographic token of claim 19, wherein said smart card is configured to store value in an electronic cash scheme.

22. The cryptographic token of claim 21, wherein said cryptographic operations include authenticating that a balance of said stored value has been decreased.

23. The cryptographic token of claim 17, wherein said cryptographic operations include asymmetric private key operations.

24. The cryptographic token of claim 23 wherein said cryptographic operations include exponential key agreement operations.

25. The cryptographic token of claim 23, wherein said cryptographic operations include DSA signing operations.

26. The cryptographic token of claim 23, wherein said cryptographic operations include ElGamal private key operations.

27. The cryptographic token of claim 23, wherein said asymmetric private key operations include RSA private key operations.

28. The cryptographic token of claim 27 wherein said private key operations include Chinese Remainder Theorem operations.

29. The cryptographic token of claim 17, wherein said cryptographic operations include symmetric encryption operations.

30. The cryptographic token of claim 17, wherein said cryptographic operations include symmetric decryption operations.

31. The cryptographic token of claim 17, wherein said cryptographic operations include symmetric authentication operations using said secret key.

32. The cryptographic token of claim 17, wherein said cryptographic operations include authenticating a payment.

33. The cryptographic token of claim 17, wherein said cryptographic operations include securing a broadcast communications signal.

34. The cryptographic token of claim 33, wherein said cryptographic operations include decrypting a satellite broadcast.

35. A method for securely managing and using a private key in a computing environment where information about said private key may leak to attackers, comprising the steps of:

(a) using a first private key, complementary to a public key, to perform first asymmetric cryptographic operation;

(b) reading at least a portion of said first private key from a memory;

(c) transforming said read portion of said first private key to produce a second private key:

(i) said second private key usable to perform a subsequent asymmetric cryptographic operation in a manner that remains complementary to said public key, and

(ii) said transformation enabling said asymmetric cryptographic operations to be performed in a manner such that information leaked during said

first asymmetric cryptographic operation does not provide incrementally useful information about said second private key;

(d) obtaining a datum;

(e) using said second private key to perform said subsequent asymmetric cryptographic operation on said datum.

36. The method of claim 35 where said asymmetric cryptographic operation includes a digital signing operation.

37. The method of claim 36 where said signing operation is an RSA operation.

38. The method of claim 36 where said signing operation is an DSA operation.

39. The method of claim 36 where said signing operation is an ElGamal operation.

40. The method of claim 35 where said asymmetric cryptographic operation includes a decryption operation.

41. The method of claim 40 where said decryption operation is an RSA operation.

42. The method of claim 40 where said decryption operation is an ElGamal operation.

43. The method of claim 35 where at least two of said steps are performed in an order different than (a), (b), (c), (d), (e).

44. The method of claim 35 further comprising the step, after at least said step (c), of replacing said private key in said memory with said second private key.

45. The method of claim 35, performed in a smart card.

46. The method of claim 35, further comprising the steps of: prior to at least said step (c), incrementing a counter stored in a nonvolatile memory and verifying that said counter has not exceeded a threshold value; and after at least said step (c) has completed successfully, decreasing a value of said counter.

47. A method for performing cryptographic transactions while protecting a stored cryptographic key against compromise due to leakage attacks, comprising the steps of:

(a) retrieving a stored private cryptographic key stored in a memory, said stored key having been used in a previous cryptographic transaction;

(b) using a first cryptographic function to derive from said stored key an updated key, about which useful information about said stored key obtained through monitoring of leaked information is effectively uncorrelated to said updated key;

(c) replacing said stored key in said memory with said updated key;

(d) using an asymmetric cryptographic function, cryptographically processing a datum with said updated key; and

(e) sending said cryptographically processed datum to an external device having a public key corresponding to said stored key.

48. The method of claim 47 where said stored key includes a first plurality of parameters, and where said updated key includes a second plurality of parameters.

49. The method of claim 48 where no secret value within said first plurality of parameters is included within said second plurality of parameters.

50. The method of claim 49 where said first plurality of parameters is different than said second plurality of parameters, yet a predetermined relationship among said first plurality of parameters is also maintained among said second plurality of parameters.

51. The method of claim 50 where said relationship among said plurality of parameters is an arithmetic function involving at least two of said plurality of parameters.

52. The method of claim 51 where said arithmetic function is the sum of said parameters.

53. The method of claim 51 where said relationship includes a bitwise combination of said parameters.

54. The method of claim 53 where said bitwise combination is an exclusive OR.

55. The method of claim 47 where said step (b) includes using pseudorandomness to derive said updated key.

56. A method for implementing a private key operation for an asymmetric cryptographic system with resistance to leakage attacks against said cryptographic system, comprising the steps of:

(a) encoding a portion of a private key as at least two component parts, such that an arithmetic function of said parts yields said portion;

(b) modifying said component parts to produce updated component parts, but where said arithmetic function of said updated parts still yields said private key portion;

(c) obtaining a message for use in an asymmetric private key cryptographic operation;

(d) separately applying said component parts to said message to produce an intermediate result;

(e) deriving a final result from said intermediate result such that said final result is a valid result of applying said private key to said message; and

(f) repeating steps (b) through (e) a plurality of times.

57. The method of claim 56 where said private key portion includes an exponent, and where said intermediate result represents the result of raising said message to the power of said exponent, modulo a second key portion.

58. The method of claim 57 where said private key operation is configured for use with an RSA cryptosystem.

59. The method of claim 57 where said private key operation is configured for use with an ElGamal cryptosystem.

60. The method of claim 56 where said private key operation is configured for use with a DSA cryptosystem.

61. The method of claim 60 where said private key is represented by secret parameters a_k and k whose product, modulo a predetermined DSA prime q for said private key, yields said private key portion.

62. The method of claim 56 implemented in a smart card.

63. The method of claim 56 where said private key is configured for use with an elliptic curve cryptosystem.

64. A method for performing cryptographic transactions in a cryptographic token while protecting a stored cryptographic key against compromise due to leakage attacks, including the steps of:

(a) retrieving said stored key from a memory;

(b) cryptographically processing said key, to derive an updated key, by executing a cryptographic update function that:

(i) prevents partial information about said stored key from revealing useful information about said updated key, and

(ii) also prevents partial information about said updated key from revealing useful information about said stored key;

(c) replacing said stored key in said memory with said updated key;

(d) performing a cryptographic operation using said updated key; and

(e) repeating steps (a) through (d) a plurality of times.

65. The method of claim 64 where said cryptographic update function of said step (b) includes a one-way hash operation.

66. The method of claim 64 where said cryptographic operation of said step (d) is a symmetric cryptographic operation; and comprising the further step of sending a result of said cryptographic operation to a party capable of rederiving said updated key.

67. The method of claim 64 further comprising the step, prior to said step (a), of receiving from a second party a symmetric authentication code and a parameter; and said where said step (b) includes iterating a cryptographic transformation a number of times determined from said parameter; and where said step (d) includes performing a symmetric message authentication code verification operation.

68. he method of claim 66 where said step (d) of performing said cryptographic operation includes using said updated key to encrypt a datum.

69. The method of claim 66 where said updated key contains unpredictable information such that said updated key is not stored in its entirety anywhere outside of said cryptographic token; and where the result of said step (d) is independent of said unpredictable information.

70. The method of claim 64 where said step (c) of replacing said stored key includes:
(i) explicitly erasing a region of said memory containing said stored key; and
(ii) storing said updated key in said region of memory.

71. The method of claim 64 performed within a smart card.

FIG. 1

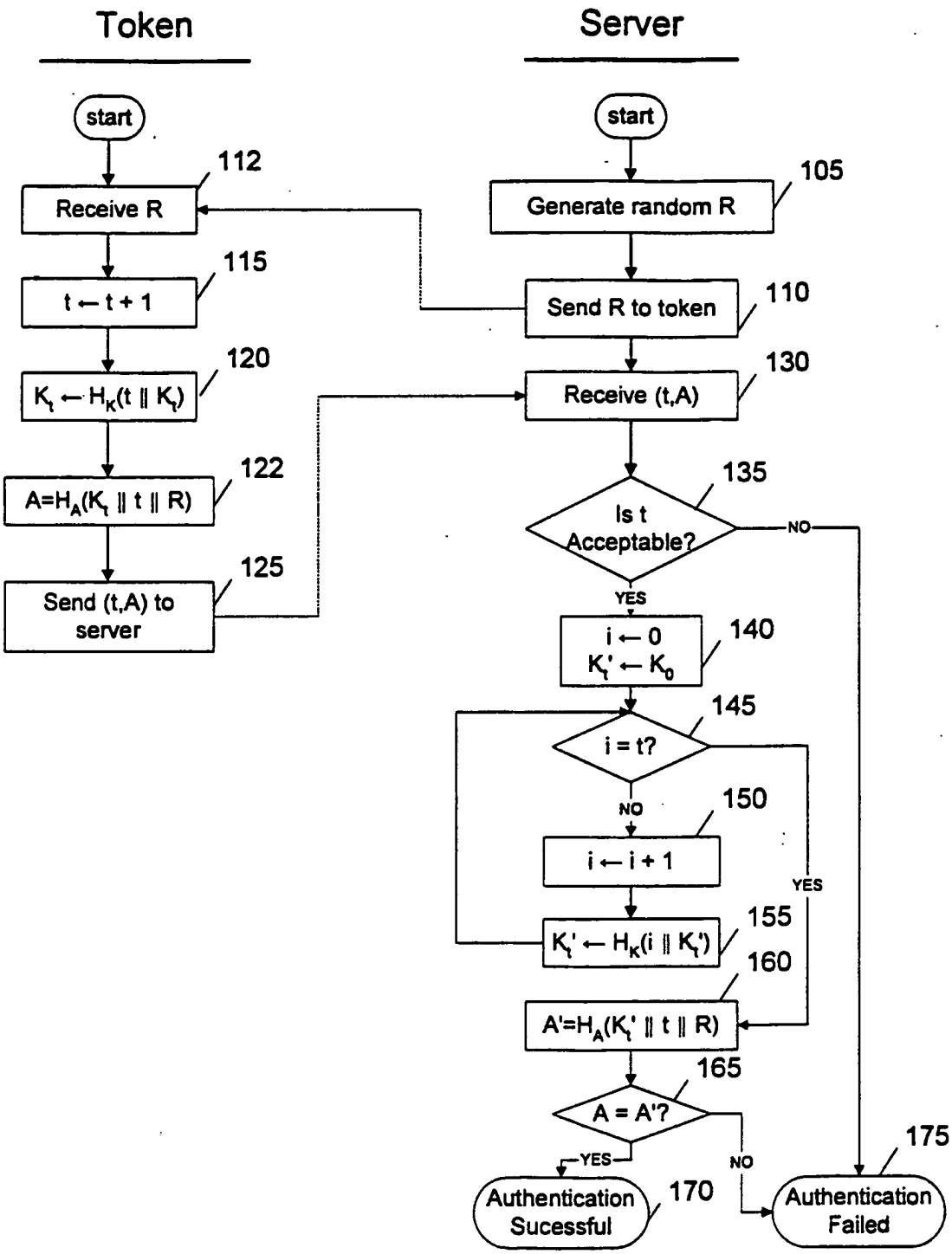


FIG. 2

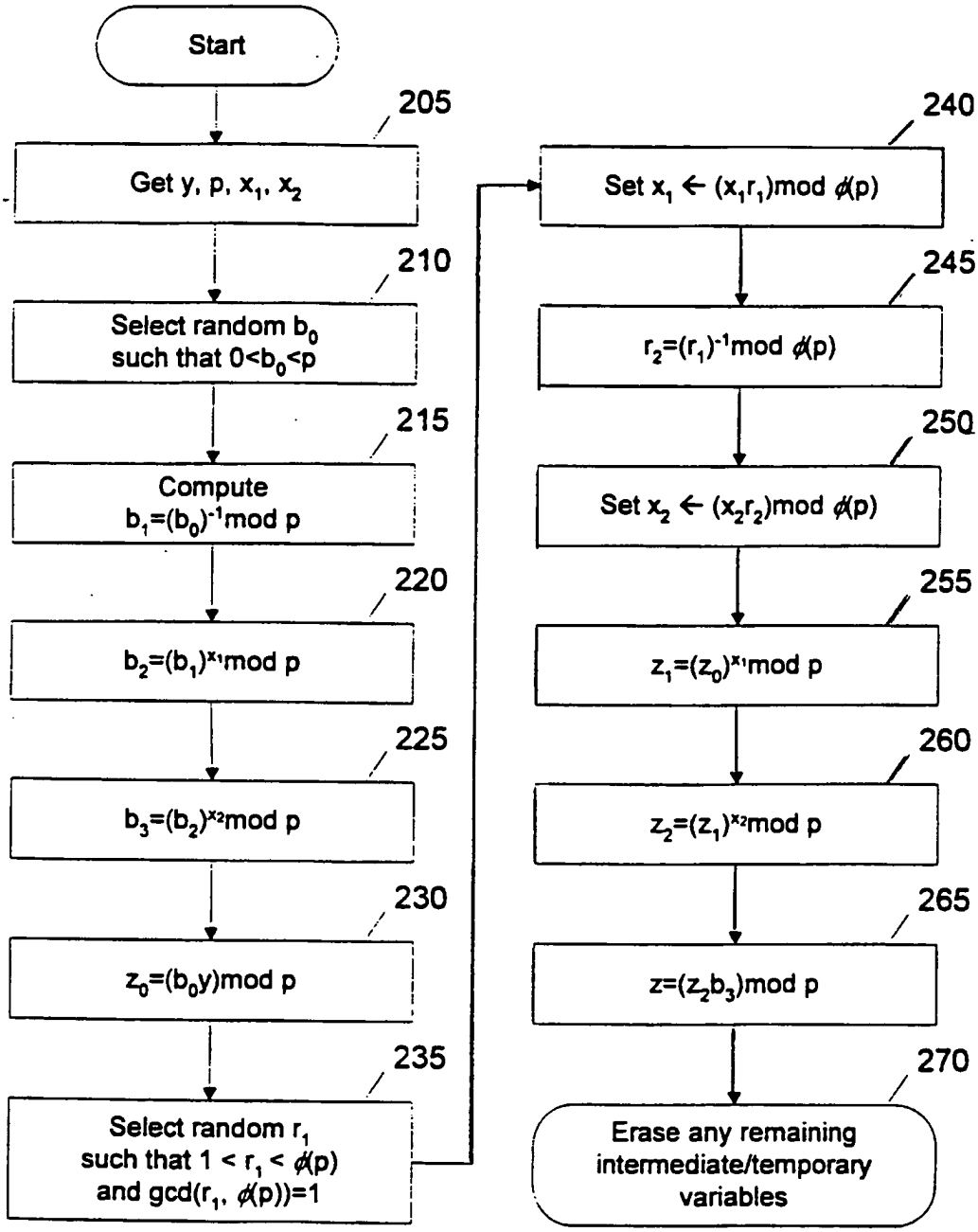


FIG. 3

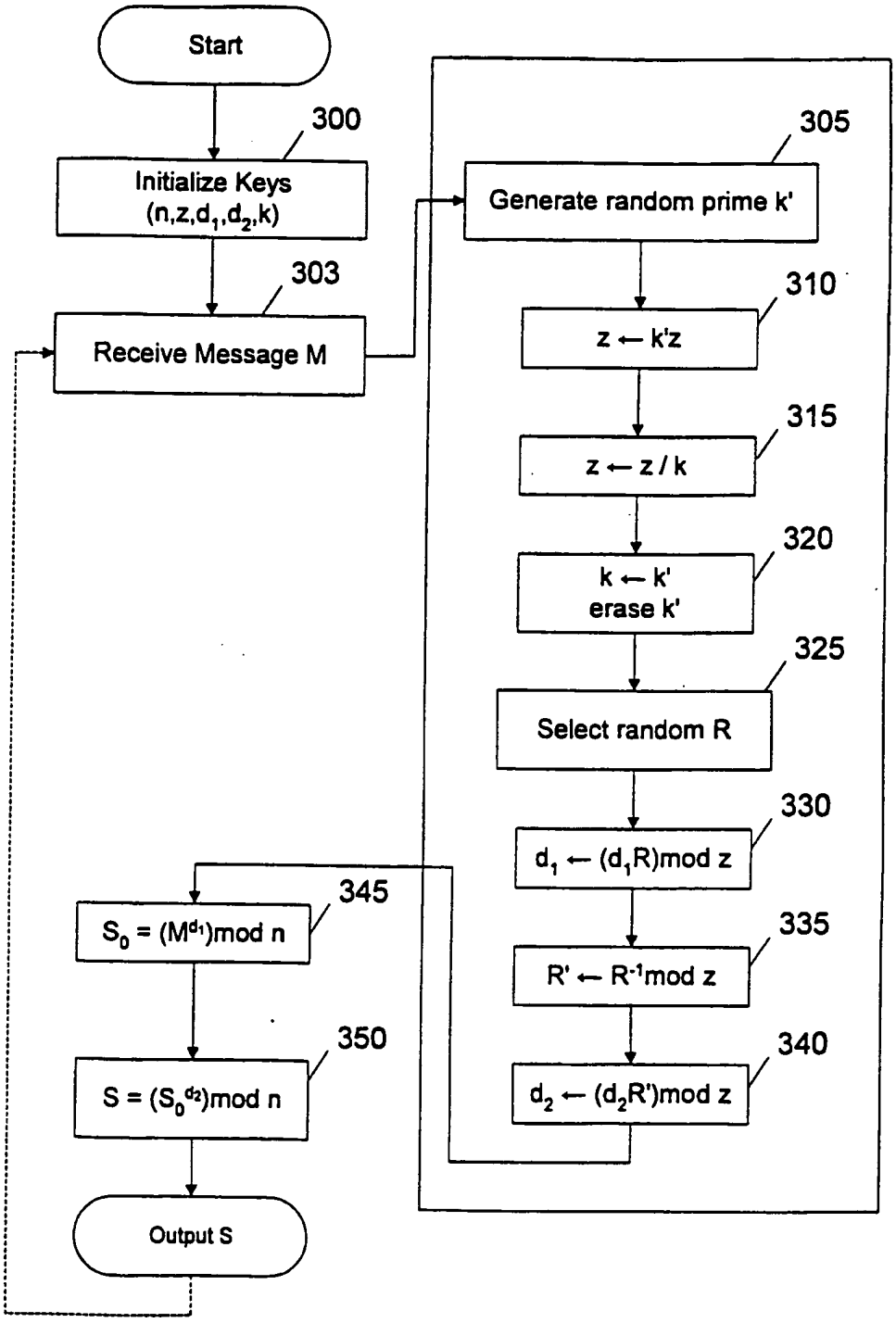
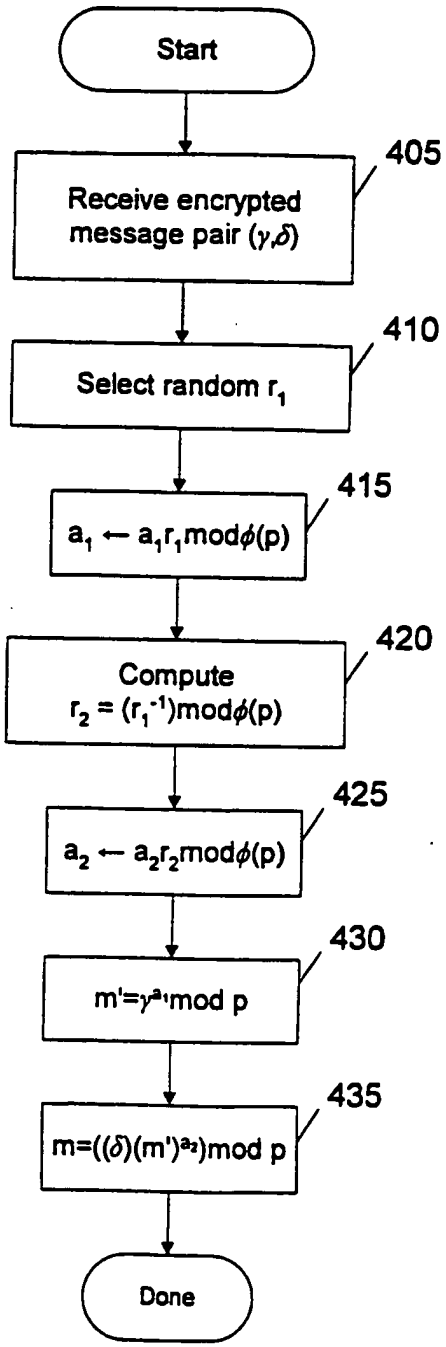


FIG. 4



INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/27896

A. CLASSIFICATION OF SUBJECT MATTER
IPC(6) :HO4 L 9/30
US CL :380/30,49
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 380/30,49

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Please See Extra Sheet.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 4,799,258 A (DAVIES et al) 17 January 1989, abstract, col.4, lines 43-50, col.7,lines 15-33, col.8,lines 12-19	17-23,25- 45
Y	US 5,546,463 A (CAPUTO et al) 13 August 1996, abstract, col.2, lines, 60-65, col.5, lines 39-50,53-58, col.6, lines 7-12	17-23,25-45
A,P	US 5,848,159 A (COLLINS et al.) 08 December 1998, abstract, col.1, lines 56-67, col.4, lines 33-44, col.5, lines 52-67, col.6, lines 24-30	1-16,46-71

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means.		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search	Date of mailing of the international search report
30 MARCH 1999	06 MAY 1999
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-0040	Authorized officer GAIL HAYES Telephone No. (703) 305-9711

Form PCT/ISA/210 (second sheet)(July 1992) ★

B. FIELDS SEARCHED
Electronic data bases consulted (Name of data base and where practicable terms used):

APS
search terms:token,smart card,tamper proof, tamper resistant,leak-resistant,RSA, public key, private key, chinese remainder theorem, diffie hellman,dsa,des

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 :
H04N 7/167, G06F 1/00 | **A1**

(11) International Publication Number: **WO 99/48296**

(43) International Publication Date: 23 September 1999 (23.09.99)

(21) International Application Number: PCT/US99/05734

(22) International Filing Date: 16 March 1999 (16.03.99)

(30) Priority Data:
60/078,053 16 March 1998 (16.03.98) US

(71) Applicant: INTERTRUST TECHNOLOGIES CORPORATION [US/US]; 460 Oakmead Parkway, Sunnyvale, CA 94086 (US).

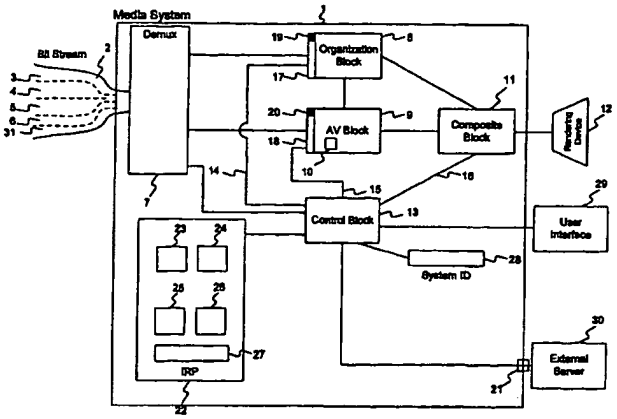
(72) Inventors: SHAMOON, Talal, G.; 533 Bryant Street #5, Palo Alto, CA 94301 (US). HILL, Ralph, D.; 224 Dover Street, Los Gatos, CA 94032 (US). RADCLIFFE, Chris, D.; 3654 Farm Hill Boulevard, Redwood City, CA 94061 (US). HWA, John, P.; 503 Lower Vinters Circle, Fremont, CA 94539 (US).

(74) Agents: GARRETT, Arthur, S. et al.; Finnegan, Henderson, Farabow, Garrett & Dunner, L.L.P., 1300 I Street, Washington, DC 20005-3315 (US).

(81) Designated States: CA, CN, JP, KR, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: METHODS AND APPARATUS FOR CONTINUOUS CONTROL AND PROTECTION OF MEDIA CONTENT



(57) Abstract

A novel method and apparatus for protection of streamed media content is disclosed. The apparatus includes control means for governance of content streams or objects, decryption means for decrypting content streams or objects under control of the control means, and feedback means for tracking actual use of content streams or objects. The control means may operate in accordance with rules received as part of the streamed content, or through a side-band channel. The rules may specify allowed uses of the content, including whether or not the content can be copied or transferred, and whether and under what circumstances received content may be "checked out" of one device and used in a second device. The rules may also include or specify budgets, and a requirement that audit information be collected and/or transmitted to an external server. The apparatus may include a media player designed to call plugins to assist in rendering content. A "trust plugin" and its use are disclosed so that a media player designed for use with unprotected content may render protected content without the necessity of requiring any changes to the media player. The streamed content may be in a number of different formats, including MPEG-4, MP3, and the RMFF format.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

METHODS AND APPARATUS FOR CONTINUOUS CONTROL AND PROTECTION OF MEDIA CONTENT

FIELD OF THE INVENTION

This invention relates generally to computer and/or electronic security. More particularly, this invention relates to systems and methods for protection of information in streamed format.

BACKGROUND

Streaming digital media consists generally of sequences of digital information received in a "stream" of packets, and designed to be displayed or rendered. Examples include streamed audio content, streamed video, etc.

Digital media streams are becoming an increasingly significant means of content delivery, and form the basis for several adopted, proposed or de facto standards. The acceptance of this format, however, has been retarded by the ease with which digital media streams can be copied and improperly disseminated, and the consequent reluctance of content owners to allow significant properties to be distributed through streaming digital means. For this reason, there is a need for a methodology by which digital media streams can be protected.

SUMMARY OF THE INVENTION

Consistent with the invention, this specification describes a new architecture for protection of information provided in streamed format. This architecture is described in the context of a generic system which resembles a system to render content encoded pursuant to the MPEG-4 specification (ISO/IEC 14496.1), though with certain modifications, and with the proviso that the described system may differ from the MPEG-4 standard in certain respects. A variety of different embodiments is described, including an MPEG-4 embodiment and a system designed to render content encoded pursuant to the MP3 specification (ISO/IEC TR 11172).

According to aspects of the invention, this architecture involves system design aspects and information format aspects. System design aspects include the incorporation of content protection functionality, control functionality, and feedback enabling control functionality to monitor the activities of the system. Information format aspects include the incorporation of rule/control information into information streams, and the protection of content through mechanisms such as encryption and watermarking.

Systems and methods consistent with the present invention perform content protection and digital rights management. A streaming media player consistent with the present invention includes a port designed to accept a digital bit stream. The digital bit stream includes content, which is encrypted at least in part, and a secure container including control information designed to control use of the content, including at least one key suitable for decryption of at least a portion of the content. The media player also includes a control arrangement including a means for opening secure containers and extracting cryptographic keys, and means for decrypting the encrypted portion of the content.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate an embodiment of the invention and, together with the description, serve to explain the advantages and principles of the invention. In the drawings,

FIG. 1 shows a generic system consistent with the present invention;

FIG. 2 shows an exemplary Header 201 consistent with the present invention;

FIG. 3 shows a general encoding format consistent with the present invention;

FIG. 4 illustrates one manner for storing a representation of a work consistent with the present invention;

FIG. 5 shows an example of a control message format;

FIG. 6 is a flow diagram illustrating one embodiment of the steps which take place using the functional blocks of FIG. 1;

FIG. 7 illustrates a form wherein the control messages may be stored in Control Block 13;

FIG. 8 shows MPEG-4 System 801 consistent with the present invention;

FIG. 9 shows an example of a message format;

FIG. 10 illustrates an IPMP table consistent with the present invention;

FIG. 11 illustrates a system consistent with the present invention;

FIG. 12 illustrates one embodiment of the DigiBox format;

FIG. 13 shows an example of a Real Networks file format (RMFF);

FIG. 14 shows an RNPFF format consistent with the present invention;

FIG. 15 illustrates the flow of changes to data in the Real Networks file format in an architecture consistent with the present invention;

FIG. 16 illustrates a standard Real Networks architecture;

FIG. 17 shows an exemplary architecture in which a trust plugin operates within the overall Real Networks architecture;

FIG. 18 shows a bit stream format consistent with the principles of the present invention;

FIG. 19 shows one embodiment of protection applied to the MP3 format;

FIG. 20 illustrates one embodiment of an MP3 player designed to process and render protected content;

FIG. 21 illustrates the flow of data in one embodiment in which a protected MPEG-4 file may be created consistent with the present invnetion;

FIG. 22 illustrates the flow of data in one embodiment in which control may be incorporated into an existing MPEG-4 stream consistent with the present invention;

FIG. 23 shows a system consistent with the principles of the present invention;

FIG. 24 shows a system consistent with the principles of the present invention;

FIG. 25 shows an example of an aggregate stream consistent with the present invention;

FIG. 26 illustrates a Header CMPO 2601 consistent with the present invention;

FIG. 27 shows exemplary Content Management Protection Objects consistent with the principles of the present invention; and

FIG. 28 shows an example of a CMPO Data Structure 2801 consistent with the present invention.

DETAILED DESCRIPTION

Reference will now be made in detail to implementations consistent with the principles of the present invention as illustrated in the accompanying drawings.

The following U.S. patents and applications, each of which is assigned to the assignee of the current application, are hereby incorporated in their entirety by reference: Ginter, et al., "Systems and Methods for Secure Transaction Management and Electronic Rights Protection," U.S. Patent Application Serial No. 08/964,333, filed on November 4, 1997 ("Ginter '333"); Ginter, et al., "Trusted Infrastructure Support Systems, Methods and Techniques for Secure electronic commerce, Electronic Transactions, Commerce Process Control Automation, Distributed Computing, and Rights Management, " U.S. Patent Application Serial No. 08/699,712, filed on August 12, 1996 ("Ginter '712") ; Van Wie, et al, "Steganographic Techniques for Securely Delivering Electronic Digital Rights Management Information Over Insecure Communications Channels, U.S. Patent Application Serial No. 08/689,606, filed on August 12, 1996 ("Van Wie") ; Ginter, et. al "Software Tamper Resistance and Secure Communication," U.S. Patent Application Serial No. 08/706,206, filed on August 30, 1996 ("Ginter, '206"); Shear, et al, "Cryptographic Methods, Apparatus & Systems for Storage Media Electronic Rights Management in

Closed & Connected Appliances, " U.S. Patent Application Serial No. 08/848,077, filed on May 15, 1997 ("Shear"); Collberg et al, "Obfuscation Techniques for Enhancing Software Security, " U.S. Patent Application Serial No. 09/095,346, filed on June 9, 1998 ("Collberg"); Shear, "Database Usage Metering and Protection System and Method," U.S. Patent No. 4,827,508, issued on May 2, 1989 ("Shear Patent").

FIG. 1 illustrates Media System 1, which is capable of accepting, decoding, and rendering streamed multimedia content. This is a generic system, though it includes elements based on the MPEG-4 specification. Media System 1 may include software modules, hardware (including integrated circuits) or a combination. In one embodiment, Media System 1 may include a Protected Processing Environment (PPE) as described in the Ginter '333 application.

In FIG. 1, Bit Stream 2 represents input information received by System 1. Bit Stream 2 may be received through a connection to an external network (e.g., an Internet connection, a cable hookup, radio transmission from a satellite broadcaster, etc.), or may be received from a portable memory device, such as a DVD player.

Bit Stream 2 is made up of a group of related streams of information, including Organization Stream 3, Audio Stream 4, Video Stream 5, Control Stream 6, and Info Stream 31. Each of these streams is encoded into the overall Bit Stream 2. Each of these represents a category of streams, so that, for example, Video Stream 5 may be made up of a number of separate Video Streams.

These streams correspond generally to streams described in the MPEG-4 format as follows:

Organization Stream 3 corresponds generally to the BIFS stream and the OD ("Object Descriptor") stream.

Audio Stream 4 and Video Stream 5 correspond generally to the Audio and Video streams.

Control Stream 6 corresponds generally to the IPMP stream.

Audio Stream 4 includes compressed (and possibly encrypted) digital audio information. This information is used to create the sound rendered and output by Media System 1. Audio Stream 1 may represent multiple audio streams. These multiple streams may act together to make up the audio output, or may represent alternative audio outputs.

Video Stream 5 includes compressed (and possibly encrypted) digital video information. This information is used to create the images and video rendered and output by Media System 1. Video Stream 5 may represent multiple video streams. These multiple streams may act together to make up the video output, or may represent alternative

video outputs.

Organization Stream 3 includes organizational information and metadata related to the work to be rendered. This information may include a tree or other organizational device which groups audio and video streams into objects. This information may also include metadata associated with the entire work, the objects, or the individual streams.

Control Stream 6 includes control information, divided generally into header information and messages. The header information includes an identifier for each discrete message. The content of the messages, which will be described further below, may include cryptographic keys and rules governing the use of content.

Info Stream 31 carries additional information associated with the content in other components of Bit Stream 2, including but not limited to graphics representing cover art, text for lyrics, coded sheet music or other notation, independent advertising content, concert information, fan club information, and so forth. Info Stream 31 can also carry system management and control information and/or components, such as updates to software or firmware in Media System 1, algorithm implementations for content-specific functions such as watermarking, etc.

Each of these streams is made up of packets of information. In one exemplary embodiment, each packet is 32 bytes in length. Since a single communications channel (e.g., a cable, a bus, an infrared or radio connection) contains packets from each of the streams, packets need to be identified as belonging to a particular stream. In a preferred embodiment, this is done by including a header which identifies a particular stream and specifies the number of following packets which are part of that stream. In another embodiment, each packet may include individual stream information.

Exemplary Header 201 is shown in FIG. 2. This header may generally be used for the Organization, Audio and Video Streams. A header for the Control Stream is described below. Header 201 includes Field 202, which includes a bit pattern identifying Header 201 as a header. Field 203 identifies the particular type of stream (e.g., Audio Stream, Organization Stream, Control Stream, etc.) Field 204 contains an Elementary Stream Identifier (ES_ID), which is used to identify the particular stream, and may be used in cases where multiple streams of a particular stream type may be encountered at the same time. Field 207 contains a time stamp, which is used by the system to synchronize the various streams, including rendering of the streams. Composite Block 11 may, for example, keep track of the elapsed time from the commencement of rendering. Time Stamp 207 may be used by Composite Block 11 to determine when each object is supposed to be rendered. Time Stamp 207 may therefore specify an elapsed time from commencement of rendering,

and Composite Block 11 may use that elapsed time to determine when to render the associated object.

Field 205 contains a Governance Indicator. Field 206 identifies the number of following packets which are part of the identified stream. In each case, the relevant information is encoded in a binary format. For example, Field 202 might include an arbitrary sequence of bits which is recognized as indicating a header, and Field 203 might include two bits, thereby allowing encoding of four different stream types.

Returning to FIG. 1, System 1 includes Demux 7, which accepts as input Bit Stream 2 and routes individual streams (sometimes referred to as Elementary Streams or "ESs") to appropriate functional blocks of the system.

Bit Stream 2 may be encoded in the format illustrated in FIG. 3. In this figure, Header 301 is encountered in the bit stream, with Packet 302 following, and so on through Packet 308.

When Demux 7 encounters Header 301, Demux 7 identifies Header 301 as a header and uses the header information to identify Packets 302-305 as organization stream packets. Demux 7 uses this information to route these packets to Organization Block 8. Demux 7 handles Header 306 in a similar manner, using the contained information to route Packets 307 and 308 to AV ("Audio Video") Block 9.

AV Block 9 includes Decompressor 10, which accepts Elementary Streams from Audio Stream 4 and Video Stream 5 and decompresses those streams. As decompressed, the stream information is placed in a format which allows it to be manipulated and output (through a video display, speakers, etc.). If multiple streams exist (e.g., two video streams each describing an aspect of a video sequence), AV Block 9 uses the ES_ID to assign each packet to the appropriate stream.

Organization Block 8 stores pointer information identifying particular audio streams and video streams contained in a particular object, as well as metadata information describing, for example, where the object is located, when it is to be displayed (e.g., the time stamp associated with the object), and its relationship to other objects (e.g., is one video object in front of or behind another video object). This organization may be maintained hierarchically, with individual streams represented at the lowest level, groupings of streams into objects at a higher level, complete scenes at a still higher level, and the entire work at the highest level.

FIG. 4 illustrates one manner in which Organization Block 8 may store a representation of a work. In this Figure, Tree 401 represents an entire audiovisual work. Branch 402 represents a high-level organization of the work. This may include, for

example, all of the video or possibly the audio and video associated with a particular scene.

Sub-Branch 403 represents a group of related video objects. Each such object may include an entire screen, or an individual entity within the screen. For example, Sub-Branch 403 may represent a background which does not change significantly from one shot to the next. If the video is moving between two points of reference (e.g., a conversation, with the camera point of view changing from one face to the other), Sub-Branch 404 could represent a second background, used in the second point of view.

Nodes 405 and 406 may represent particular video objects contained within the related group. Node 405 could, for example, represent a distant mountain range, while Node 406 represents a tree immediately behind one of the characters.

Each of the nodes specifies or contains a particular ES_ID, representing the stream containing the information used by that node. Node 405, for example, contains ES_ID 407, which identifies a particular video stream which contains compressed (and possibly encrypted) digital information representing the mountain range.

Composite Block 11 accepts input from Organization Block 8 and from AV Block 9. Composite Block 11 uses the input from Organization Block 8 to determine which specific audiovisual elements will be needed at any given time, and to determine the organization and relationship of those elements. Composite Block 11 accepts decompressed audiovisual objects from AV Block 9, and organizes those objects as specified by information from Organization Block 8. Composite Block 11 then passes the organized information to Rendering Device 12, which might be a television screen, stereo speakers, etc.

Control Block 13 stores control messages which may be received through Control Stream 6 and/or may be watermarked into or steganographically encoded in other streams, including Audio Stream 4 and Video Stream 5. One control message format is illustrated by FIG. 5, which shows Control Message 501. Control Message 501 is made up of Header 502 and Message 503. Header 502 consists of Field 508, which includes a bit pattern identifying the following information as a header; Stream Type Field 509, which identifies this as a header for the organization stream; ID Field 504, which identifies this particular control message; Pointer Field 505, which identifies those ESs which are controlled by this message; Time Stamp Field 507, which identifies the particular portion of the stream which is controlled by this control message (this may indicate that the entirety of the stream is controlled); and Length Field 506, which specifies the length (in bytes) of Message 503. Message 503 may include packets following Header 502, using the general format shown in FIG. 3. In the example shown, Control Message 501 carries the unique ID 111000,

encoded in ID Field 504. This control message controls ESs 14 and 95, as indicated by Pointer Field 505. The associated Message contains 1,024 bytes, as indicated by Length Field 506.

In an alternate embodiment, the association of control to content may be made in Organization Block 8, which may store a pointer to particular control messages along with the metadata associated with streams, objects, etc. This may be disadvantageous, however, in that it may be desirable to protect this association from discovery or tampering by users. Since Control Block 13 will generally have to be protected in any event, storing the association in this block may make protection of Organization Block 8 less necessary.

Control Block 13 implements control over System 1 through Control Lines 14, 15 and 16, which control aspects of Organization Block 8, AV Block 9 and Composite Block 11, respectively. Each of these Control Lines may allow two-way communication.

Control Lines 14 and 15 are shown as communicating with AV Block Stream Flow Controller 18 and with Organization Block Stream Flow Controller 17. These Stream Flow Controllers contain functionality controlled by Control Block 13. In the embodiment illustrated, the Stream Flow Controllers are shown as the first stage in a two-stage pipeline, with information being processed by the Stream Flow Controller and then passed on to the associated functional block. This allows isolation of the control functionality from the content manipulation and display functionality of the system, and allows control to be added in without altering the underlying functionality of the blocks. In an alternate embodiment, the Stream Flow Controllers might be integrated directly into the associated functional blocks.

Stream Flow Controllers 17 and 18 contain Cryptographic Engines 19 and 20, respectively. These Cryptographic Engines operate under control of Control Block 13 to decrypt and/or cryptographically validate (e.g., perform secure hashing, message authentication code, and/or digital signature functions) the encrypted packet streams received from Demux 7. Decryption and validation may be selective or optional according to the protection requirements for the stream.

Cryptographic Engines 19 and 20 may be relatively complex, and may, for example, include a validation calculator that performs cryptographic hashing, message authentication code calculation, and/or other cryptographic validation processes. In addition, as is described further below, additional types of governance-related processing may also be used. In one alternative embodiment, a single Stream Flow Controller may be used for both Organization Stream 3 and Audio/Video Streams 4-5. This may reduce the cost of and space used by System 1. These reductions may be significant, since System 1

may contain multiple AV Blocks, each handling a separate Audio or Video Stream in parallel. This alternative may, however, impose a latency overhead which may be unacceptable in a real-time system.

If the Stream Flow Controllers are concentrated in a single block, they may be incorporated directly into Demux 7, which may handle governance processing prior to routing streams to the functional blocks. Such an embodiment would allow for governed decryption or validation of the entirety of Bit Stream 2, which could occur prior to the routing of streams to individual functional blocks. Encryption of the entirety of Bit Stream 2 (as opposed to individual encryption of individual ESs) might be difficult or impossible without incorporating stream controller functionality into Demux 7, since Demux 7 might otherwise have no ability to detect or read the header information necessary to route streams to functional blocks (that header information presumably being encrypted).

As is noted above, each of the individual streams contained in Bit Stream 2 may be individually encrypted. An encrypted stream may be identified by a particular indicator in the header of the stream, shown in FIG. 2 as Governance Indicator 205.

When a header is passed by Demux 7 to the appropriate functional block, the stream flow controller associated with that block reads the header and determines whether the following packets are encrypted or otherwise subject to governance. If the header indicates that no governance is used, the stream flow controller passes the header and the packets through to the functional blocks with no alteration. Governance Indicator 205 may be designed so that conventionally encoded content (e.g., unprotected MPEG-4 content) is recognized as having no Governance Indicator and therefore passed through for normal processing.

If a stream flow controller detects a set governance indicator, it passes the ES_ID associated with that stream and the time stamp associated with the current packets to Control Block 13 along Control Line 14 or 15. Control Block 13 then uses the ES_ID and time stamp information to identify which control message(s) are associated with that ES. Associated messages are then invoked and possibly processed, as may be used for governance purposes.

A simple governance case is illustrated by FIG. 6, which shows steps which take place using the functional blocks of FIG. 1. In Step 601, Demux 7 encounters a header, and determines that the header is part of the AV stream. In Step 602, Demux 7 passes the header to AV Stream Controller 18. In Step 603, AV Stream Controller 18 reads the header and determines that the governance indicator is set, thereby triggering further processing along Path 604. In Step 605, AV Stream Controller 18 obtains the ES_ID and

time stamp from the header and transmits these to Control Block 13, along Control Line 15. In Step 606, Control Block 13 looks up the ES_ID and determines that the ES_ID is associated with a particular control message. In Step 611, Control Block 13 uses the time stamp information to choose among control messages, if there is more than one control message associated with a particular ES. In Step 607, Control Block 13 accesses the appropriate control message, and obtains a cryptographic key or keys for decryption and/or validation. In Step 608, Control Block 13 passes the cryptographic key(s) along Control Line 15 to AV Stream Controller 18. In Step 609, AV Stream Controller 18 uses the cryptographic key as an input to Cryptographic Engine 20, which decrypts and/or validates the packets following the header as those packets are received from Demux 7. In Step 610, the decrypted packets are then passed to AV Block 9, which decompresses and processes them in a conventional manner.

Time stamp information may be useful when it is desirable to change the control message applicable to a particular ES. For example, it may be useful to encode different portions of a stream with different keys, so that an attacker breaking one key (or even a number of keys) will not be able to use the content. This can be done by associating a number of control messages with the same stream, with each control message being valid for a particular period. The time stamp information would then be used to choose which control message (and key) to use at a particular time. Alternatively, one control message may be used, but with updated information being passed in through the Control Stream, the updates consisting of a new time stamp and a new key.

In an alternative embodiment, Control Block 13 may proactively send the appropriate keys to the appropriate stream flow controller by using time stamp information to determine when a key will be will be needed. This may reduce overall latency.

Control Line 16 from FIG. 1 comes into play once information has been passed from Organization Block 8 and AV Block 9 to Composite Block 11, and the finished work is prepared for rendering through Rendering Device 12. When Composite Block 11 sends an object to Rendering Device 11, Composite Block 11 sends a start message to Control Block 13. This message identifies the object (including any associated ES_IDs), and specifies the start time of the display (or other rendering) of that object. When an object is no longer being rendered, Composite Block 11 sends an end message to Control Block 13, specifying that rendering of the object has ended, and the time at which the ending occurred. Multiple copies of a particular object may be rendered at the same time. For this reason, start and stop messages sent by Composite Block 11 may include an assigned instance ID, which specifies which instance of an object is being rendered.

Control Block 13 may store information relating to start and stop times of particular objects, and/or may pass this information to external devices (e.g., External Server 30) through Port 21. This information allows Control Block 13 to keep track not only of which objects have been decrypted, but of which objects have actually been used. This may be used, since System 1 may decrypt, validate, and/or decompress many more objects than are actually used. Control Block 13 can also determine the length of use of objects, and can determine which objects have been used together. Information of this type may be used for sophisticated billing and auditing systems, which are described further below.

Control Line 16 may also be used to control the operation of Composite Block 11. In particular, Control Block 13 may store information specifying when rendering of a particular object is valid, and may keep track of the number of times an object has been rendered. If Control Block 13 determines that an object is being rendered illegally (i.e., in violation of rules controlling rendering), Control Block 13 may terminate operation of Composite Block 11, or may force erasure of the illegal object.

In an alternate embodiment, the level of control provided by Control Line 16 may at least in part be provided without requiring the presence of that line. Instead, Control Block 13 may store a hash of the organization information currently valid for Organization Block 8. This hash may be received through Control Stream 6, or, alternatively, may be generated by Control Block 13 based on the information contained in Organization Block 8.

Control Block 13 may periodically create a hash of the information currently resident in Organization Block 8, and compare that to the stored hash. A difference may indicate that an unauthorized alteration has been made to the information in Organization Block 8, thereby potentially allowing a user to render information in a manner violative of the rules associated with that information. In such an event, Control Block 13 may take appropriate action, including deleting the information currently resident in Organization Block 8.

If System 1 is designed so that Organization Block 8 controls the use of content by Composite Block 11, so that content cannot be rendered except as is specified by the organization information, Control Block 13 may be able to control rendering of information through verifying that the current Organization Block contents match the hash which has been received by Control Block 13, thereby eliminating at least one reason for the presence of Control Line 16.

Control Block 13 may also be responsible for securely validating the origin, integrity, authenticity, or other properties of received content, through cryptographic

validation means such as secure hashing, message authentication codes, and/or digital signatures.

System 1 may also include an Inter-Rights Point, indicated as IRP 22. IRP 22 is a protected processing environment (e.g., a PPE) in which rules/controls may be processed, and which may store sensitive information, such as cryptographic keys. IRP 22 may be incorporated within Control Block 13, or may be a separate module. As is illustrated, IRP 22 may include CPU 23 (which can be any type of processing unit), Cryptographic Engine 24, Random Number Generator 25, Real Time Clock 26, and Secure Memory 27. In particular embodiments, some of these elements may be omitted, and additional functionality may be included.

Governance Rules

Control messages stored by Control Block 13 may be very complex. FIG. 7 illustrates the form in which the control messages may be stored in Control Block 13, consisting of Array 717. Column 701 consists of the address at which the control messages are stored. Column 702 consists of the identifier for each control message. This function may be combined with that of Column 701, by using the location information of Column 701 as the identifier, or by storing the message in a location which corresponds to the identifier. Column 703 consists of the ES_IDs for each stream controlled by the control message. Column 704 consists of the message itself. Thus, the control message stored at location 1 has the ID 15, and controls stream 903.

In a simple case, the message may include a cryptographic key, used to decrypt the content associated with the stream(s) controlled by the message. This is illustrated by Cryptographic Key 705 from FIG. 7. Cryptographic keys and/or validation values may also be included to permit cryptographic validation of the integrity or origin of the stream.

In a more complex case, the message may include one or more rules designed to govern access to or use of governed content. Rules may fall into a number of categories.

Rules may require that a particular aspect of System 1, or a user of System 1, be verified prior to decryption or use of the governed content. For example, System 1 may include System ID 28, which stores a unique identifier for the system. A particular rule contained in a control message may specify that a particular stream can only be decrypted on a system in which System ID 28 contains a particular value. This is illustrated at row 2 in FIG. 7, in which the message is shown as consisting of a rule and commands. The rule may be implicit, and therefore may not be stored explicitly in the table (e.g. the table may store only the rule, the rule - specific functions (commands) invoked by the rule, or only the functions).

In this case, when Stream Controller 18 encounters a Header for stream 2031 containing a set governance indicator, Stream Controller 18 passes the associated ES_ID (2031) to Control Block 13. Control Block 13 then uses the ES_ID to identify Control Message 20 which governs stream 2031. Control Message 20 includes Rule 706, which includes (or invokes) Commands 707, and an Authorized System ID 708. Authorized System ID 708 may have been received by System 1, either as part of Control Message 20, or as part of another control message (e.g., Control Message 9), which Control Message 20 could then reference in order to obtain access to the Authorized System ID. Such a case might exist, for example, if a cable subscriber had pre-registered for a premium show. The cable system might recognize that registration, and authorize the user to view the show, by sending to the user an ID corresponding to the System ID.

When Rule 706 is invoked, corresponding Commands 707 access System ID 28 and obtain the system ID number. The commands then compare that number to Authorized System ID 708, specified by Rule 706. If the two numbers match, Commands 707 release Cryptographic Key 709 to Stream Controller 18, which uses Cryptographic Key 709 to decrypt the stream corresponding to ES_ID 2031. If the two numbers do not match, Commands 707 fail to release Cryptographic Key 709, so that Stream Controller 18 is unable to decrypt the stream.

In order to carry out these functions, in one embodiment, Control Block 13 includes, or has access to, a processing unit and memory. The processing unit is preferably capable of executing any of the commands which may be included or invoked by any of the rules. The memory will store the rules and association information (ID of the control message and IDs of any governed ESs).

Since the functions being carried out by Control Block 13 are sensitive, and involve governance of content which may be valuable, Control Block 13 may be partially or completely protected by a barrier which resists tampering and observation. As is described above, the processing unit, secure memory, and various other governance-related elements may be contained in IRP 22, which may be included in or separate from Control Block 13.

Control Block 13 may also carry out somewhat more complex operations. In one example, a control message may require that information from System 1 not only be accessed and compared to expected information, but stored for future use. For example, a control message might allow decryption of a Stream, but only after System ID 28 has been downloaded to and stored in Control Block 13. This would allow a control message to check the stored System ID against System ID 28 on a regular basis, or perhaps on every

attempted re-viewing of a particular Stream, thereby allowing the control message to insure that the Stream is only played on a single System.

Control Block 13 may also obtain information dynamically. For example, System 1 may include User Interface 29, which can include any type of user input functionality (e.g., hardware buttons, information displayed on a video screen, etc.) A particular rule from a control message may require that the user enter information prior to allowing decryption or use of a stream. That information may, for example, be a password, which the Rule can then check against a stored password to insure that the particular user is authorized to render the stream.

Information obtained from the user might be more complicated. For example, a rule might require that the user input payment or personal information prior to allowing release of a cryptographic key. Payment information could, for example, constitute a credit card or debit card number. Personal information could include the user's name, age, address, email address, phone number, etc. Entered information could then be sent through Port 21 to External Server 30 for verification. Following receipt of a verification message from External Server 30, the Rule could then authorize release of a cryptographic key. Alternatively, Control Block 13 may be designed to operate in an "off-line" mode, storing the information pending later hookup to an external device (or network). In such a case, Control Block 13 might require that a connection be made at periodic intervals, or might limit the number of authorizations which may be obtained pending the establishment of an external connection.

In a somewhat more complex scenario, a control message may include conditional rules. One particular example is illustrated by row 4 of the table shown in FIG. 7, in which Control Message 700 is shown as controlling streams 49-53. Control Message 700 further consists of Rule 710, Commands 711 and Cryptographic Keys 712-716. There could, of course, be a number of additional cryptographic keys stored with the message.

In this case, Rule 710 specifies that a user who agrees to pay a certain amount (or provide a certain amount of information) may view Stream 49, but all other users are required to view Stream 50, or a combination of Streams 49 and 50. In this case, Stream 49 may represent a movie or television program, while Stream 50 represents advertisements. In one embodiment, different portions of Stream 49 may be decrypted with different keys so that, for example, a first portion is decrypted with Key 712, a second portion is decrypted with Key 713, a third portion is decrypted with Key 714, and so on. Rule 710 may include all keys used to decrypt the entirety of Stream 49. When the user initially attempts to access the video encoded in Stream 49, Rule 710 could put up a

message asking if the user would prefer to use pay for view mode or advertising mode. If the user selects pay for view mode, Rule 710 could store (or transmit) the payment information, and pass Cryptographic Key 712 to Stream Controller 18. Stream Controller 18 could use Cryptographic Key 712 to decrypt the first stream until receipt of a header indicating that a different key is needed to decrypt the following set of packets. Upon request by Stream Controller 18, Control Block 13 would then check to determine that payment had been made, and then release Cryptographic Key 713, which would be used to decrypt the following packets, and so on. Rule 710 could additionally release Cryptographic Key 716, corresponding to Organization Stream 52, which corresponds to video without advertisements.

If, on the other hand, the user had chosen the advertising mode, Rule 710 could release Cryptographic Key 712 to Stream Controller 18 to allow decryption of Stream 49. Rule 710 could also authorize decryption of Stream 50 which contains the advertisements. Rule 710 could further release Cryptographic Key 715 to Organization Block 8. Cryptographic Key 715 matches Organization Stream 51. Organization Stream 51 references the video from Stream 49, but also references advertisements from Stream 50. Rule 710 would refuse to release Cryptographic Key 716, which corresponds to Organization Stream 52, which corresponds to the video without advertisements.

In operation, Control Block 13 could monitor information from Composite Block 11 over Control Line 16. That information could include the identity of each object actually rendered, as well as a start and stop time for the rendering. Control Block 13 could use this information to determine that an advertisement had actually been rendered, prior to releasing Cryptographic Key 713 for decryption of the second portion of video from Stream 49. This feedback loop allows Control Block 13 to be certain that the advertisements are not only being decrypted, but are also being displayed. This may be necessary because Composite Block 11 may be relatively unprotected, thereby allowing an unscrupulous user to remove advertisements before viewing.

A variety of additional relatively complex scenarios are possible. For example, rules from Control Block 13 could customize the programming for a particular geographic location or a particular type of viewer, by using information on the location or the viewer to control conditional decryption or use. This information could be stored in System 1 or entered by the user.

In another example, shown at row 5 of Array 717, Rule 719 may specify Budget 718, which may include information relating to the number of uses available to the user, the amount of money the user has to spend, etc. In operation, Rule 719 may require that

Budget 718 be securely stored and decremented each time a budgeted activity occurs (e.g., each time the associated work is played). Once the budget reaches zero, Rule 719 may specify that the work may no longer be played, or may display a message to the user indicating that the user may obtain additional budget by, for example, entering a credit card number or password, or contacting an external server.

In another example, a rule may control the ability of a user to copy a work to another device. The rule may, for example, specify that the user is authorized to use the governed work on more than one device, but with only one use being valid at any time. The rule may specify that an indication be securely stored regarding whether the user has "checked out" the work. If the user copies the work to another device (e.g., through Port 21), the rule may require that the work only be transmitted in encrypted form, and that the relevant control messages be transmitted along with it. The rule can further require that an indicator be securely set, and that the indicator be checked each time the user attempts to use or copy the work. If the indicator is set, the rule might require that the work not be decrypted or used, since the user only has the right to use the work on one device at a time, and the indicator establishes that the work is currently "checked out" to another device and has not been checked back in.

The receiving device may include the same type of indicator, and may allow the user to use the work only as long as the indicator is not set. If the user desires to use the work on the original device, the two devices may communicate, with the indicator being set in the second and reset in the first. This allows the work to be stored in two locations, but only used in one.

In another embodiment, the same result may be reached by copying the relevant control message from one device to the other, then erasing it from the original device. Because the control message includes keys used for decryption, this would insure that the work could only be used in one device at a time.

In one embodiment, this technique may be used to communicate digital media files (e.g., music, video, etc.) from a personal computer to a consumer electronics device without allowing the user to make multiple choices for simultaneous use. Thus, a larger, more sophisticated device (e.g., a personal computer), could download a file, then "check out" the file to a portable device lacking certain functions present in the personal computer (e.g., a hand-held music player).

Rules may also be used to specify that an initial user may transfer the file to another user, but only by giving up control over the file. Such rules could operate similarly to the

technique described above for transferring a file from one device to another, or could require that the original file be entirely erased from the original device after the transfer.

Rules in Control Block 13 may be added or updated through at least two channels. New rules may be obtained through Control Stream 6. If a control message contains an identifier corresponding to a control message already present in Control Block 13, that control message (including contained rules) may overwrite the original control message. A new rule may, for example, be identical to an existing rule, but with a new time stamp and new keys, thereby allowing decryption of a stream which had been encrypted with multiple keys. System 1 may be designed so that certain rules may not be overwritable. This may be enforced by designating certain positions in Array 717 as non-overwritable, or by providing a flag or other indicator to show that a particular rule cannot be overwritten or altered. This would allow for certain types of superdistribution models, including allowing a downstream distributor to add rules without allowing the downstream distributor to remove or alter the rules added by upstream distributors.

In addition, new rules may be encoded into Organization Stream 3, Audio Stream 4, or Video Stream 5, in the form of a watermark or steganographic encoding.

New rules may also be obtained through Port 21. Port 21 may connect to an external device (e.g., a smart card, portable memory, etc.) or may connect to an external network (e.g., External Server 30). Rules may be obtained through Port 21 either in an ad hoc manner, or as a result of requests sent by Control Block 13. For example, Control Message 14 (FIG. 7, row 6) may include a rule specifying that a new rule be downloaded from a particular URL, and used to govern Stream 1201.

Control messages, including rules, may be encoded using secure transmission formats such as DigiBoxes. A DigiBox is a secure container means for delivering a set of business rules, content description information, content decryption information and/or content validation information. One or more DigiBoxes can be placed into the headers of the media content or into data streams within the media.

FIG. 12 illustrates one embodiment of the DigiBox format and the manner in which that format is incorporated into a control message. Control Message 1201 is made up of Control Message Header 1202 and Control Message Contents 1203. As is described elsewhere, Control Message Header 1202 may include information used by Demux 7 (FIG. 1) to appropriately route the message to Control Block 13.

Control Message Contents 1203 of Control Message 1201 consists of DigiBox 1204, and may also include additional information. DigiBox 1204 consists of DigiBox Header 1205, Rules 1206 and Data 1207. Rules 1206 may include one or more rules. Data

1207 may include various types of data, including ES_ID 1208, Cryptographic Key 1209, and Validation Data 1210. Data 1207 may also include cryptographic information such as a specification of the encryption algorithm, chaining modes used with the algorithm, keys and initialization vectors used by the decryption and chaining.

Initialization vectors contained within Data 1207 are similar to cryptographic keys, in that they constitute input to the original encryption process and therefore are necessary for decryption. In one well-known prior art embodiment, the initialization vectors may be generated by starting with a base initialization vector (a 64 bit random number) and xor'ing in the frame number or start time for the content item.

Validation Data 1210 contained within Data 1207 may include cryptographic has or authentication values, cryptographic keys for calculating keyed authentication values (e.g., message authentication codes), digital signatures, and/or public key certificates used in validating digital certificates.

Thus, the DigiBox may incorporate the information described above as part of the control message, including the rules, the stream ID and the cryptographic keys and values.

In an alternative embodiment, DigiBox Header 1205 may be designed so that it can be read by Demux 7 and routed to Control Block 13. In such an embodiment, DigiBox 1204 would itself constitute the entirety of the control message, thus obviating the need to nest DigiBox 1204 within Control Message 1201.

Some or all of the contents of DigiBox 1204 will generally be encrypted. This may include Rules 1206, Data 1207, and possibly some or all of Header 1205. System 1 may be designed so that a DigiBox may only be decrypted (opened) in a protected environment such as IRP 22. In an alternate embodiment, Control Block 13 may directly incorporate the functionality of IRP 22, so that the DigiBox may be opened in Control Block 13 without the necessity of routing the DigiBox to IRP 22 for processing. In one embodiment, the cryptographic key used to decrypt DigiBox 1204 may be stored in IRP 22 (or Control Block 13), so that the DigiBox can only be opened in that protected environment.

Rules 1206 are rules governing access to or use of DigiBox Data 1207. In one embodiment, these rules do not directly control the governed streams. Since Cryptographic Key 1209 can only be accessed and used through compliance with Rules 1206, however, Rules 1206 in fact indirectly control the governed streams, since those streams can only be decrypted through use of the key, which can only be obtained in compliance with the rules. In another embodiment, Data 1207 may include additional rules, which may be extracted from the DigiBox and stored in a table such as Array 717 of FIG. 7.

The rules governing access to or use of a DigiBox may accompany the DigiBox, (as shown in FIG. 12) or may be separately transmitted, in which event Rules 1206 would contain a pointer or reference to the rules used to access Data 1207. Upon receipt of a DigiBox, Control Block 13 may receive rules separately through Control Stream 6, or may request and receive rules through Port 21.

Pipelined Implementation

One potential drawback to the system illustrated in FIG.1 consists of the fact that the system introduces complexity and feedback into a pipelined system designed to render content in real time. The rendering pipeline generally consists of Demux 7, Organization Block 8 and AV Block 9, Composite Block 11 and Rendering Device 12. Because content is received in a streamed fashion, and must be rendered in real time, pipelined processing must occur in a highly efficient manner, under tight time constraints. A failure to process within the time available may mean that output to Rendering Device 12 may be interrupted, or that incoming Bit Stream 2 may overflow available buffers, thereby causing the loss of some portion of the incoming data.

An alternative embodiment of System 1 is designed to address these problems, although at a possible cost in the ability to use standard system components and a possible cost in overall system security. This alternative embodiment is illustrated in FIG. 11, which shows System 1101.

System 1101 is similar to System 1 from FIG. 1 in many respects. It receives Bit Stream 1102, which consists of Organization Stream 1103, Audio Stream 1104, Video Stream 1105 and Control Stream 1106. These streams are received by Demux 1107, which passes Organization Stream 1103 to Organization Block and passes Audio Stream 1104 and Video Stream 1105 to AV Block 1109. Organization Block 1108 and AV Block 1109 operate similarly to their counterparts in FIG. 1, and pass information to Composite Block 1110, which organizes the information into a coherent whole and passes it to Rendering Device 1111. Streams sent to Organization Block 1108 are decrypted and/or validated by Stream Flow Controller 1112, and streams sent to AV Block 1109 are decrypted and/or validated by Stream Flow Controller 1113.

System 1101 differs from System 1, however, in that control and feedback are distributed, and integrated directly into the processing and rendering pipeline. System 1101 thus lacks a separate control block, and also lacks a feedback path back from the Composite Block 1110.

In System 1101, control is exercised directly at Organization Block 1108 and AV Block 1109. As in System 1, cryptographic keys are received through Control Stream 1106

(in an alternative embodiment, the keys could be incorporated directly into header or other information in Organization Stream 1103 or Audio/Video Streams 1104 and 1105). Those keys are included in a data format which includes information regarding the stream type of the encrypted content and, if multiple stream types are possible, an identifier for the particular controlled stream.

When Demux 1107 encounters a key in Control Stream 1106, it reads the information relating to the stream type, and routes the key to the appropriate stream flow controller. If Demux 1107 encounters a key designated for decryption or validation of Organization Stream 1103, for example, it routes that key to Stream Flow Controller 1112.

Stream Flow Controller 1112 stores received keys in Storage Location 1114. Storage Location 1114 stores the keys and also stores an indicator of the controlled stream ID.

Stream Flow Controller 1112 includes Cryptographic Engine 1115, which uses the received keys to decrypt and/or validate encrypted and/or protected portions of Organization Stream 1103. The keys may themselves be received in an encrypted manner, in order to provide some degree of security. In such a case, Stream Flow Controller may use a variety of techniques to decrypt the key, including using stored information as a key, or as a key seed. That stored information could, for example, constitute a "meta-key" provided earlier through Bit Stream 1102 or through a separate port.

Stream Flow Controller 1113, associated with AV Block 1109, contains a corresponding Storage Location 1116 and Cryptographic Engine 1117, and operates in a manner similar to the operation described for Stream Flow Controller 1112.

This implementation avoids the latency penalty which may be inherent in the necessity for communication between stream flow controllers and a separate control block.

This alternate implementation may also eliminate the feedback channel from the composite block (FIG.1, Control Line 16). This feedback channel may be used in order to insure that the content being passed from Composite Block 11 to Rendering Device 12 is content that has been authorized for rendering. In the alternate embodiment shown in FIG.11, this feedback channel does not exist. Instead, this implementation relies on the fact that Composite Block 1110 depends upon information from Organization Block 1108 to determine the exact structure of the information being sent to Rendering Device 1111. Composite Block 1110 cannot composite information in a manner contrary to the organization dictated by Organization Block 1108.

In one embodiment, this control by Organization Block 1108 may be sufficient to obviate the need for any feedback, since Organization Block 1108 may be designed so that

it accepts information only through Stream Controller 1112, and Stream Controller 1112 may be designed so that it only decrypts or validates information under the control of rules stored in Storage Location 1114.

In such an embodiment, security may be further increased by incorporating Secure Memory 1118 into Organization Block 1108. Secure Memory 1118 may store a copy or hash of the organization tree validly decrypted by Stream Controller 1112, and in current use in Main Organization Block Memory 1119. Organization Block 1108 may be used to periodically compare the organization tree stored in Main Organization Block Memory 1119 to the tree stored in Secure Memory 1118. If a discrepancy is spotted, this may indicate that an attacker has altered the organization tree stored in Main Organization Block 1119, thereby possibly allowing for the rendering of content in violation of applicable rules. Under such circumstances, Organization Block 1108 may be used to take protective measures, including replacing the contents of Main Organization Block Memory 1119 with the contents of Secure Memory 1118.

MPEG-4 Implementation

The generic system described above may be embodied in an MPEG-4 system, as illustrated in FIG. 8, which shows MPEG-4 System 801.

MPEG-4 System 801 accepts MPEG-4 Bit Stream 802 as input. MPEG-4 Bit Stream 802 includes BIFS Stream 803, OD Stream 804, Audio Stream 805, Video Stream 806 and IPMP Stream 807. These streams are passed to Demux 808, which examines header information and routes packets as appropriate, to BIFS 809, AVO 810, OD 811 or IPMP System 812.

IPMP System 812 receives IPMP messages through IPMP Stream 807. Those messages may include header information identifying the particular message, as well as an associated IPMP message. The IPMP message may include control information, which may include a cryptographic key, validation information, and/or may include complex governance rules, as are described above.

Stream Controllers 813, 814 and 815 act to decrypt, validate, and/or govern streams passed to BIFS 809, AVO 810 and OD 811, respectively.

OD 811 holds object descriptors, which contain metadata describing particular objects. This metadata includes an identifier of the particular Elementary Stream or streams which include the object, and may also include a pointer to a particular IPMP message which governs the object. Alternatively, the relationship between IPMP messages and particular objects or streams may be stored in a table or other form within IPMP System 812.

IPMP System 812 may exercise control over other functional blocks through Control Lines 816, 817, 818 and 819, each of which may transmit control/governance signals from IPMP System 812 and information or requests from other functional blocks to IPMP System 812. The information requests may include an ES_ID and a time stamp, which IPMP System 812 may use to determine which particular message (e.g., key) should be used and when.

In an alternative embodiment, IPMP System 812 may exercise control over Composite and Render 821 by receiving a hash of the currently valid BIFS tree (possibly through IPMP stream 807), and periodically checking the hash against the BIFS tree stored in BIFS 809. Because BIFS 809 controls the manner in which Composite and Render 821 renders information, if IPMP System 812 confirms that the current BIFS tree is the same as the authorized tree received through BIFS Stream 803, IPMP System 812 can confirm that the proper content is being rendered, even without receiving feedback directly from Composite and Render 821. This may be necessary, since BIFS 809 may communicate with Port 822, which may allow a user to insert information into BIFS 809, thereby creating a possibility that a user could insert an unauthorized BIFS tree and thereby gain unauthorized access to content.

When a stream controller receives encrypted or otherwise governed information, it may send the ES_ID and time stamp directly to IPMP System 812. Alternatively, it may send this information to OD 811, which may reply with the ID of the IPMP message which governs that object or stream. The stream controller can then use that IPMP message ID to request decryption, validation, and/or governance from IPMP System 812. Alternatively, OD 811 can pass the IPMP ID to IPMP System 812, which can initiate contact with the appropriate stream controller.

IPMP System 812 may obtain IPMP information through two channels other than IPMP Stream 807. The first of these channels is Port 820, which may be directly connected to a device or memory (e.g., a smart card, a DVD disk, etc.) or to an external network (e.g., the Internet). An IPMP message may contain a pointer to information obtainable through Port 812, such as a URL, address on a DVD disk, etc. That URL may contain specific controls needed by the IPMP message, or may contain ancillary required information, such as, for example, information relating to the budget of a particular user.

IPMP System 812 may also obtain IPMP information through OD updates contained in OD Stream 804. OD Stream 804 contains metadata identifying particular objects. A particular OD Message may take the format shown in FIG. 9. In this figure, OD Message 901 includes Header 902, which identifies the following packets as part of the OD

stream, and indicates the number of packets. OD Message 901 further consists of Message 903, which includes a series of Pointers 904 and associated Metadata 905. Each Pointer 904 identifies a particular Elementary Stream, and the associated metadata is applicable to that stream. Finally, OD Message 901 may contain an IPMP Pointer 906, which identifies a particular IPMP message.

In aggregate, the information contained in OD Message 901 constitutes an object descriptor, since it identifies and describes each elementary stream which makes up the object, and identifies the IPMP message which governs the object. OD Message 901 may be stored in OD 811, along with other messages, each constituting an object descriptor.

Object descriptors stored in OD 811 may be updated through OD Stream 804, which may pass through a new object descriptor corresponding to the same object. The new object descriptor then overwrites the existing object descriptor. This mechanism may be used to change the IPMP message which controls a particular object, by using a new object descriptor which is identical to the existing object descriptor, with the exception of the IPMP pointer.

OD Stream 804 can also carry IPMP_DescriptorUpdate messages. Each such message may have the same format as IPMP messages carried on the IPMP stream, including an IPMP ID and an IPMP message.

IPMP_DescriptorUpdate messages may be stored in a table or array in OD 811, or may be passed to IPMP System 812, where they may overwrite existing stored IPMP messages, or may add to the stored messages.

Since IPMP information may be separately conveyed through the OD stream or the IPMP stream, MPEG-4 System 801 may be designed so that it only accepts information through one or the other of these channels.

In another embodiment, the existence of the two channels may be used to allow multi-stage distribution, with governance added at later stages, but with no risk that later alterations may override governance added at an earlier stage.

Such a system is illustrated in FIG. 10. In this Figure, IPMP System 812 includes IPMP Table 1002, which has slots for 256 IPMP messages. This table stores the IPMP_ID implicitly, as the location at which the information is stored, shown in Column 1003. The IPMP message associated with IPMP_ID 4, for example, is stored at slot 4 of IPMP Table 1002.

Each location in IPMP Table 1002 includes Valid Indicator 1004 and Source Indicator 1005. Valid Indicator 1004 is set for a particular location when an IPMP message is stored at that location. This allows IPMP System 812 to identify slots which are

unfilled, which otherwise might be difficult, since at start-up the slots may be filled with random information. This also allows IPMP System 812 to identify messages which are no longer valid and which may be replaced. Valid Indicator 1004 may store time stamp information for the period during which the message is valid with IPMP System 812 determining validity by checking the stored time stamp information against the currently valid time.

Source Indicator 1005 is set based on whether the associated IPMP message was received from IPMP Stream 807 or from OD Stream 804.

These indicators allow IPMP System 812 to establish a hierarchy of messages, and to control the manner in which messages are added and updated. IPMP System 812 may be designed to evaluate the indicators for a particular location once a message is received corresponding to that location. If the valid indicator is set to invalid, IPMP System 812 may be designed to automatically write the IPMP message into that slot. If the valid indicator is set to valid, IPMP System 812 may then be designed to check the source indicator. If the source indicator indicates that the associated message was received through OD Stream 804, IPMP System 1812 may be designed to overwrite the existing message with the new message. If, however, the source indicator indicates that the associated message was received through IPMP Stream 807, IPMP System 812 may be designed to check the source of the new message. That check may be accomplished by examining the header associated with the new message, to determine if the new message was part of OD Stream 804 or part of IPMP Stream 807. Alternatively, IPMP System 812 may derive this information by determining whether the message was received directly from Demux 808 or through OD 811.

If the new message came through IPMP Stream 807, IPMP System 812 may be designed to store the new message in Table 1002, overwriting the existing message. If the new message came through OD Stream 804, on the other hand, IPMP System 812 may be designed to reject the new message.

This message hierarchy can be used to allow for a hierarchy of control. A studio, for example, may encode a movie in MPEG-4 format. The studio may store IPMP messages in the IPMP stream. Those messages may include a requirement that IPMP System 812 require that a trailer for another movie from the same studio be displayed prior to the display of the feature movie. IPMP System 812 could be used to monitor the beginning and end of rendering of the trailer (using feedback through Control Line 819) to ensure that the entire trailer plays, and that the user does not fast-forward through it.

The movie studio could encrypt the various elementary streams, including the IPMP stream. The movie studio could then provide the movie to a distributor, such as a cable channel. The movie studio could provide the distributor with a key enabling the distributor to decrypt the OD stream (or could leave the OD stream unencrypted), and the ability to insert new messages in that stream. The cable channel could, for example, include a rule in the OD stream specifying that the IPMP system check to determine if a user has paid for premium viewing, decrypt the movie if premium viewing has been paid for, but insert advertisements (and require that they be rendered) if premium viewing has not been paid for).

The cable channel would therefore have the ability to add its own rules into the MPEG-4 Bit Stream, but with no risk that the cable channel would eliminate or alter the rules used by the movie studio (e.g., by changing the trailer from a movie being promoted by the studio to a rival movie being promoted by the cable channel). The studio's rules could specify the types of new rules which would be allowed through the OD stream, thereby providing the studio a high degree of control.

This same mechanism could be used to allow superdistribution of content, possibly from one user to another. A user could be provided with a programming interface enabling the insertion of messages into the OD stream. A user might, for example, insert a message requiring that a payment of $1.00 be made to the user's account before the movie can be viewed. The user could then provide the movie to another user (or distribute it through a medium whereby copying is uncontrolled, such as the Internet), and still receive payment. Because the user's rules could not overrule the studio's rules, however, the studio could be certain that its rules would be observed. Those might include rules specifying the types of rules a user would be allowed to add (e.g., limiting the price for redistribution).

MPEG-4 System 801 may also be designed to include a particular type of IPMP system, which may be incompatible with IPMP systems that may be designed into other MPEG-4 systems. This may be possible because the MPEG-4 standard does not specify the format of the information contained in the IPMP stream, thereby allowing different content providers to encode information in differing manners.

IPMP System 812 in MPEG-4 System 801 may be designed for an environment in which differing IPMP formats exist. That system may scan the IPMP stream for headers that are compatible with IPMP System 812. All other headers (and associated packets) may be discarded. Such a mechanism would allow content providers to incorporate the same IPMP message in multiple formats, without any concern that encountering an unfamiliar format would cause an IPMP system to fail. In particular, IPMP headers can

incorporate an IPMP System Type Identifier. Those identifiers could be assigned by a central authority, to avoid the possibility that two incompatible systems might choose the same identifier.

IPMP System 801 might be designed to be compatible with multiple formats. In such a case, IPMP System 801 might scan headers to locate the first header containing an IPMP System Identifier compatible with IPMP System 801. IPMP System 801 could then select only headers corresponding to that IPMP System Identifier, discarding all other headers, including headers incorporating alternate IPMP System Identifiers also recognized by the IPMP system.

Such a design would allow a content provider to provide multiple formats, and to order them from most to least preferred, by including the most preferred format first, the second most preferred format second, and so on. Since IPMP System 801 locks onto the first compatible format it finds, this ordering in IPMP Stream 801 would insure that the IPMP system chose the format most desired by the content provider.

Even if different IPMP formats are used, content will probably be encoded (and encrypted) using a single algorithm, since sending multiple versions of content would impose a significant bandwidth burden. Thus, ordinarily it will be necessary for content to be encrypted using a recognized and common encryption scheme. One such scheme could use the DES algorithm in output feedback mode.

This method of screening IPMP headers, and locking onto a particular format may also be used to customize an MPEG-4 bit Stream for the functional capabilities of a particular MPEG-4 system. Systems capable of rendering MPEG-4 content may span a considerable range of functionality, from high-end home theaters to handheld devices. Governance options suitable for one type of system may be irrelevant to other systems.

For example, MPEG-4 System 801 may include a connection to the Internet through Port 820, whereas a second MPEG-4 system (for example a handheld Walkman-like device) may lack such a connection. A content provider might want to provide an option to a viewer, allowing the viewer to see content for free in return for providing information about the viewer. The content provider could insert a rule asking the user whether the user wants to view the content at a cost, or enter identification information. The rule could then send the information through a port to the Internet, to a URL specified in the rule. A site at that URL could then evaluate the user information, and download advertisements targeted to the particular user.

Although this might be a valuable option for a content provider, it obviously makes no sense for a device which is not necessarily connected to the Internet. It would make no

sense to present this option to the user of a non-connected device, since even if that user entered the information, the rule would have no way to provide the information to an external URL or download the advertisements. In such a case, the content provider might prefer to require that the user watch preselected ads contained in the original MPEG-4 bit stream.

Header information in the IPMP stream could be used to customize an MPEG-4 bit stream for particular devices. As with the IPMP System Type information, IPMP Header information could include MPEG-4 System Types. These could include 8 or 16-bit values, with particular features represented by bit maps. Thus, the presence of a bit at position 2, for example, could indicate that a device includes a persistent connection to the Internet.

An IPMP system could then evaluate the headers, and lock on to the first header describing functionality less than or equal to the functionality contained in the MPEG-4 device in which the IPMP system is embedded. If the header constituted a complete match for the functionality of the MPEG-4 device, the IPMP system could then cease looking. If the header constitutes less than a complete match (e.g., a header for a system which has an Internet connection, but lacks a digital output port, when the system includes both), the IPMP system can lock on to that header, but continue to scan for closer matches, locking on to a closer match if and when one is found.

The IPMP messages identified by a particular header would be those suited for the particular functionality of the MPEG-4 device, and would allow for customization of the MPEG-4 bit stream for that functionality. In the context of the example given above, the IPMP system for an MPEG-4 device containing an Internet connection would lock on to a particular header, and would download the IPMP messages characterized by that header. Those messages would prompt the user for information, would provide that information to the URL, and would authorize decryption and rendering of the movie, with the advertisements inserted at the appropriate spot.

In the case of an MPEG-4 device without an Internet connection, on the other hand, the IPMP system would lock onto a set of headers lacking the bit indicating an Internet connection, and would download the rules associated with that header. Those rules might not provide any option to the user. The rules might allow decryption of the content, but would also specify decryption of an additional ES from the MPEG-4 stream. That additional ES would contain the advertisements, and the IPMP system would require decryption and rendering of the advertisements, checking Control Line 819 to make certain that this had occurred. In the case of the system with the Internet connection, however, the rules allowing decryption and requiring rendering of the ES containing the advertisements

would never be loaded, since those rules would be contained within messages identified by the wrong type of header. The advertisement ES would therefore never be decrypted and would be ignored by the MPEG-4 device.

FIG. 21 illustrates one manner in which a protected MPEG-4 file may be created. In this figure, CreateBox 2101 represents a DigiBox creation utility, which accepts keys and rules. In one embodiment, CreateBox 2101 may pass these keys and rules to IRP 2102 and receive DigiBox 2103 from IRP 2102. In another embodiment, IRP 2102 may be incorporated into CreateBox 2101, which accepts keys and rules and outputs DigiBox 2103.

DigiBox 2103 contains governance rules, initialization vectors and keys. DigiBox 2103 is passed from CreateBox 2101 to Bif Encoder 2104. Bif Encoder 2104 may be conventional, with the exception that it is designed to accept and process DigiBoxes such as DigiBox 2103. Bif Encoder 2104 also accepts a .txt file containing a scene graph, and initial object descriptor commands.

Bif Encoder 2104 outputs a .bif file, containing the scene graph stream (in compressed binary form) and a .od file, containing the initial object descriptor commands, the object descriptor stream, and DigiBox 2103.

Bif Encoder 2104 passes the .bif file and the .od file to Mux 2105. Mux 2105 also accepts compressed audio and video files, as well as a .scr file that contains the stream description. Mux 2105 creates IPMP streams, descriptors and messages, encrypts the content streams, interleaves the received streams, and outputs Protected MPEG-4 Content File 2106, consisting of Initial Object Descriptor 2107 and Encrypted Content 2108. Initial Object Descriptor 2107 contains DigiBox 2103, as well as other information. Encrypted Content 2108 may include a scene graph stream (i.e., a BIFS stream), an object descriptor stream, IPMP streams, and encrypted content streams.

If DigiBox 2103 contains all keys and rules necessary to render all of the content, it may be unnecessary for Mux 2105 to create any IPMP streams. If additional keys or rules may be necessary for at least a portion of the content, Mux 2105 may incorporate those rules and keys into one or more additional DigiBoxes, and incorporate those DigiBoxes either in the IPMP stream or in the OD update stream.

FIG. 22 illustrates one manner in which control may be incorporated into an existing MPEG-4 stream. In this figure, Unprotected MPEG-4 Content File 2201 includes Initial Object Descriptor 2202 and Content 2203. The content may include a scene description stream (or BIF stream), an object descriptor stream, a video stream, an audio stream, and possibly additional content streams.

Unprotected MPEG-4 Content File 2201 is passed to Repackager 2204, which also accepts keys and rules. Repackager 2204 passes the keys and rules to IRP 2205, and receives DigiBox 2206 in return, containing keys, rules and initialization vectors. In an alternate embodiment, IRP 2205 may be incorporated directly into Repackager 2204.

Repackager 2204 demuxes Unprotected MPEG-4 Content File 2201. It inserts DigiBox 2206 into the Initial Object Descriptor and encrypts the various content streams. Repackager 2204 also adds the IPMP stream, if this is necessary (including if additional DigiBoxes are necessary).

Repackager 2204 outputs Protected MPEG-4 Content File 2207, consisting of Initial Object Descriptor 2208 (including DigiBox 2206) and Encrypted Content 2209 (consisting of various streams, including the IPMP streams, if necessary).

Real Networks Implementation

In one embodiment, the elements described above may be used in connection with information encoded in compliance with formats established by Real Networks, Inc.

The Real Networks file format (RMFF) is illustrated in FIG. 13. This format includes a block of headers at the beginning (Header 1301), followed by a collection of content packets (Content 1302), followed by an index used for seek and goto operations (Index 1303). Each file can contain several streams of different types. For each stream, there is a "Media Properties Header" (1304) used to describe the format of the media content (e.g., compression format) and provide stream specific information (e.g., parameters for the decompressor).

Real Networks streams can be protected by inserting a DigiBox into Header 1301 and encrypting the data packets contained in Content 1302. The altered format is illustrated in FIG.14, which shows Header 1401, including Media Properties Headers 1402 and 1403, which in turn contain DigiBoxes 1404 and 1405, respectively. The format also includes encrypted Content 1406 and Index 1407.

In one embodiment, the declared type of the data is changed from the standard Real Networks format to a new type (e.g., RNWK_Protected.) The old type is then saved. Changing the type forces the Real Networks player to load a "Trust Plugin," since this Plugin is registered as the only decoder module that can process streams of type "RNWK-Protected." The Trust Plugin opens the DigiBox, gets approval from the user, if it is needed, determines the original content type, loads a decoder plugin for the original content, and then decrypts and/or validates the content, passing it to the content decoder plugin to be decompressed and presented to the user.

In one embodiment, the specific alterations made to the Real Networks file format are the following:

- Increase the preroll time to force larger buffers on playback. In a current embodiment, an increase of 3 seconds is used. Larger buffers are needed because of the extra steps needed to decrypt the content.
- Modify each stream-specific header by changing the mime type to "RNWK-Protected", saving the old mime type in the decoder specific information and adding a content identifier and DigiBox to the decoder specific information. The DigiBox contains the key, initialization vector (IV), version information, and watermarking instructions. The key, IV and content identifier are generated automatically, or can be provided as command-line parameters. The same key, IV and content identifier are used for every stream.
- Content packets are selectively encrypted. In one embodiment, content packets whose start time in milliseconds is in the first half-second of each 5 seconds (i.e., starttime % 5000 < 500) are encrypted. This encrypts approximately one-tenth of the content reducing encryption and decryption costs, and damages the content, sufficiently to prevent resale. The encryption algorithm can be DES using output-feedback mode or any similar algorithm. The initialization vector is computed for each packet by xoring the stream's IV with the packet's start time in milliseconds. Some information unique to the stream should also be xored into the IV. In one embodiment, the same IV is used for multiple packets whenever two or more streams have packets with the same start time. This usually happens for the first packet in each stream since they usually have start time 0. Other than the first packet, it is rare to have two packets have the same start time.

In one embodiment, these changes to the Real Networks file format are accomplished as is shown in FIG. 15. As is illustrated, RMFF file 1501 is formatted in the standard Real Networks RMFF format. This file is passed to Packager 1502. Also passed to Packager 1502 is Rights File 1503. Packager 1503 generates Protected RMFF File 1504, which includes various alterations as described above and as listed in FIG. 15, including the incorporation of one or more DigiBoxes in the header, encryption of the content, modification of the mime type, etc.

In one embodiment, the trust plugin described above is illustrated in FIGs. 16 and 17. FIG. 16 illustrates the standard Real Networks architecture. File 1601 (e.g., a streaming audio file in Real Networks format) is provided to Real Networks G2 Client

Core 1602. File 1601 may be provided to RealNetworks G2 Client Core 1602 from Server 1603, or through Direct Connection 1604.

Upon receipt of File 1601, Real Networks G2 Client Core 1602 accesses a rendering plugin appropriate to File 1601, based on information which is obtained from the header associated with File 1601. Rendering Plugins 1605 and 1606 are shown. If File 1601 is of a type which cannot be rendered by either Rendering Plugin 1605 or Rendering Plugin 1606, Real Networks G2 Client Core 1602 may attempt to access an appropriate plugin, e.g., by asking for the user's assistance or by accessing a site associated with the particular file type.

Rendering Plug-In 1605 or 1606 processes File 1601 in a conventional manner. This processing most likely includes decompression of File 1601, and may include other types of processing useful for rendering the content. Once this processing is complete (keeping in mind that the content is streamed, so that processing may be occurring on one set of packets at the same time that another set of packets is being rendered), File 1601 is passed back to Real Networks G2 Client Core 1602, which then passes the information to Rendering Device 1607. Rendering Device 1607 may, for example, be a set of stereo speakers, a television receiver, etc.

FIG. 17 illustrates the manner in which a trust plugin operates within the overall Real Networks architecture. Much of the architecture illustrated in FIG. 17 is the same as that illustrated in FIG. 16. Thus, File 1701 is provided to Real Networks G2 Client Core 1702 through Server 1703 or through Direct Connection 1704. The file is processed by Real Networks G2 Client Core 1702, using plugins, including Rendering Plugins 1705 and 1706, and is then passed to Rendering Device 1707.

FIG. 17 differs from FIG. 16 in its incorporation of Trust Plugins 1708 and 1709, and IRP 1710. When initially registered with Real Networks G2 Client Core 1702, Trust Plugins 1708 and 1709 inform Real Networks G2 Client Core 1702 that they can process content of type RNWK-Protected. Whenever Real Networks G2 Client Core 1702 encounters a stream of this type, it is then enabled to create an instance of the trust plugin to process the stream, e.g., Trust Plugin 1708. It then passes the stream to the trust plugin.

The stream passed to Trust Plugin 1708 may be in the format shown in FIG. 14. In such a case, Trust Plugin 1708 extracts DigiBox 1404 from Media Properties Header 1402. It also extracts the content id and original mime type from Media Properties Header 1402. The Trust Plugin first checks to see if any other stream with the same content identifier has been opened. If so, then DigiBox 1404 is not processed further. Instead, the key and IV from the box for this other stream are used. This avoids the time cost of opening a second

box. Also, this ensures that a user is only asked to pay once even if there are multiple protected streams. By sharing content ids, keys, and IVs, several files can be played with the user only paying once. This is useful when SMIL is used to play several RMFF files as a single presentation.

In an alternate and possibly more secure embodiment, this check is not performed, and the key and IV from the current DigiBox are used even if another stream with the content identifier has already been opened.

If no other stream has been identified with the same content identifier, Trust Plugin 1708 passes DigiBox 1404 to IRP 1710. IRP 1710 may be a software process running on the same computer as Real Networks G2 Client Core and Trust Plugin 1708. IRP 1710 may run in a protected environment or may incorporate tamper resistance techniques designed to render IRP 1710 resistant to attack.

IRP 1708 may process DigiBox 1404 and extract a cryptographic key and an IV, which may then be passed to Trust Plugin 1708. Trust Plugin 1708 may then use this information to decrypt Encrypted Contents 1406.

Trust Plugin 1708 uses the original mime type information extracted from Media Properties Header 1402 to create an instance of the rendering plugin to be used for the content (e.g., Rendering Plugin 1705). Once this is done, Trust Plugin 1708 behaves like an ordinary rendering plugin to the Real Networks G2 Client Core 1702, in that Real Networks G2 Client Core 1702 passes streamed information to Trust Plugin 1708, which decrypts that information and passes it to Rendering Plugin 1705. From the perspective of Real Networks G2 Client Core 1702, Trust Plugin 1708 constitutes the appropriate rendering pluin, and the core is not aware that the information is being passed by Trust Plugin 1708 to a second plugin (e.g., Rendering Plugin 1705).

Similarly, from the point of view of Rendering Plugin 1705, Trust Plugin 1708 behaves like Real Networks G2 Client Core 1702. Thus although Rendering Plugin 1705 receives decrypted stream information from Trust Plugin 1708, Rendering Plugin 1705 operates exactly as if the information had been received directly from Real Networks G2 Client Core 1702. In this manner, content formatted for Rendering Plugin 1705 may instead be first processed by Trust Plugin 1708, without requiring any alteration to Real Networks G2 Client Core 1702 or Rendering Plugin 1705.

Trust Plugin 1708 may also perform other processing that may be helpful for security purposes. For example, Trust Plugin 1708 may watermark the decrypted file prior to passing it to Rendering Plugin 1705, keeping in mind that the watermark algorithm must be such that it will survive decompression of the file by Rendering Plugin 1705.

MP3 Embodiment

The techniques described above can also be applied to MP3 streaming content.

The MP-3 specification does not define a standard file format, but does define a bit stream, which is illustrated in FIG.18. In FIG. 18, MP-3 Bit Stream 1801 includes Content 1802. Content 1802 is divided into frames, shown as Frame 1803, Frame 1804 and Frame 1805. The dots between Frame 1804 and 1805 symbolize the fact that Content 1802 may include a large number of frames.

Each frame includes its own small header, shown in FIG. 18 as Headers 1806, 1807 and 1808.

Many MP3 players support a small trailer defined by the ID3 V1 specification, shown as Trailer 1809. This is a 128 byte trailer for carrying fields like artist, title and year, shown as Fields 1810, 1811 and 1812. The ID3 V1 trailer is ignored by players not designed to read such trailers, since it does not appear to be valid MP3 data.

FIG. 19 shows one embodiment of protection applied to the MP3 format. This protected format constitutes File 1908 and includes the following items:

- Unencrypted MP3 Content 1912. This is the first information encountered by a player, and will be rendered by any standard MP3 player. It can include a message to the user indicating that the content is protected and providing instructions as to how the content can be accessed (e.g., a URL for a trust plugin, instructions on payment mechanisms, etc.) Unencrypted MP3 Content 1912 may include a "teaser," consisting of an initial portion of the content (e.g., 30 seconds), which is rendered at no cost, thereby allowing a user to sample the content prior to making a decision to purchase it.
- Encrypted MP-3 Content 1901, which may include thousands of MP-3 frames. In one embodiment, the first eight frames out of every 32 frames are encrypted. Thus, one-quareter of the frames are rendered unuseable unless a player is able to decrypt them. In practice, this may render the content un-sellable or unuseable, without imposing excessive encryption or decryption costs. To further reduce encryption and decryption costs, only 32 bytes in each frame are encrypted. In a current embodiment, these are the first 32 bytes after the header and CRC information. In a different embodiment, a different 32 bytes may be encrypted in every frame. In a current embodiment, the content is encrypted with the DES using algorithm output-feedback mode. The initial IV for the file is randomly generated and then xored with the frame number to generate a unique IV for each frame.

Many alternate embodiments may exist, including encrypting more or less information, and using different encryption algorithms.

- ID3 V1 Trailer 1902, including 128 bytes.

- Content ID 1903, including 16 bytes. This is used by the player application to avoid opening DigiBoxes which it has already opened.
- DigiBox 1904, which may comprise approximately 18K bytes. It includes Key 1909, IV 1910 and Watermarking Instructions 1911. Watermarking Instructions 1911 may be used in a process of watermarking the associated content.
- Address 1905, which contains the address in the file of Content ID 1903 and consists of 4 bytes.
- Trust ID 1906, which identifies this trusted MP-3 file and consists of 16 bytes.
- ID3 V1 Trailer 1907, which is a copy of Trailer 1902.

A conventional MP3 player encountering File 1908 would be unable to render Content 1901, since at least a portion of that content is encrypted. Such a player would most likely read through to Trailer 1902 and cease processing at that point. A conventional player looking for the ID3 trailer information will seek to the end and find it.

FIG. 20 illustrates one embodiment of an MP3 player designed to process and render protected content. This figure shows MP3 Player 2001, which includes Buffer 2006 and Decompressor 2007, and renders content to Rendering Device 2008. In one embodiment, this is a modified version of a player distributed by Sonique.

Player 2001 obtains Protected MP3 File 2002 through any standard interface. Protected MP3 File 2002 may have the format illustrated in FIG. 19.

When Player 2001 is asked to play Protected MP3 File 2002, Player 2001 first calls Trust Plug-In 2003, which includes Approval Function 2009 and Decrypt Function 2005. Trust Plugin 2003 calls Approval Function 2009 to determine if Protected MP3 File 2002 is protected and whether authorization exists to play the file. Approval Function 2009 is first given a pointer to Protected MP3 File 2002. It then checks Protected MP3 File 2002 for the presence of Trust ID 1906. If Trust ID 1906 is not found, Approval Function 2009 returns an indicator that the file is not protected. Player 2001 then proceeds to render the file as a normal MP3 file.

If Trust ID 1906 is found, Approval Function 2009 checks Content ID 1903 to see if it matches the Content ID of a file that has already been opened.

If Protected MP3 File 2002 has not been previously opened, DigiBox 1904 is retrieved by Approval Function 2009, and is passed to IRP 2004, which may include software running in a protected environment, or incorporating tamper resistance. IRP 2004 attempts to open DigiBox 1904 in compliance with the rules associated with that DigiBox. One such rule may require, for example, that the user indicate assent to pay for use of the content. If DigiBox 1904 cannot be opened (e.g., the user refuses to pay) a value is

returned to Approval Function 2009 indicating that the file is protected and may not be played.

If DigiBox 1904 is opened in compliance with applicable rules, the key and IV are retrieved and passed to Decrypt Function 2005. The key and IV are stored with the content id for later re-use and Decrypt Function 2005 is initialized. This may improve overall system performance, since it reduces the number of times a DigiBox must be opened. Each such action may introduce significant latency.

On the other hand, storing this information in unprotected memory may reduce overall system security. Security may be enhanced either by not storing this information (thereby requiring that each DigiBox be opened, even if the corresponding file has already been opened through another DigiBox), or by storing this information in a protected form or in a secure location.

The stored key, IV and content id are referenced when Approval Function 2009 first checks Content ID 1903 to determine if it matches the Content ID of an already opened file. If the new Content ID matches a stored Content ID, Decrypt Function 2005 is reinitialized using the stored key and IV corresponding to the matching content id an value indicating that this is a protected file for which play is authorized is returned to Approval Function 2009.

Once Protected MP3 File 2002 has been opened, each time Player 2001 needs a packet, Player 2001 reads it into Buffer 2006, strips off the header and CRC and passes the remaining data and a frame number to Decrypt Function 2005, which decrypts the frame if necessary, and returns it to Player 2001.

In a current embodiment, although audio content is encrypted, headers or trailers are not encrypted. This allows the Player 2001 to process information in headers or trailers without intervention from Approval Function 2009 or Decrypt Function 2005. This allows Player 2001 to place information such as playing time, artist and title into a playlist display, and initialize Decompressor 2007, without any action required from Trust Plugin 2003.

Commerce Appliance Embodiment

This section will describe an embodiment, comprising a Commerce Appliance architecture designed to allow persistent control of digital works in consumer electronics devices. Although this is described as a separate embodiment, it should be understood that the features of this embodiment may be combined with, or supplant, the features of any of the embodiments provided elsewhere in this description.

In one embodiment, this section will describe modifications to the MPEG-4 standard designed to support the association of persistent rules and controls with MPEG-4

content, as well as elements necessary for a Commerce Appliance to use such content. This is intended, however, merely as an example.

In one embodiment, shown in FIG. 23, each Commerce Appliance 2301 includes a CMPS ("Content Management and Protection System") 2302. Each CMPS is responsible for governing the use of controlled content, including decrypting the content and ensuring that the content is only used as permitted by associated rules.

Each governed digital work is associated with one or more CMPOs (Content Management Protection Object), e.g., CMPOs 2303. Each CMPO may specify rules governing the use of the digital work, and may include keys used to decrypt the work.

CMPOs may be organized in an hierarchical fashion. In one embodiment, a content aggregator (e.g., a cable channel, a web site, etc.) may specify a Channel CMPO ("CCMPO") used to associate certain global rules with all content present on that channel. Each independent work may in turn have an associated Master CMPO ("MCMPO") used to associate rules applicable to the work as a whole. Each object (or Elementary Stream, in MPEG-4) may have associated with it a CMPO containing rules governing the particular object.

In one exemplary application, Commerce Appliance 2301 may be an MPEG-4 player containing CMPS 2302. Upon receipt of a user command to play a particular work, CMPS 2302 may download a MCMPO associated with the work and obtain rules, which may include conditions required for decryption and viewing of the work. If the rules are satisfied, CMPS 2302 may use keys from the MCMPO to decrypt any Elementary Streams ("ES"), and may pass the decrypted ESs into the buffers. Composition and rendering of the MPEG-4 work may thereafter proceeds according to the MPEG-4 standard, except that any storage location or bus which may contain the work in the clear must be secure, and CMPS 2302 may have the ability to govern downstream processing, as well as to obtain information regarding which AVOs were actually released for viewing.

In a variation, the process of obtaining and governing the work may include downloading a CCMPO which applies rules governing this and other works. If rules contained in the CCMPO are satisfied, CMPS 2302 may obtain a key used to decrypt the MCMPO associated with the particular work to be viewed.

In another variation, a CMPO may be associated with each ES. In this variation, the MCMPO supplies one or more keys for decryption of each CMPO, and each CMPO may in turn supply a key for decryption of the associated ES.

Commerce Appliance 2301 is a content-rendering device which includes the capability of supporting distributed, peer management of content related rights by securely

applying rules and controls to govern the use of content. Commerce Appliance 2301 may include general-purpose functions devoted to acquisition and managed rendering of content (e.g., a DVD (and/or any other optical disk format) player is able to play a DVD (and/or any other optical disk format) disk and output content to a television.) Commerce Appliance 2301 may make use of any of the means for protecting and using digital content on high capacity optical disk, in one non-limiting example, a DVD disk, as described in the aforementioned Shear patent application.

Commerce Appliance 2301 also includes special-purpose functions relating to other management and protection of content functions. These special-purpose functions may be supported by one or more embedded or otherwise included CMPS 2302 in the form of a single CMPS or a cooperative CMPS arrangement, and may include a user interface (e.g., User Interface 2304) designed to display control-related information to the user and/or to receive control-related information and directions from the user. Commerce Appliance 2301 may also be designed so that it is networkable with other Commerce Appliances (e.g., a set-top box connected to a DVD player and a digital television) and/or with other devices, such as a computer arrangement, which may also include one or more CMPSs.

An important form of Commerce Appliance specifically anticipates secure coupling on a periodic or continual fashion with a computer managed docking environment (e.g., a standalone computer or other computer managed device which itself may be a Commerce Appliance) where the one or more CMPSs of the Commerce Appliance interoperate with the docking environment to form a single user arrangement whose performance of certain functions and/or certain content usage events is enabled by such inter-operation through, at least in part, cooperation between CMPSs and content usage management information of the Commerce Appliance and the trust environment capabilities of the docking environment, (e.g., further one or more CMPSs and content usage management information, such as, for example, information provided by use of CI).

An exemplary Commerce Appliance may be designed to comply with the emerging MPEG-4 standard for the formatting, multiplexing, transmission, compositing, and rendering of video and other types of information.

Commerce Appliance 2301 may be any computing device, one non-limiting example of which is a Personal Computer (PC) that includes MPEG-4 software (and/or hardware) for rendering content. In accordance with the present invention, the PC may also use one or more CMPSs as described herein.

The commerce appliance function is not restricted to streamed channel content but may include various browser-type applications consisting of aggregated composite content

such as still imagery, text, synthetic and natural video and audio and functional content such as applets, animation models and so on. these devices include browsers, set-top boxes, etc.

Content Management and Protection System (CMPS)

Each commerce appliance includes one or more CMPS (e.g., CMPS 2302). The CMPS is responsible for invocation and application of rules and controls, including the use of rules and controls to govern the manner in which controlled content is used.

Particular functions of CMPS 2302 include the following:

(a) Identification and interpretation of rules.

CMPS 2302 must determine which rules are to be applied, and must determine how those rules are to be interpreted in light of existing state information. In one embodiment, this requires that CMPS 2302 obtain and decrypt one or more CMPOs 2303 associated with a work.

(b) Identification of content associated with particular rules.

CMPS 2302 must determine which content is governed by particular one or more rules. This may be accomplished by obtaining information from one or more CMPOs 2303 and/or other CI. In one embodiment, a CCMPO may identify a set of works, a MCMPO may identify a particular work and a CMPO may identify a particular ES or Audio Visual Object ("AVO").

(c) Decryption of content as allowed by the rules.

CMPS 2302 may be designed so that all content is routed through CMPS 2302 for decryption, prior to reinsertion into the data flow required by the relevant standard. In the case of MPEG-4, for example, the output from Demux 2305 may be fed into CMPS 2302. CMPS 2302 may then decrypt the content and, if relevant rules and controls are satisfied, feed the content into the MPEG-4 buffers. From that point, the data flow associated with the content may be as described by MPEG-4.

(d) Control of content based on rules.

CMPS 2302 may be used to control usage of content after the initial decryption, for example, through the use of secure event management as described in the incorporated Ginter '333 patent application. In the case of MPEG-4 systems, this may require that CMPS 2302 exercise control over hardware and/or software which performs the following functions: demuxing (performed by Demux 2305), decompression/buffering/decode into AVOs (performed by Scene Descriptor Graph 2306, AVO Decode 2307 and Object Descriptors 2308), scene rendering (performed in Composite and Render 2309).

CMPS 2302 may also be used to control use and consequences according to: (1) generational copy protection rules such as the CGMS and/or SGMS standards; (2) various Conditional Access control methods, such as those proposed and/or implemented by NDS as described in MPEG-4 document M2959, DAVIC "Copyright Control Framework" document, and in other publications; (3) a Rights Management Language, such as those proposed in the Ginter '333 patent application and/or as described by U.S. Patent No. 5,638, 443 to Stefik, et al.; (4) use policies described in accordance with AT&T's Policy Maker, as described by Blaze, Feigenbaum, and Lacy; (5) the CCI layer bits for IEEE 1394 serial bus transmission as specified by the DTDG subgroup of the DVD Copy Protection Technical Working Group and/or as implemented by the Hitachi, Intel, Matsushita, Sony and Toshiba proposed standard (hereafter "the five company proposal"); (6) controls transmitted using any secure container technology such as, for example, IBM Cryptolope; (7) any other means for specifying use rules and consequences.

(e) Monitoring use of content.

CMPS 2302 may be used to monitor content to: (i) ensure that rules are being complied with; (ii) ensure that no attempts are being made to tamper with the system or protected content; and (iii) record information used by rules, including usage information needed for payment purposes.

(f) Updating user budgets.

CMPS 2302 may be used to update user or other budgets to reflect usage.

(g) Exhaust information.

CMPS 2302 may be used to output payment and usage information ("exhaust information") to external processes, including one or more Commerce Utility Systems.

(h) Hardware identification and configuration.

(i) Obtaining new, additional, and/or augmented rules from an external process, one non-limiting example of which is a Rights and Permission Clearinghouse as described in the incorporated Shear patent application.

(j) Receiving keys, digital credentials, such as certificates, and/or administrative information, from certifying authorities, deployment managers, clearinghouses, and/or other trusted infrastructure services.

(k) Securely sending and/or receiving user and/or appliance profiling and/or attribute information.

(l) Securely identifying a user or a member of a class of users who requests content and/or CMPO and/or CMPS usage.

(m) Securely certifying or otherwise guaranteeing the authenticity of application code, for example certifying within CMPO 2301 and/or CMPS 2302 that application code containing rules and/or other application information, such as information written in Java code for conditional execution within a Commerce Appliance, and/or that executes at least in part outside of CMPO 2301 and/or CMPS 2302, has not been altered and/or has been delivered by a guaranteed (e.g., trusted) party.

(n) Securely processing independently delivered CI, such as described in the incorporated Ginter '333 patent application, to perform content usage control that protects the rights of plural, independent parties in a commerce value chain.

(o) Securely performing watermarking (including, for example fingerprinting) functions, for example as described in the Ginter '333 patent application and as incorporated herein, for example including interpreting watermarking information to control content usage and/or to issue an event message, wherein such event message may be reported back to a remote authority, such as, for example, a MCMPO rights clearinghouse management location.

CMPS 2302 may be used to identify and record the current hardware configuration of the Commerce Appliance and any connected devices (e.g., which loudspeakers are available, identification of attached monitors, including whether particular monitors have digital output ports, etc.) If attached devices (such as loudspeakers) also include CMPSs, the CMPSs may be used to communicate for purposes of coordination (e.g., a CMPS in a set-top box and/or loudspeaker arrangement may communicate with a CMPS in a downstream digital television or other display device to establish which CMPS will be responsible for governance or the nature of cooperative governance through a virtual rights process, said process optionally involving a rights authority server that may find, locate, provide, aggregate, distribute, and/or manage rights processes, such as described in the aforementioned Shear patent application, for employing plural CMPSs, for example, for a single user content processing and usage arrangement).

The present invention includes arrangements comprising plural Commerce Appliances and/or CMPSs in one or more user locations, non-limiting examples of which include a home, apartment, loft, office, and/or vehicle, such as a car, truck, sports utility vehicle, boat, ship, or airplane, that may communicate among themselves at least occasionally and may comprise a virtual network that operates in a logically cooperative manner, through at least in part the use of such CMPSs, to ensure optimal commercial flexibility and efficiency and the enforcement of rights of commerce value chain participants, including financial and copyright rights of providers, infrastructure rights of

appliance providers, societal rights of government and/or societal bodies, and privacy rights of all parties, including consumers. Information related to interaction among such a network of value chain participants, including content usage auditing, content usage consequence, and CI specification, can be securely, variably reported to parties having right to such information, through, at least in part, use of such CMPSs, for example, as described in the aforementioned Ginter '712 patent application regarding the information reporting functioning of VDE nodes.

In one embodiment, shown in FIG. 24, CMPS 2401 consists of special-purpose hardware and resident software or firmware. These include the following:

(a) One or more processors or microcontrollers e.g. CPU 2402. CPU 2402 controls the overall processing of CMPS 2401, including execution of any necessary software.

(b) One or more external communications ports, e.g., Port 2403. Port 2403 communicates with External Network 2404, which may include LANs, WANs or distributed networks such as the Internet. External communications ports may also include one or more IEEE 1394 serial bus interfaces.

(c) Memory 2405. Types of memories which may be included in Memory 2405-- and examples of the information they may store -- are the following:

i. ROM 2406. ROM 2406 may include any information which is permanently stored in CMPS 2401, such as (1) CMPS Operating System 2407 and/or CMPS BIOS 2408, (2) Rules/Controls 2409 which are permanently stored in the CMPS; (3) Control Primitives 2410 which may be used to build rules or controls; (4) Keys 2411 associated with the CMPS, including a Public/Private Key Pair; (5) one or more Certificates 2412 designed to identify CMPS 2401 and/or the device, including version information; (6) Hardware Signature Information 2413 used to check for tampering (e.g., a hashed signature reflecting the expected hardware state of the device).

ii. RAM 2414. RAM 2414 may hold current state information needed by CMPS 2401, as well as information temporarily stored by CMPS 2401 for later use. Information stored in RAM 2414 may include the following: (1) Software 2415 currently executing in CPU 2402; (2) CMPOs 2416 which are currently active; (3) Content Object Identification 2417 of those content objects which are currently active (in an MPEG 4 system this would constitute, for example, an identification of active AVOs); (4) Rules 2418 which are currently active; (5) State Information 2419 regarding the current state of use of content, including an identification of any higher-order organization (in an MPEG-4 system this would constitute an identification of the scene descriptor tree and the current

state of composition and rendering); (6) Stored Exhaust Information 2420 relating to use and/or the user, designed for external transmission; (7) Updated Budget Information 2421; (8) Content 2422; (9) Active Content Class Information 2423; and (10) Active User Identification 2424, including identification characteristic information.

iii. NVRAM 2425 (e.g., flash memory). This type of memory may hold information which is persistent but changeable, including at least some: (1) Budget Information 2426; (2) User Information 2427, such as identification, credit card numbers; preferred clearinghouses and other Commerce Utility Systems; (3) User Preferences 2428, such as preferences, profiles, and/or attribute information; and (4) Appliance Information 2429, such as attribution and/or state information.

The types of information described above and stored in CMPS Memory 2405 may be stored in alternative of the above memory types, for example, certain budget information may be located in ROM, information regarding specific one or more clearinghouses may be stored in ROM, certain active information may be moved into NVRAM, etc.

Budget information may include stored budgets made up of, for example:

(1) electronic cash;

(2) pre-authorized uses (e.g., based on a prepayment, the user has the right to watch 12 hours of programming).

(3) Security budgets related to patterns reflecting abnormal and/or unauthorized usage, for example, as described in the incorporated Shear patent, wherein such budgets restrict and/or report certain cumulative usage conduct.

(4) electronic credit, including credit resulting from usage events such as attention to promotional material and/or the playing of multiple works from one or more classes of works (e.g., certain publisher's works) triggering a credit or cash refund event and/or a discount on future playing of one or more of such publisher's works, such as other works provided by such publisher.

User information may include the following types of information for one or more authorized users of the Commerce Appliance:

(1) Name, address, telephone number, social security number or other identifier

(2) Information used to authenticate the user, which may include a user selected password and/or biometric data, such as fingerprints, retinal data, etc.

(3) User public/private key pair

(4) User attribute and/or profiling information.

iv. Removable Memory 2430. This may include any type of removable memory storage device, such as smart cards, floppy disks or DVD disks. If the commerce appliance is designed to play content received on removable memory devices (e.g., a DVD player), that capability may be used for purposes of the CMPS.

Memory 2405 may include a protected database, in which certain control, budget, audit, security, and/or cryptographic information is stored in secure memory, with complete information stored in an encrypted fashion in unsecure memory.

(d) Encryption/Decryption Engine 2431. CMPS 2401 must include a facility for decrypting received information, including content and CMPOs and/or other. CMPS 2401 may also include a facility for encrypting information if such information is to be transmitted outside the secure boundaries of CMPS 2401. This may include exhaust sent to clearinghouses or other external repositories; and content sent across unsecured buses for usage, such as content sent across IEEE 1394 Serial Bus 2432 to a computer central processing arrangement or to a viewing device such as a monitor, wherein a receiving CMPS may be employed to control such content's usage, including, for example, decrypting such content, as appropriate. Encryption/Decryption Engine 2431 may include a Random Number Generator 2433 used for the creation of keys or key pairs that can be used to identify and assure the uniqueness of CMPSs and support the opening of secure communication channels between such secure content control secure encryption/decryption arrangements.

(e) Secure Clock/Calendar 2434. CMPS 2401 may include Secure Clock/Calendar 2434 designed to provide absolute information regarding the date and time of day, information regarding elapsed absolute time, and/or relative timing information used to determine the elapsed time of operations performed by the system. Secure Clock/Calendar 2434 may include Battery Back Up 2435. It may further include Sync Mechanism 2436 for synchronization with outside timing information, used to recover the correct time in the event of a power loss, and/or to check for tampering.

(f) Interface 2437 to blocks used for content rendering and display. This interface is used for controlling rendering and display, based on rules, and for obtaining feedback information, which may be used for budgeting purposes or for providing information to outside servers (e.g., information on which content was actually displayed, which choices the user invoked, etc.) In the case of an MPEG-4 player such as is shown in

FIG. 23, this may include control over Commerce Appliance circuitry which handles, for example, buffering, the scene descriptor graph, AVO decode, object descriptors and composite and rendering (e.g., Control Lines 2310, 2311 and 2312).

Feedback Path 2313 from Composite and Render block 2309 may allow CMPS 2302 to determine whether and when content has actually been released to the viewer. For example, Composite and Render block 2309 can issue a start event to CMPS 2302 when an AVO object is released for viewing, and can issue a stop event to CMPS 2302 when the AVO object is no longer being viewed.

Feedback from Composite and Render block 2309 may also be used to detect tampering, by allowing CMPS 2302 to match the identification of the objects actually released for viewing with the identification of the objects authorized for release. Start and end time may also be compared with the expected elapsed time, with a mismatch possibly indicative of the occurrence of an unauthorized event.

In one embodiment, the following protocol may be used for feedback data:

start <id>, T, <instance number><clock time><rendering options>

Sent if elementary stream <id> is reachable in the SD-graph at time *T*, but not at time *T-1*.

end <id>, T, <instance number><clock time><rendering options>

T constitutes presentation time, clock time constitutes the wall clock time, including day and date information, and rendering options may include such information as QoS and rate of play (e.g., fast forward).

Sent if elementary stream **<id>** is reachable in the SD-graph at time *T-1* but not at time *T*. A SD-graph stream is reachable if, during traversal of the SD-graph for display update, the renderer encounters a node that the SD-graph update stream <id> created or modified. This implies that all nodes in the tree need an update history list. This list need not be as large as the number of streams. Further, it can be labeled to indicate if the CMPS will be watching for stream, if not labeled it will not record them. An AV elementary stream is reachable if the stream's content was rendered.

For SD-graph update streams, the object instance number is ignored. For AV streams, the instance number can be used to disambiguate the case where the display shows two or more instances of the same data stream simultaneously. Instance numbers do not have to count up. In this case, they are simply a unique id that allows the CMPS to match a start event with an end event.

In a second embodiment, CMPS 2302 may include some special purpose hardware in combination with general purpose hardware which is also used for other functions of the

device. In this embodiment, care must be taken to ensure that commercially trusted CMPS functions are performed in a secure and tamper-resistant manner, despite the use of general purpose hardware. Each of the elements recited above may include dedicated CMPS functions and general purpose device functions:

(a) CPU/microcontroller. This may include one or more devices. If more than one device is included (e.g., a CPU and a DSP, a math coprocessor or a commerce coprocessor), these devices may be included within the same package, which may be rendered tamper-resistant, or the devices may communicate on a secure bus. The CPU may include two modes: a secure CMPS mode, and an unsecure general purpose mode. The secure CMPS mode may allow addressing of secure memory locations unavailable to the processor in general purpose mode. This may be accomplished, for example, by circuitry which remaps some of the available memory space, so that, in unsecure mode, the CPU cannot address secure memory locations.

(b) External communications ports. If the device, for example, a Commerce Appliance, is capable of receiving content or other information through a communications port (e.g., a cable connection, an Internet connection), this communications port can be used for CMPS purposes. In such a case, CMPS accesses to the external communications port is preferably designed to avoid or minimize interference with the use of such port for receipt of content.

(c) Memory. In some applications and embodiments, it is possible to operate a Commerce Appliance without NVRAM, wherein information that may be needed for CMPS operation that would employ NVRAM would be loaded into RAM, as required. ROM, RAM and NVRAM may be shared between CMPS uses and general uses. This can be accomplished in any of the following ways, or in a combination of these ways: (1) Some memory space may be rendered off-limits to general purpose uses, for example by remapping; (2) the entirety of the memory may be rendered secure, so that even portions of the memory being used for non-secure purposes cannot be observed or changed except in a secure and authorized manner; (3) CMPS information may be stored in an encrypted fashion, though this requires at least some RAM to be secure, since the CMPS will require direct access to unencrypted information stored in RAM.

(d) Encryption/decryption engine. Encryption and decryption functions, including key generation, may be handled by special purpose software running on a general purpose processor arrangement, particularly, for example, a floating point processor or DSP arrangement. That processor arrangement may also be used for purposes of decompressing and displaying content and/or for handling watermarking/fingerprinting

insertion and/or reading. Alternatively, the device may include native encryption and decryption functions. For example, various emerging standards may require at least some degree of encryption and decryption of content designed to be passed across unsecure buses within and among devices such as DVD players, such as the "five company proposal" and other IEEE 1394 related initiatives. Circuitry designed to perform such encryption and decryption may also be usable for CMPS applications.

(e) Secure clock/calendar. The underlying device may already require at least some clock information. MPEG-4, for example, requires the use of clock information for synchronization of Elementary Streams. A secure CMPS clock can also be used for such purposes.

In a third embodiment, CMPS 2302 can be primarily software designed to run on a general purpose device which may include certain minimal security-related features. In such a case, CMPS 2302 may be received in the same channel as the content, or in a side-band channel. An I-CMPO and/or other CI may specify a particular type of CMPS, which Commerce Appliance 2301 must either have or acquire (e.g., download from a location specified by the I-CMPO), or CMPS 2302 may be included, for example, with an I-CMPO.

A software CMPS runs on the CPU of the Commerce Appliance. This approach may be inherently less secure than the use of dedicated hardware. If the Commerce Appliance includes secure hardware, the software CMPS may constitute a downloadable OS and/or BIOS which customizes the hardware for a particular type of commerce application.

In one embodiment, a software CMPS may make use of one or more software tamper resistance means that can materially "harden" software. These means include software obfuscation techniques that use algorithmic means to make it very difficult to reverse engineer some or all of a CMPS, and further make it difficult to generalize from a reverse engineering of a given one or more CMPS. Such obfuscation is preferably independent of source code and object code can be different for different CMPSs and different platforms, adding further complexity and separation of roles. Such obfuscation can be employed "independently" to both CI, such as an CMPO, as well as to some or all of the CMPS itself, thus obscuring both the processing environment and executable code for a process. The approach is also applicable for integrated software and hardware implementation CMPS implementations described above. Other tamper resistance means can also be employed, including using "hiding places" for storing certain state information in obscure and unexpected locations, such as locations in NV memory used for other purposes, and data hiding techniques such as watermarking/fingerprinting.

Association of CMPS With a Commerce Appliance

A CMPS may be permanently attached to a particular device, or may be partially or fully removable. A removable CMPS may include software which is securely loaded into a Commerce Appliance, and/or removable hardware. A removable CMPS may be personalized to one or more particular users, including user keys, budget information, preferences, etc., thereby allowing different users to use the same Commerce Appliance without commingling budgets and/or other rights, etc.

A CMPS may be designed for operation with certain types of content and/or for operation with certain types of business models. A Commerce Appliance may include more than one type of CMPS. For example, a Commerce Appliance designed to accept and display content pursuant to different standards may include one CMPS for each type of format. In addition, a Commerce Appliance may include a CMPS provided by a particular provider, designed to preferentially display certain types of content and to preferentially bill for such content through a particular channel (e.g., billing to one or more particular credit cards and/or using a particular one or more clearinghouses).

Source of Rules

The CMPS must recognize those rules which are to be applied to particular content. Such rules may be received by the CMPS from a variety of sources, depending on the particular embodiment used:

(a) CMPO. The rules may be included within a CMPO (e.g., CMPO 2303) and/or other CI. The CMPO and/or other CI may be incorporated within a content object or stream (as, e.g., a header on an MPEG-4 ES), and/or may be contained within a dedicated content object or stream encoded and received as per the underlying standard (e.g., an MPEG-4 CMPO ES), and/or may be received outside the normal content stream, in which event it may not be encoded as per the underlying standard (e.g., a CMPS received as an encrypted object through a sideband channel).

(b) CMPS. Rules may be permanently and/or persistently stored within a CMPS, e.g., Rules 2409. A CMPS may include default rules designed to handle certain situations, for example, where no CMPO and/or other necessary CI is received (e.g., content encoded under an earlier version of the standard which did not incorporate CMPOs, including MPEG-4 version 1). Complete rules which are stored within the CMPS may be directly or indirectly invoked by a CMPO and/or other CI. This may occur through the CI identifying particular rules through a pointer, and/or it may occur through the CI identifying itself and the general class of control it requires, with the CMPS then applying particular rules specific to that CMPS.

Rule "primitives" may also be stored within the CMPS (e.g., Control Primitives 2410). The CMPO and/or other CI may invoke these primitives by including a sequence of macro-type commands, each of which triggers a sequence of CMPS primitives.

(c) User. The user may be given the ability to create rules relating to the particular user's preferences. Such rules will generally be allowed to further restrict the use of content, but not to expand the use of content beyond that which would otherwise be allowed. Examples include: (a) rules designed to require that certain types of content (e.g., adult movies) only be accessible after entry of a password and/or only to certain CMPS users (e.g. adults, not children, as, for example, specified by parents and/or a societal body such as a government agency); (b) rules designed to require that only particular users be allowed to invoke operations requiring payment beyond a certain limit and/or aggregate payment over a certain amount.

The user may be allowed to create templates of rules such as described in the aforementioned Ginter '333 patent application (and incorporated herein). In addition, a CMPS arrangement, and/or a particular CMPO and/or other CI, may restrict the rules the user is allowed to specify. For example, a CI may specify that a user can copy a work, but cannot add rules to the work restricting the ability of a recipient to make additional copies (or to be able to view, but only after a payment to the first user). User supplied one or more rules may govern the use of – including privacy restrictions related to -- payment, audit, profiling, preference, and/or any other kind of information (e.g., information result as a consequence of the use of a CMPS arrangement, including, for example, use of secured content). Such user supplied one or more rules can be associated with the user and/or one or more Commerce Appliances in a user arrangement, whether or not the information is aggregated according to one or more criteria, and whether or not user and/or appliance identification information is removed during aggregation and/or subsequent reporting, distribution, or any other kind of use.

The ability to allow the user to specify rules allows the CMPS to subsume (and thereby replace) V-chips, since a parent can use content rating information to specify precisely what types of information each viewer will be allowed to watch (e.g., violent content can only be displayed after entry of a certain password and/or other identifier, including, for example, insertion of a removable hardware card (smart or rights card) possessed by a user).

(d) External network source. The rules may be stored on an external server. Rules may be addressed and downloaded by the CMPS if necessary (e.g., either the CMPO and/or other CI and/or the CMPS contains a pointer to certain rules location(s), such as one

or more URLs). In addition, content providers and/or clearinghouses may broadcast rules designed for general applicability. For example, a content provider might broadcast a set of rules providing a discount to any user participating in a promotional event (e.g., by providing certain user information). Such rules could be received by all connected devices, could be received by certain devices identified as of interest by the content provider (e.g., all recent viewers of a particular program, as identified by exhaust information provided by the CMPS to a clearinghouse and/or all members having certain identity characteristics such as being members of one or more classes) and/or could be posted in central locations.

Example Embodiment

In one embodiment, a set of MPEG-4 Elementary Streams may make up a work. The Elementary Streams may be encrypted and multiplexed together to form an Aggregate Stream. One or more CMPOs may be present in such stream, or may otherwise be associated with the stream. Options are as follows:

1. Content may be streamed or may be received as static data structures.

2. A Work may be made up of a single stream or data structure, or of many separately addressable streams or data structures, each of which may constitute an Object.

3. If a Work is made up of separately addressable streams or data structures, those streams or data structures may be multiplexed together into an Aggregate Stream, or may be received separately.

4. If streams or data structures are multiplexed together into an Aggregate Stream, the streams or data structures may be encrypted prior to such multiplexing. The Aggregate Stream itself may be encrypted, whether or not the underlying streams or data structures are encrypted. The following possibilities therefore exist: (a) individual streams/data structures are unencrypted (in the clear), the Aggregate Stream is unencrypted; (b) individual streams/data structures are unencrypted prior to multiplexing, the Aggregate Stream is encrypted following multiplexing; (c) individual streams/data structures are encrypted prior to multiplexing, the Aggregate Stream is not encrypted following multiplexing; or (d) individual streams/data structures are encrypted prior to multiplexing, the Aggregate Stream is encrypted following multiplexing.

5. A CMPO may be associated with a channel (CCMPO), a work (MCMPO) or an individual Object (CMPO).

6. A CMPO may be received prior to the controlled data, may be received contemporaneously with the data, or may be received after the data (in which event use of the data must wait until the CMPO has been received).

7. A CMPO may be received as part of an Aggregate Stream or separately.

8. If a CMPO is received as part of the Aggregate Stream, it may be multiplexed together with the individual streams or data structures, or may constitute a separate stream or data structure.

9. If a CMPO is multiplexed within the Aggregate Stream, it may be encrypted or nonencrypted. If encrypted, it may be encrypted prior to multiplexing, and/or encrypted after multiplexing, if the entire Aggregate Stream is encrypted.

10. If a CMPO is received as part of the Aggregate Stream, it may be (a) a part of the stream or data structure which holds the content (e.g., a header); (b) a separate stream or data structure encoded pursuant to the same format as the streams or data structures which hold the content (e.g., an MPEG-4 ES) or (c) a separate stream or data structure encoded under a different format designed for CMPOs.

11. If a CMPO is a part of the stream or data structure which holds the content, it may be (a) a header which is received once and then persistently maintained for control of the content; (b) a header which is received at regular intervals within the stream or data structure; or (c) data distributed throughout the stream or data structure.

These various scenarios give rise to different requirements for demultiplexing and decryption of the CMPOs. FIG. 25 illustrates the following embodiment:

1. Aggregate Stream 2501 is made up of multiplexed ESs (e.g., ES 2502 and 2503). A combination of such ESs makes up a single work. Aggregate Stream 2501 is generated by a cable aggregator and received by a user's set-top box as one of a number of channels.

2. CCMPOs 2504 corresponding to each channel are sent along the cable in Header 2505 at regular intervals (e.g., once per second). When the set-top box is turned on, it polls each channel, and downloads all current CCMPOs. These are stored persistently, and are changed only if a new CCMPO is received which differs from prior CCMPOs.

3. When the user selects a channel, the set-top box addresses the associated CCMPO. The CCMPO may specify, for example, that content in this particular channel may only be accessed by subscribers to the channel. A CMPS within the set-top box accesses a user profile persistently stored in NVRAM and determines that the user is a subscriber. The CMPS deems the CCMPO rule to have been satisfied.

4. The CMPS obtains an identifier for the MCMPO associated with the work (video) currently streaming on the channel and a key for the MCMPO. If works are received serially on the channel (e.g., a television channel in which one work is provided at a time), the received MCMPO identifier may include don't care bits so that it can address any MCMPO currently on the channel.

5. The CMPS begins demuxing of Aggregate Stream 2501 (this may occur in parallel with the preceding step), and obtains the MCMPO, which is encoded into an ES multiplexed within the Aggregate Stream (e.g., MCMPO 2506). Although each ES within Aggregate Stream 2501 has been encrypted, Aggregate Stream 2501 was not encrypted following multiplexing. This allows the CMPS to demultiplex Aggregate Stream 2501 without decrypting the entire Aggregate Stream.

6. The CMPS identifies the ES which constitutes the MCMPO (e.g., ES 2503). The CMPS downloads one complete instance of MCMPO 2506 into an internal buffer, and uses the key received from CCMPO 2504 to decrypt MCMPO 2506.

7. The CMPS determines which rules are applied by MCMPO 2506. MCMPO 2506 might, for example, include a rule stating that the user can view the associated work with advertisements at a low fee, but must pay a higher fee for viewing the work without advertisements.

8. The CMPS generates an options menu, and displays that menu on the screen for the user. The menu specifies the options, including the cost for each option. Additional options may be specified, including payment types.

9. The user uses a remote control pointing device to choose to view the work at a lower cost but with advertisements. The user specifies that payment can be made from an electronic cash budget stored in the CMPS.

10. The CMPS subtracts the specified amount from the budget persistently stored in NVRAM, and generates and encrypts a message to a server associated with the cable. The message transfers the required budget to the server, either by transferring electronic cash, or by authorizing a financial clearinghouse to transfer the amount from the user's account to the cable provider's. This message may be sent immediately, or may be buffered to be sent later (e.g., when the user connects the device to the Internet). This step may be taken in parallel with decryption of the content.)

11. The CMPS obtains from MCMPO 2506 a set of keys used to decrypt the Elementary Streams associated with the work (e.g., ES 2502). The CMPS also obtains identifiers for the specific ESs to be used. Since the user has indicated that advertisements are to be included, the MCMPO identifies ESs associated with the advertisements, and identifies a Scene Descriptor Graph which includes advertisements. A Scene Descriptor Graph which does not include advertisements is not identified, and is not passed through by the CMPS.

12. The CMPS passes the decrypted ESs to the MPEG-4 buffers. The normal process of MPEG-4 decoding, compositing and rendering then takes place. The Composite

and Render block outputs Start and Stop events for each object released for viewing. The CMPS monitors this information and compares it to the expected events. In particular, the CMPS confirms that the advertisements have been released for viewing, and that each operation has occupied approximately the expected amount of time.

In another embodiment, a set-top box containing a CMPS (e.g., CMPS 2302 from FIG. 23) may have a cable input (e.g., carrying M4 Bit Streams 2314 and CMPOs 2303). The cable may carry multiple channels, each made up of two sub-channels, with one sub-channel carrying MPEG-4 ESs (e.g., M4 Bit Streams 2314), and the other sub-channel carrying CMPOs (e.g., CMPOs 2303). The sub-channel carrying CMPOs 2303 could be routed directly to CMPS 2302, with the ES channel being routed to a decryption block (operating under control of the CMPS, e.g., CR&D 2315), and then to the MPEG-4 buffers (e.g., buffers associated with Scene Descriptor Graph 2306, AVO Decode 2307 and Object Descriptors 2308). In this case, if the ESs are not encrypted, they proceed unchanged through the decryption block and into the buffers. This may occur, for example, if the ESs are being broadcast for free, with no restrictions, and/or if they are public domain information, and/or they were created prior to inclusion of CMPOs in the MPEG-4 standard.

Such an embodiment might include timing synchronization information in the CMPO sub-channel, so that CMPOs can be synchronized with the associated ESs.

The concept of incorporating two separate streams, one consisting of control information and connected directly to the CMPS, and the other consisting of ESs, may support a high degree of modularization, such that the formats of CMPOs, and particular types of CMPS's, may be changed without alteration to the underlying ES format. For example, it may be possible to change the CMPO format without the necessity for reformatting content ESs. To take another example, it may be possible to upgrade a Commerce Appliance by including a new or different CMPS, without the necessity for any changes to any of the circuitry designed to demultiplex, composite and render the content ESs. A user might obtain a CMPS on a smart card or other removable device, and plug that device into a Commerce Appliance. This could be done to customize a Commerce Appliance for a particular application or for particular content.

CMPS Interface to a CE Device

A CMPS may be designed to present a standardized interface between the general-purpose functionality of a consumer electronics device and any relevant CMPOs and/or other CI and protected content. For example, a CMPS could be designed to accept CI and encrypted ESs, and output decrypted ESs into the device's buffers. In such a case, the

manufacturer of the device would be able to design the device in compliance with the specification (e.g., MPEG-4), without concern about commerce-related extensions to the standard, which extensions might differ from provider to provider. All such extensions would be handled by the CMPS.

Initialization

1. Initialization of the CMPS.

A CMPS may be used to identify the capabilities of the Commerce Appliance in which a CMPS is installed. A CMPS permanently associated with a particular Commerce Appliance may have such information designed-in when the CMPS is initially installed (e.g., stored in ROM 2406 shown in FIG.24). A CMPS which is removable may be used to run an initialization operation in order to obtain information about the device's capabilities. Such information may be stored in a data structure stored in NVRAM 2425. Alternatively, some or all of such information may be gathered each time the device is turned on, and stored in RAM 2414.

For example, a DVD player may or may not contain a connection to an external server and/or process. A CMPO and/or other CI stored on a DVD (and/or any other format optical disk) inserted into a DVD (or any other format optical disk) player may include rules predicated on the possibility of outputting information to a server (e.g., content is free if user identification information is output), or may require a direct connection in order, for example, to download keys used to decrypt content. In such a case, the CMPS arrangement may determine the hardware functionality which is expected by or required by the CMPO, and compare that to the hardware actually present. If the CMPS determines that the CMPO and/or other CI requires a network connection , and that the DVD player does not include such a connection, the CMPS may take a variety of steps, including: (1) if the network connection is required for some options but not others, causing only those options which are possible to be displayed to the user; (2) informing the user that necessary hardware is missing; or (3) causing a graceful rejection of the disk, including informing the user of the reason for the rejection.

To take another example, a CMPO and/or other CI may include a business model which allows the user to choose among quality levels (or other forms of variations of a given work, for example, longer length and/or greater options), with a higher price being charged if the user selects a higher level of quality (e.g., music may be played at low resolution for free, but requires a payment in order to be played at a higher resolution). In such a case, the Commerce Appliance may not include loudspeakers which are capable of outputting sound at the higher resolution. The CMPS arrangement preferably identifies this situation, and either eliminates the higher resolution output as an option for the user, or

informs the user that this option costs more but provides no additional benefit given the Commerce Appliance's current functionality or given the Commerce Appliance not being docked in a user arrangement that provides higher quality loudspeakers.

If the Commerce Appliance may be hooked up to external devices (e.g., loudspeakers, display, etc.), the CMPS will require some mechanism for identifying and registering such devices. Each device may be used to make standard ID and capability information available at all times, thereby allowing the CMPS to poll all connected devices at regular intervals, including, for example, authenticating CMPS arrangements within one or more of each such connected devices. Using another approach, all devices could be used to output CMPS identification information upon power-on, with later connected devices being used to output such information upon establishment of the connection. Such identification information may take the form, for example, of authentication information provided under the "five company arrangement", such authentication methods are herein incorporated by reference.

As discussed earlier, a Commerce Appliance may be connected to multiple devices each containing its own CMPS arrangement (e.g., a DVD player may be connected to a digital TV) In such cases, the CMPSs must be able to initiate secure communication (e. g., using a scheme, for example, like the "five company proposal" for IEEE 1394 serial bus) and determine how the CMPSs will interact with respect to content communication between CMPSs and, in certain embodiments, regarding cooperative governance of such content such as describing in the incorporated Shear patent application. In one embodiment, the first CMPS arrangement to receive content might govern the control process by downloading an initial CMPO and/or other CI, and display one or more of the rules to the user, etc. The second CMPS arrangement might recognize that it has no further role to play, either as a result of a communication between the two CMPS arrangements, or as a result of changes to the content stream created by the first CMPS arrangement (which decrypted the content, and may have allowed demuxing, composition and rendering, etc.)

The relationship between upstream and downstream CMPSs arrangements may be complicated if one device handles certain aspects of MPEG-4 rendering, and the other handles other aspects. For example, a DVD player might handle demuxing and buffering, transferring raw ESs to a digital TV, which then handles composition and rendering, as well as display. In such a case, there might be no back-channel from the composition and rendering block to the upstream CMPS arrangement. CMPS arrangements are preferably designed to handle stand-alone cases (a DVD (or any other optical disk) player with a CMPS arrangement attached to a dumb TV with no CMPS), multiple CMPS arrangement

cases in which one CMPS arrangement handles all of the processing (a DVD (or other optical disk) player which handles everything through composition and rendering, with a video stream output to the digital TV (in one non-limiting example, via an IEEE 1349 serial bus) (that output stream would be encrypted as per the "five company proposal" for copy protection using IEEE 1394 serial bus transmission)) and/or shared processing between two or more CMPSs arrangements regarding some, or in certain cases, all, of such processing.

2. Initialization of a particular content stream.

The CMPS may be designed so that it can accept initialization information which initializes the CMPS for a particular content stream or channel. This header, which may be a CMPO and/or other CI, may contain information used by the CMPS to locate and/or interpret a particular content stream as well as CI associated with that stream. This initial header may be received through a sideband channel, or may be received as a CI ES such as a CMPO ES.

In one example, shown in FIG. 26, Header CMPO 2601 may include the following information:

(a) Stream/Object/CMPO ID 2602, which identifies the content streams/objects governed by Header CMPO 2601 and/or identification of CMPOs associated with each such content stream or object.

In one embodiment, Header CMPO 2601 identifies other CMPOs which contain rules and keys associated with particular content streams. In another embodiment, Header CMPO 2601 directly controls all content streams, by incorporating the keys and rules associated with such streams. In the latter case, no other CMPOs may be used.

In one embodiment, Header CMPO 2601 may be one or more CMPOs, CCMPOs, MCMPOs, and/or other CI.

(b) One or CMPO Keys 2603 for decrypting each identified CMPO.

(c) Work-Level Control 2604, consisting of basic control information associated with the work as a whole, and therefore potentially applicable to all of the content streams which make up the work. This basic control information may include rules governing the work as a whole, including options to be presented to the user.

(d) In one embodiment of this embodiment, a header CMPO may be updatable to contain User/Site Information 2605 regarding a particular user or site currently authorized to use certain content, as well as one or more rule sets under which the user has gained such authorization. A header CMPO associated with a work currently being viewed may be stored in RAM or NVRAM. This may include updated information. In one

embodiment, the CMPO may also store header CMPOs for certain works viewed in the past. In one embodiment, header CMPOs may be stored in non-secure memory, with information sufficient to identify and authenticate that each header CMPO had not been changed.

In one such header CMPO embodiment of this embodiment, the header CMPO operates as follows:

(a) The header CMPO is received by a CMPS arrangement. In the case of previously unreceived content which has now become available, the header CMPO may be received at an input port. In the case of content which is already available, but is not currently being used (e.g., a set-top box with 500 channels, of which either 0 or 1 are being displayed at any given time), CCMPOs for each channel may be buffered by the CMPS arrangement for possible use if the user invokes particular content (e.g., switches to a particular channel).

In either case, the header CMPO must include information which allows a CMPS arrangement to identify it as a header CMPO.

(b) The CMPS arrangement obtains business-model information held in the clear in the header CMPO. Business-model information may include, for example, a statement that content can be viewed for free if advertisements are included, or if the user authorizes Nielson-type information, user and/or audience measurement information, for example, content may be output to a server or otherwise copied once, but only at a price.

(c) The CMPS arrangement either accepts the business model, if the user has authorized it to accept certain types of models (e.g., the user has programmed the CMPS arrangement to always accept play with advertisements for free), rejects the business model, if the user has instructed that the particular model always be rejected, or displays the business model to the user (e.g., by presenting options on the screen).

(d) If a business model has been accepted, the CMPS arrangement then decrypts the remainder of the header CMPO. If the Commerce Appliance contains a live output connection to an external server (e.g., Internet connection, back-channel on a set-top box, etc.), and if latency problems are handled, decryption of these keys can be handled by communicating with the external server, each side authenticating the other, establishment of a secure channel, and receipt of a key from the server. If the Commerce Appliance is not at least occasionally connected to an external server, decryption may have to be based on one or more keys securely stored in the Commerce Appliance.

(e) Once a header CMPO has been decrypted, the CMPS arrangement acquires information used to identify and locate the streams containing the content, and

keys which are used to decrypt either the CMPOs associated with the content, or to directly decrypt the content itself.

(f) In one embodiment of this header embodiment, the header CMPO may contain a data structure for the storage of information added by the CMPS arrangement. Such information may include the following:

(1) Identification of user and/or Commerce Appliance and/or CMPS arrangement. In this embodiment, such information may be stored in a header CMPO in order to provide an audit trail in the event the work (including the header CMPO) is transferred (this only works if the header CMPO is transferred in a writable form). Such information may be used to allow a user to transfer the work to other Commerce Appliances owned by the user without the payment of additional cost, if such transfers are allowed by rule information associated with the header CMPO. For example, a user may have a subscription to a particular cable service, paid for in advance by the user. When a CMPS arrangement downloads a header CMPO from that cable service, the CMPS arrangement may store the user's identification in the header CMPO. The CMPS arrangement may then require that the updated header CMPO be included if the content is copied or transferred. The header CMPO could include a rule stating that, once the user information has been filled in, the associated content can only be viewed by that user, and/or by Commerce Appliances associated with that user. This would allow the user to make multiple copies of the work, and to display the work on multiple Commerce Appliances, but those copies could not be displayed or used by non-authorized users and/or on non-authorized Commerce Appliances. The header CMPO might also include a rule stating that the user information can only be changed by an authorized user (e.g., if user 1 transfers the work to user 2, user 2's CMPS arrangement can update the user information in the header CMPO, thereby allowing user 2 to view the work, but only if user 2 is also a subscriber to the cable channel).

(2) Identification of particular rules options governing use. Rule sets included in header CMPOs may include options. In certain cases, exercise of a particular option might preclude later exercise of a different option. For example, a user might be given the choice to view an unchanged work for one price, or to change a work and view the changed work for a higher price. Once the user decides to change the work and view the changed work, this choice is preferably stored in the header CMPO, since the option of viewing the original unchanged work at the lower price is no longer available. The user might have further acquired the right, or may now be presented with the option for the right, to further distribute the changed work at a mark-up in cost resulting in third party

derived revenue and usage information flowing to both the user and the original work stakeholder(s).

(3) Historical usage information. The header CMPO may include information relating to the number and types of usages. For example, if the underlying work is copied, the header CMPO may be updated to reflect the fact that a copy has been made, since a rule associated with the work might allow only a single copy (e.g., for backup and/or timeshifting purposes). To take another example, a user might obtain the right to view a work one time, or for a certain number of times. The header CMPO would then be updated to reflect each such use.

Usage information may be used to determine if additional uses are authorized by rules associated with the header CMPO. Such information may also be used for audit purposes. Such information may also be provided as usage information exhaust, reported to an external server. For example, a rule may specify that a work may be viewed for free, but only if historical usage information is downloaded to a server.

Content Management Protection Objects (CMPO)

The Content Management and Protection Object ("CMPO") is a data structure which includes information used by the CMPS to govern use of certain content. A CMPO may be formatted as a data structure specified by a particular standard (e.g., an MPEG-4 ES), or may be formatted as a data structure not defined by the standard. If the CMPO is formatted as a data structure specified by the standard, it may be received in the channel utilized by the standard (e.g., as part of a composite MPEG-4 stream) or may be received through some other, side-band method. If the CMPO is formatted as a data structure not specified by the relevant standard, it is provided and decoded using some side-band method, which may include receipt through the same port as formatted content and/or may include receipt through a separate port.

Content may be controlled at virtually any level of granularity. Three exemplary levels will be discussed herein: "channel," "work," and "object."

A "channel" represents an aggregation of works. The works may be available for selection by the user (e.g., a web site, or a video library) or may be received serially (e.g., a cable television channel).

A "work" represents a single audio-visual, textual or other work, intended to be consumed (viewed, read, etc.) by a user as an integrated whole. A work may, for example, be a movie, a song, a magazine article, a multimedia product such, for example, as sophisticated videogame. A work may incorporate other works, as, for example, in a multimedia work which incorporates songs, video, text, etc. In such a case, rights may be

associated

An "object" represents a separately addressable portion of a work. An object may be, for example, an individual MPEG-4 AVO, a scene descriptor graph, an object descriptor, the soundtrack for a movie, a weapon in a videogame, or any other logically definable portion.

Content may be controlled at any of these levels (as well as intermediate levels not discussed herein). The preferred embodiment mechanism for such control is a CMPO or CMPO arrangement (which comprises one or more CMPOs, and if plural, then plural, cooperating CMPOs). CMPOs and CMPO arrangements may be organized hierarchically, with a Channel CMPO arrangement imposing rules applicable to all contained works, a MCMPO or an SGCMPO imposing rules applicable to all objects within a work, and a CMPO arrangement imposing rules applicable to a particular object.

In one embodiment, illustrated in FIG. 27, a CMPS may download CCMPO 2701. CCMPO 2701 may include one or more Rules 2702 applicable to all content in the channel, as well as one or more Keys 2703 used for decryption of one or more MCMPOs and/or SGCMPOs. MCMPO 2704 may include Rules 2705 applicable to a single work and/or works, one or more classes and/or more users and/or user classes, and may also include Keys 2706 used to decrypt CMPOs. CMPO 2707 may include Rules 2708 applicable to an individual object, as well as Key 2709 used to decrypt the object.

As long as all objects are subject to control at some level, there is no requirement that each object be individually controlled. For example, CCMPO 2701 could specify a single rule for viewing content contained in its channel (e.g., content can only be viewed by a subscriber, who is then might be free to redistribute the content with no further obligation to the content provider). In such a case, rules would not necessarily be used for MCMPOs (e.g. Rules 2705), SGCMPOs, or CMPOs (e.g., Rules 2708). In one embodiment, MCMPOs, SGCMPOs, and CMPOs could be dispensed with, and CCMPO 2701 could include all keys used to decrypt all content, or could specify a location where such keys could be located. In another embodiment, CCMPO 2701 would supply Key 2703 used to decrypt MCMPO 2704. MCMPO 2704 might include keys used to decrypt CMPOs (e.g., Keys 2706), but might include no additional Rules 2705. CMPO 2707 might include Key 2709 used to decrypt an object, but might include no additional Rules 2708. In certain embodiments, there may be no SGCMPOs.

A CMPO may be contained within a content data structure specified by a relevant standard (e.g., the CMPO may be part of a header in an MPEG-4 ES.) A CMPO may be contained within its own, dedicated data structure specified by a relevant standard (e.g., a

CMPO ES). A CMPO may be contained within a data structure not specified by any content standard (e.g., a CMPO contained within a DigiBox).

A CCMPO may include the following elements:

(a) ID 2710. This may take the following form: <channel ID>< CMPO type><CMPO ID><version number>. In the case of hierarchical CMPO organization (e.g., CCMPOs controlling MCMPOs controlling CMPOs), CMPO ID 2711 can include one field for each level of the hierarchy, thereby allowing CMPO ID 2711 to specify the location of any particular CMPO in the organization. ID 2710 for a CCMPO may, for example, be 123-000-000. ID 2712 for a MCMPO of a work within that channel may, for example, be 123-456-000, thereby allowing the specification of 1,000 MCMPOs as controlled by the CCMPO identified as "123." CMPO ID 2711 for a CMPO associated with an object within the particular work may, for example, be 123-456-789, thereby allowing the specification of 1,000 CMPOs as associated with each MCMPO.

This method of specifying CMPO IDs thereby conveys the exact location of any CMPO within a hierarchy of CMPOs. For cases in which higher levels of the hierarchy do not exist (e.g., a MCMPO with no associated CCMPO), the digits associated with that level of the hierarchy may be specified as zeroes.

(b) Rules 2702 applicable to all content in the channel. These may be self-contained rules, or may be pointers to rules obtainable elsewhere. Rules are optional at this level.

(c) Information 2713 designed for display in the event the user is unable to comply with the rules (e.g., an advertisement screen informing the user that a subscription is available at a certain cost, and including a list of content available on the channel).

(d) Keys 2703 for the decryption of each MCMPO controlled by this CCMPO. In one embodiment, the CCMPO includes one or more keys which decrypt all MCMPOs. In an alternate embodiment, the CCMPO includes one or more specific keys for each MCMPO.

(e) A specification of a CMPS Type (2714), or of hardware/software necessary or desirable to use the content associated with this channel.

The contents of a MCMPO may be similar to those of a CCMPO, except that the MCMPO may include rules applicable to a single work, and may identify CMPOs associated with each object.

The contents of each CMPO may be similar to those of the MCMPO, except that the CMPO may include rules and keys applicable to a single object.

The contents of an SGCMPO may be similar to those of the CCMPO, except that the MCMPO may include rules applicable to only certain one or more classes of rights, certain one or more classes of works, and/or to one or more certain classes of users and/or user arrangements (e.g. CMPO arrangements and/or their devices).

In another embodiment, shown in FIG. 28, CMPO Data Structure 2801 may be defined as follows:

CMPO Data Structure 2801 is made up of elements. Each element includes a self-contained item of information. The CMPS parses CMPO Data Structure, one element at a time.

Type Element 2802 identifies the data structure as a CMPO, thereby allowing the CMPS to distinguish it from a content ES. In an exemplary embodiment, this element may include 4 bits, each of which may be set to "1" to indicate that the data structure is a CMPO.

The second element is CMPO Identifier 2803, which is used to identify this particular CMPO and to convey whether the CMPO is part of a hierarchical organization of CMPOs and, if so, where this CMPO fits into that organization.

CMPO Identifier 2803 is divided into four sub-elements, each of three bits. These are shown as sub-elements A, B, C and D. The first sub-element (2803 A) identifies the CMPO type, and indicates whether the CMPO is governed or controlled by any other CMPO:

100: this is a top-level CMPO (associated with a channel or an aggregation of works) and is not controlled by any other CMPO.

010: this is a mid-level CMPO (associated with a particular work) and is not controlled by any other CMPO.

110: this is a mid-level CMPO, and is controlled by a top-level CMPO.

001: this is a low-level CMPO (associated with an object within a work) and is not controlled by any other CMPO. This case will be rare, since a low-level CMPO will ordinarily be controlled by at least one higher-level CMPO.

011: this is a low-level CMPO, and is controlled by a mid-level CMPO, but not by a top-level CMPO.

111: this is a low-level CMPO, and is controlled by a top-level CMPO and by a mid-level CMPO.

The second sub-element of CMPO ID 2803 (sub-element B) identifies a top-level CMPO. In the case of a top-level CMPO, this identifier is assigned by the creator of the CMPO. In the case of a mid-level or low-level CMPO which is controlled by a top-level

CMPO, this sub-element contains the identification of the top-level CMPO which performs such control. In the case of a mid-level or low-level CMPO which is not controlled by a top-level CMPO, this sub-element contains zeroes.

The third sub-element of CMPO ID 2803 (sub-element C) identifies a mid-level CMPO. In the case of a top-level CMPO, this sub-element contains zeroes. In the case of a mid-level CMPO, this sub-element contains the identification of the particular CMPO. In the case of a low-level CMPO which is controlled by a mid-level CMPO, this sub-element contains the identification of the mid-level CMPO which performs such control. In the case of a low-level CMPO which is not controlled by a mid-level CMPO, this sub-element contains zeroes.

The fourth sub-element of CMPO ID 2803 (sub-element D) identifies a low-level CMPO. In the case of a top-level or mid-level CMPO, this sub-element contains zeroes. In the case of a low-level CMPO, this sub-element contains the identification of the particular CMPO.

Following the identifier element is Size Element 2804 indicating the size of the CMPO data structure. This element contains the number of elements (or bytes) to the final element in the data structure. This element may be rewritten if alterations are made to the CMPO. The CMPS may use this size information to determine whether the element has been altered without permission, since such an alteration might result in a different size. For such purposes, the CMPS may store the information contained in this element in a protected database. This information can also be used to establish that the entire CMPO has been received and is available, prior to any attempt to proceed with processing.

Following Size Element 2804 are one or more Ownership/Control Elements containing ownership and chain of control information (e.g., Ownership/Control Elements 2805, 2806 and 2807). In the first such element (2805), the creator of the CMPO may include a specific identifier associated with that creator. Additional participants may also be identified in following elements (e.g., 2806, 2807). For example, Element 2805 could identify the creator of the CMPO, Element 2806 could identify the publisher of the associated work and Element 2807 could identify the author of the work.

A specific End Element 2808 sequence (e.g., 0000) indicates the end of the chain of ownership elements. If this sequence is encountered in the first element, this indicates that no chain of ownership information is present.

Chain of ownership information can be added, if rules associated with CMPO 2801 permit such additions. If, for example, a user purchases the work associated with CMPO 2801, the user's identification may be added as a new element in the chain of ownership

elements (e.g., a new element following 2807, but before 2808). This may be done at the point of purchase, or may be accomplished by the CMPS once CMPO 2801 is encountered and the CMPS determines that the user has purchased the associated work. In such a case, the CMPS may obtain the user identifier from a data structure stored by the CMPS in NVRAM.

Following the ownership element chain are one or more Handling Elements (e.g., 2809, 2810) indicating chain of handling. These elements may contain the identification of any CMPS which has downloaded and decoded CMPO 2801, and/or may contain the identification of any user associated with any such CMPS. Such information may be used for audit purposes, to allow a trail of handling in the event a work is determined to have been circulated improperly. Such information may also be reported as exhaust to a clearinghouse or central server. Chain of handling information preferably remains persistent until reported. If the number of elements required for such information exceeds a specified amount (e.g., twenty separate user identifiers), a CMPS may refuse to allow any further processing of CMPO 2801 or the associated work until the CMPS has been connected to an external server and has reported the chain of handling information.

The last element in the chain of handling elements (e.g., 2811) indicates the end of this group of elements. The contents of this element may, for example, be all zeroes.

Following the chain of handling elements may be one or more Certificate Elements (e.g., 2812, 2813) containing or pointing to a digital certificate associated with this CMPO. Such a digital certificate may be used by the CMPS to authenticate the CMPO. The final element in the digital certificate chain is all zeroes (2814). If no digital certificate is present, a single element of all zeroes exists in this location.

Following the Certificate Elements may be a set of Governed Object Elements (e.g., 2815, 2816, 2817, 2818) specifying one or more content objects and/or CMPOs which may be governed by or associated with CMPO 2801. Each such governed object or CMPO is identified by a specific identifier and/or by a location where such object or CMPO may be found (e.g., these may be stored in locations 2815 and 2817). Following each such identifier may be one or more keys used to decrypt such CMPO or object (e.g., stored in locations 2816 and 2818). The set of identifiers/keys ends with a termination element made up of all zeroes (2819).

Following the set of elements specifying identifiers and/or keys may be a set of Rules Elements (e.g., 2820, 2821, 2822) specifying rules/controls and conditions associated with use of the content objects and/or CMPOs identified in the Governed Objects chain (e.g., locations 2815 and 2817). Exemplary rules are described below. Elements may

contain explicit rules or may contain pointers to rules stored elsewhere. Conditions may include particular hardware resources necessary to use associated content objects or to satisfy certain rules, or particular types of CMPS's which are necessary or preferred for use of the associated content objects.

Following the rules/controls and conditions elements may be a set of Information Elements 2823 containing information specified by the creator of the CMPO. Among other contents, such information may include content, or pointers to content, programming, or pointers to programming.

The CMPO ends with Final Termination Element 2824.

In one embodiment, the rules contained in Rules Elements 2820-2822 of CMPO 2801 may include, for example, the following operations:

(1) Play. This operation allows the user to play the content (though not to copy it) without restriction.

(2) Navigate. This allows the user to perform certain types of navigation functions, including fast forward/rewind, stop and search. Search may be indexed or unindexed.

(3) Copy. Copy may be allowed once (e.g., time-shifting, archiving), may be allowed for a specified number of times and/or may be allowed for limited period of time, or may be allowed for an unlimited period of time, so long as other rules, including relevant budgets, are not violated or exceeded. A CMPS arrangement may be designed so that a Copy operation may cause an update to an associated CMPO (e.g., including an indication that the associated content has been copied, identifying the date of copying and the site responsible for making the copy), without causing any change to any applicable content object, and in particular without requiring that associated content objects be demuxed, decrypted or decompressed. In the case of MPEG-4, for example, this may require the following multi-stage demux process:

(i) the CMPS arrangement receives a Copy instruction from the user, or from a header CMPO.

(ii) CMPO ESs associated with the MPEG-4 stream which is to be copied are separated from the content stream in a first demux stage.

(iii) CMPOs are decrypted and updated by the CMPS arrangement. The CMPOs are then remuxed with the content ESs (which have never been demuxed from each other), and the entire stream is routed to the output port without further alteration.

This process allows a copy operation to take place without requiring that the content streams be demuxed and decrypted. It requires that the CMPS arrangement include

two outputs: one output connected to the digital output port (e.g., FIG. 23 line 2316, connecting to Digital Output Port 2317), and one output connected to the MPEG-4 buffers (e.g., FIG. 23, lines 2310, 2311, 2312), with a switch designed to send content to one output or the other (or to both, if content is to be viewed and copied simultaneously) (e.g., Switch 2319). Switch 2319 can be the only path to Digital Output Port 2317, thereby allowing CMPS 2302 to exercise direct control over that port, and to ensure that content is never sent to that port unless authorized by a control. If Digital Output Port 2317 is also the connector to a digital display device, CMPS 2302 will also have to authorize content to be sent to that port even if no copy operation has been authorized.

In one example embodiment, the receiving device receiving the information through Digital Output Port 2317 may have to authenticate with the sending device (e.g., CMPS 2302). Authentication may be for any characteristic of the device and/or one or more CMPSs used in conjunction with that device. Thus, for example, a sending appliance may not transmit content to a storage device lacking a compatible CMPS.

In another non-limiting example, CMPS 2302 can incorporate session encryption functionality (e.g., the "five company arrangement") which establishes a secure channel from a sending interface to one or more external device interfaces (e.g., a digital monitor), and provided that the receiving interface has authenticated with the sending interface, encrypts the content so that it can only be decrypted by one or more authenticated 1394 device interfaces. In that case, CMPS 2302 would check for a suitable IEEE 1394 serial bus interface , and would allow content to flow to Digital Output Port 2317 only if (a) an authorized Play operation has been invoked, a secure channel has been established with the device and the content has been session-encrypted, or (b) an authorized Copy or Retransmit operation has been invoked, and the content has been treated as per the above description (i.e., the CMPO has been demuxed, changed and remuxed, the content has never been decrypted or demuxed).

This is only possible if CMPOs are separately identifiable at an early demux stage, which most likely requires that they be stored in separate CMPO ESs. If the CMPOs are stored as headers in content ESs, it may be impossible to identify the CMPOs prior to a full demux and decrypt operation on the entirety of the stream.

(4) Change. The user may be authorized to change the content.

(5) Delete. This command allows the user to delete content which is stored in the memory of the Consumer Appliance. This operation operates on the entire work. If the user wishes to delete a portion of a work, the Change operation must be used.

(6) Transfer. A user may be authorized to transfer a work to a third party.

This differs from the Copy operation in that the user does not retain the content or any rights to the content. The Transfer operation may be carried out by combining a Copy operation and a Delete operation. Transfer may require alteration of the header CMPO associated with the work (e.g., adding or altering an Ownership/Control Element, such as Elements 2805-2807 of FIG. 28), so as to associate rights to the work with the third party.

These basic operations may be subject to modifications, which may include:

i. Payment. Operations may be conditioned on some type of user payment. Payment can take the form of cash payment to a provider (e.g., credit card, subtraction from a budget), or sending specified information to an external site (e.g., Nielson-type information).

ii. Quality of Service. Operations may specify particular quality of service parameters (e.g., by specifying a requested QoS in MPEG-4), including: requested level of decompression, requested/required types of display, rendering devices (e.g., higher quality loudspeakers, a particular type of game controller).

iii. Time. Operations may be conditioned such that the operation is only allowed after a particular time, or such that the price for the operation is tied to the time (e.g., real-time information at a price, delayed information at a lower price or free, e.g., allowing controlled copies but only after a particular date).

iv. Display of particular types of content. Operations may be conditioned on the user authorizing display of certain content (e.g., the play operation may be free if the user agrees to allow advertisements to be displayed).

In all of these cases, a rule may be modified by one or more other rules. A rule may specify that it can be modified by other rules or may specify that it is unmodifiable. If a rule is modifiable, it may be modified by rules sent from other sources. Those rules may be received separately by the user or may be aggregated and received together by the user.

Data types which may be used in an exemplary MPEG-4 embodiment may include the following:

a. CMP Data Stream.

The CMP-ds is a new elementary stream type that has all of the properties of an elementary stream including its own CMPO and a reference in the object descriptors. Each CMP-ds stream has a series of one or more *CMP Messages*. A CMP_Message has four parts:

1. **Count**: [1...*n*] CMPS types supported by this IP ES. Multiple CMPS systems may be supported, each identified by a unique *type*. (There may have

to be a central registry of types.)

2. CMPS_type_identifiers: [1...*n*] identifiers, each with an offset in the stream and a length. The offset points to the byte in the CMPO where the data for that CMPS type is found. The length is the length in bytes of this data.

3. Data segments: One segment for each of the *n* CMPS types encoded in a format that is proprietary to the CMPS supplier.

4. CMP_Message_URL: That references another CMP_Message. (This is in keeping with the standard of using URLs to point to streams.)

b. CMPO.

The CMPO is a data structure used to attach detailed CMP control to individual elementary streams. Each **CMPO** contains:

1. CMPO_ID: An identifier for the content under control. This identifier must *uniquely* identify an elementary stream.

2. CMPO_count: [1...*n*] CMPS types supported by this **CMPO**.

3. CMPS_type_identifiers: [1...*n*] identifiers, each with an offset in the stream and a length. The offset points to the byte in the CMPO where the data for that CMPS type is found. The length is the length in bytes of this data.

4. Data segments: *n* data segments. Each data segment is in a format that is proprietary to the CMPS supplier.

5. CMPO_URL: An optional URL that references an additional CMPO that adds information to the information in this CMPO. (This is a way of dynamically adding support for new CMPSs.)

c. Feedback Event

The feedback events come in two forms: start and end. Each feedback event contains three pieces of information:

1. Elementary_stream_ID

2. Time: in presentation time

3. Object_instance_number

User Interface.

Commerce Appliance 2301 may include User Interface 2304 designed to convey control-related information to the user and to receive commands and information from the user. This interface may include special purpose displays (e.g., a light which comes on if a current action requires payment), special purpose buttons (e.g., a button which accepts the payment or other terms required for display of content), and/or visual information presented on screen.

Example of Operation in an MPEG-4 Context

1. User selects a particular work or channel. The user may, for example, use a remote control device to tune a digital TV to a particular channel.

2. Selection of the channel is communicated to a CMPS arrangement, which uses the information to either download a CCMPO or to identify a previously downloaded CCMPO (e.g., if the CMPS arrangement is contained in a set-top box, the set-top box may automatically download CCMPOs for every channel potentially reachable by the box).

3. The CMPS arrangement uses the CCMPO to identify rules associated with all content found on the channel. For example, the CCMPO may specify that content may only be viewed by subscribers, and may specify that, if the user is not a subscriber, an advertisement screen should be put up inviting the user to subscribe.

4. Once rules specified by the CCMPO have been satisfied, the CCMPO specifies the location of a MCMPO associated with a particular work which is available on the channel. The channel CMPO may also supply one or more keys used for decryption of the MCMPO.

5. The CMPS arrangement downloads the MCMPO. In the case of an MPEG-4 embodiment, the MCMPO may be an Elementary Stream. This Elementary Stream must be identifiable at a relatively early stage in the MPEG-4 decoding process.

6. The CMPS arrangement decrypts the MCMPO, and determines the rules used to access and use the content. The CMPS arrangement presents the user with a set of options, including the ability to view for free with advertisements, or to view for a price without advertisements.

7. The user selects view for free with advertisements, e.g., by highlighting and selecting an option on the screen using a remote control device.

8. The CMPS arrangement acquires one or more keys from the MCMPO and uses those keys to decrypt the ESs associated with the video. The CMPS arrangement identifies two possible scene descriptor graphs, one with and one without advertisements. The CMPS arrangement passes the scene descriptor graph with advertisements through, and blocks the other scene descriptor graph.

9. The CMPS arrangement monitors the composite and render block, and checks to determine that the advertisement AVOs have actually been released for viewing. If the CMPS arrangement determines that those AVOs have not been released for viewing, it puts up an error or warning message, and terminates further decryption.

CMPS Rights Management In Provider And Distribution Chains

In addition to consumer arrangements, in other embodiments one or more CMPSs

may be used in creating, capturing, modifying, augmenting, animating, editing, excerpting, extracting, embedding, enhancing, correcting, fingerprinting, watermarking, and/or rendering digital information to associate rules with digital information and to enforce those rules throughout creation, production, distribution, display and/or performance processes.

In one non-limiting example, a CMPS, a non-exhaustive example of which may include a least a secure portion of a VDE node as described in the aforementioned Ginter et al., patent specification, is incorporated in video and digital cameras, audio microphones, recording, playback, editing, and/or noise reduction devices and/or any other digital device. Images, video, and/or audio, or any other relevant digital information may be captured, recorded, and persistently protected using at least one CMPS and/or at least one CMPO. CMPSs may interact with compression/decompression, encryption/decryption, DSP, digital to analog, analog to digital, and communications hardware and/or software components of these devices as well.

In another non-exhaustive example, computer animation, special effects, digital editing, color correcting, noise reduction, and any other applications that create and/or use digital information may protect and/or manage rights associated with digital information using at least one CMPS and/or at least one CMPO.

Another example includes the use of CMPSs and/or CMPOs to manage digital assets in at least one digital library, asset store, film and/or audio libraries, digital vaults, and/or any other digital content storage and management means.

In accordance with the present applications, CMPSs and/or CMPOs may be used to manage rights in conjunction with the public display and/or performance of digital works. In one non-exhaustive example, flat panel screens, displays, monitors, TV projectors, LCD projectors, and/or any other means of displaying digital information, may incorporate at least one hardware and/or software CMPS instance that controls the use of digital works. A CMPS may allow use only in conjunction with one or more digital credentials, one example of which is a digital certificate, that warrant that use of the digital information will occur in a setting, location, and/or other context for public display and/or performance. Non-limiting examples of said contexts include theaters, bars, clubs, electronic billboards, electronic displays in public areas, or TVs in airplanes, ships, trains and/or other public conveyances. These credentials may be issued by trusted third parties such as certifying authorities, non-exhaustive examples of which are disclosed in the aforementioned Ginter '712 patent application.

Additional MPEG-4 Embodiment Information

This work is based on the MPEG-4 description in the version 1 Systems Committee Draft (CD), currently the most complete description of the evolving MPEG-4 standard.

This section presents the structural modifications to the MPEG-4 player architecture and discusses the data lines and the concomitant functional changes. Figure 23 shows the functional components of the original MPEG-4 player. Content arrives at Player 2301 packaged into a serial stream (e.g., MPEG-4 Bit Stream 2314). It is demultiplexed via a sequence of three demultiplexing stages (e.g., Demux 2305) into elementary streams. There are three principle types of elementary streams: AV Objects (AVO), Scene Descriptor Graph (SDG), and Object Descriptor (OD). These streams are fed into respective processing elements (e.g., AVO Decode 2307, Scene Descriptor Graph 2306, Object Descriptors 2308). The AVOs are the multimedia content streams such as audio, video, synthetic graphics and so on. They are processed by the player's compression/coding subsystems. The scene descriptor graph stream is used to build the scene descriptor graph. This tells Composite and Render 2309 how to construct the scene and can be thought of as the "script." The object descriptors contain description information about the AVOs and the SD-graph updates.

To accommodate a CMPS (e.g., CMPS 2302) and to protect content effectively, the player structure must be modified in several ways:

- Certain data paths must be rerouted to and from the CMPS
- Certain buffers in the SDG, AVO decode and Object descriptor modules must be secured
- Feedback paths from the user and the composite and render units to the CMPS must be added

In order for CMPS 2302 to communicate with the MPEG-4 unit, and for it to effectively manage content we must specify the CMPO structure and association protocols and we must define the communication protocols over the feedback systems (from the compositor and the user.)

The structural modifications to the player are shown in Figure 23. The principal changes are:

- All elementary streams are now routed through CMPS 2302.
- Direct communication path between Demux 2305 and CMPS 2302.
- A required "Content Release and Decrypt" Module 2315 in CMPS 2302.

- The addition of a feedback loop (e.g., Line 2313) from Composite and Render 2309 to CMPS 2302.
- Bi-directional user interaction directly with the CMPS 2302, through Line 2316.

Furthermore, for M4v2P, CMP-objects are preferably associated with all elementary streams. Elementary streams that the author chooses not to protect are still marked by an "unprotected content" CMPO. The CMPOs are the primary means of attaching rules information to the content. Content here not only refers to AVOs, but also to the scene descriptor graph. Scene Descriptor Graph may have great value and will thus need to be protected and managed by CMPS 2302.

The direct path from Demux 2305 to CMPS 2302 is used to pass a CMPS specific header, that potentially contains business model information, that communicates business model information at the beginning of user session. This header can be used to initiate user identification and authentication, communicate rules and consequences, and initiate up-front interaction with the rules (selection of quality-of-service (QoS), billing, etc.) The user's communication with CMPS 2302 is conducted through a *non-standardized* channel (e.g., Line 2316). The CMPS designer may provide an independent API for framing these interactions.

Feedback Path 2313 from Composite and Render block 2309 serves an important purpose. The path is used to cross check that the system actually presented the user with a given scene. Elementary streams that are processed by their respective modules may not necessarily be presented to the user. Furthermore, there are several fraud scenarios wherein an attacker could pay once and view multiple times. The feedback path here allows CMPS 2302 to cross check the rendering and thereby perform a more accurate accounting. This feedback is implemented by forcing the Composite and Render block 2309 to issue a *start event* that signals the initiation of a given object's rendering that is complemented by a *stop event* upon termination. The feedback signaling process may be made optional by providing a CMP-notification flag that may be toggled to indicate whether or not CMPS 2302 should be notified. All CMPOs would be required to carry this flag.

The final modification to the structure is to require that the clear text buffers in the AVO, SDG and Object Descriptor processors and in the Composite-and-Render block be secured. This is to prevent a pirate from stealing content in these buffers. As a practical matter, this may be difficult, since tampering with these structures may well destroy synchronization of the streams. However, a higher state of security would come from placing these buffers into a protected processing environment.

CMPS 2302 *governs* the functioning of Player 2301, consistent with the following:

- Communication mechanism between CMPS 2302 and the MPEG-4 player (via CMPOs)
- A content release and decryption subsystem
- Version authentication subsystem
- Sufficient performance so as not to interfere with the stream processing in the MPEG-4 components

CMPS 2302 may have a bi-directional side-channel that is external to the MPEG-4 player that may also be used for the exchange of CMP information. Furthermore, the CMPS designer may choose to provide a user interface API that provides the user with the ability to communicate with the content and rights management side of the stream management (e.g., through Line 2316).

Encrypted content is decrypted and released by CMPS 2302 as a function of the rules associated with the protected content and the results of user interaction with CMPS 2302. Unencrypted content is passed through CMPS 2302 and is governed by associated rules and user interaction with CMPS 2302. As a consequence of these rules and user interaction, CMPS 2302 may need to transact with the SDG and AVO coding modules (e.g., 2310, 2311) to change scene structure and/or the QoS grade.

Ultimately, the CMPS designer may choose to have CMPS 2302 generate audit trail information that may be sent to a clearinghouse authority via CMPS Side Channel Port 2318 or as encrypted content that is packaged in the MPEG-4 bit stream.

The MPEG-4 v1 Systems CD uses the term "object" loosely. In this document, "object" is used to specifically mean a data structure that flows from one or more of the data paths in Figure 23.

Using multiple SD-graph update streams, each with its own CMPO, allows an author to apply arbitrarily specific controls to the SD-graph. For example, each node in the SD-graph can be created or modified by a separate SD-graph update stream. Each of these streams will have a distinct CMPO and ID. Thus, the CMPS can release and decrypt the creation and modification of each node and receive feedback information for each node individually. The practical implications for controlling release and implementing consequences should be comparable to having a CMPO on each node of the SD-graph, without the costs of having a CMPO on each SD-graph node.

Principles consistent with the present invention may be illustrated using the following examples:

In the first example, there is a bilingual video with either an English or French soundtrack. The user can choose during playback to hear either the English or French. The

basic presentation costs $1. If the French soundtrack is presented there is a $0.50 surcharge. If the user switches back and forth between French and English, during a single viewing of the presentation, the $0.50 surcharge will occur only once.

In this example, there will be four elementary streams:

The Scene Description Graph Update stream will have a CMPO. The CMPO will imply a $1.00 fee associated with the use of the content. The scene description graph displays the video, English audio and puts up a button that allows the user to switch to French. If the user clicks that button, the English stops, the French picks up from that point and the button changes to a switch-to-English button. (Optionally, there may be a little dialog at the beginning to allow the user to select the initial language. This is all easy to do in the SD graph.)

The Video Stream with the CMPO will say that it can only be released if the scene description graph update stream above is released.

The English Audio Stream will be similar to the Video stream.

The French Audio Stream will be similar to the Video stream but there is a $.50 charge it if is seen in the feedback channel. (The CMPS must to not count twice if the user switches between the two in a single play of the presentation.)

An important requirement is that the ID for the SD-graph update stream appears in the feedback path (e.g., Feedback Path 2313). This is so CMPS 2302 knows when the presentation stops and ends so that CMPS 2302 can correctly bill for the French audio.

The rules governing the release of the video and audio streams may include some variations. The rules for these streams, for example, may state something like "if you don't see the id for the scene description graph update stream X in the feedback channel, halt release of this stream." If the main presentation is not on the display, then the video should not be. This ties the video to this one presentation. Using the video in some other presentation would require access to the original video, not just this protected version of it.

In a second example, an author wants to have a presentation with a free attract sequence or "trailer". If the user clicks the correct button the system moves into the for-fee presentation, which is organized as a set of "acts".

Multiple SD-graph update streams may update a scene description graph. Multiple SD-graph update streams may be open in parallel. The time stamps on the ALUs in the streams are used to synchronize and coordinate.

The trailer and each act are represented by a separate SD-graph update stream with a separate CMPO. There is likely an additional SD-graph update stream that creates a simple

root node that is invisible and silent. This node brings in the other components of the presentation as needed.

The foregoing description of implementations of the invention has been presented for purposes of illustration and description. It is not exhaustive and does not limit the invention to the precise form disclosed. Modifications and variations are possible in light of the above teachings or may be acquired from practicing of the invention. For example, the described implementation includes software but the present invention may be implemented as a combination of hardware and software or in hardware alone. The invention may be implemented with both object-oriented and non-object-oriented programming systems. The scope of the invention is defined by the claims and their equivalents.

We claim:

1. A streaming media player providing content protection and digital rights management, including:

a port configured to receive a digital bit stream, the digital bit stream including:

content which is encrypted at least in part, and

a secure container including control information for controlling use of the content, including at least one key suitable for decryption of at least a portion of the content; and

a control arrangement including:

means for opening secure containers and extracting cryptographic keys, and

means for decrypting the encrypted portion of the content.

2. The player of Claim 1 in which the digital bit stream includes at least two sub-streams which have been muxed together, at least one of the sub-streams including compressed information, and

wherein the player further includes:

a demux designed to separate and route the sub-streams;

a decompression unit configured to decompress at least one of the sub-streams, the decompression unit and the demux being connected by a pathway for the transmission of information; and

a rendering unit designed to process decompressed content information for rendering.

3. The player of Claim 2, further including:

a stream controller operatively connected to the decompression unit, the stream controller including decryption functionality configured to decrypt at least a portion of a sub-stream and pass the decrypted sub-stream to the decompression unit.

4. The player of Claim 3, further including:

a path between the control arrangement and the stream controller to enable the control arrangement to pass at least one key to the stream controller for use with the stream controller's decryption functionality.

5. The player of Claim 4, further including:

a feedback path from the rendering unit to the control arrangement to allow the control arrangement to receive information from the rendering unit regarding the identification of objects which are to be rendered or have been rendered.

6. The player of Claim 1, wherein the digital bit stream is encoded in MPEG-4 format.

7. The player of Claim 1, wherein the digital bit stream is encoded in MP3 format.

8. The player of Claim 4, wherein the control arrangement contains a rule or rule set associated with governance of at least one sub-stream or object.

9. The player of Claim 8, wherein the rule or rule set is delivered from an external source.

10. The player of Claim 9, wherein the rule or rule set is delivered as part of the digital bit stream.

11. The player of Claim 8, wherein the rule or rule set specifies conditions under which the governed sub-stream or object may be decrypted.

12. The player of Claim 8, wherein the rule or rule set governs at least one aspect of access to or use of the governed sub-stream or object.

13. The player of Claim 12, wherein the governed aspect includes making copies of the governed sub-stream or object.

14. The player of Claim 12, wherein the governed aspect includes transmitting the governed sub-stream or object through a digital output port.

15. The player of Claim 14, wherein the rule or rule set specifies that the governed sub-stream or object can be transferred to a second device, but rendering of the governed sub-stream or object must be disabled in the first device prior to or during the transfer.

16. The player of Claim 15, wherein the second device includes rendering capability, lacks at least one feature present in the streaming media player, and is at least somewhat more portable than the streaming media player.

17. The player of Claim 11, wherein the control arrangement contains at least two rules governing access to or use of the same governed sub-stream or object.

18. The player of Claim 17, wherein a first of the two rules was supplied by a first entity, and the second of the two rules was supplied by a second entity.

19. The player of Claim 18, wherein the first rule controls at least one aspect of operation of the second rule.

20. The player of Claim 12, wherein the governed aspect includes use of at least one budget.

21. The player of Claim 12, wherein the governed aspect includes a requirement that audit information be provided.

22. The player of Claim 1, wherein the control arrangement includes tamper resistance.

23. A digital bit stream including:

content information that is compressed and at least in part encrypted; and

a secure container including

governance information for the governance of at least one aspect of access to or use of at least a portion of the content information; and

a key for decryption of at least a portion of the encrypted content information.

24. The digital bit stream of Claim 23, wherein the content information is encoded in MPEG-4 format.

25. The digital bit stream of Claim 23, wherein the content information is encoded in MP3 format.

26. A method of rendering a protected digital bit stream including:

receiving the protected digital bit stream,

passing the protected digital bit stream to a media player,

the media player reading first header information identifying a plugin used to process the protected digital bit stream, the first header information indicating that a first plugin is required;

the media player calling the first plugin;

the media player passing the protected digital bit stream to the first plugin;

the first plugin decrypting at least a portion of the protected digital bit stream;

the first plugin reading second header information identifying a second plugin necessary in order to render the decrypted digital bit stream;

the first plugin calling the second plugin;

the first plugin passing the decrypted digital bit stream to the second plugin;

the second plugin processing the decrypted digital bit stream, the processing including decompressing at least a portion of the decrypted digital bit stream;

the second plugin passing the decrypted and processed digital bit stream to the media player; and

the media player enabling rendering of the decrypted and processed digital bit stream,

whereby the first plugin may be used in an architecture not designed for multiple stages of plugin processing.

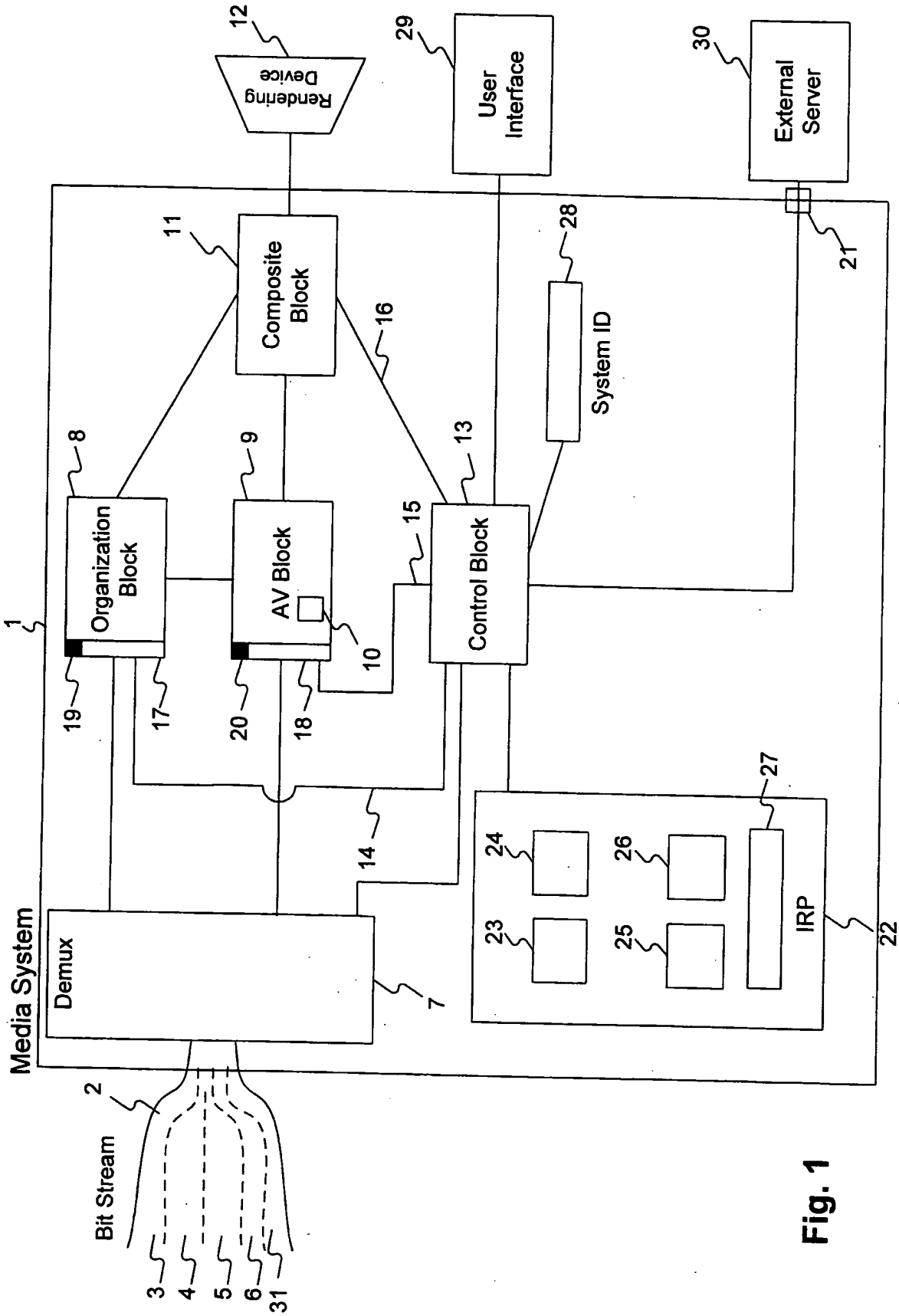

Media System
1
Rendering Device
12
User Interface
29
External Server
30
21
Composite Block
11
16
System ID
28
Organization Block
8
19
17
AV Block
9
20
18
10
Control Block
13
15
14
Demux
7
IRP
22
27
24
26
23
25
Bit Stream
2
3
4
5
6
31

Fig. 1

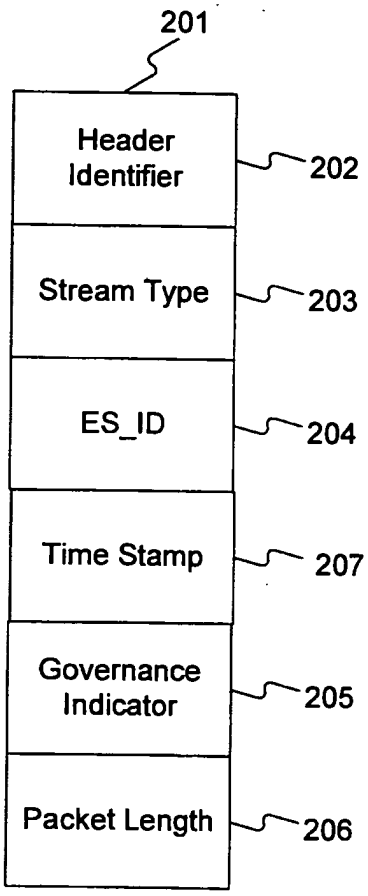

201
Header Identifier
202
Stream Type
203
ES_ID
204
Time Stamp
207
Governance Indicator
205
Packet Length
206

Fig. 2

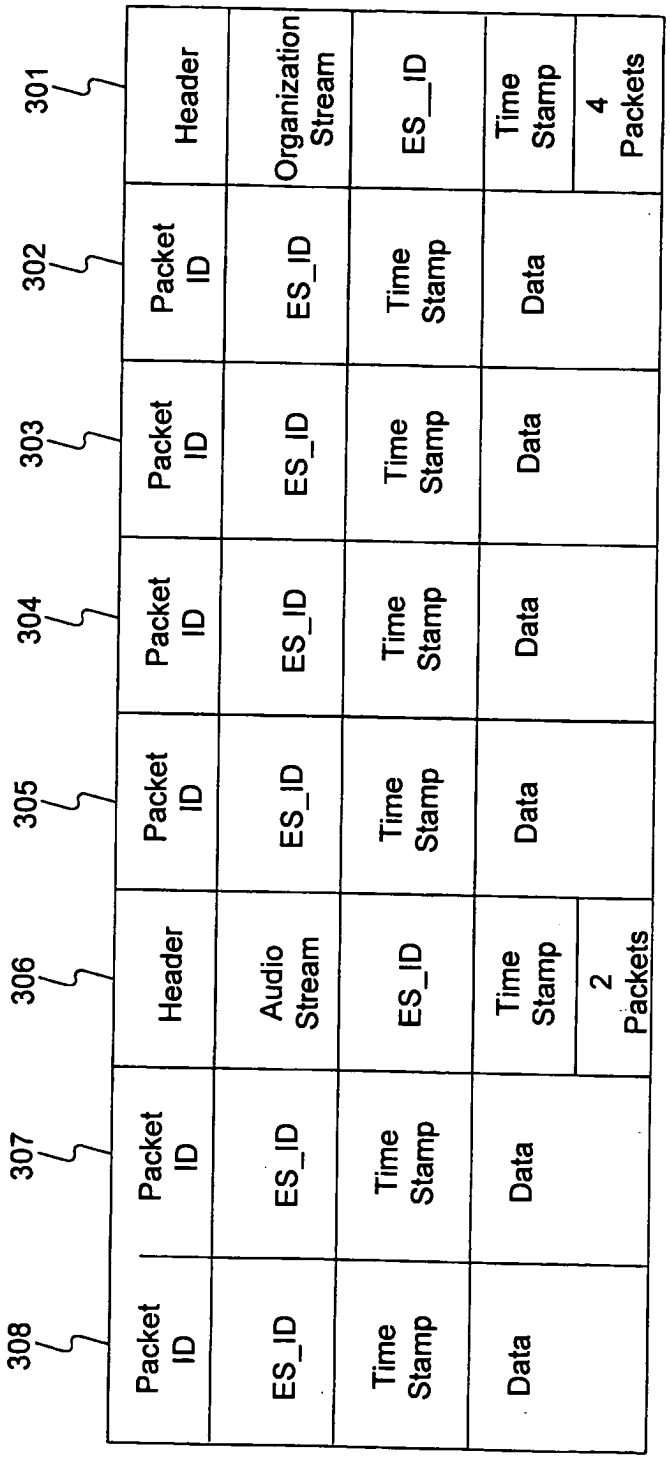

301
302
303
304
305
306
307
308
Header
Packet ID
Packet ID
Packet ID
Packet ID
Packet ID
Header
Packet ID
Packet ID
Organization Stream
ES_ID
ES_ID
ES_ID
ES_ID
Audio Stream
ES_ID
ES_ID
ES_ID
Time Stamp
Time Stamp
Time Stamp
Time Stamp
Time Stamp
Time Stamp
Time Stamp
Time Stamp
Data
Data
Data
Data
Data
Data
Data
Data
4 Packets
2 Packets

Fig. 3

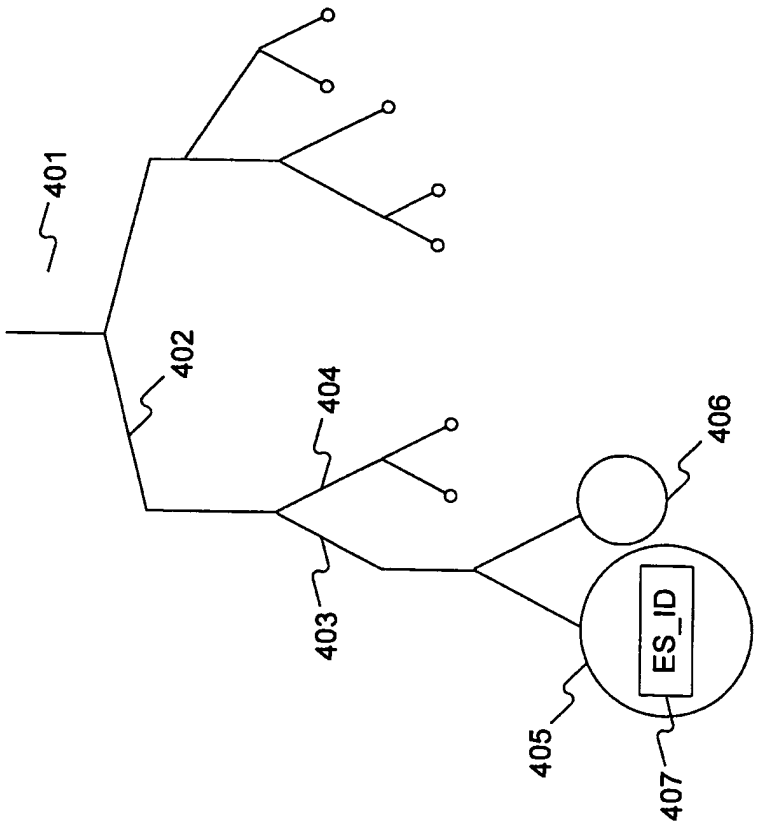


Fig. 4

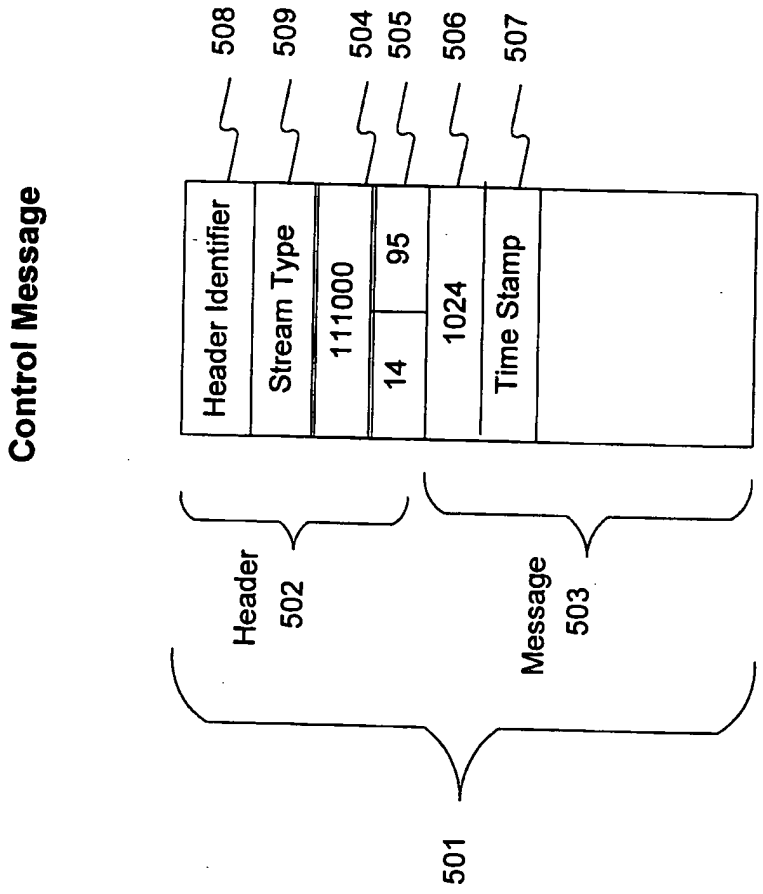

Control Message
508
509
504
505
506
507
Header Identifier
Stream Type
111000
14
95
1024
Time Stamp
Header
502
Message
503
501

Fig. 5

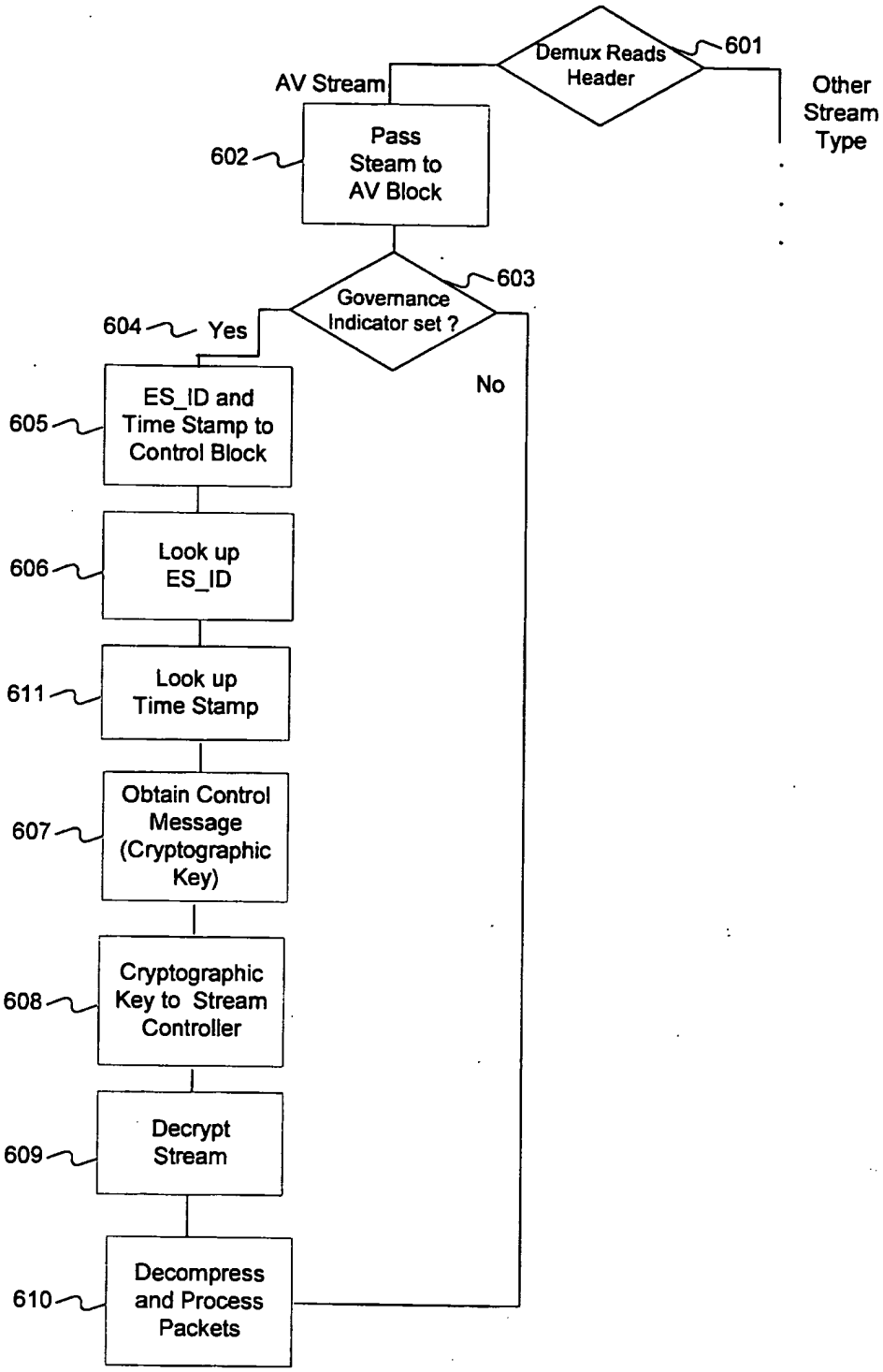

Demux Reads Header
601
AV Stream
Other Stream Type
Pass Steam to AV Block
602
Governance Indicator set ?
603
604
Yes
No
ES_ID and Time Stamp to Control Block
605
Look up ES_ID
606
Look up Time Stamp
611
Obtain Control Message (Cryptographic Key)
607
Cryptographic Key to Stream Controller
608
Decrypt Stream
609
Decompress and Process Packets
610

Fig. 6

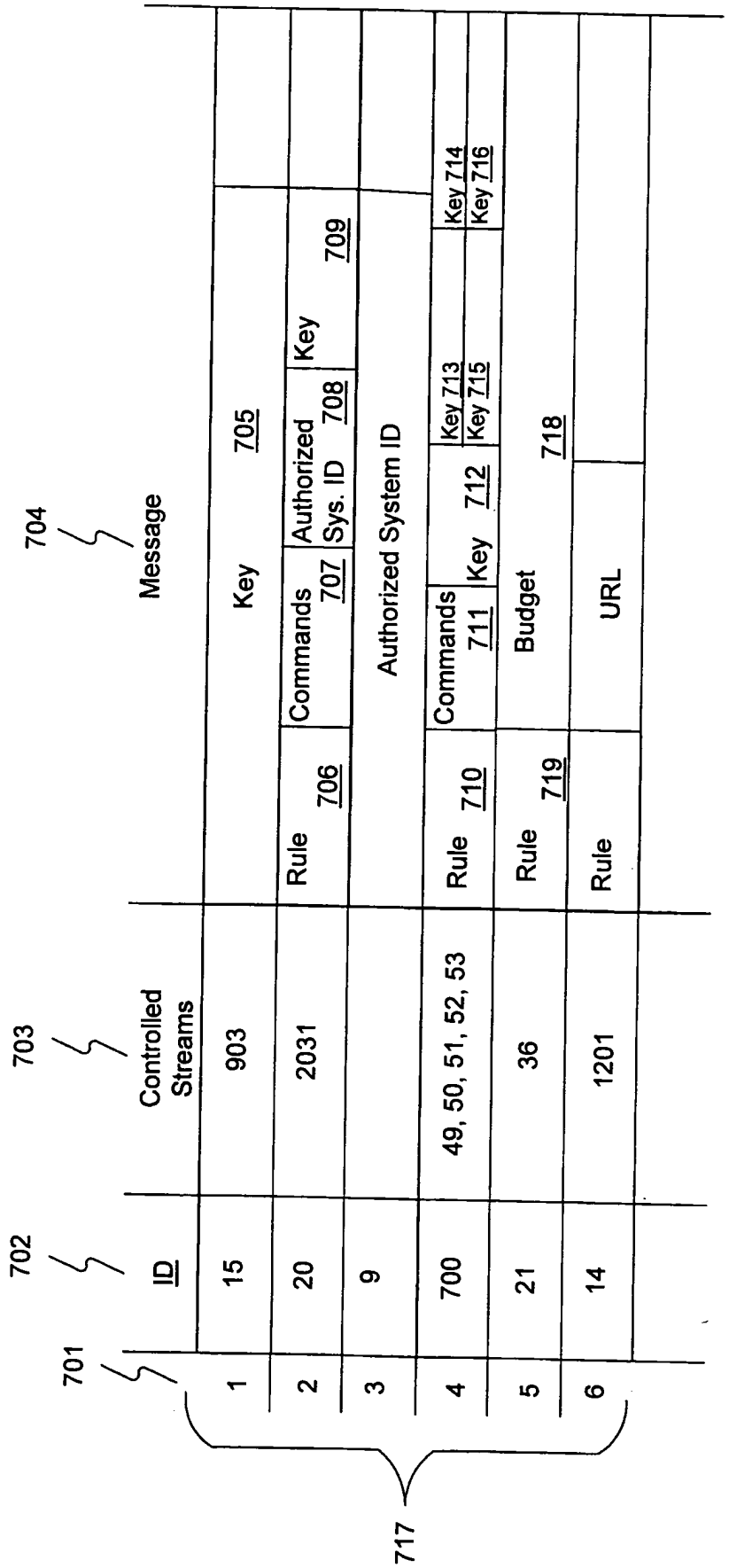

701
702
703
704
717
ID
Controlled Streams
Message
15
20
9
700
21
14
903
2031
49, 50, 51, 52, 53
36
1201
Key 705
Rule 706
Commands 707
Authorized Sys. ID 708
Key 709
Authorized System ID
Rule 710
Commands 711
Key 712
Key 713
Key 715
Key 714
Key 716
Rule 719
Budget 718
Rule
URL

Fig. 7

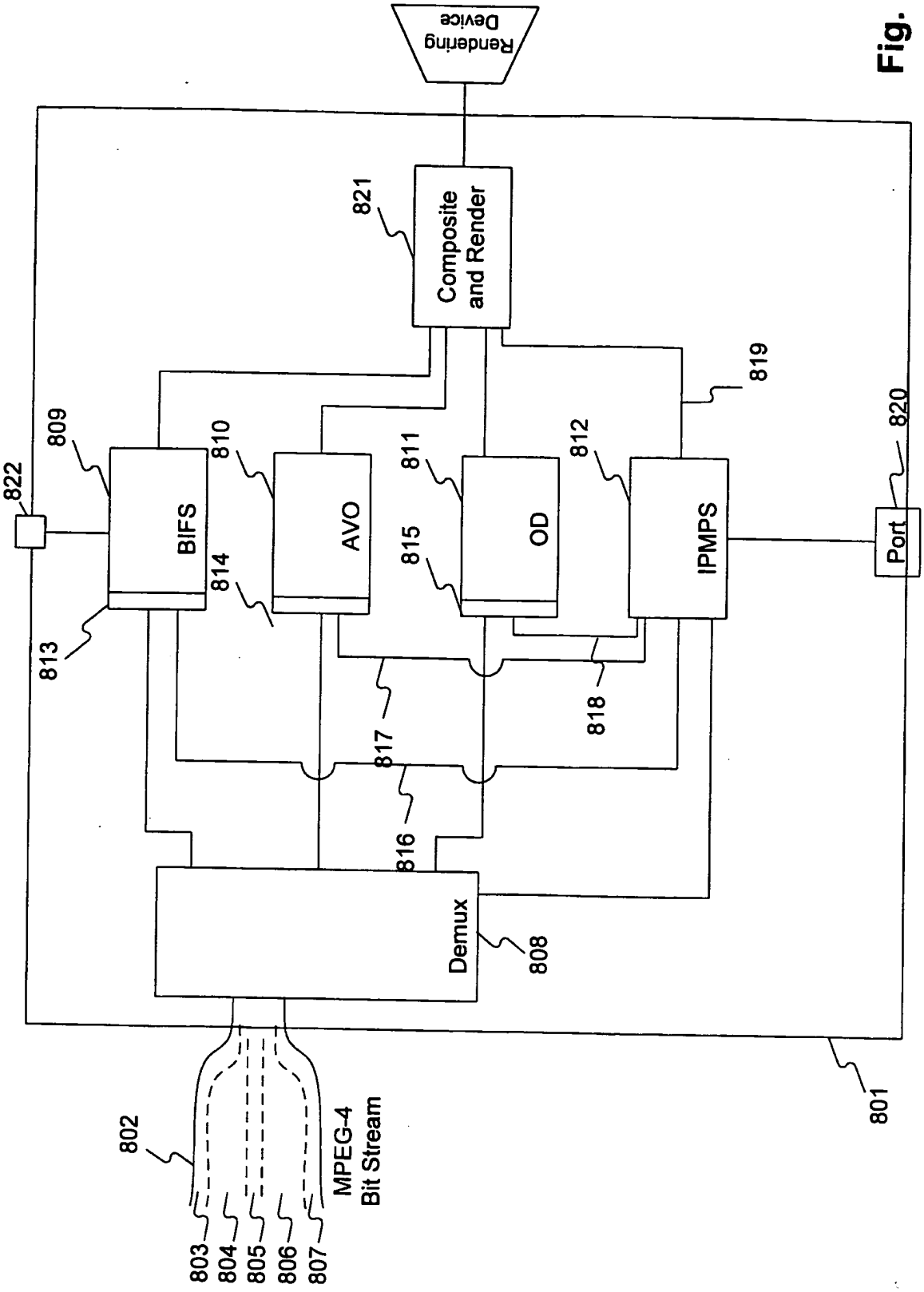

Rendering Device
Composite and Render
821
819
820
809
810
811
812
822
BIFS
AVO
OD
IPMPS
Port
814
815
813
818
817
816
Demux
808
801
802
MPEG-4 Bit Stream
803
804
805
806
807

Fig. 8

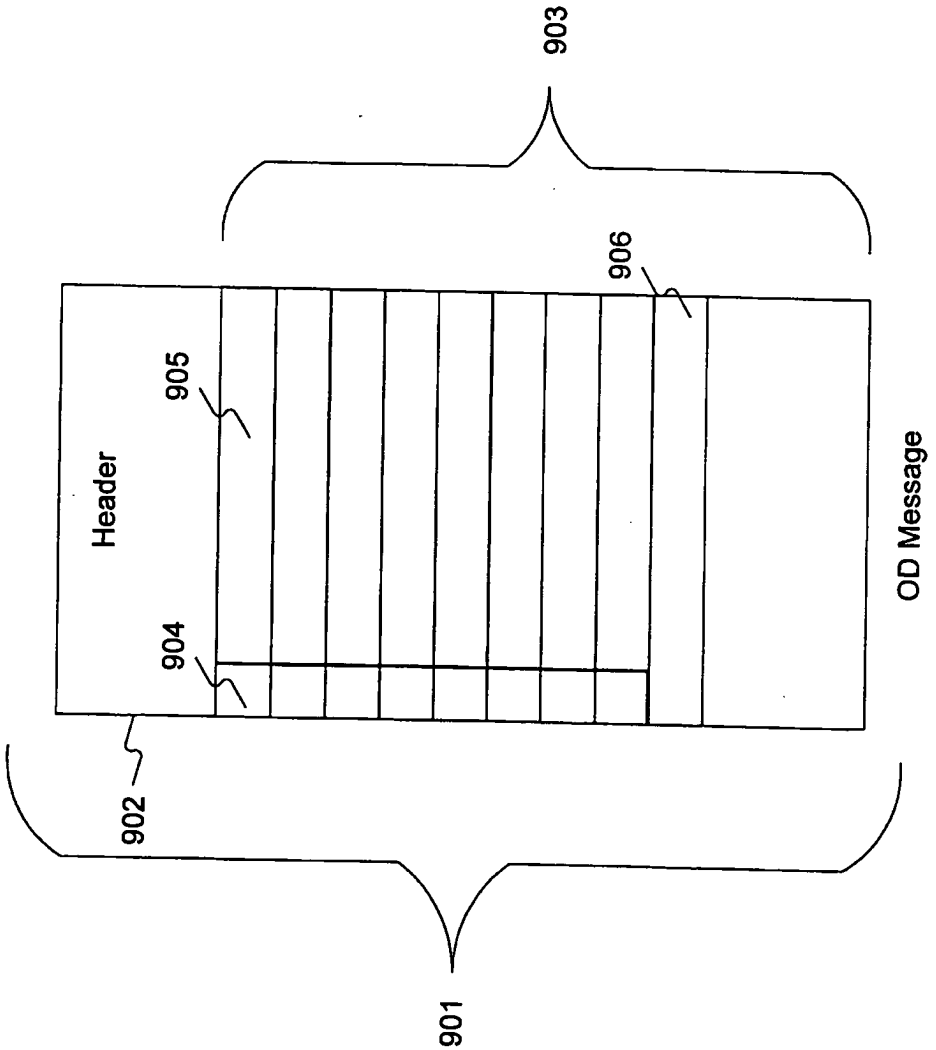


Fig. 9

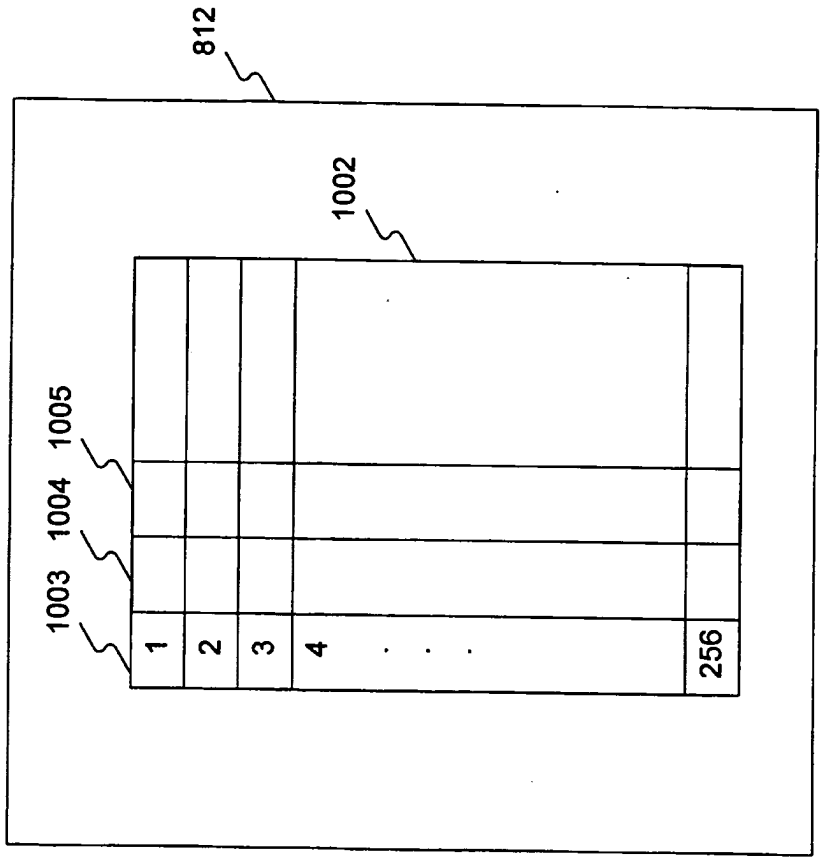


Fig. 10

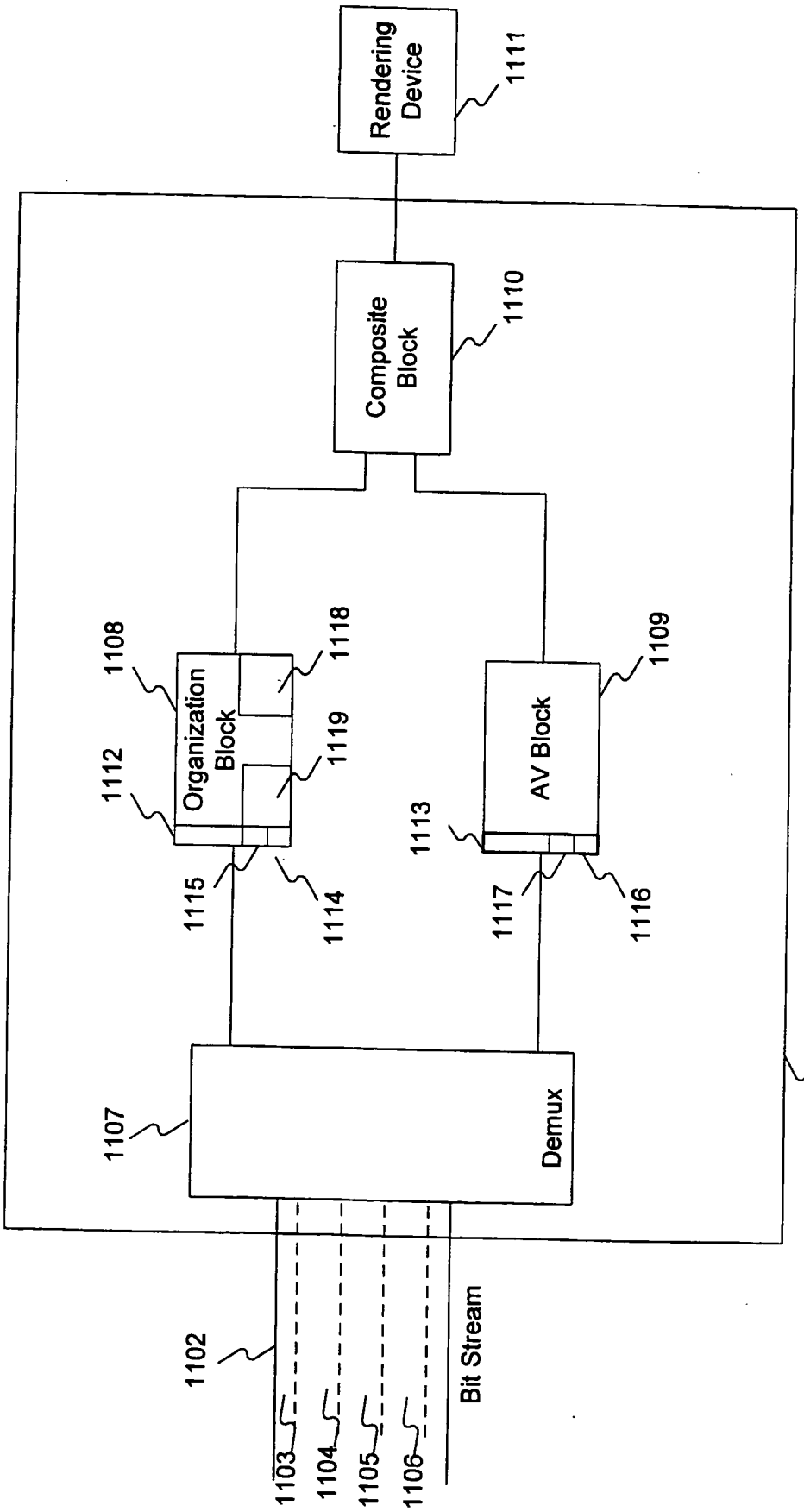

Rendering Device
1111
Composite Block
1110
Organization Block
1108
1118
1119
1112
1115
1114
AV Block
1109
1113
1117
1116
Demux
1107
1101
1102
Bit Stream
1103
1104
1105
1106

Fig. 11

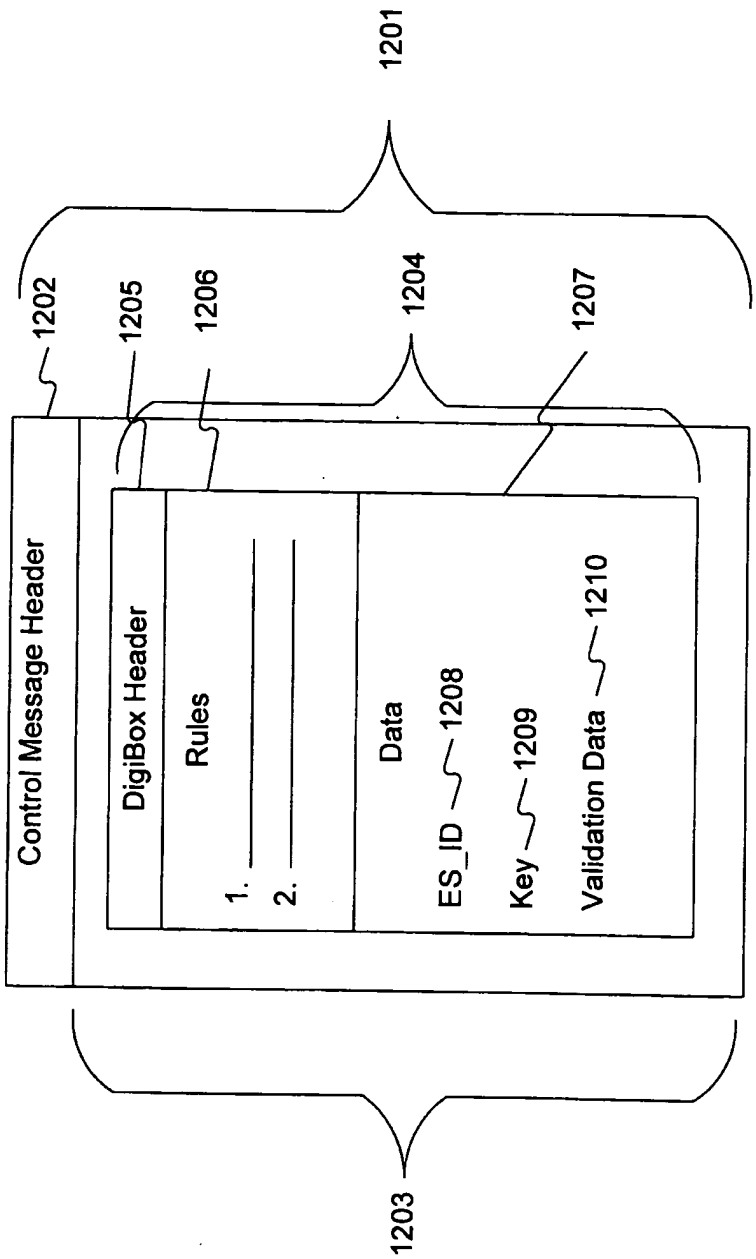

1201
1202
1205
1206
1204
1207
Control Message Header
DigiBox Header
Rules
1.
2.
Data
ES_ID
1208
Key
1209
Validation Data
1210
1203

Fig. 12

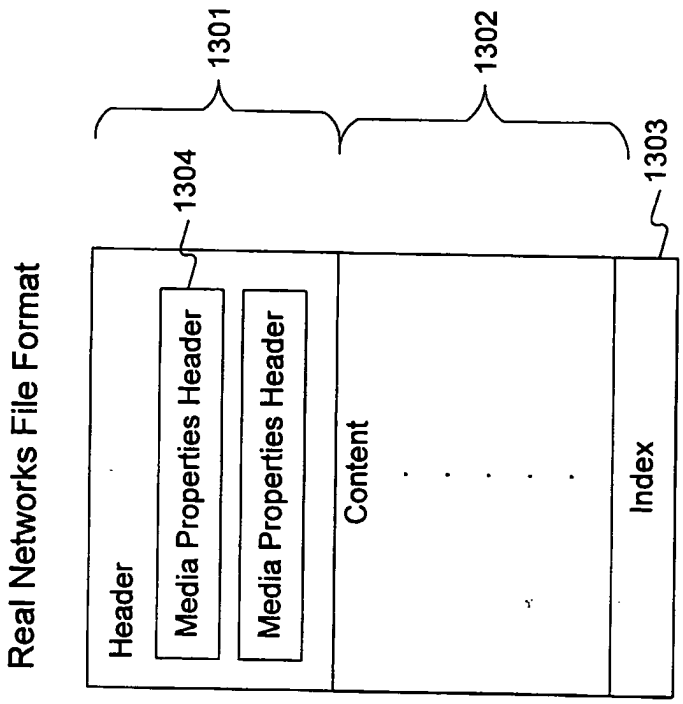

Real Networks File Format
Header
Media Properties Header
Media Properties Header
Content
Index
1301
1302
1303
1304

Fig. 13

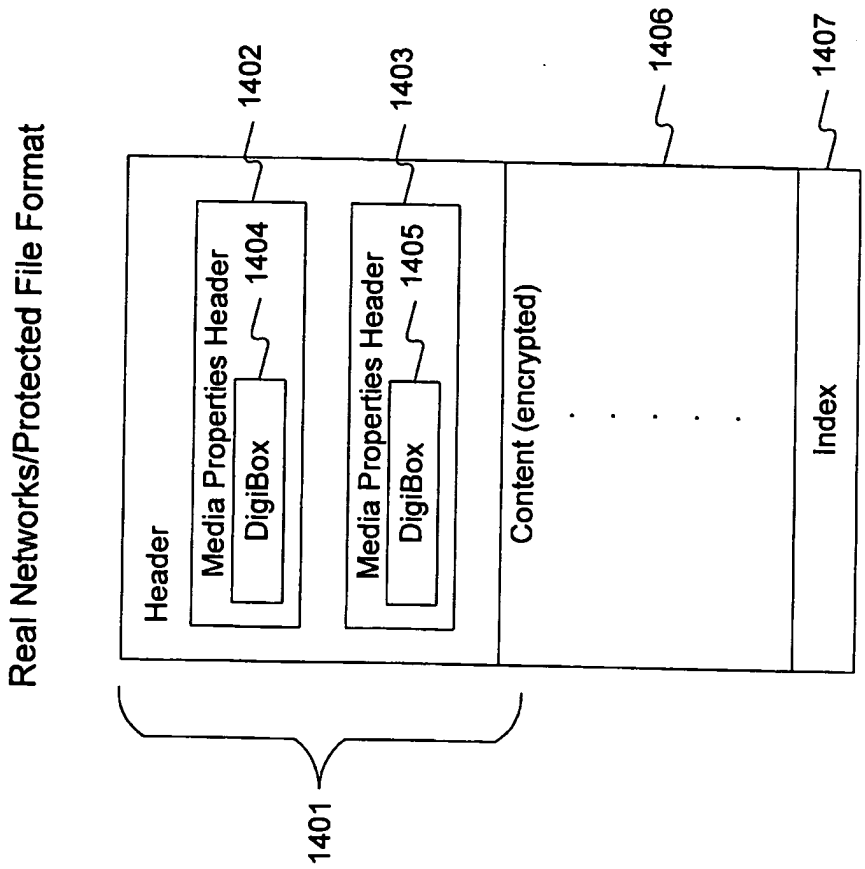

Real Networks/Protected File Format
1402
1403
1406
1407
Header
Media Properties Header
1404
DigiBox
Media Properties Header
1405
DigiBox
Content (encrypted)
Index
1401

Fig. 14

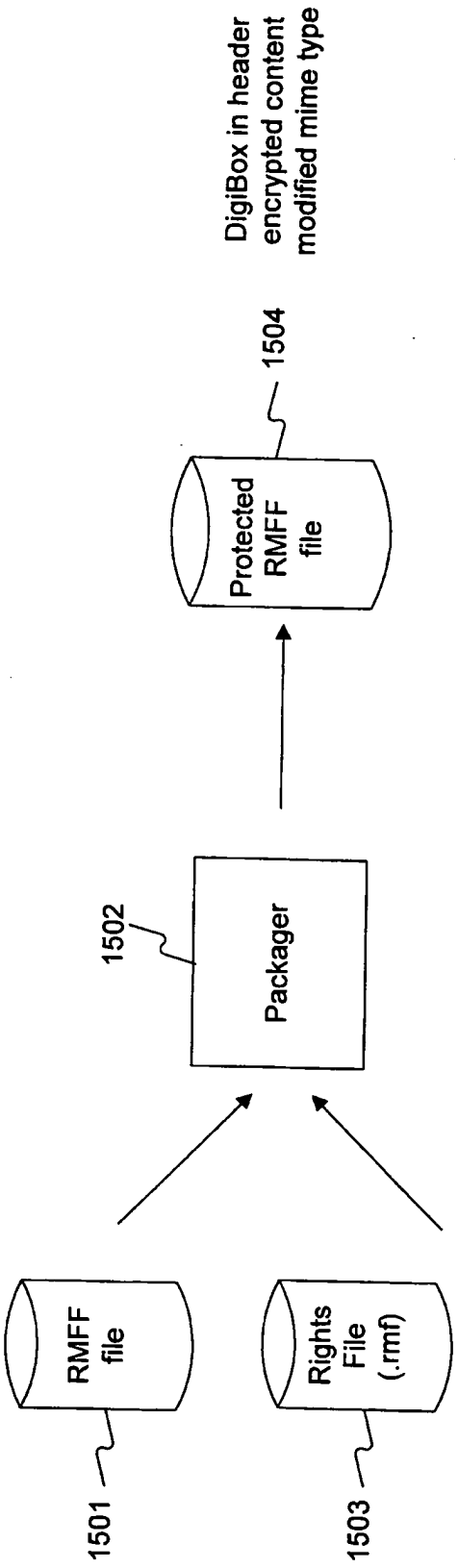

DigiBox in header
encrypted content
modified mime type
1504
Protected RMFF file
1502
Packager
RMFF file
Rights File (.rmf)
1501
1503

Fig. 15

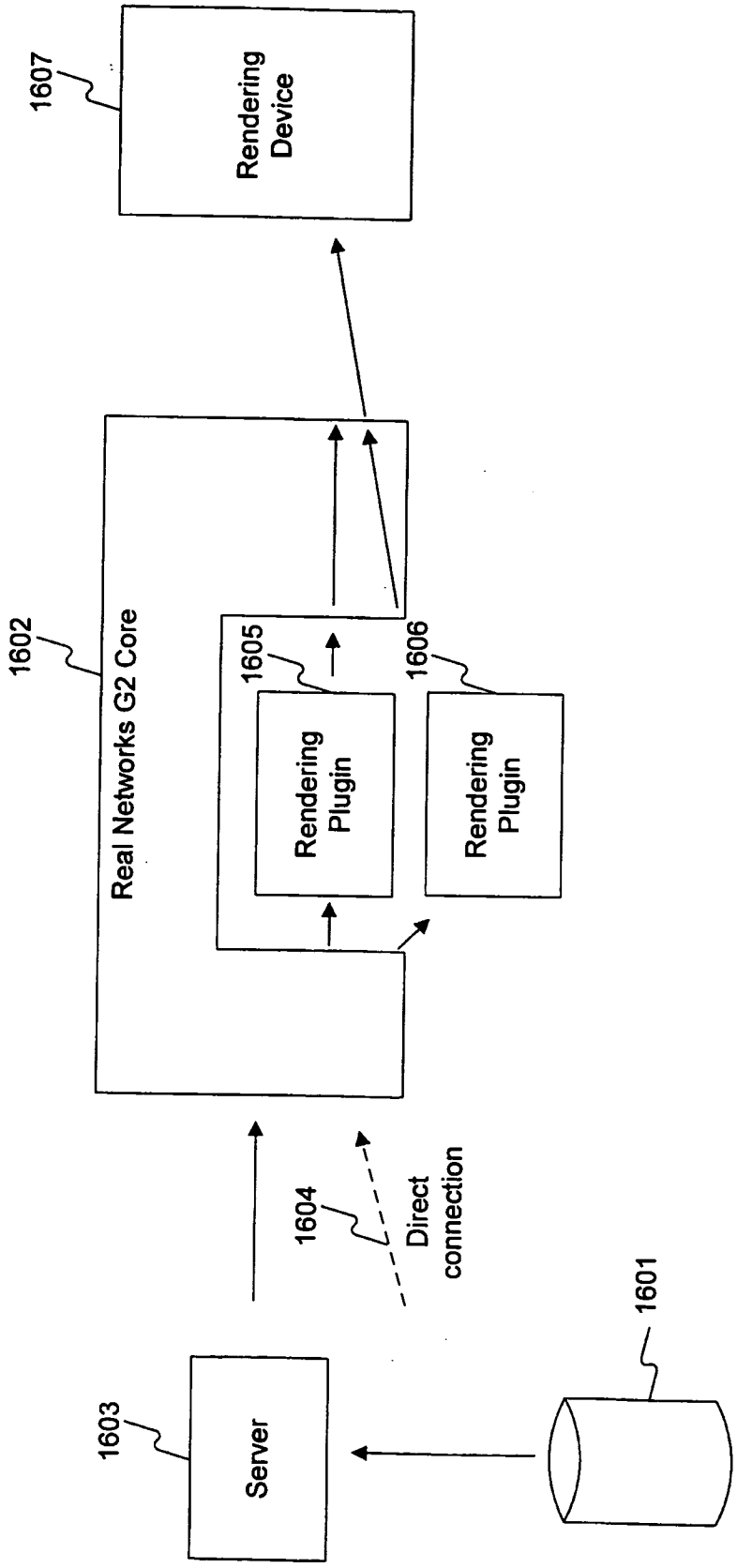

1607
Rendering Device
1602
Real Networks G2 Core
1605
Rendering Plugin
1606
Rendering Plugin
1604
Direct connection
1601
1603
Server

Fig. 16

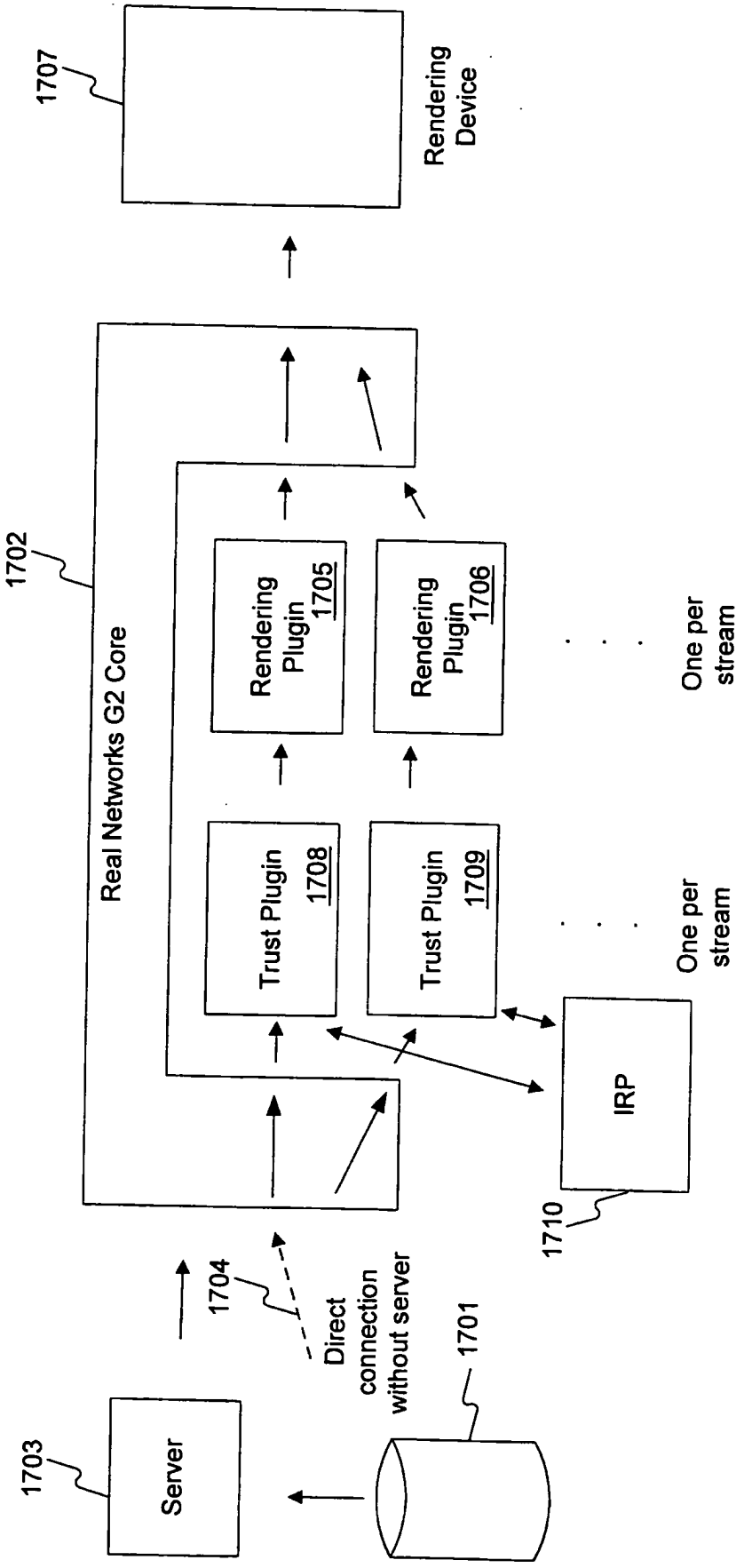

1707
Rendering Device
1702
Real Networks G2 Core
Rendering Plugin 1705
Rendering Plugin 1706
One per stream
Trust Plugin 1708
Trust Plugin 1709
One per stream
IRP
1710
1704
Direct connection without server
1701
1703
Server

Fig. 17

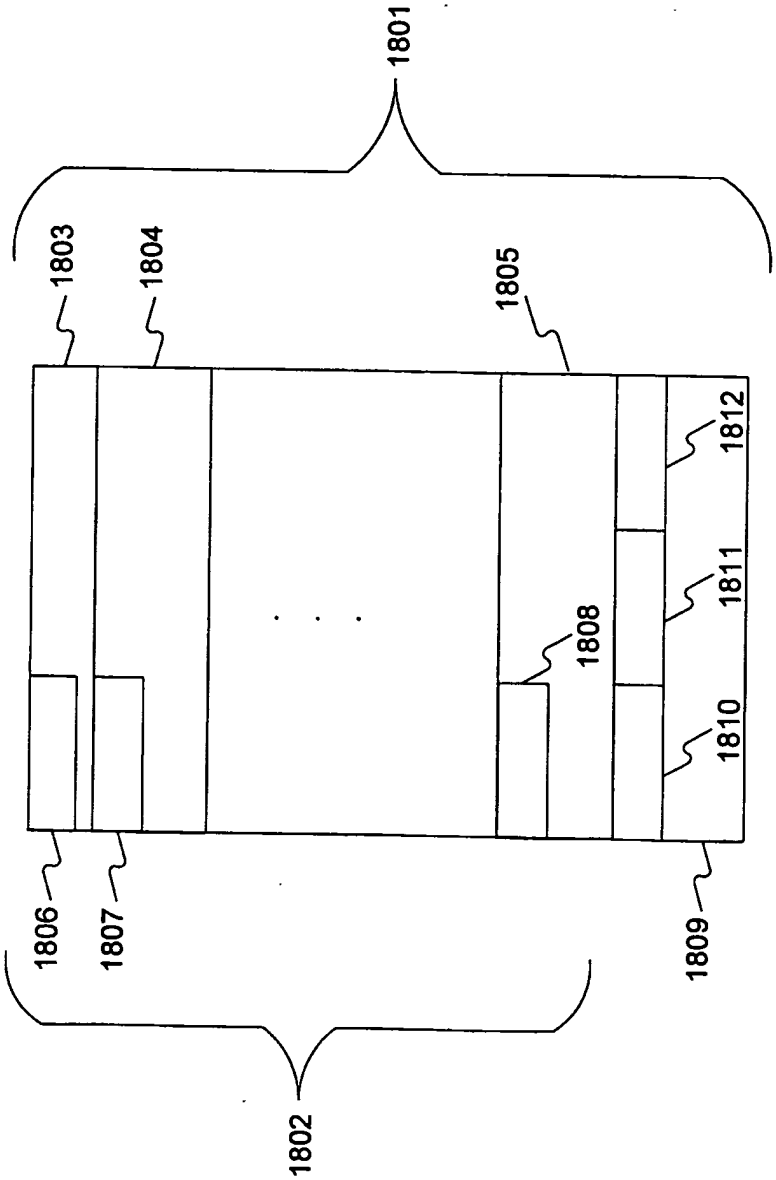
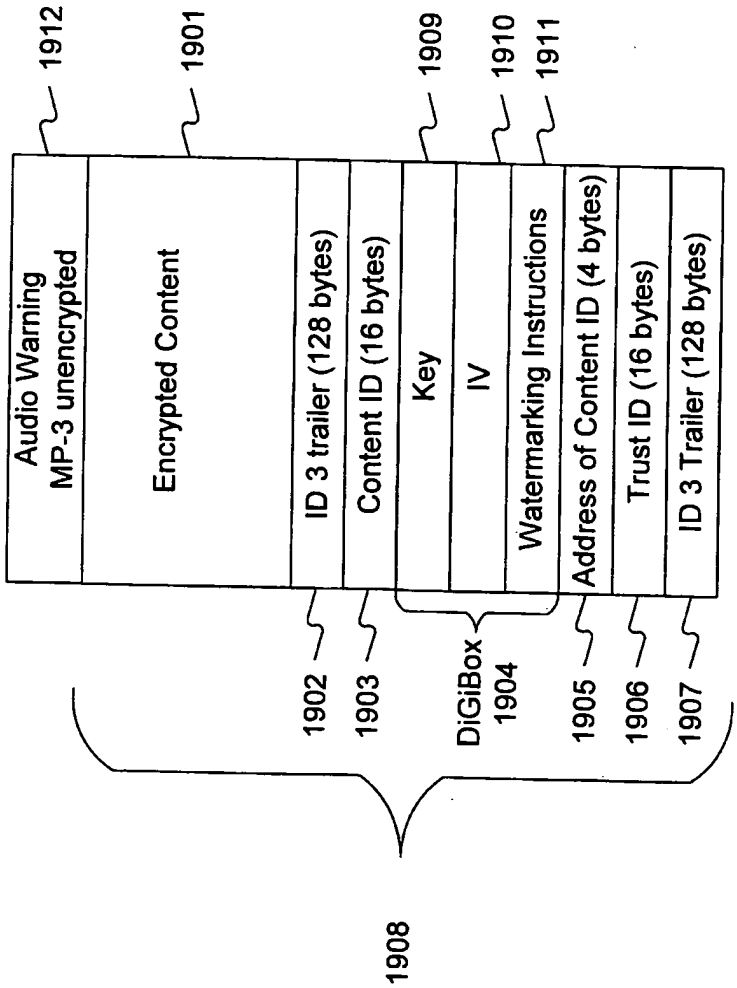


Fig. 18



Protected MP3 Format

Fig. 19

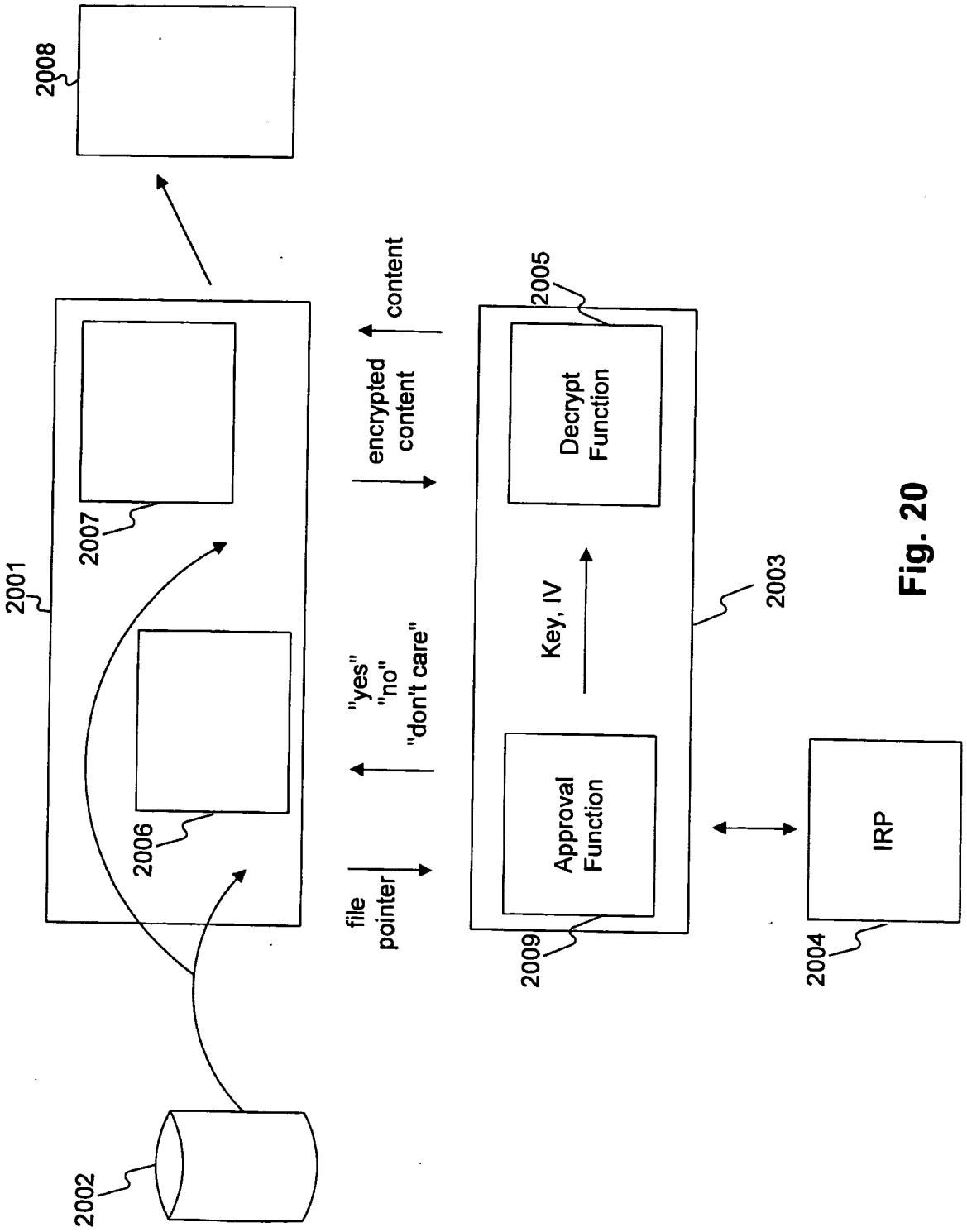

2008
2001
2007
2006
2002
content
encrypted content
"yes"
"no"
"don't care"
file pointer
2005
Decrypt Function
Key, IV
Approval Function
2009
2003
IRP
2004

Fig. 20

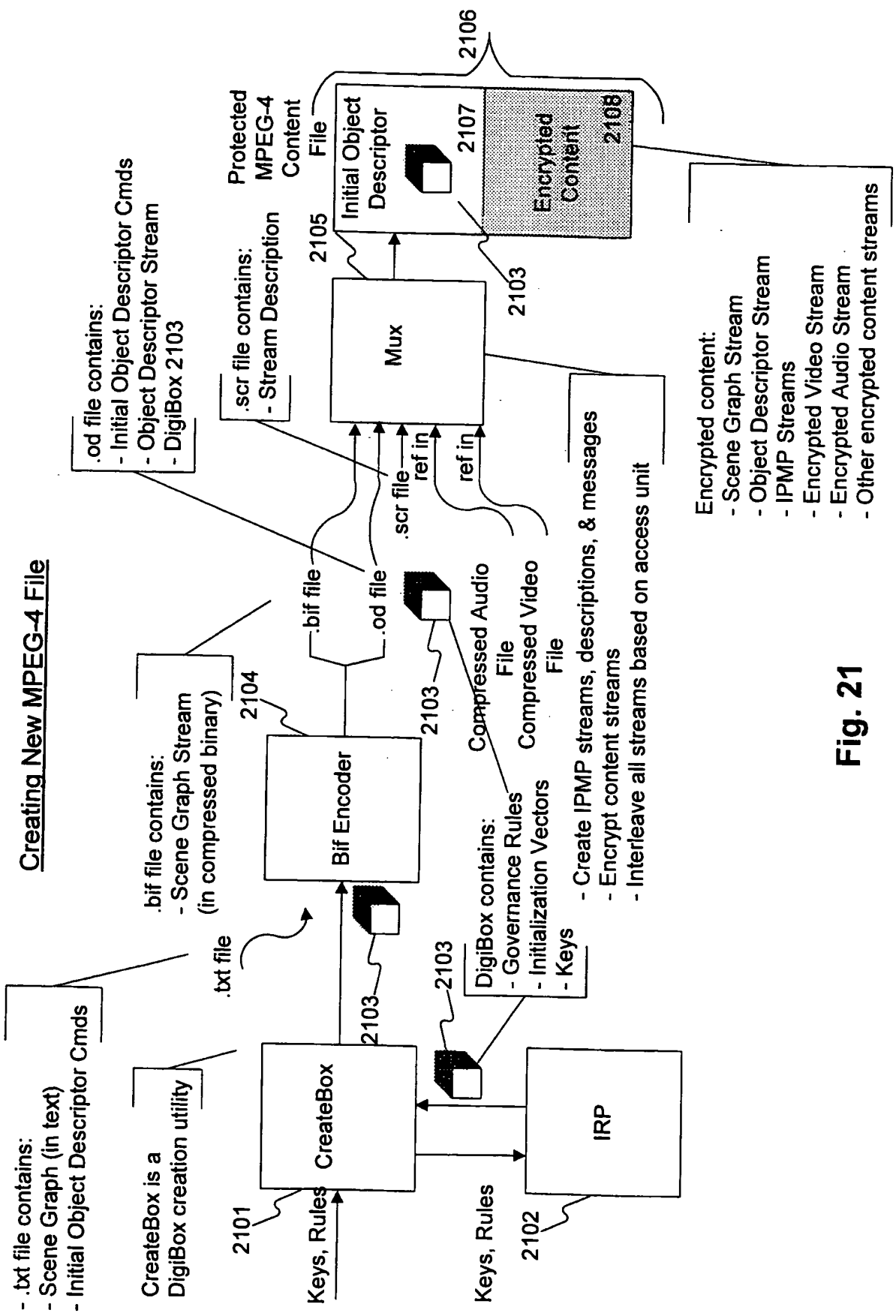

Creating New MPEG-4 File
- .txt file contains:
- Scene Graph (in text)
- Initial Object Descriptor Cmds
CreateBox is a DigiBox creation utility
CreateBox
2101
Keys, Rules
IRP
2102
Keys, Rules
.txt file
.bif file contains:
- Scene Graph Stream
(in compressed binary)
2104
Bif Encoder
2103
2103
2103
DigiBox contains:
- Governance Rules
- Initialization Vectors
- Keys
- Create IPMP streams, descriptions, & messages
- Encrypt content streams
- Interleave all streams based on access unit
Compressed Audio File
Compressed Video File
.bif file
.od file
.scr file
ref in
ref in
.od file contains:
- Initial Object Descriptor Cmds
- Object Descriptor Stream
- DigiBox 2103
.scr file contains:
- Stream Description
Mux
2103
2105
Protected MPEG-4 Content File
2106
Initial Object Descriptor
2107
Encrypted Content
2108
Encrypted content:
- Scene Graph Stream
- Object Descriptor Stream
- IPMP Streams
- Encrypted Video Stream
- Encrypted Audio Stream
- Other encrypted content streams

Fig. 21

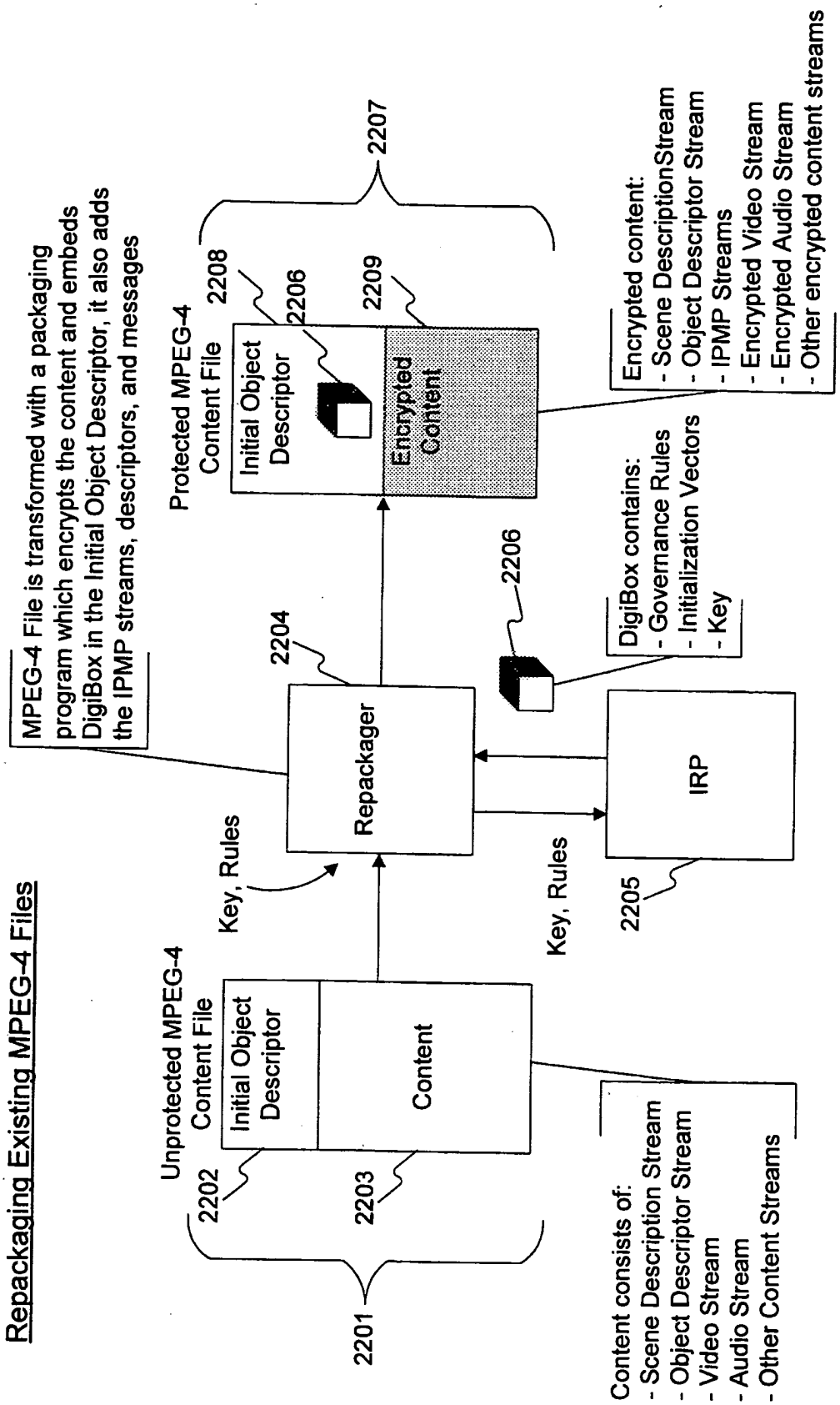

Repackaging Existing MPEG-4 Files
Unprotected MPEG-4 Content File
Initial Object Descriptor
Content
2202
2203
2201
Content consists of:
- Scene Description Stream
- Object Descriptor Stream
- Video Stream
- Audio Stream
- Other Content Streams
Key, Rules
Repackager
2204
MPEG-4 File is transformed with a packaging program which encrypts the content and embeds DigiBox in the Initial Object Descriptor, it also adds the IPMP streams, descriptors, and messages
Key, Rules
IRP
2205
2206
DigiBox contains:
- Governance Rules
- Initialization Vectors
- Key
Protected MPEG-4 Content File
Initial Object Descriptor
2208
2206
Encrypted Content
2209
2207
Encrypted content:
- Scene DescriptionStream
- Object Descriptor Stream
- IPMP Streams
- Encrypted Video Stream
- Encrypted Audio Stream
- Other encrypted content streams

Fig. 22

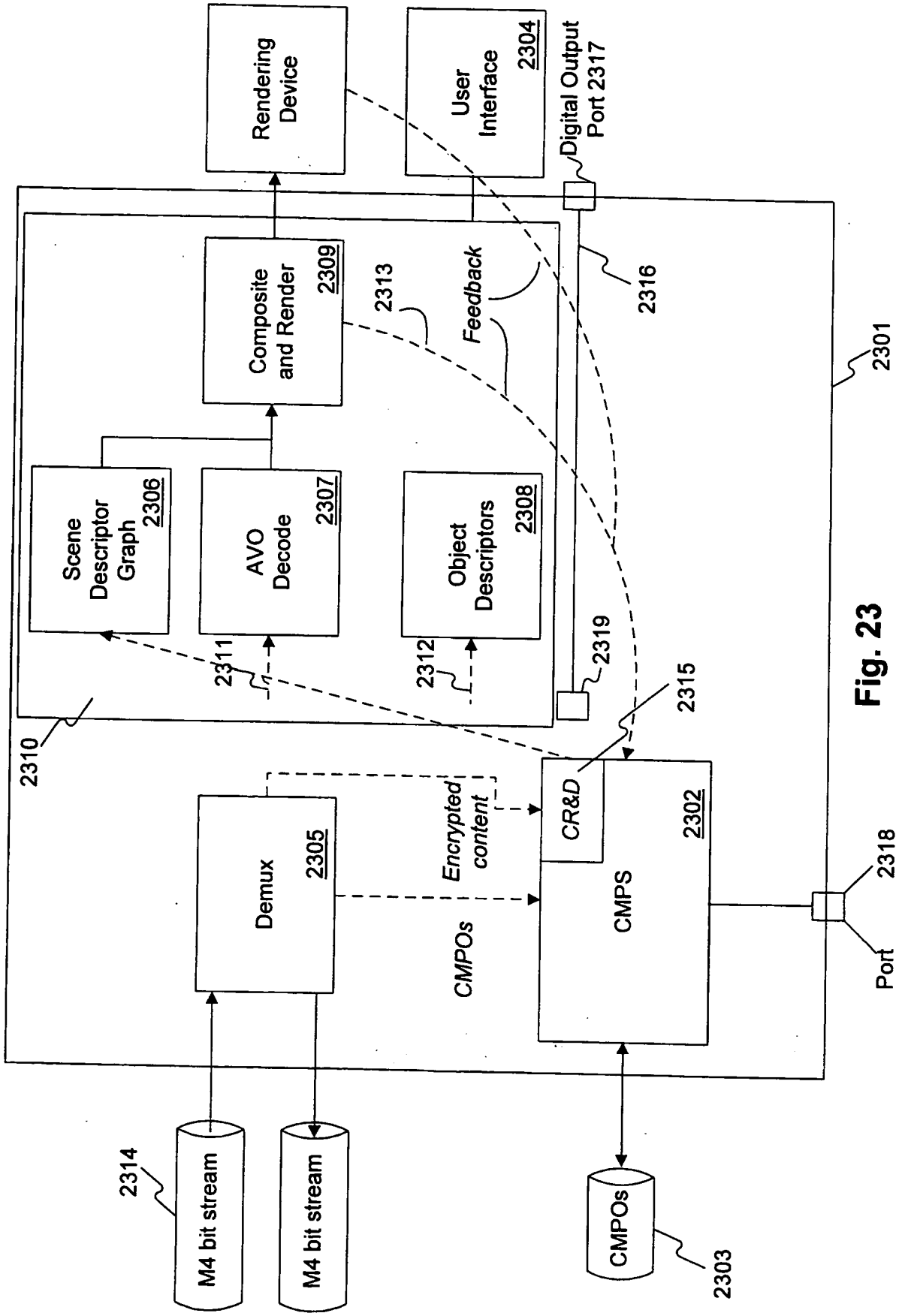

Rendering Device
User Interface 2304
Digital Output Port 2317
Composite and Render 2309
2313
Feedback
2316
2301
Scene Descriptor Graph 2306
AVO Decode 2307
Object Descriptors 2308
2311
2312
2319
2315
2310
CR&D
2302
Demux 2305
Encrypted content
CMPS
2318
CMPOs
Port
2314
M4 bit stream
M4 bit stream
CMPOs
2303

Fig. 23

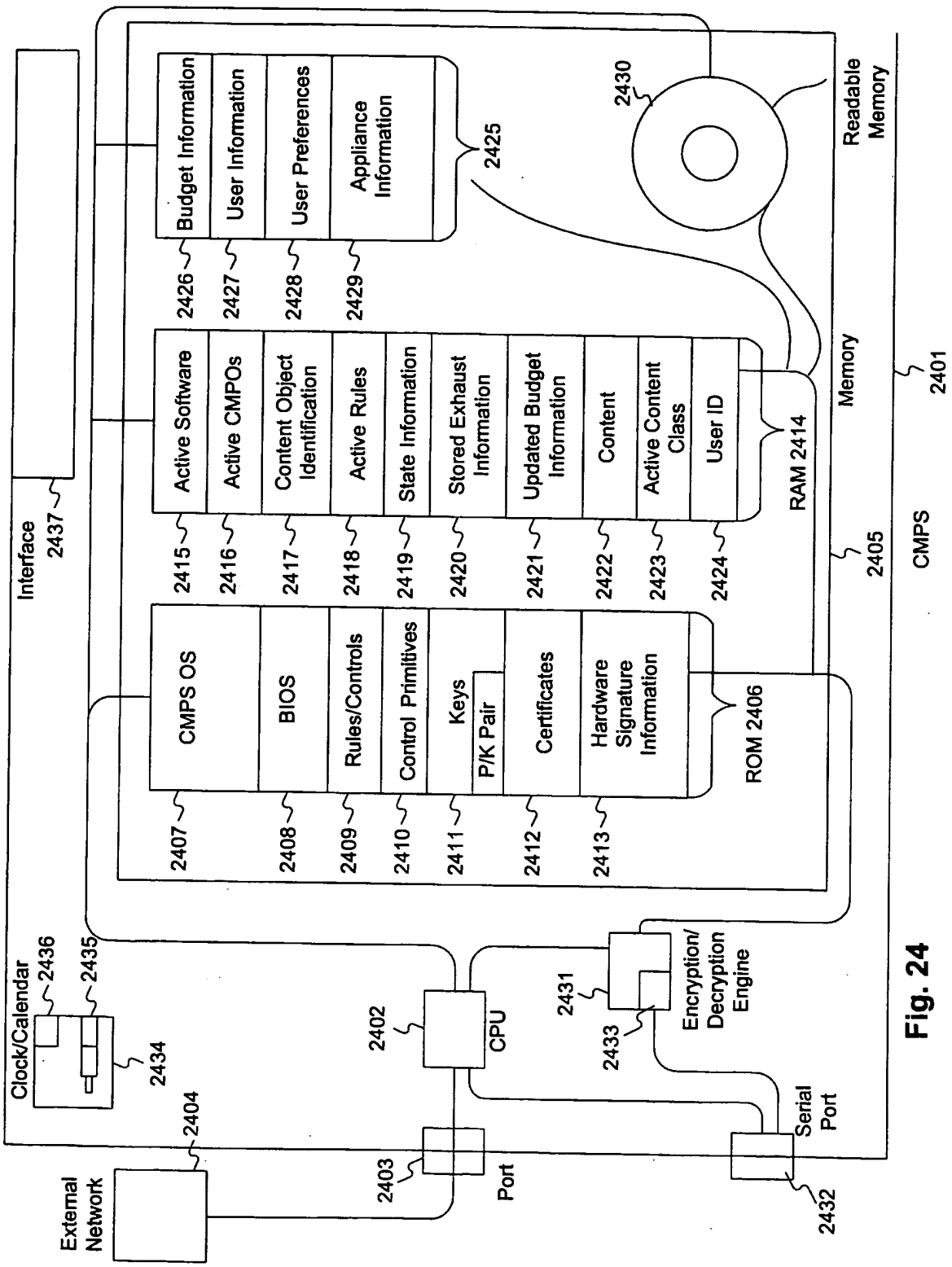

Budget Information
User Information
User Preferences
Appliance Information
2425
2426
2427
2428
2429
2430
Readable Memory
Active Software
Active CMPOs
Content Object Identification
Active Rules
State Information
Stored Exhaust Information
Updated Budget Information
Content
Active Content Class
User ID
RAM 2414
Memory
2401
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
Interface
2437
CMPS OS
BIOS
Rules/Controls
Control Primitives
Keys
P/K Pair
Certificates
Hardware Signature Information
ROM 2406
CMPS
2405
2407
2408
2409
2410
2411
2412
2413
Clock/Calendar
2436
2435
2434
2404
External Network
2403
Port
2402
CPU
2431
2433
Encryption/ Decryption Engine
Serial Port
2432

Fig. 24

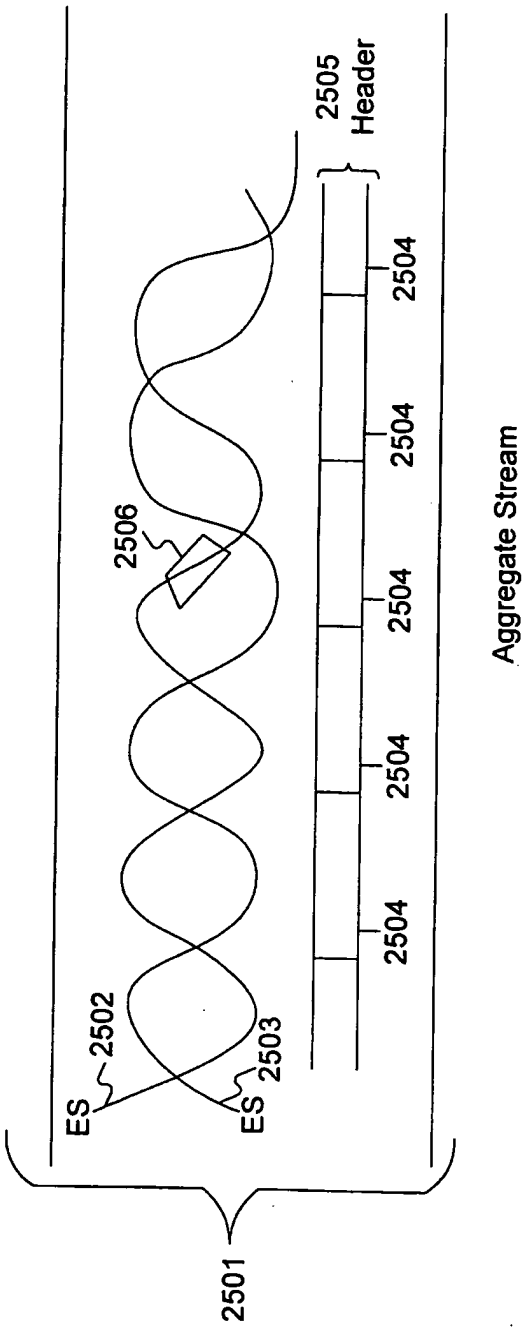

2501
ES
2502
ES
2503
2506
2505
Header
2504
2504
2504
2504
2504
Aggregate Stream

Fig. 25

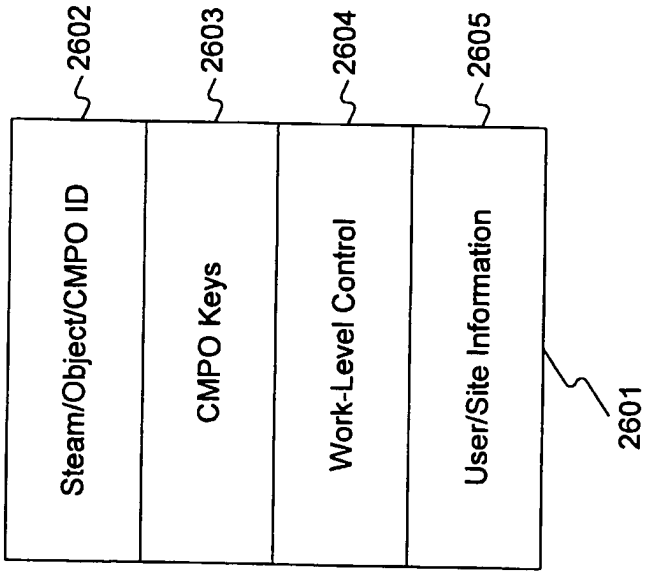

2602
2603
2604
2605
Steam/Object/CMPO ID
CMPO Keys
Work-Level Control
User/Site Information
2601

Fig. 26

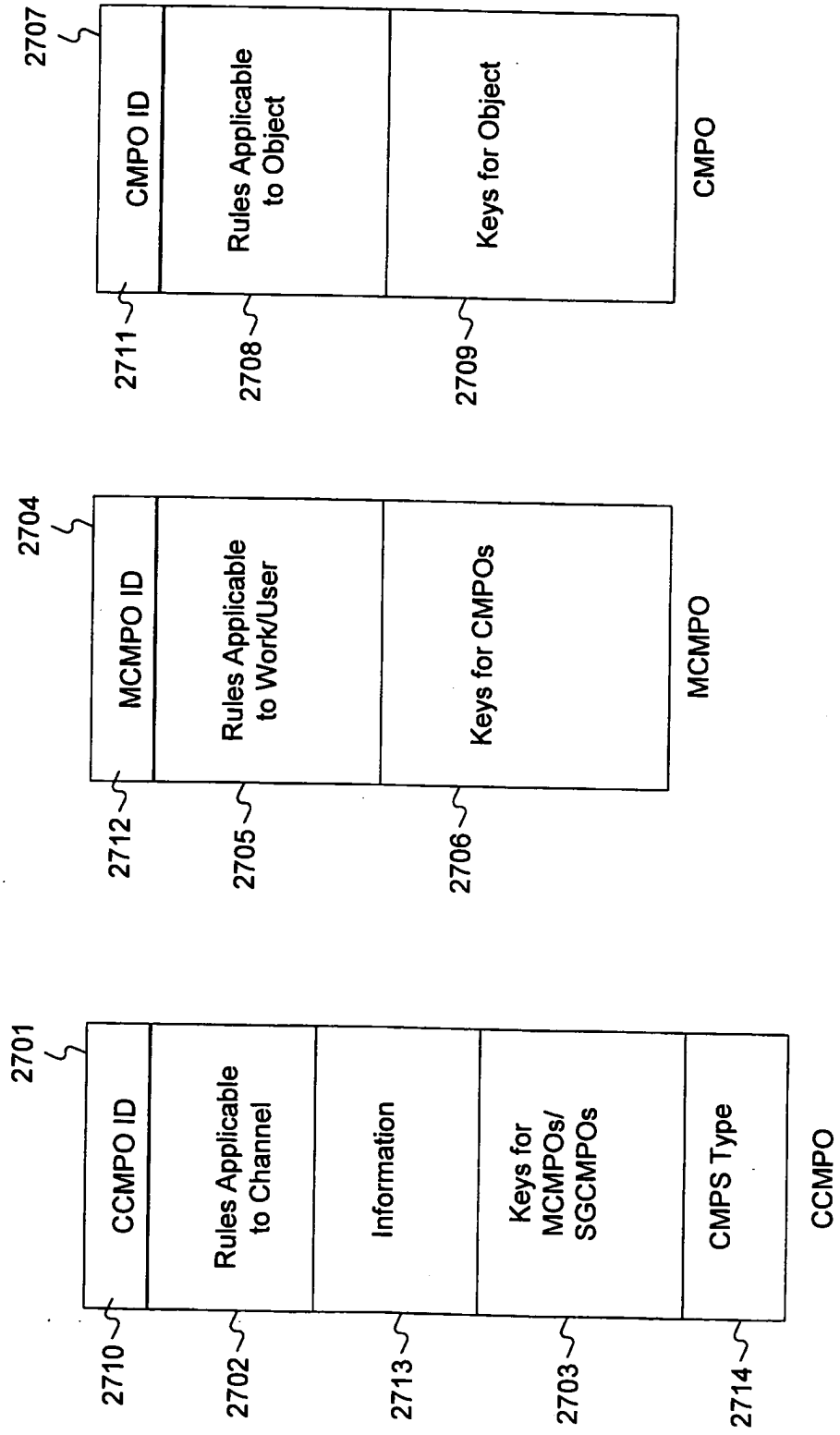

2707
CMPO ID
Rules Applicable to Object
Keys for Object
CMPO
2711
2708
2709
2704
MCMPO ID
Rules Applicable to Work/User
Keys for CMPOs
MCMPO
2712
2705
2706
2701
CCMPO ID
Rules Applicable to Channel
Information
Keys for MCMPOs/ SGCMPOs
CMPS Type
CCMPO
2710
2702
2713
2703
2714

Fig. 27

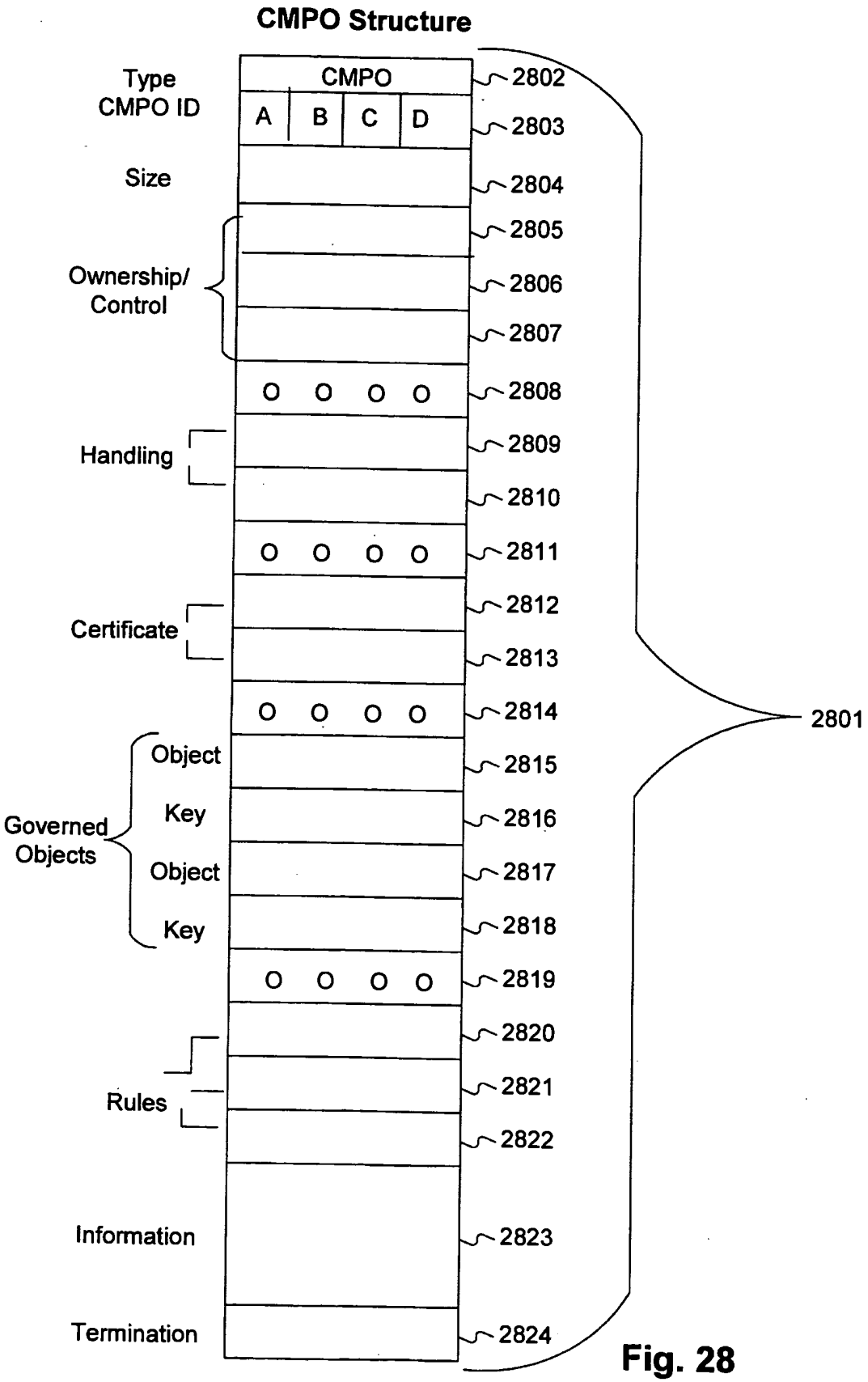

CMPO Structure
Type
CMPO ID
CMPO
A
B
C
D
Size
Ownership/
Control
Handling
Certificate
Governed
Objects
Object
Key
Object
Key
Rules
Information
Termination
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2801

Fig. 28

INTERNATIONAL SEARCH REPORT

Inter ›nal Application No
PCT/US 99/05734

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 H04N7/167 G06F1/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 H04N G06F G11B

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 763 936 A (LG ELECTRONICS INC) 19 March 1997	1-4, 6-14, 17-20, 22-25
A	see abstract see column 6, line 27 - column 8, line 4 see column 9, line 6 - column 11, line 43 see column 16, line 47 - column 18, line 38 see figures 4,6A,6B,7 see figures 10,16	5,15,16, 21,26

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
1 July 1999	09/07/1999

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Hampson, F

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Inte: ɔnal Application No
PCT/US 99/05734

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 714 204 A (LG ELECTRONICS INC) 29 May 1996	1-4, 6-14, 22-26
A	see abstract see page 6, line 14 - page 8, line 45 see figures 7,19A,B,20	5,15-21
A	EP 0 715 246 A (XEROX CORP) 5 June 1996 see abstract see page 3, line 46 - page 8, line 27 see figures 1-3,4A,4B	1-26
A	WO 97 25816 A (SONY CORP ;INOUE HAJIME (US); LEE CHUEN CHIEN (US); SONY ELECTRONI) 17 July 1997 see abstract see page 7, line 10 - page 10, line 7 see figures 2,3	1-26
A	EP 0 800 312 A (MATSUSHITA ELECTRIC IND CO LTD) 8 October 1997 see abstract see column 50, line 20 - column 51, line 53	1,6,7, 23-25

1

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No
PCT/US 99/05734

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0763936 A	19-03-1997	CN 1150738 A	28-05-1997
		JP 9093561 A	04-04-1997
		US 5799081 A	25-08-1998
EP 0714204 A	29-05-1996	CN 1137723 A	11-12-1996
		JP 8242438 A	17-09-1996
		US 5757909 A	26-05-1998
EP 0715246 A	05-06-1996	US 5638443 A	10-06-1997
		JP 8263439 A	11-10-1996
WO 9725816 A	17-07-1997	AU 1344097 A	01-08-1997
		CN 1209247 A	24-02-1999
		EP 0882357 A	09-12-1998
		US 5889919 A	30-03-1999
EP 0800312 A	08-10-1997	WO 9714249 A	17-04-1997
		CN 1168054 A	17-12-1997
		EP 0789361 A	13-08-1997
		JP 10079174 A	24-03-1998

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 1/00 — A1

(11) International Publication Number: **WO 99/60461**

(43) International Publication Date: 25 November 1999 (25.11.99)

(21) International Application Number: PCT/GB98/03828

(22) International Filing Date: 18 December 1998 (18.12.98)

(30) Priority Data:
09/080,030 15 May 1998 (15.05.98) US

(71) Applicant: INTERNATIONAL BUSINESS MACHINES CORPORATION [US/US]; New Orchard Road, Armonk, NY 10504 (US).

(71) Applicant *(for MC only)*: IBM UNITED KINGDOM LIMITED [GB/GB]; North Harbour, Portsmouth, P.O. Box 41, Hampshire PO6 3AU (GB).

(72) Inventors: BERSTIS, Viktors; 5194 Cuesta Verde, Austin, TX 78746 (US). HIMMEL, Maria, Azua; 6403 Rain Creek Parkway, Austin, TX 78759 (US).

(74) Agent: BOYCE, Conor; IBM United Kingdom Limited, Intellectual Property Law, Hursley Park, Winchester, Hampshire SO21 2JN (GB).

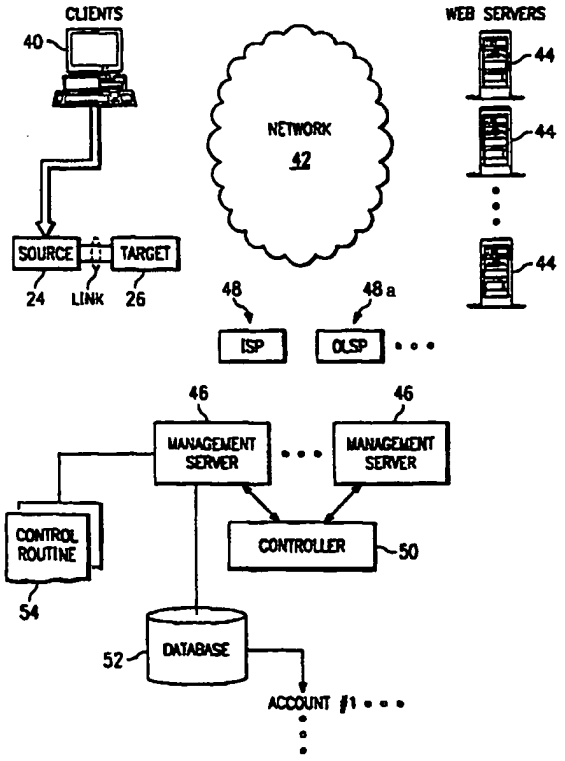
(81) Designated States: CN, CZ, IL, IN, JP, KR, PL, SG, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.

(54) Title: ROYALTY COLLECTION METHOD AND SYSTEM FOR USE OF COPYRIGHTED DIGITAL MATERIALS ON THE INTERNET

(57) Abstract

A method, system and computer program product to facilitate royalty collection with respect to online distribution of electronically published material over a computer network. In one embodiment, a method for managing use of a digital file (that includes content subject to copyright protection on behalf of some content provider) begins by establishing a count of a number of permitted copies of the digital file. In response to a given protocol, a copy of the digital file is then selectively transferred from a source to a target. Thus, for example, the source and target may be located on the same computer with the source being a disk storage device and the target being a rendering device (e.g., a printer, a display, a sound card or the like). The method logs an indication each time the digital file is transferred from the source to a target rendering device, and the count is decremented upon each transfer. When the count reaches a given value (e.g., zero), the file is destroyed or otherwise prevented from being transferred from the source device. The indications logged are transferred to a management server to facilitate payment of royalties to the content provider.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

ROYALTY COLLECTION METHOD AND SYSTEM FOR USE OF COPYRIGHTED DIGITAL MATERIALS ON THE INTERNET

BACKGROUND OF THE INVENTION

Technical Field

The present invention relates generally to managing collection of royalties for electronically-published material distributed over a computer network.

Description of the Related Art

The World Wide Web is the Internet's multimedia information retrieval system. In the Web environment, client machines effect transactions to Web servers using the Hypertext Transfer Protocol (HTTP), which is a known application protocol providing users access to files (e.g., text, graphics, images, sound, video, etc.) using a standard page description language known as Hypertext Markup Language (HTML). HTML provides basic document formatting and allows the developer to specify "links" to other servers and files. In the Internet paradigm, a network path to a server is identified by a so-called Uniform Resource Locator (URL) having a special syntax for defining a network connection. Use of an HTML-compatible browser (e.g., Netscape Navigator or Microsoft Internet Explorer) at a client machine involves specification of a link via the URL. In response, the client makes a request to the server (sometimes referred to as a "Web site") identified in the link and, in return, receives in return a document or other object formatted according to HTML.

One of the technical advantages of the World Wide Web is the ease with which digital content (e.g., graphics, sound, video, movies and the like) may be transmitted and distributed to many users. Indeed, copying a digital file is as easy as clicking on a computer mouse. Copyright laws afford a copyright owner the exclusive right to reproduce the copyrighted work in copies, to distribute such copies, and to publicly perform and display the work. Each time a digital file is transferred over the Internet and copied onto a user s memory, the copyright owner's exclusive reproduction right is implicated (and possibly violated). Likewise, transmission of the copyrighted work over the physical wire is tantamount to a distribution. Indeed, in an open system (e.g., a personal computer accessing the World Wide Web through an Internet Service Provider (ISP)), copies of copyrighted materials can undergo unlimited further copying and transmission without the ability of the owner to collect appropriate compensation (e.g., royalties).

Many publishers or other content providers naturally are hesitant to make their copyrighted works available over the Internet due to the ease with which these materials may be copied and widely disseminated without adequate compensation. Presently, Internet commerce remains highly unregulated, and there is no central authority for managing collection and allocation of content provider royalties. Moreover, while publishers and content rights societies and organizations are attempting to address the legal and logistical issues, the art has yet to develop viable technical solutions.

One technique that has been proposed involves wrapping a copyrighted work in a copy protection "environment" to facilitate charging users for use of that information obtained from the Internet or World Wide Web. This approach, called COPINET, links a copyright protection mechanism with a copyright management system, and it is described in *Charging, paying and copyright - information access in open networks*, Bennett et al., 19th International Online Information Meeting Proceedings, Online Information 1995 pp. 13-23 (Learned Information Europe Ltd.). Publishers in such a system can determine an appropriate level of protection while monitoring use and managing the chain of rights. This approach is also said to provide protection for digital material even after delivery to the user workstation. In particular, copyright material is "wrapped" (by encryption) and "unwrapped" as a result of a specific authorization provided by a trusted subsystem. Material thus is only "visible" to the environment and thus any subsequent user actions, such as "save" or "copy", result in the protected material, or material derived from it, remaining in a protected state when outside the environment.

Although the above-described approach provides some advantages, it does not address the problem of managing the collection of royalties and/or the allocating of such payments to content providers. Moreover, it is not an accepting solution in the context of an open PC architecture such as implemented in the public Internet. It also requires the use of a separate trusted subsystem to generate the authorizations for particular content transfers, which is undesirable.

Other known techniques for managing use of content over the Internet typically involve electronic "wallets" or smart cards. Known prior art systems of this type are illustrated, for example, in U.S. Patent Nos. 5,590,197 and 5,613,001. These systems involve complex hardware and encryption schemes, which are expensive and difficult to implement in practice. They are not readily adaptable to provide general royalty payment schemes for Internet content usage.

Thus, there remains a need to provide improved methods and systems for collecting royalties on the Internet as a result of use of copyrighted content.

The present invention solves this important problem.

SUMMARY OF THE INVENTION

An object of this invention is to enable a pair of "certified" devices (e.g., a storage device and a rendering device) to operate within the context of a given security protocol and thereby manage copies of a digital file and associated copy control information.

Still another object of this invention is to enable a copyright proprietor to maintain a degree of control over copyrighted content even after that content has been fetched from a server and downloaded to a client machine, e.g., in a Web client-server environment.

A particular object of the present invention is to manage the number of copies of a digital file that may be made within a Web appliance having a secure disk storage and that is connectable to the Internet using a dialup network connection.

A still further object of this invention is to restrict a number of copies of a digital file that may be made at a given Web client machine connected to the World Wide Web.

It is yet another object of this invention to enable a publisher of an electronic document to control the number of copies of such document that may be made on the Internet by permitted users.

It is a more general object of this invention to manage permissible use of copyrighted content on the Internet and World Wide Web.

It is still another more general object of this invention to manage collection of information to facilitate payment of appropriate compensation to content providers and publishers arising from use of their copyrighted content on the Internet.

Another object of this invention is to manage the charging of users for information obtained from the Internet or World Wide Web.

A still further object of this invention is to facilitate royalty collection as a result of electronically published material distributed

online over a computer network (e.g., the public Internet, an intranet, an extranet or other network).

One embodiment of the invention is a method for managing copies of a digital file, which includes content subject to copyright protection, on behalf of some content provider (e.g., an author, publisher or other). It is assumed that a given usage scheme has been established with respect to the file as defined in copy control information associated with the file. Thus, for example, the copy control information may define a set of payment options including, without limitation, prepayment (for "n" copies), pay-per-copy (as each copy is made), IOU (for copies made offline), or some other payment option. The copy control information may also include other data defining how the file is managed by the scheme including: a count of the number of permitted copies, a count of the number of permitted pay-per-copy versions, copyright management information, payee information, an expiration date (after which copying is no longer permitted), and the like.

The present invention assumes the existence of a pair of devices, a "source" and a "target", that have been or are certified to use the scheme. Typically, the "source" is a storage device while the "target" is a rendering device. An illustrative storage device may be disk storage, system memory, or the like. An illustrative rendering device may be a printer, a display, a sound card or the like. The source and target devices may both be storage devices (e.g., a Web server and a client disk storage). In either case, each of the devices comprising the pair is "certified" (typically upon manufacture) to operate under a given security protocol. Under the protocol, the devices include appropriate circuitry and/or software, as the case may be, to facilitate the establishment of a secure link between the storage and rendering devices. Each device requires the other to validate itself and thus prove that the device can be trusted to manage the content (namely, the digital file) sought to be protected.

When the technique is implemented in an "open" client-server environment, hardware devices (e.g., microcontrollers) preferably are used in the storage and rendering devices to facilitate generation of the secure link. When the technique is implemented in a "closed" Web appliance environment, the secure link may be established and managed using software resident in the control routines associated with the storage and rendering devices. The secure link may be established and managed in software under such conditions because, in the Web appliance environment, it is possible to readily disable the secure link in the event of tampering with the appliance housing or other circuitry. Regardless of the environment, the secure link is first established

between the "certified" storage and rendering devices. Thereafter, the digital file, together with at least part of its copy control information, is transferable between the storage and rendering devices in accordance with the particular usage and payment scheme being utilized. Thus, for example, if a prepayment scheme is implemented and an expiration date (associated therewith) has not occurred, a given number of copies of the file may be transferred between the storage and rendering devices. The prepayment funds are collected at a central location and then redistributed to the copyright proprietor or some third party.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a representative system in which the present invention is implemented;

Figure 2 is a simplified block diagram of a source device and a target device connected by a channel over which a digital file is transferred according to the present invention;

Figure 3 is an illustrative example of a source device connected to a set of target rendering devices in a client computer;

Figure 4 is a block diagram of a representative copyright management system according to the present invention;

Figure 5 is a flowchart of a preferred method of managing a digital file according to the present invention;

Figure 6A is pictorial representation of a data processing system unit connected to a conventional television set to form a "Web" appliance;

Figure 6B is a pictorial representation of a front panel of the data processing system unit;

Figure 6C is a pictorial representation of a rear panel of the data processing system unit;

Figure 6D is a pictorial representation of a remote control unit associated with the data processing system unit; and

Figure 7 is a block diagram of the major components of the data processing system unit.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

A representative system in which the present invention is implemented is illustrated in **Figure 1**. A plurality of Internet client machines **10** are connectable to a computer network Internet Service Provider (ISP) **12** via a "resource" such as a dialup telephone network **14**. As is well known, the a dialup telephone network usually has a given, limited number of connections **16a-16n**. ISP **12** interfaces the client machines **10** to the remainder of the network **18**, which includes a plurality of Internet server machines **20**. A client machine typically includes a suite of known Internet tools (e.g., Web browser **13**) to access the servers of the network and thus obtain certain services. These services include one-to-one messaging (e-mail), one-to-many messaging (bulletin board), on-line chat, file transfer and browsing. Various known Internet protocols are used for these services. Thus, for example, browsing is effected using the Hypertext Transfer Protocol (HTTP), which provides users access to multimedia files using Hypertext Markup Language (HTML). The collection of servers that use HTTP comprise the World Wide Web, which is the Internet's multimedia information retrieval system.

As will be described in more detail below, the present invention may be implemented in hardware and/or in software. The software implementation is particularly useful when the client machine is an Internet or Web appliance, such as illustrated in **Figures 6A-6D**. In the case of the software implementation, a client machine has associated therewith a software routine **15** designed to perform one or more of the functions of the digital file copy protection method, as will be described. The software is preferably a client application (although it may be implemented with the browser as a plug-in, or with a client-side proxy, or as a standalone application). Alternatively, the agent is built into the browser, or it is implemented as a Java applet or standalone application. Thus, as used herein, in this particular embodiment, the software **15** is any application running on a client machine **10** that performs the copy protection/royalty management task(s) on behalf of the user(s) of that client according to the present invention.

The discussion which follows primarily uses the words "copying" or "copies" to describe the control of the further exercise of a copyright right for a particular work. The reader should understand that "copying" could include other types of rendering of the work for different devices. That is, "copying" in a printer would entail printing on paper or another substrate. Copying on a display is presenting an image on the screen. Copying in an audio device would be the performance of an audio portion of the work. Each of these devices both storage devices, e.g., hard disks, tapes in CDR, and rendering devices, e.g., prints, display graph, audio

player, movie player, should be equipped with the present invention so that the copies are controlled throughout the systems and networks until their final rendering place.

The present invention is a method for managing copies of a digital file, which includes content subject to copyright protection, on behalf of some content provider (e.g., an author, publisher or other). It is assumed that a given payment scheme has been established with respect to the file. Thus, for example, such payment schemes include, without limitation, prepayment (for "n" copies), pay-per-copy (as each copy is made), IOU (for copies made offline), or some other payment option. In a prepayment option, a user prepays funds for the right to obtain copies of the digital file. In a pay-per-copy (or "pay as you go") option, the user pays for each copy of the digital file when the file is copied. In an IOU scheme, the user makes copies of the digital file (e.g., while the client machine is not connected to the network) and generates an IOU (or many IOUs) that are then submitted to a clearinghouse or other payment entity when the user later goes online. Other payment schemes (such as a combination of the above options) may also be implemented.

The payment scheme is preferably defined in copy control information associated with the file and established by the author, publisher or some other third party. Thus, for example, the copy control information may also include a count of the number of permitted copies, a count of the number of permitted pay-per-copy versions, a count of the number of copies that may be made under an IOU payment option, copyright management information identifying the author, publisher and/or other license or use restrictions, information about a bank or other financial institution that handles use payments and their reconciliation, one or more expiration dates (after which copying is no longer permitted), and the like.

The copy control information associated with a given file thus defines a usage scheme for the file because it includes information that controls how the content may be used, how such use is paid for, over what period the content may be used, and other such information. A particular usage scheme (or some portion thereof) may also be implemented in the devices between which the file is transferred, although preferably such restrictions are defined by the content provider.

According to the present invention as illustrated in **Figure 2**, the present invention assumes the existence of a pair of devices, a "source" **24** and a "target" **26**, that have been or are certified to use the scheme. In particular, devices that implement the inventive scheme preferably include a device certificate that is not accessible (and thus is free from tampering) and stored therein. The certificate evidences that the device

is capable of understanding a given security protocol useful in carrying out the protection scheme. A representative security protocol is CSS, or the Content Scrambling System protocol, available commercially from Matsushita Corp. Thus, for example, if the source device is a disk storage, the device certificate is typically stored inside a secure chip within the device control hardware. Typically, each of the devices is "certified" upon manufacture, although this is not a requirement.

As also illustrated in **Figure 2**, a channel **28** is established between the source and target devices over which copies of a digital file (that is subject to the scheme) are communicated in a secure fashion. Thus, prior to transfer of the digital file, the channel **28** is first established between the devices to ensure that the copy restrictions (such as set forth in the copy control information) may be enforced. Typically, this is accomplished by having each device (in accordance with the security protocol implemented) require the other device (of the pair) to verify that its device certificate is valid. An appropriate message exchange may be used for this purpose as defined in the protocol. Once the secure link has been established, each of the devices can be trusted to control the digital file in accordance with the file's copy control information.

Typically, the "source" **24** is a storage device while the "target" **26** is a rendering device. An illustrative storage device may be disk storage, system memory, or the like. An illustrative rendering device may be a printer, a display, a sound card or the like. The source and target devices may both be storage devices (e.g., a Web server and a client disk storage).

When the technique is implemented in an "open" client-server environment, hardware devices (e.g., microcontrollers) are used in the storage and rendering devices to facilitate generation and management of the secure link. When less security may be tolerated, some of these functions may be implemented in software. When the technique is implemented in a "closed" Web appliance environment (**Figures 6A-6D**), the secure link may be established in whole or in part using software resident in the control routines associated with the storage and rendering devices. The secure link may be established in software under such conditions because, in the Web appliance environment, it is possible to readily disable the secure link in the event of tampering with the appliance housing or other circuitry. Regardless of the environment, the secure link is first established between the "certified" storage and rendering devices. Thereafter, the digital file, together with at least part of its copy control information, is transferable between the storage and rendering devices in accordance with the particular usage scheme defined, for example, by the copy control information. Thus, for example, if a

prepayment scheme is implemented and an expiration date (associated therewith) has not occurred, a given number of copies of the file may be transferred between the storage and rendering devices.

Thus, as illustrated in **Figure 2** in simplified form, the digital file copy protection method and system of the present invention involves a "source" device **24** (or one or more of such devices), and a set of one or more "target" devices **26**a-n connected via the secure channel or link **28**. The physical characteristics of the channel, of course, depend on whether the source and target devices are located in the same machine or are in separate machines connected via a network. In a network connection, the link may be a conventional TCP/IP connection. Channel **28** may be a physically secure channel (such as a https connection), but this is not required as the given security protocol in the certified devices establishes a secure link. According to the invention, once the link is established, one or more digital files are transferred (under the control of a control routine or mechanism) between the certified devices in an predictable, auditable manner so that (a) a controlled number of file transfers can be made, and (b) the precise number of file transfers (and their particular use) may be readily documented to facilitate dissemination of royalties or some such other consideration, typically to providers of such content. Generalizing, prior to transfer of a given digital file (or set of files, or file component) from the source to the target via the secure link, that transfer must first be authorized, and the transfer itself is then capable of being associated with some royalty payment then due to a content provider for use of such file. The scheme thus facilitates implementation of a generalized copyright management/royalty collection and distribution scheme.

As previously mentioned, the source **24** and target **26** may be located on the same computer. **Figure 3** illustrates this particular connection for a disk storage subsystem **24'** and the target rendering devices, namely printer **26a'**, display **26b'** and sound card **26c'**. The illustrated computer is a Web appliance, in which case the secure link may be established (as noted above) using software. Thus, in this example, each source and/or target device includes appropriate control software (part of software **15** as described above) to facilitate creation of the secure channel. Although not meant to be limiting, one convenient mechanism to create the channel involves each of the devices to generate a random number **30**, which numbers are then supplied to a key generation algorithm **32** in a known manner to generate a secret of "private" key **34**. The key **34** may be generated for each digital file to be transferred over the link **28**, or a signal key may be used for a set of such files, or even for a particular browsing session. To create the secure channel, the software resident on the disk storage encrypts the digital file as it leaves the source device.

The target device then decrypts the digital file using the key prior to rendering. In this way, the digital file cannot be readily intercepted as it is being transferred between these devices. As noted above, each of the source and target devices may also include secure chips or other known hardware devices to facilitate or augment such secure transfer of the digital file between the devices.

The particular mechanism for securing the channel between the source and target may be quite varied, and the present invention contemplates the use of any now known or later-developed technique, system or method for securing such communications. Thus, for example, another technique that may be used would be a public key cryptosystem.

Figure 4 is a block diagram illustrating a representative copyright royalty management system implemented according to the present invention. In this system, it is assumed that client computers **40** access the computer network **42** (e.g., the public Internet, an intranet, an extranet, or other computer network) to obtain access to Web-like documents supported on Web servers **44**. One or more management servers **46** are connectable to the system via an access provider **48**, and a control management server **50** may be used to facilitate scaling of the architecture if required. Control management server **50** may be controlled by a regulatory or rights agency that has responsibility for managing collection and distribution of copyright royalties.

A given management server includes a database **52** and appropriate control routines **54** for establishing a royalty account **55** for content providers. It is envisioned (although not required) that given content providers will subscribe to a royalty collection service implemented by the present invention and perhaps pay a fee (e.g., a commission or service charge) for the service provided. A given content provider thus may subscribe to the service to receive royalty payments for the use of his or her copyrighted content by users of the client machines. To this end, control routines **54** are used to establish an account for each of a set of given content providers, with each account including a representation of a given royalty value (which may be $0 when the account is established). A control routine then adjusts the given royalty value in a given provider account in response to receipt of an indication that a given digital file associated with the given content provider has been transferred from a source **24** to a target rendering device **26** in a given client computer **40**. Periodically, the content provider account is adjusted for any service or processing fees, and the remainder of the account is then distributed to the content provider. In the situation where the content provider is willing to allow his or her content (a given digital file) to be used with charges for such use paid later, a given bit may be set in the file's copy

control information indicating such preference. Other data in the copy control information may be used to set or control other content provider preferences with respect to use of the file within the context of the inventive scheme.

Figure 5 is a flowchart of one method of managing royalty account collection with respect to a particular digital file when a prepayment option is utilized. In this representative example, the digital file is an image (i.e. a .jpeg file) having a copyright owned by a given content proprietor or provider. Of course, the principles of the present invention are designed to be implemented collectively with many such digital files, and the following description is thus merely representative of one type of basic payment scheme. The routine assumes initially that a usage or payment account has been established for a given client computer (or a user of that computer). This is step 60 in the flowchart. It is also assumed that a royalty account has been established for the content provider at one of the management servers as previously described. This is step 62 in the flowchart. One of ordinary skill will appreciate that steps 60 and 62 need not be in any particular sequence. Step 60 typically involves the user prepaying some amount of funds into an account from which payments may be withdrawn, although this is not required.

At step 64, a count is established by a control routine for the particular digital file. Typically, this is a count of a number of permitted copies of the digital file that may be transferred from the source to one or more target devices according to the present invention. This number, as noted above, is typically identified in the file's copy control information. The count is usually a positive integer, which is then decremented (by the control routine) down to zero as permitted or authorized copies are made. Alternatively, of course, the count may begin at zero (or any other arbitrary number), which is then incremented (by the control routine) to the threshold value identified in the copy count information. As noted above, the count may be set by the copyright proprietor, by a system operator, by a Webmaster, by hardware constraints, or by any other party or entity having authority and/or ability to set the count. Under certain circumstances, e.g., where a prepaid user account is used, it may be unnecessary to use an explicit count as the number of copies transferred may simply depend on the royalty assessed per copy. Thus, the "count" as used herein may be expressed explicitly or implicitly. The digital file may be stored on the client already, or it may be available from a Web server or other storage or archive. The particular location from which the digital file is sourced initially does not matter. Step 64 assumes, however, that the image is located already at the source device. If the file is not present at the source, it may be

necessary to obtain it (although, conceptually, the "source" may be broadly construed as the original or initial location of the file).

At step 66, a test is done repeatedly to determine whether a request for the image has been received. If not, the routine cycles on step 66 and waits for such a request. If the outcome of the test at step 66 is positive, then the routine continues at step 68 by testing whether the given client computer (which generated the request) is authorized to effect the transfer. Step 68 may comprise a simple comparison of the user's account balance and the royalty amount to be assessed. If the user's account balance is large enough, the transfer may be allowed. Or, step 68 may simply test whether the count has a value indicating that further copies may be made. More typically, step 68 will require that the count be non-zero (in the situation where the count is positive and decremented to zero) and the user have sufficient funds allocated to pay the royalty assessment for use of the image. The step 68 may also test whether a given expiration date set in the copy count information has past.

If the outcome of the test at step 68 is negative, the transfer is not authorized, and the routine branches to step 70 to so notify the user of the client machine. Such notification may be in the form of an error or "access denied" message or the like. The user may be informed merely that a preset expiration date has passed or that his or her prepaid account is exhausted and requires more funds. If, however, the outcome of the test at step 68 is positive, the digital file may be transferred to the target. The routine then branches to step 72 to initiate the copy transfer. Preferably, all bytes of the file must be transferred before the transfer is considered valid. At step 74, the control routine count is adjusted (e.g., decremented) and/or a given charge is allocated against the user's account. The given charge may be equal to the royalty or use charge, or some fixed percentage thereof (e.g., 105%) reflecting that royalty plus some service charge). At step 76, the appropriate content provider account is adjusted by the amount of the royalty payment (plus or minus appropriate service fees or other charges).

Neither step 74 nor step 76 need occur at the time of the file transfer. Typically, the account adjustments will take place in batch at a given time. Thus, for example, where the Web client is a Web appliance connected to the computer network via a dialup connection, the account information may be transferred to the management server upon establishing a given connection (e.g. perhaps once each day). Other variations regarding the timing of delivery of this information are, of course, within the scope of the present invention.

The present invention thus provides numerous advantages. Certified source and target devices first establish a secure link between themselves. Upon transfer of the file copy between source and target, the control routine records an appropriate indication thereof in the copy count, and the central authority is notified of the transfer of the digital file. Such notification may occur upon transfer of the digital file between the source and target devices, or at some later time (e.g., upon dialup connection of the computer to the network). Royalty accounts are then managed at a central authority; to facilitate distribution of royalties to content owners/publishers. When the copy count reaches the authorized limit (as set in the copy control information), the control routine destroys the file or otherwise prevents further copying of the digital file.

Thus, in one embodiment, the user establishes a "prepaid" account from which royalty or usage payments are drawn against as files are copied/transmitted. The system detects use of the file and, preferably, allows only a certain number of copies of the file to be made before the document is destroyed or otherwise rendered inaccessible (from the client machine). The resulting copyright management infrastructure is robust, secure, scaleable and easily managed.

In one embodiment of this invention as described above, the Internet client is a data processing system or a so-called "Web appliance" such as illustrated in **Figures 6A-6D** and 7. **Figure 6A** is a pictorial representation of the data processing system as a whole. Data processing system **100** in the depicted example provides, with minimal economic costs for hardware to the user, access to the Internet. Data processing system **100** includes a data processing unit **102**. Data processing unit **102** is preferably sized to fit in typical entertainment centers and provides all required functionality, which is conventionally found in personal computers, to enable a user to "browse" the Internet. Additionally, data processing unit **102** may provide other common functions such as serving as an answering machine or receiving facsimile transmissions.

Data processing unit **102** is connected to television **104** for display of graphical information. Television **104** may be any suitable television, although color televisions with an S-Video input will provide better presentations of the graphical information. Data processing unit **102** may be connected to television **104** through a standard coaxial cable connection. A remote control unit **106** allows a user to interact with and control data processing unit **102**. Remote control unit **106** allows a user to interact with and control data processing unit **102**. Remote control unit **106** emits infrared (IR) signals, preferably modulated at a different frequency than the normal television, stereo, and VCR infrared remote

control frequencies in order to avoid interference. Remote control unit **106** provides the functionality of a pointing device (such as a mouse, glidepoint, trackball or the like) in conventional personal computers, including the ability to move a cursor on a display and select items.

Figure 6B is a pictorial representation of the front panel of data processing unit **102**. The front panel includes an infrared window **108** for receiving signals from remote control unit **106** and for transmitting infrared signals. Data processing unit **102** may transmit infrared signals to be reflected off objects or surfaces, allowing data processing unit **102** to automatically control television **104** and other infrared remote controlled devices. Volume control **110** permits adjustment of the sound level emanating from a speaker within data processing unit **102** or from television **104**. A plurality of light-emitting diode (LED) indicators **112** provide an indication to the user of when data processing unit **102** is on, whether the user has messages, whether the modem/phone line is in use, or whether data processing unit **102** requires service.

Figure 6C is a pictorial representation of the rear panel of data processing unit **102**. A three wire (ground included) insulated power cord **114** passes through the rear panel. Standard telephone jacks **116** and **118** on the rear panel provide an input to a modem from the phone line and an output to a handset (not shown). The real panel also provides a standard computer keyboard connection **120**, mouse port **122**, computer monitor port **124**, printer port **126**, and an additional serial port **128**. These connections may be employed to allow data processing unit **102** to operate in the manner of a conventional personal computer. Game port **130** on the rear panel provides a connection for a joystick or other gaming control device (glove, etc.). Infrared extension jack **132** allows a cabled infrared LED to be utilized to transmit infrared signals. Microphone jack **134** allows an external microphone to be connected to data processing unit **102**.

Video connection **136**, a standard coaxial cable connector, connects to the video-in terminal of television **104** or a video cassette recorder (not shown). Left and right audio jacks **138** connect to the corresponding audio-in connectors on television **104** or to a stereo (not shown). If the user has S-Video input, then S-Video connection **140** may be used to connect to television **104** to provide a better picture than the composite signal. If television **104** has no video inputs, an external channel 3/4 modulator (not shown) may be connected in-line with the antenna connection.

Figure 6D is a pictorial representation of remote control unit **106**. Similar to a standard telephone keypad, remote control unit **106** includes buttons **142** for Arabic numerals 0 through 9, the asterisk or "star" symbol

(*), and the pound sign (#). Remote control unit also includes "TV" button **144** for selectively viewing television broadcasts and "Web" button **146** for initiating "browsing" of the Internet. Pressing "Web" button **146** will cause data processing unit **102** to initiate modem dial-up of the user's Internet service provider and display the start-up screen for an Internet browser.

A pointing device **147**, which is preferably a trackpoint or "button" pointing device, is included on remote control unit **106** and allows a user to manipulate a cursor on the display of television **104**. "Go" and "Back" buttons **148** and **150**, respectively, allow a user to select an option or return to a previous selection. "Help" button **151** causes context-sensitive help to be displayed or otherwise provided. "Menu" button **152** causes a context-sensitive menu of options to be displayed, and "Update" button **153** will update the options displayed based on the user's input, while home button **154** allows the user to return to a default display of options. "PgUp" and "PgDn" buttons **156** and **158** allows the user to change the context of the display in display-sized blocks rather than by scrolling. The message button **160** allows the user to retrieve messages.

In addition to, or in lieu of, remote control unit **106**, an infrared keyboard (not shown) with an integral pointing device may be used to control data processing unit **102**. The integral pointing device is preferably a trackpoint or button type of pointing device. A wired keyboard (also not shown) may also be used through keyboard connection **120**, and a wired pointing device such as a mouse or trackball may be used through mouse port **122**. When a user has one or more of the remote control unit **106**, infrared keyboard, wired keyboard and/or wired pointing device operable, the active device locks out all others until a prescribed period of inactivity has passed.

Referring now to **Figure 7**, a block diagram for the major components of data processing unit **102** is portrayed. As with conventional personal computers, data processing unit **102** includes a motherboard **202** containing a processor **204** and memory **206** connected to system bus **280**. Processor **205** is preferably at least a **486** class processor operating at or above **100** MHz. Memory **206** may include cache memory and/or video RAM. Processor **205**, memory **206**, and system bus **208** operate in the same manner as corresponding components in a conventional data processing system.

Video/TV converter **210**, located on motherboard **202** and connected to system bus **208**, generates computer video signals for computer monitors, a composite television signal, and an S-Video signal. The functionality of Video/TV converter **210** may be achieved through a Trident TVG**9685** video

chip in conjunction with an Analog Devices AD722 converter chip. Video/TV converter 210 may require loading of special operating system device drivers.

Keyboard/remote control interface unit 212 on motherboard 202 receives keyboard codes through controller 214, regardless of whether a wired keyboard/pointing device or an infrared keyboard/remote control is being employed. Infrared remote control unit 106 transmits signals which are ultimately sent to the serial port as control signals generated by conventional mouse or pointing device movements. Two buttons on remote control unit 106 are interpreted identically to the two buttons on a conventional mouse, while the remainder of the buttons transmit signals corresponding to keystrokes on an infrared keyboard. Thus, remote control unit 106 has a subset of the function provided by an infrared keyboard.

Connectors/indicators 216 on motherboard 202 provide some of the connections and indicators on data processing unit 102 described above. Other connections are associated with and found on other components. For example, telephone jacks 116 and 118 are located on modem 222. The power indicator within connectors/indicators 216 is controlled by controller 214.

External to motherboard 202 in the depicted example are power supply 218, hard drive 220, modem 222 and speaker 224. Power supply 218 is a conventional power supply except that it receives a control signal from controller 214 which effects shut down of all power to motherboard 202, hard drive 220 and modem 222. Power supply 218, in response to a signal from controller 214, is capable of powering down and restarting data processing unit 102.

Controller 214 is preferably one or more of the 805x family controllers. Controller 214 receives and processes input from infrared remote control 106, infrared keyboard, wired keyboard, or wired mouse. When one keyboard or pointing device is used, all others are locked out (ignored) until none have been active for a prescribed period. Then the first keyboard or pointing device to generate activity locks out all others. Controller 214 also directly controls all LED indicators except that indicating modem use. As part of the failure recovery system, controller 214 specifies the boot sector selection during any power off-on cycle.

Hard drive 220 contains operating system and applications software for data processing unit 102, which preferably includes IBM DOS 7.0, a product of International Business Machines Corporation in Armonk, New York; an operating system 221 such as Windows 3.1 (or higher), a product

of Microsoft Corporation in Redmond, Washington; and a browser 223 such as Netscape Navigator (Version 1.0 or higher), a product of Netscape Communications Corporation in Mountain View, California. Hard drive 220 may also support an SMTP mechanism to provide electronic mail, an FTP mechanism to facilitate file transfers from Internet FTP sites, and other Internet protocol mechanisms, all in a known manner. Hard drive 220 is not generally accessible to the user of the Web appliance.

Modem 222 may be any suitable modem used in conventional data processing systems, but is preferably a 33.6 kbps modem supporting the V.42bis, V.34, V.17 Fax, MNP 1-5, and AT command sets. Modem 222 is connected to a physical communication link 227, which, in turn, in connected or connectable to the Internet (not shown).

Those skilled in the art will recognize that the components depicted in Figures 6A-6D and 7 and described above may be varied for specific applications or embodiments. Such variations in which the present invention may be implemented are considered to be within the spirit and scope of the present invention.

According to the invention, the client machine (typically the hard drive 220) also includes a proxy 225. Preferably, the proxy is implemented in software and includes a cache 227 associated therewith. The cache may be integral to the proxy or logically associated therewith. The cache preferably has a size up to several hundred megabytes, which is substantially larger than the standard cache associated with a browser such as Netscape Navigator. The client machine also includes a protocol stack 229 (e.g., a TCP/IP protocol stack) and a sockets mechanism 231, which are used to support communications in a known manner. According to the invention, the proxy 225 is advantageously located on the client along with the browser. Thus, the proxy is sometimes referred to as a "client side" proxy.

Preferably, the proxy starts up when the Web appliance is booted up. Connectivity between the proxy and the browser is achieved using the sockets mechanism by configuring the browser to pass the HTTP requests to the proxy. To send an HTTP GET request, the browser creates a packet (including the URL and other information) and then opens a socket using the sockets mechanism. The packet is then sent to the IP address/port number to service the HTTP request. Thus, when the browser issues an HTTP GET request, it binds to the socket and sends the request. The request is then intercepted and processed by the proxy instead of being sent directly over the network, all in the manner previously described.

Although in the preferred embodiment the client machine is a Web "appliance", this is not a requirement of the present invention. Thus, a client machine **10** may be a personal computer such as a desktop of notebook computer, e.g., an IBM® or IBM-compatible machine running under the OS/2® operating system, an IBM ThinkPad® machine, or some other Intel **x86** or Pentium®-based computer running Windows **95** (or the like) operating system.

A representative server platform comprises an IBM RISC System/**6000** computer (a reduced instruction set of so-called RISC-based workstation) running the AIX (Advanced Interactive Executive Version **4.1** and above) Operating System **21** and Server program(s) **22**. The platform **20** also includes a graphical user interface (GUI) **23** for management and administration. It may also include an application programming interface (API) **24**. HTTP GET requests are transferred from the client machine to the server platform, typically via the dial-up computer network, to obtain documents or objects formatted according to HTML or some other markup language. While the above platform is useful, any other suitable hardware/operating system/server software may be used.

One of the preferred implementations of the client side or server side mechanisms of the invention is as a set of instructions (program code) in a code module resident in the random access memory of the computer. Until required by the computer, the set of instructions may be stored in another computer memory, for example, in a hard disk drive, or in a removable memory such as an optical disk (for eventual use in a CD ROM) or floppy disk (for eventual use in a floppy disk drive), or downloaded via the Internet or other computer network.

In addition, although the various methods described are conveniently implemented in a general purpose computer selectively activated or reconfigured by software, one of ordinary skill in the art would also recognize that such methods may be carried out in hardware, in firmware, or in more specialized apparatus constructed to perform the required method steps.

As used herein, "Web client" should be broadly construed to mean any computer or component thereof directly or indirectly connected or connectable in any known or later-developed manner to a computer network, such as the Internet. The term "Web server" should also be broadly construed to mean a computer, computer platform, an adjunct to a computer or platform, or any component thereof. Of course, a "client" should be broadly construed to mean one who requests or gets the file, and "server" is the entity which downloads the file. Moreover, although the present invention is described in the context of the Hypertext Markup Language

(HTML), those of ordinary skill in the art will appreciate that the invention is applicable to alternative markup languages including, without limitation, SGML (Standard Generalized Markup Language) and XML (Extended Markup Language).

In addition, the term "Web appliance" should be broadly construed to cover the display system illustrated in **Figures 6A-6D**, as well as any other machine in which a browser application is associated with some television class or other display monitor. Moreover, while the preferred embodiment is illustrated in the context of a dial-up network, this is not a limitation of the present invention. There may be other "bottleneck" resources in a direct connect network that could be managed indirectly by using this approach.

CLAIMS

1. A method for managing use of a digital file, comprising the steps of:

establishing a secure link between a pair of devices, each of the devices being certified to operate under a given security protocol;

establishing a usage scheme defining one or more conditions under which the digital file may be transferred between the pair of devices; and

transferring one or more copies of the digital file over the secure link between the pair of devices in accordance with the established usage scheme.

2. The method as described in Claim 1 wherein the pair of devices include a storage device and a rendering device.

3. The method as described in Claim 2 wherein the storage device and the rendering device are located in a computer.

4. The method as described in Claim 2 wherein the storage device is located in a first computer and the rendering device is located in a second computer and the secure link is established over a computer network connecting the first and second computers.

5. The method as described in Claim 4 wherein the second computer is a personal computer and the rendering device includes circuitry for establishing the secure link.

6. The method as described in Claim 4 wherein the second computer is a Web appliance and the rendering device includes software for establishing the secure link.

7. The method as described in Claim 2 wherein the rendering device is selected from a group of rendering devices consisting essentially of a printer, a display, and a sound card.

8. The method as described in Claim 1 further including the step of establishing an account representing a given monetary value.

9. The method as described in Claim 8 further including the step of allocating a given charge against the given monetary value when a copy of the digital file is transferred between the pair of devices.

10. The method as described in Claim 9 further including the step of associating the given charge with a content provider account to facilitate the payment of the given consideration to the provider of the digital file.

11. The method as described in Claim 1 wherein the usage scheme includes a given payment method.

12. A method for managing use of digital material in a computer network, comprising the steps of:

establishing an account for a given client computer including a representation of a given monetary value;

establishing an account for a given content provider including a representation of a given royalty value;

establishing a count of a number of permitted copies of a digital file;

in response to a given protocol, transferring a copy of the digital file from a source to a target associated with the given client computer;

adjusting the given monetary value in the account of the given client computer; and

adjusting the given royalty value in the account of the given content provider.

13. The method as described in Claim 12 wherein the given protocol includes the steps of:

determining whether a given client computer requesting transfer of the digital file is authorized to effect the transfer;

if the client is authorized to effect the transfer of the digital file, determining whether the count has a given value; and

if the count has the given value, transferring the digital file from the source to the target.

14. The method as described in Claim 13 wherein the given value is a non-zero value.

15. The method as described in Claim 13 wherein the given protocol further includes the step of adjusting the count after a copy of the digital file has been transferred.

16. The method as described in Claim 15 wherein the count is decremented.

17. The method as described in Claim 12 wherein the source and target are located in the given client computer connected to the computer network.

18. The method as described in Claim 17 wherein the source is a disk storage device and the target is a device selected from a group of rendering devices consisting essentially of a printer, a display, and a sound card.

19. The method as described in Claim 12 wherein the source is located on a first computer and the target is located on a second computer connected to the first computer via the computer network.

20. A method for managing use of digital material in a computer network including a Web client connectable to a Web server, comprising the steps of:

establishing a count of a number of permitted copies of a digital file located at a source device in the Web client;

in response to a given protocol, transferring one or more copies of the digital file from the source device to a set of one or more target rendering devices in the Web client; and

for each such transfer from the source device to one of the target rendering devices, logging an indication that the digital file has been transferred to facilitate payment of a given consideration to a provider of the digital file.

21. The method as described in Claim 20 wherein the Web client is a Web appliance and the source device is a secure disk storage.

22. The method as described in Claim 21 wherein each target rendering device is a device selected from a group of target rendering devices consisting essentially of a printer, a display, and a sound card.

23. The method as described in Claim 20 wherein the Web client is connected to the Web server via a non-secure connection.

24. The method as described in Claim 23 wherein the given protocol further includes the step of establishing a secure channel between the source device and a target rendering device prior to transferring the digital file.

25. The method as described in Claim 24 wherein the step of establishing a secure channel includes generating a secret key shared by the source device and the target rendering device.

26. The method as described in Claim 25 wherein the source device encrypts the digital file with the secret key as the source device transfers the digital file to the target rendering device, and wherein the target rendering device decrypts the digital file with the secret key upon receipt.

27. A computer program product in computer-readable media for use in a Web client having a source device and one or more target rendering devices, the computer program product comprising:

means for establishing a count of a number of permitted copies of a digital file located at the source device;

means, responsive to a given protocol, for transferring one or more copies of the digital file from the source device to the one or more target rendering devices;

means, responsive to each transfer, for logging an indication that the digital file has been transferred to facilitate payment of a given consideration to a provider of the digital file; and

means responsive to the logging means for adjusting the count.

28. The computer program product as described in Claim 27 further including means responsive to a given occurrence for transferring the indication to a central authority.

29. The computer program product as described in Claim 28 wherein the given occurrence is establishing a dialup connection between the Web client and an Internet Service Provider.

30. A computer system connected to a computer network and including a source device and one or more target rendering devices, comprising:

a processor;

an operating system;

an application for managing use of digital material, comprising:

means for establishing a count of a number of permitted copies of a digital file located at the source device;

means, responsive to a given protocol, for transferring one or more copies of the digital file from the source device to the one or more target rendering devices;

means, responsive to each transfer, for logging an indication that the digital file has been transferred to facilitate payment of a given consideration to a provider of the digital file; and

means responsive to the logging means for adjusting the count.

31. The computer system as described in Claim 30 wherein the application further includes means for restricting transfer of the digital file when the count reaches a given value.

32. A data processing system, comprising:

a remote control unit; and

a base unit connectable to a monitor for providing Internet access under the control of the remote control unit, the base unit comprising:

a processor having an operating system;

a browser application run by the operating system;

a secure disk storage in which a digital file is stored;

one or more target rendering devices; and

means for restricting a number of copies of the digital file that may be transferred between the secure disk storage and the one or more target rendering devices.

33. The data processing system as described in Claim 32 wherein the restricting means includes means responsive to a given occurrence for transmitting an indication of a number of copies of the digital file that were transferred between the secure disk storage and the one or more target rendering devices during a given time interval.

34. The data processing system as described in Claim 33 wherein the given occurrence is a dialup connection of the data processing system to an Internet Service Provider.

35. A management server for use in managing collection and allocation of royalties among content providers, the management server connected in a computer network to an access provider servicing a plurality of Web client appliances receiving dialup access to Web content, the management server comprising:

means for establishing an account for each of set of given content providers, each account including a representation of a given royalty value; and

means for adjusting the given royalty value in the account of the given content provider in response to receipt of an indication that a given digital file associated with the given content provider has been transferred from a source to a target rendering device in a given Web client appliance.

36. A copy management system, comprising:

a first device and a second device, each of which is certified to operate under a given security protocol;

means for establishing a secure link between the first and second devices; and

means responsive to establishment of the secure link for managing transfer of a permitted number of copies of a digital file between the first and second devices in accordance with copy control information restrictions associated with the digital file.

FIG. 1

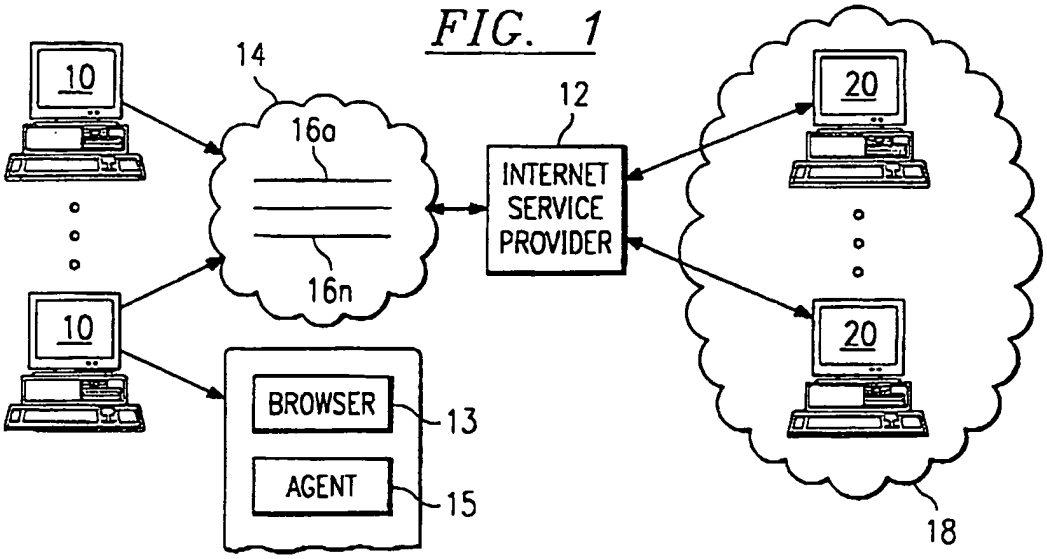


FIG. 2

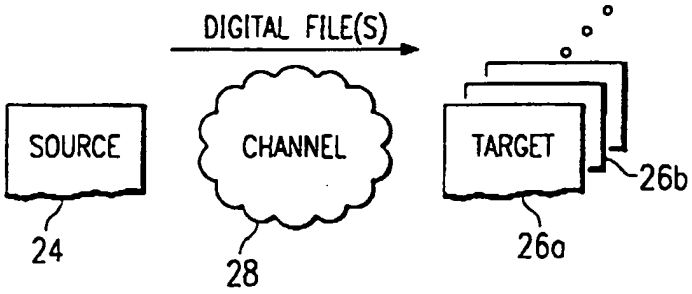


FIG. 3

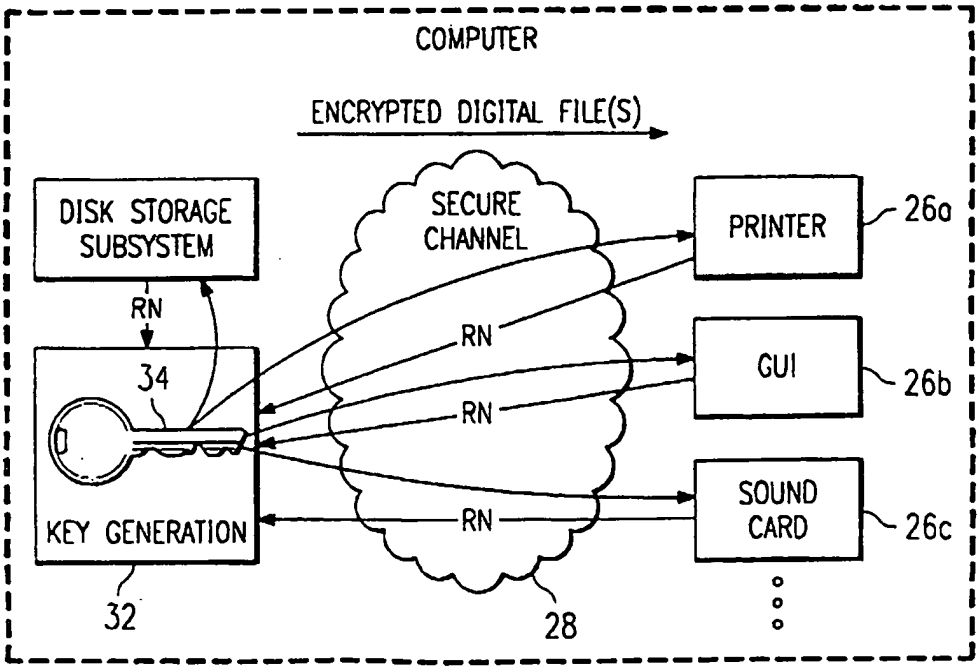
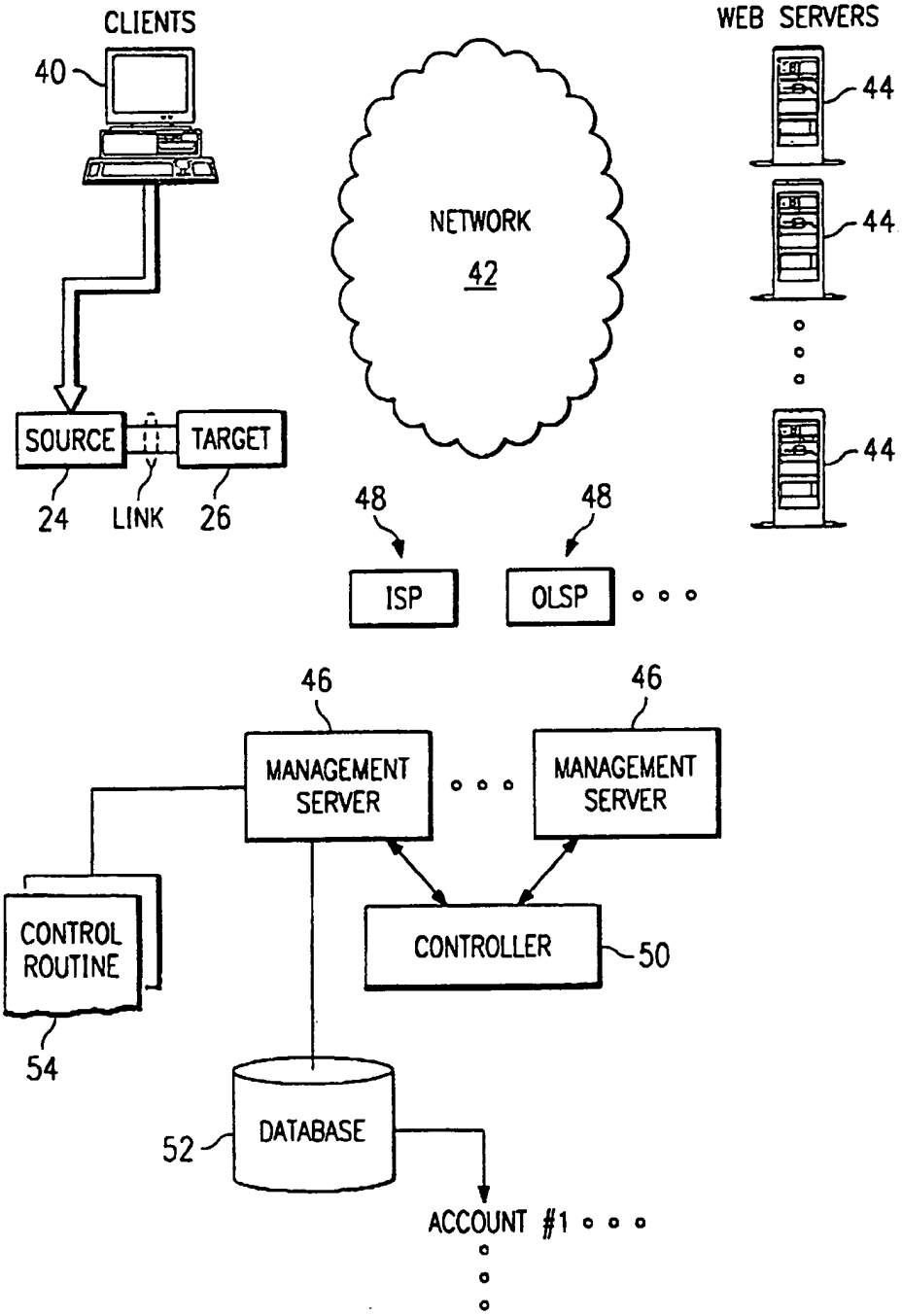


FIG. 4


CLIENTS
40
WEB SERVERS
44
44
44
NETWORK
42
SOURCE
TARGET
24
LINK
26
48
48
ISP
OLSP
46
46
MANAGEMENT SERVER
MANAGEMENT SERVER
CONTROL ROUTINE
54
CONTROLLER
50
DATABASE
52
ACCOUNT #1

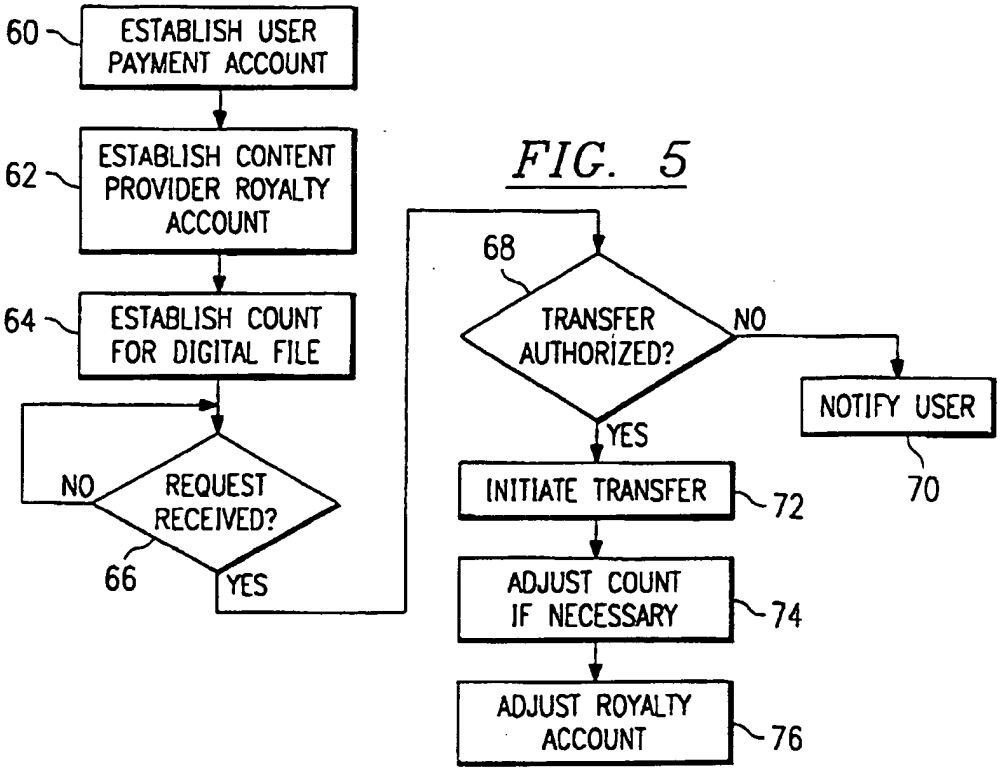
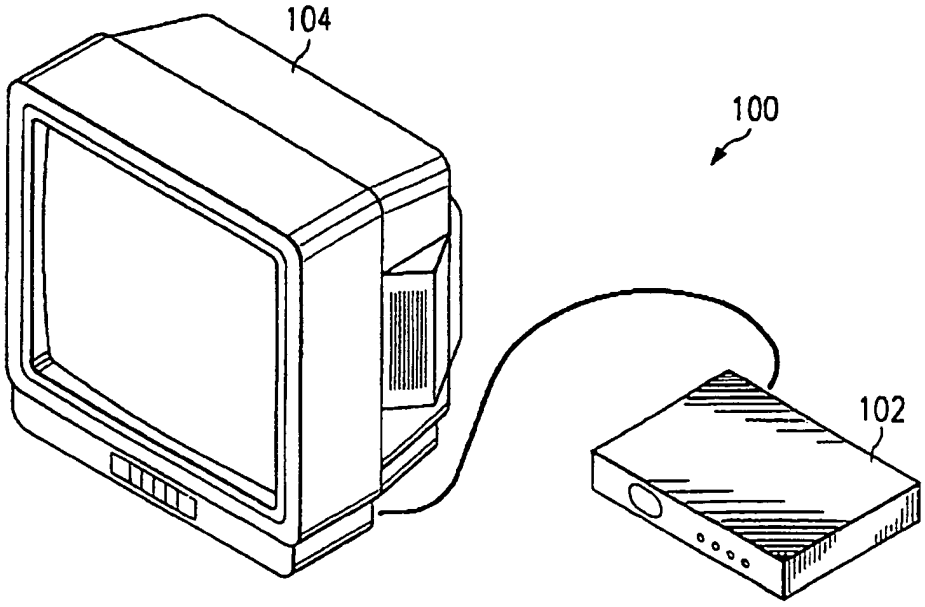

60
ESTABLISH USER PAYMENT ACCOUNT
62
ESTABLISH CONTENT PROVIDER ROYALTY ACCOUNT
64
ESTABLISH COUNT FOR DIGITAL FILE
NO
REQUEST RECEIVED?
66
YES
68
TRANSFER AUTHORIZED?
NO
NOTIFY USER
70
YES
INITIATE TRANSFER
72
ADJUST COUNT IF NECESSARY
74
ADJUST ROYALTY ACCOUNT
76

FIG. 5


104
100
102

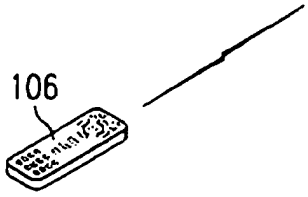

106

FIG. 6A

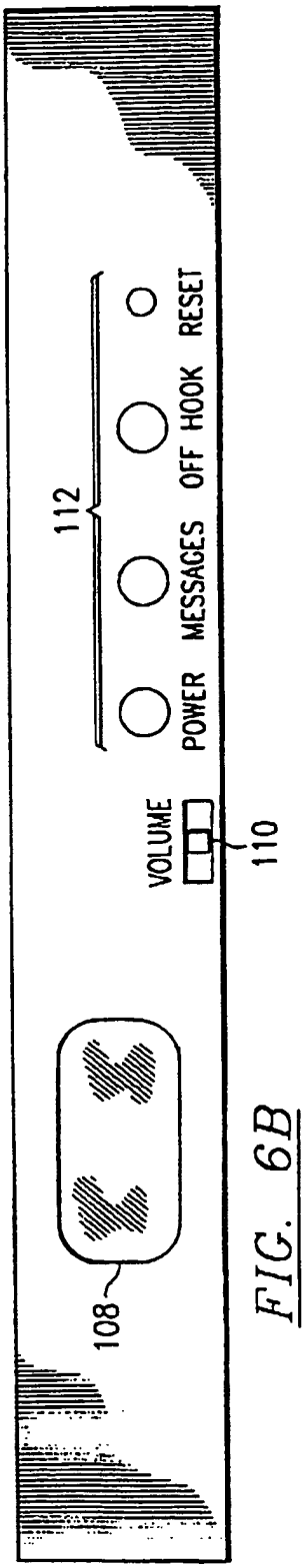

112
POWER
MESSAGES
OFF HOOK
RESET
VOLUME
110
108

FIG. 6B

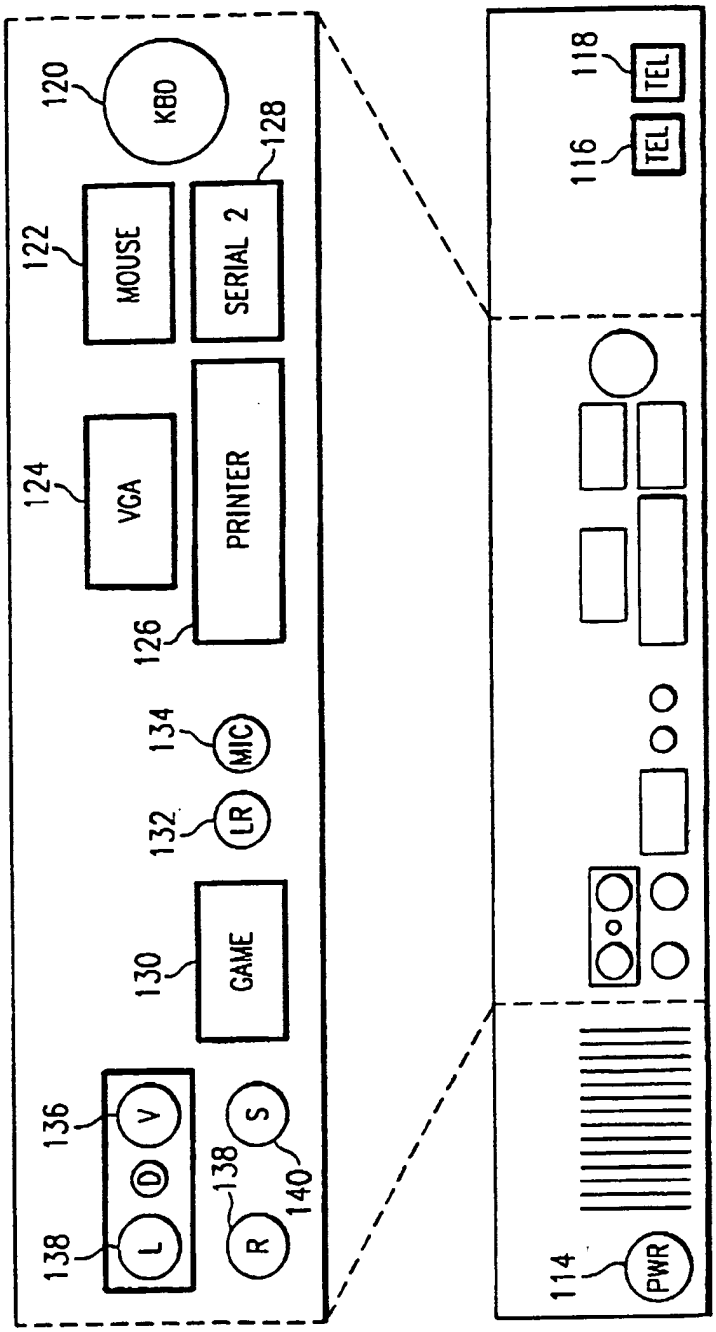

120
KBD
122
MOUSE
128
SERIAL 2
124
VGA
126
PRINTER
134
MIC
132
LR
130
GAME
136
V
138
L
138
140
S
R
118
TEL
116
TEL
114
PWR

FIG. 6C

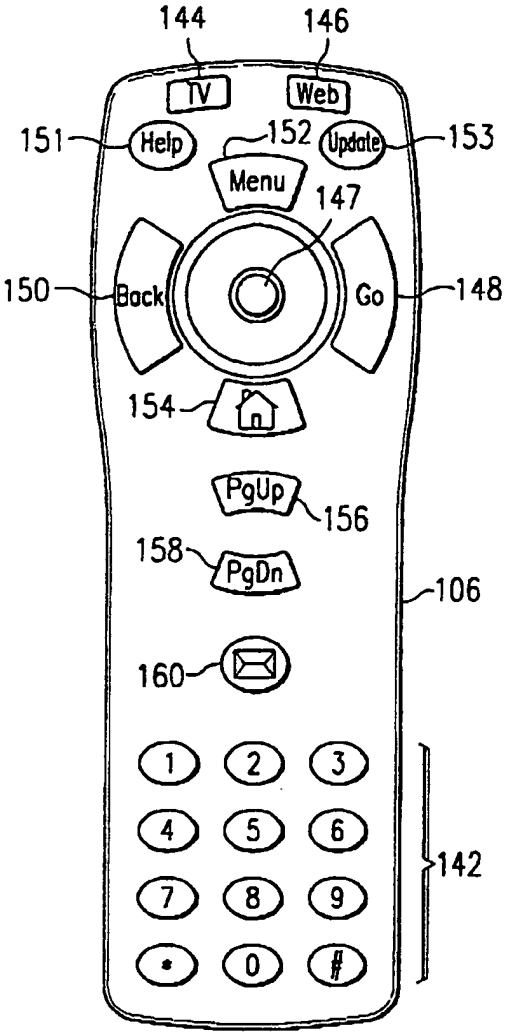
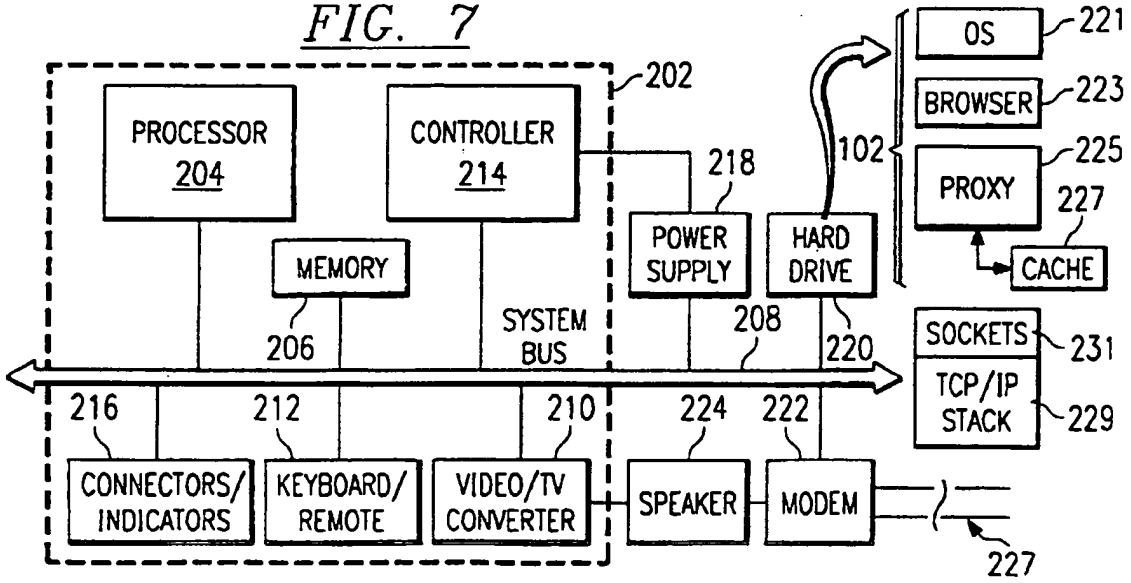

144
146
TV
Web
151
Help
152
Update
153
Menu
147
150
Back
Go
148
154
PgUp
156
158
PgDn
106
160
1
2
3
4
5
6
7
8
9
*
0
#
142

FIG. 6D

FIG. 7


202
PROCESSOR
204
CONTROLLER
214
218
102
OS
221
BROWSER
223
PROXY
225
227
CACHE
MEMORY
206
SYSTEM
BUS
POWER
SUPPLY
HARD
DRIVE
208
220
SOCKETS
231
TCP/IP
STACK
229
216
212
210
224
222
CONNECTORS/
INDICATORS
KEYBOARD/
REMOTE
VIDEO/TV
CONVERTER
SPEAKER
MODEM
227

INTERNATIONAL SEARCH REPORT

Int. ional Application No
PCT/GB 98/03828

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F1/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5 532 920 A (HARTRICK THOMAS V ET AL) 2 July 1996 see figures 1,2 see column 6, line 44 - column 7, line 15 see column 14, line 51 - column 16, line 25	1-20, 23-25, 27-33, 35,36
Y	EP 0 798 906 A (SUN MICROSYSTEMS INC) 1 October 1997 see figures 1-3 see column 4, line 21 - column 5, line 33	1-20, 23-25, 27-33, 35,36

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
31 March 1999	08/04/1999

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Weiss, P

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

Int ional Application No
PCT/GB 98/03828

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5532920 A	02-07-1996	EP 0567800 A	03-11-1993
		JP 2659896 B	30-09-1997
		JP 6103286 A	15-04-1994
EP 0798906 A	01-10-1997	US 5761421 A	02-06-1998
		JP 10105529 A	24-04-1998

Form PCT/ISA/210 (patent family annex) (July 1992)

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04L 9/32 | A2

(11) International Publication Number: **WO 99/60750**

(43) International Publication Date: 25 November 1999 (25.11.99)

(21) International Application Number: PCT/FI99/00432

(22) International Filing Date: 18 May 1999 (18.05.99)

(30) Priority Data:
981132 20 May 1998 (20.05.98) FI

(71) Applicant *(for all designated States except US)*: NOKIA NETWORKS OY [FI/FI]; Keilalahdentie 4, FIN–02150 Espoo (FI).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: USKELA, Sami [FI/FI]; Puistokaari 8 B 12, FIN–00200 Helsinki (FI).

(74) Agent: KOLSTER OY AB; Iso Roobertinkatu 23, P.O. Box 148, FIN–00121 Helsinki (FI).

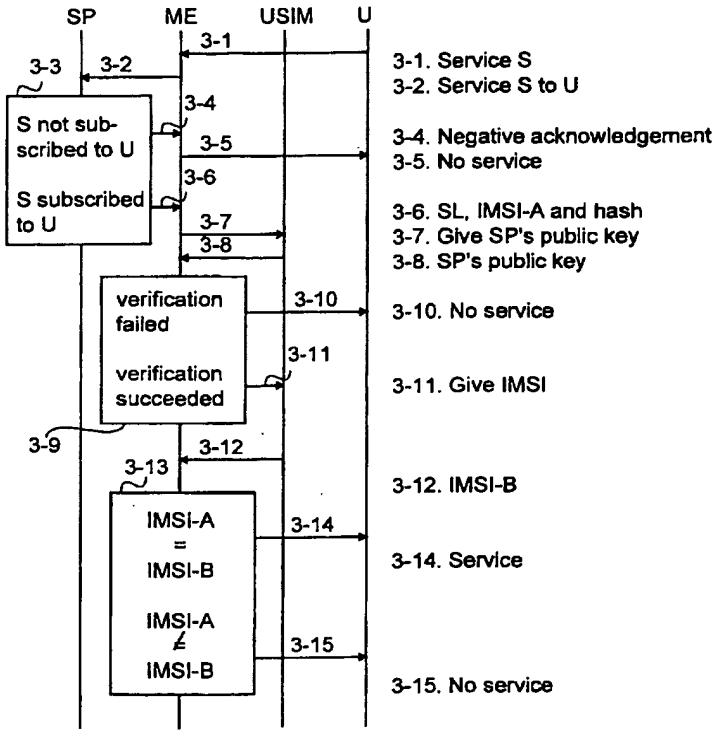
(81) Designated States: AE, AL, AM, AT, AT (Utility model), AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, CZ (Utility model), DE, DE (Utility model), DK, DK (Utility model), EE, EE (Utility model), ES, FI, FI (Utility model), GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SK (Utility model), SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
In English translation (filed in Finnish).
Without international search report and to be republished upon receipt of that report.

(54) Title: PREVENTING UNAUTHORIZED USE OF SERVICE

(57) Abstract

A method, a system, a network element and an apparatus of a telecommunication system for preventing unauthorized use of a service. The method, in which a service request is received from a user and the service is generated by means of a service logic, is characterized in that to prevent unauthorized use of the service, authentication data is appended to the service logic (3–6), the user requesting the service is authenticated by means of the authentication data (3–9), and the service logic is executed (3–14) only if the authentication succeeds.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

PREVENTING UNAUTHORIZED USE OF SERVICE

BACKGROUND OF THE INVENTION

The invention relates to preventing unauthorized use of services and especially to preventing unauthorized use of the services in a mobile communication system.

Mobile communication systems were developed, because there was a need to allow people to move away from fixed telephone terminals without affecting their reachability. The services offered through mobile stations have developed along with the mobile communication systems. At the moment, various new forms of service are being planned for the current and particularly for the future third-generation mobile communication systems, such as Universal Mobile Telecommunication System (UMTS) and International Mobile Telecommunication 2000 (IMT-2000). UMTS is being standardized by ETSI (European Telecommunications Standards Institute), whereas ITU (International Telecommunications Union) is standardizing the IMT-2000 system. These future systems are very similar in basic features. The following will describe in greater detail the IMT-2000 system whose architecture is illustrated in Figure 1.

Like all mobile communication systems, IMT-2000 produces wireless data transmission services to mobile users. The system supports roaming, in other words, IMT-2000 users can be reached and they can make calls anywhere within the IMT-2000 system coverage area. IMT-2000 is expected to fulfil the need for a wide range of future services, such as virtual home environment (VHE). With the virtual home environment, an IMT-2000 user has access to the same services everywhere within the coverage area of the system. According to present knowledge, a flexible implementation of various services and especially supporting roaming requires the loading of certain service logics into the terminal of the user and/or the serving network. A serving network is the network through which the service provider offers his service to the end-user. A service logic is a program, partial program, script or applet related to the service. The service is generated by means of the service logic by executing at least the service logic and the functions defined in it. A service can also comprise several service logics.

A problem with the arrangement described above is that it does not in any way verify that the user really has the right to use the service. It is

especially easy to copy and make unauthorized use of services in which the service logic is loaded into the terminal and/or serving network.

BRIEF DESCRIPTION OF THE INVENTION

Thus, it is an object of the invention to develop a method and an apparatus implementing the method so as to solve the above-mentioned problem. The object of the invention is achieved by a method, a system, a network element and an apparatus characterized by what is stated in the independent claims. The term apparatus refers here to a network element of the serving network, a terminal or any other corresponding service platform, into which the service logic can be loaded. The preferred embodiments of the invention are set forth in the dependent claims.

The invention is based on the idea of forming a service logic of two parts: user authentication and the actual service logic. The data required for user authentication is appended to the service logic, and the user is always authenticated before executing the actual service logic. This provides the advantage that an unauthorized use and copying of the service logic can be prevented. Only the users, to whom the service is subscribed and who thus have the right to use the service, can use it.

In a preferred embodiment of the invention, the service provider is always verified before the service is executed. This improves considerably the security of the user and a possible service platform into which the service logic is loaded. This ensures that the service logic truly originates from the service provider.

In a preferred embodiment of the invention, subscriber identification used to individualise a user is used in user authentication. This provides the advantage that subscriber authentication is simple, but reliable.

In a preferred embodiment of the invention, the service logic is saved with its user and authentication data in the memory of the service platform where it is loaded, and for a new user, only the authentication data of the new user is loaded. This provides the advantage that the service logic need not be loaded several times consecutively, which reduces the network load.

BRIEF DESCRIPTION OF THE DRAWINGS

In the following, the invention will be described in more detail in connection with preferred embodiments and with reference to the attached drawings in which

Figure 1 illustrates the IMT-2000 architecture,

Figure 2 shows a flow chart of the service platform functions in a first preferred embodiment of the invention,

Figure 3 is a signalling diagram of a second preferred embodiment of the invention, and

Figure 4 shows the operation of a network element controlling a service of a service provider in a third preferred embodiment of the invention.

DETAILED DESCRIPTION OF THE INVENTION

The present invention can be applied to any data transmission system in which the user can receive the subscribed services in any terminal supporting service provision. In the following, the invention will be described using the IMT-2000 system as an example, without limiting the invention to this particular system, however. The specifications of mobile communication systems in general and those of the IMT-2000 and UMTS system in particular evolve rapidly. This evolution may require extra changes to the invention. Therefore, all terms and expressions should be interpreted as widely as possible and they are intended to describe and not to limit the invention. It is the function that is essential for the invention and not in which network element or apparatus it is executed.

Figure 1 shows the network architecture of the IMT-2000 system on a general level, because the system specifications are currently being defined. A more detailed network structure bears no essential significance with regard to the invention. Third-generation mobile communication systems separate a service provider SP and a network operator from each other. A service provider offers services to an end-user through a network SN of one or more network operators. This type of network SN is called a serving network. A service provider can offer services through a serving network SN of one or more network operators. In addition, a service provider may switch to another serving network during the service without the user noticing it. A service provider can also be a network operator. A serving network SN comprises an actual access network AN, one or more core networks CN, and an

interworking unit adapting interfaces IWU for each different type of core network. According to present knowledge, an access network comprises base stations BS and radio network controllers RNC controlling them (not shown in the figure). A core network can be a network according to the pan-European mobile communication system GSM (Global System for Mobile Communication). Connections to other networks ON are established through a core network CN.

In the example in Figure 1, a home location register with IMT-2000 enhancement HLRi and the service control node SCN have been located in the serving network SN. The enhanced home location register HLRi contains not only the home register data of the core network but also the subscriber and service data required by the IMT-2000 system. The service provider SP maintains this IMT-2000 data for the part of the services. The subscriber makes an order agreement with the service provider which then charges the subscriber for the use of the services. The service control node SCN is a service platform to which the service logic related to the service can be loaded and in which it can be executed. The service control node SCN can also take care of loading the service elsewhere in the network and forward service requests from the user to the service provider. In addition to this, the service control node SCN makes sure that the services of the home network are also available in the visited networks.

In third-generation mobile communication networks, subscriber and user are also separated. The subscriber grants the user access to the subscribed services by giving the user an identification card (IC Card), for instance a USIM card (User and Services and Identity Module). The user accesses the services with a mobile terminal MT which is connected through base stations BS to a serving network SN over a radio path. A mobile terminal MT comprises actual mobile equipment ME and a detachably connected identification card USIM, also called a subscriber identity module. It is a smart card which can be detached from the mobile terminal and with which the subscriber can use a card-controlled mobile terminal. The user is identified by the card in the mobile terminal and not by the terminal itself. According to present knowledge, the USIM card is a multi-functional card and supports mobile communication system applications and other applications, such as Java applications, healthcare applications, etc. The subscriber can subscribe to the services of several different service providers with the same subscriber

identity module USIM. The subscriber and the user can be one and the same person. The subscriber identity module USIM also contains an international mobile subscriber identity IMSI with which the subscriber can be explicitly identified and which can also be used to identify the user. The identifier of a mobile subscriber is called subscriber identity.

The terminal selection of third-generation systems will probably be extremely versatile. The terminal can be a simplified terminal for speech only or it can be a terminal providing diverse services, which acts as a service platform and supports the loading and execution of various service logics.

A mobile communication system implementing the functionality of the present invention comprises not only means required for generating and loading services according to prior art, but also means for appending authentication data to a service logic and means for authenticating the user prior to executing the service logic. Here, appending also refers to embedding data into the service logic. In addition, the system can comprise means for verifying the service provider and means for saving the service logic with its supplementary data into the memory and means for receiving plain authentication data. The means for appending authentication data and the possible means for appending verification data are preferably located together with the means required for loading the service logic of the service provider. The other means are preferably located on the service platform, for instance in the terminal or the service control point of the network operator. The means or a part of them can also be located elsewhere, for instance in the network node of the subscriber network or in the serving support node of the core network.

Figure 2 shows a flow chart of the operation according to the first preferred embodiment of the invention on a service platform which can be actual mobile equipment ME or a service control node SCN, for instance. In the first preferred embodiment of the invention, an encryption technique, known per se, based on public keys is utilized in a novel and inventive manner. One such encryption technique is RSA (Rivest Shamir Adleman public-key cryptographic algorithm) which can be used for both encryption and digital signature. In the first preferred embodiment of the invention, at least the secret key of the subscriber and the public key of the service provider are saved in the subscriber identity module USIM. If the subscriber has several key pairs, the secret key of the pair, whose public key has been entered in the subscriber data of the user, is saved. Correspondingly, a service provider can

have several key pairs of which one, for instance, is saved in the subscriber identity module and information on the pair, whose key is saved in the identity module, is entered in the subscriber data. This ensures that the secret and public key of the same pair is used. In the first preferred embodiment of the invention, the service provider is verified by a digital signature. It is generated in the first embodiment of the invention by calculating a one-way hash (one-way hash function) from the service logic, which is then encrypted. This embodiment provides the unexpected advantage that in connection with the verification of the service provider, the fact whether the service logic has been changed, is also checked. If the service logic has been changed, the hash calculated from it also changes and the service provider verification does not succeed anymore.

With reference to Figure 2, a service request concerning a service S1 is received from a user U1 in step 200. In step 201, a check is made to see whether the service logic SL1 related to the service S1 is in the memory. If it is not, the service request is forwarded to the service provider in step 202. The service provider finds the actual service logic SL1 related to the service S1 and appends to the service logic authentication data A1 required for user authentication, which in the first preferred embodiment is the public key of the subscriber. After this, the service provider calculates the hash from the actual service logic and the authentication data and appends it as verification data V1 to the service logic and encrypts the thus created file with its own secret key. The file contains the verification data V1, the authentication file A1 and the actual service logic SL1. Alternatively, the service provider could encrypt the hash with its secret key, append the encrypted hash as the verification data V1 to the file and then encrypt the file with the public key of the user. After this, in step 203, the file, i.e. the actual service logic SL1 related to the service S1, the authentication data A1 appended to it for the user U1, and the verification data V1 calculated from them, is loaded onto the service platform. In step 204, the service logic SL1 is saved and in it, the authentication data A1 and the verification data V1 related to the user U1, and, of course, information on the user U1 to which the authentication data A1 and the verification data V1 are related. The data is stored in encrypted format in the memory. Then, in the first preferred embodiment, the public key of the service provider is requested from the subscriber identity module USIM in the terminal of the user in step 205 and received in step 206, after which the encryption of the service

logic SL1, the authentication data A1 and the verification data V1 is decrypted in step 207 using the received key. In embodiments in which the service provider only has one key pair, the information on the public key of the provider can already be on the service platform and need not be separately requested. When the encryptions have been decrypted, the service provider is verified by calculating a hash from the service logic and the authentication data in step 208 and by comparing the thus calculated hash with the verification data V1 in step 209. If they are the same, the verification of the service provider succeeds, and after this, a challenge, i.e. a character string, is selected in step 210. How the challenge is selected bears no significance with regard to the invention. A simple and safe solution is to use a random number generator, whereby the challenge is a random number. The selected challenge is encrypted in step 211 with the public key of the subscriber, i.e. the authentication data A1. After this, in step 212, the encrypted challenge is sent to the subscriber identity module USIM in the terminal of the user U1, which decrypts the encrypted challenge into plain text with the secret key of the subscriber and sends the plain text back to the service platform. In step 213, the service platform receives the plain text and in step 214, it compares the original challenge with the plain text. If the character strings are the same, user authentication succeeds and the actual service logic SL1 can be executed in step 215.

If it is detected in step 209 that the calculated hash is not the same as the verification data, the service provider verification fails or the service logic has been changed. In both cases, executing the service logic would be a security risk and, therefore, it is not executed, and in step 217, all data saved for the service S1, i.e. the service logic SL1 and all appended authentication and verification data with user data, is deleted from the memory.

If it is detected in step 214 that the challenge is not the same as the plain text, authentication does not succeed and the service logic is not executed, and in step 216, the authentication data A1, verification data V1 and information on the user U1 appended to the service logic SL1 of the service S1, is deleted from the memory. This way, the actual service logic SL1 need not be loaded next time, only the authentication data and verification data.

If it is detected in step 201 that the service logic SL1 related to the service S1 is in the memory, a check is made in step 218 to see if authentication and verification data for the user U1 is appended to it. If this

data, too, is in the memory, operation continues from step 205 where the public key of the service provider is requested from the subscriber identity module USIM. From step 205 onward, operation continues as described above. This way, network resources are saved, because the once loaded data need not be loaded again.

If it is detected in step 218 that no authentication and verification data for the user U1 is appended to the service logic SL1 in the memory, in step 219, the authentication and verification data for the service S1 is requested from the service provider for the user U1. The authentication data A1 and the verification data V1 are received in step 220, after which they and information on the user U1 are appended to the service logic SL1 in step 221. After this, operation continues from step 205 where the public key of the service provider is requested from the subscriber identity module USIM. From step 205 onward, operation continues as described above.

The service platform can be actual mobile equipment ME or a network element of the serving network, such as the service control node SCN. The memory where the data and service logics are saved can also be a cache memory. In embodiments in which the service logic is saved in the memory, the service platform can comprise means for deleting the service logic from the memory for predefined reasons, for instance after a certain time period.

In embodiment in which the service logic is not saved in the memory, steps 200, 202, 203 and 205 to 215 are executed. The data deletion described in steps 216 and 217 is not done, but the actual service logic is left unexecuted.

The steps described above in Figure 2 are not in absolute chronological order and some of the steps can be executed simultaneously or deviating from the given order. Other functions can also be executed between the steps. Some of the steps, such as the service provider verification, can also be left out. The essential thing is to authenticate the user before the actual loaded service logic is executed.

Figure 3 shows signalling according to a second preferred embodiment of the invention. In the second preferred embodiment, the subscriber identification IMSI is used as the authentication data. It is also assumed that the service logic is loaded into the actual mobile equipment ME and not saved in its memory.

With reference to Figure 3, user U sends information in message 3-1 to mobile equipment ME through the user interface requesting service S. The mobile equipment ME sends the service request through the serving network to service provider SP in message 3-2. The service request contains information on the required service S and the user U requesting the service. In step 3-3, the service provider checks if the service S is subscribed to the user U. If the service S is not subscribed to the user, the service provider sends through the serving network a negative acknowledgement to the service request in message 3-4 to the mobile equipment ME which forwards the information in message 3-5 through the user interface to the user U.

If the service S is subscribed to the user, the service provider retrieves the subscriber identification IMSI-A of the user U and the service logic SL related to the service S and calculates a hash from them. After this, the service provider encrypts the service logic and the related data (IMSI-A, hash) with the secret key of the service provider. In message 3-6, the service provider sends the service logic SL, the identification IMSI-A and the hash to the mobile equipment ME. In the second preferred embodiment, after receiving the message 3-6, the mobile equipment ME requests the public key of the service provider from the subscriber identity module USIM in message 3-7. The subscriber identity module USIM sends it to the mobile equipment in message 3-8, after which the mobile equipment ME verifies the service provider in step 3-9. The mobile equipment decrypts the encryption of the service logic SL, the subscriber identification A1 and the hash with the received public key and calculates a hash from the combination of the service logic and the subscriber identification IMSI-A. If the calculated hash is not the same as that received in message 3-6, the verification fails. In such a case, the service logic is not executed and the mobile equipment ME sends information on the verification failure through the user interface to the user U in message 3-10, saying, for instance, that the service is not available.

If the hash calculated in step 3-9 and the received hash are the same, the verification succeeds and the mobile equipment ME requests the subscriber identification IMSI of the user U from the subscriber identity module USIM in message 3-11. The subscriber identity module USIM retrieves the subscriber identification IMSI-B from its memory and sends it to the mobile equipment ME in message 3-12. In step 3-13, the mobile equipment authenticates the user by checking if the IMSI-A received from the service

provider is the same as the IMSI-B received from the identity module. If the user passes the authentication in step 3-9 (i.e. IMSI-A is the same as IMSI-B), the mobile equipment ME executes the actual service logic SL and provides the service through the user interface to the user U in messages 3-14. If the values of the subscriber identifications IMSI differ from each other, authentication fails. In such a case, the mobile equipment does not execute the actual service logic, but informs the user U through the user interface in message 3-10 that the authentication failed, saying, for instance, that the service is not available.

The signalling messages described above in connection with Figure 3 are for reference only and can contain several separate messages to forward the same information. In addition, the messages can also contain other information. The messages can also be freely combined. In embodiment in which the service provider is not verified, the messages 3-7, 3-8 and 3-10 related to verification and step 3-9 are left out. Depending on the service providers, core network and mobile equipment, other network elements, to which various functionalities have been distributed, can take part in the data transmission and signalling.

Figure 4 shows a flow chart of a network element controlling a service of a service provider in a third preferred embodiment of the invention. In the third preferred embodiment, authentication and verification are only performed when a service logic is loaded into a visited (visiting) network or mobile equipment. The visited network is a network whose network element, into which the service is loaded, is a network element belonging to a provider other than the service provider. The third preferred embodiment utilizes both public key encryption and symmetrical encryption, such as DES (Data Encryption Standard). The latter encryption technique is used when the service logic is loaded into the mobile equipment. A common key is saved for it in both the subscriber identity module and the subscriber data of the user. In addition, the public key of the service provider is saved in the subscriber identity module for the service logics to be loaded into visited networks. Only encryption of the service logic with the secret key of the service provider prior to sending the service logic to the serving network or encryption with the common key prior to loading it in the mobile equipment is used as signature.

With reference to Figure 4, in step 400, a service request concerning a service S2 is received from a user U2. In step 401, a check is

made to see if the user U2 subscribes to the service S2. If the user subscribes to the service S2, a check is made in step 402 to see if the service logic SL2 related to the service U2 requires loading into the mobile equipment ME of the user. If the service logic SL2 is loaded into the mobile equipment of the user, in step 403, a common key is retrieved from the subscriber data of the user U2 for encrypting the service logic SL2 in step 404. This common key is used both as the authentication data of the user and the verification data of the service provider. Nobody else should have any information on the common key in this case. The authentication and verification are performed in connection with the decryption of the service logic. The encrypted service logic SL2 is loaded into the mobile equipment ME in step 405. The user is authenticated and the service provider is verified in the mobile equipment, for instance by sending the encrypted service logic to the subscriber identity module USIM in the mobile equipment, which decrypts the service logic using the common key in its memory and sends the plain-text service logic to the mobile equipment. When the service logic has been executed, information concerning this is received in step 406, and the subscriber is charged for the use of the service in step 407.

If it is detected in step 402 that the service logic SL2 will not be loaded into the mobile equipment, a check is made in step 408 to see if the user U2 is in the home network area. If yes, the service logic SL2 is executed in step 409, after which operation continues from step 407 in which the user is charged for the use of the service.

If it is detected in step 408 that the user is not in the home network area, in the third preferred embodiment of the invention, the service logic SL2 must be loaded into the visited network. To do this, in step 410, the public key of the user U2 is retrieved from the subscriber data for appending it as authentication data to the service logic. In step 411, the public key of the user is appended to the service logic SL2, and they are encrypted using the secret key of the service provider in step 412. The encryption also acts as the verification data. If the service provider has several key pairs of public and secret keys, the secret key of the pair whose public key has been saved in the identity module of the user is used. The encrypted service logic, to which the authentication data is appended, is loaded into the visiting network in step 413. The network element of the visiting network verifies the service provider by decrypting the service logic using the public key of the service provider and

authenticates the user, for instance in the manner described in connection with Figure 2, after which the service logic is executed. When the service logic has been executed, information concerning this is received in step 406, and the subscriber is charged for the use of the service in step 407.

If it is detected in step 411 that the requested service is not subscribed to the user, information is transmitted in step 414 that the service is not available to the user.

Above, in connection with Figure 4, it was assumed that the authentication and verification succeeded. If this is not the case, the service logic is not executed and the subscriber not invoiced. The steps described above in connection with Figure 4 are not in absolute chronological order and some of the steps can be executed simultaneously or deviating from the given order. Other functions can also be executed between the steps. Some of the steps can also be left out. The essential thing is that the authentication data is in some way appended to the service logic being loaded.

In the above embodiments, the actual service logic has been changed to ensure that the authentication and verification are done. This has been done by adding to the service logic a part taking care of the authentication and verification, which is always executed before the service logic. In some embodiments, the service logic can only be changed to ensure the authentication. In some embodiments, there is no need to change the service logic, and the authentication data and the possible verification data are appended to the service logic as separate data, and the service platform makes sure that the authentication and the possible verification are done. In these embodiments, pre-encrypted service logics can be used, which reduces the load of the network element, because encryption is done only once.

It has been assumed above in connection with Figures 2, 3 and 4 that the service provider appends the authentication data to the service logic before the encryption. The authentication data can also be appended to a pre-encrypted service logic. In such a case, the serving network or mobile equipment can also be adapted to append the authentication data to the service logic, for instance by means of the user data provided in the service request. It has been presented above that the user is authenticated only after the verification. However, the order bears no significance with regard to the invention. The user can be authenticated before the service provider is verified in embodiment in which the service provider is also verified. The data and/or

service logic also need not be encrypted unless the encryption is used for authentication and/or verification. Other alternatives for authentication, verification and possible encryption than those described above in connection with the preferred embodiments can also be used. The preferred embodiments can also be combined. The essential thing is that the user is authenticated before executing the service logic at least when the service logic is loaded into the mobile equipment or visiting network. In embodiment in which the service logic is loaded into the mobile equipment, the encryption of the service logic with the public key of the subscriber can also be used as the authentication data. The subscriber is authenticated when the identity module USIM decrypts the encryption with the secret key of the subscriber. For security's sake, it is advantageous that USIM never sends even to the mobile equipment the secret key saved in it, and the decryption with the secret key is always performed in USIM. Other data for authentication and possible verification than used in the above examples can also be used. The requirements for the authentication data and possible verification data are adequate individualization, reliability and non-repudiation. Adequate individualization means that the data specifies the user at least by subscriber.

No hardware changes are required in the structure of the serving network. It comprises processors and memory that can be utilized in functions of the invention. All changes required for implementing the invention can instead be made as additional or updated software routines in the network elements into which the service logic is loaded. An example of such a network element is the service control node. Extra memory is also needed in the network element saving the loaded service logic with its supplementary data.

The structure of the service provider also requires no hardware changes. The service provider comprises processors and memory that can be utilized in functions of the invention. All changes required for implementing the invention can be made as additional or updated software routines to achieve the functionality of the invention. Extra memory may be needed depending on the embodiment of the invention. It is, however, limited to a small amount sufficient for saving the extra authentication data and the possible verification data.

The structure of the mobile equipment requires no hardware changes. It comprises processors and memory that can be utilized in functions of the invention. All changes required for implementing the invention can

instead be made as additional or updated software routines in the mobile equipment which is adapted to function as a service platform. If the service logic is saved in the mobile equipment, extra memory is also needed.

In the subscriber identity module USIM, the extra memory possibly needed for implementing the invention is limited to a small amount sufficient for saving the extra authentication data, the possible verification data and the decryption algorithms possibly needed.

It will be understood that the above description and the figures related to it are only presented for the purpose of illustrating the present invention. The various modifications and variations of the invention will be obvious to those skilled in the art without departing from the scope or spirit of the invention disclosed in the attached claims.

CLAIMS

1. A method for preventing unauthorized use of a service in a mobile communication system, in which method

a service request is received from a user of the service, and

the service is generated by means of a service logic,

characterized in that in the method

authentication data is appended to the service logic (3-6),

the user requesting the service is authenticated by means of the authentication data (3-9), and

the service logic is executed only if the authentication succeeds (3-14).

2. A method as claimed in claim 1, **characterized** in that

verification data of the service provider is also appended to the service logic,

the service provider is verified in connection with user authentication (3-13), and

the service logic is executed only if the verification also succeeds.

3. A method as claimed in claim 2, **characterized** in that

a first hash calculated from the service logic is used as verification data of the service logic,

the service logic is loaded onto a service platform where it is executed to generate the service,

the service provider is verified on the service platform by calculating a second hash from the service logic, and

if the first and the second hash are the same, the verification succeeds,

if the first and the second hash differ, the verification fails.

4. A method as claimed in claim 2, **characterized** in that

the signature of the service provider is used as the verification data,

the service logic is signed by encrypting it with the secret key of the service provider, and

the service provider is verified by decrypting the encryption of the service logic with the public key of the service provider.

5. A method as claimed in any one of the above claims, **characterized** in that

the secret key of the subscriber is saved in the subscriber identity module (USIM) of the user of the service,

the public key of the subscriber is used as the authentication data,

a challenge encrypted with the public key of the subscriber is sent to the subscriber identity module located in the mobile equipment of the user requesting the service (207),

the challenge is decrypted into plain text with the secret key of the subscriber in the identity module,

the plain text is received from the identity module (208),

a check is made to see if the unencrypted challenge and the plain text correspond to each other (209), and

if they correspond, the authentication succeeds, and

if they do not correspond, the authentication fails.

6. A method as claimed in claims 1, 2, 3 or 4, **characterized** in that

individual identity of the subscriber is used as the authentication data,

a service request is received from the user (3-1),

the individual subscriber identity related to the user is requested (3-7),

the requested identity is received (3-8),

a check is made to see if the authentication data and the requested identity correspond to each other (3-9), and

if they correspond, the authentication succeeds, and

if they do not correspond, the authentication fails.

7. A method as claimed in any one of the above claims, **characterized** in that

the service logic is loaded onto the service platform where it is executed to generate the service, and

the authentication data is appended to the service logic in connection with the loading.

8. A method as claimed in claim 7, **characterized** in that

the service logic, the authentication data appended to it, and the data indicating the user are saved on the service platform in connection with the loading (204),

a service request is received from the user,

a check is made to see if the service logic related to the requested service is saved on the service platform (201), and

if not, the service logic is loaded (203),

if yes,

- a check is made to see if authentication data has been saved for the user requesting the service (217), and

- if yes, the user is authenticated,

- if not,

-- authentication data is requested for the user (218),

-- the authentication data and the data indicating the user are saved in the service logic (220), and

-- the user is authenticated.

9. A telecommunication system comprising

a first part (SP) to produce the service for the user by means of a service logic, and

a second part to provide the service (SN, MT) to the user of the service,

in which system the first part (SP) is adapted to identify the user requesting the service and to check, if the service is subscribed to the user, and if the service is subscribed to the user, to generate the service by loading the service logic into the second part (SN, MT) which is adapted to provide the service by executing the loaded service logic,

characterized in that

the first part (SP) is adapted to append authentication data into the service logic being loaded for user authentication, and

the second part (SN, MT) is adapted to authenticate the user and to execute the service logic only in response to a successful authentication.

10. A system as claimed in claim 9, **characterized** in that

the first part (SP) is adapted to sign the service logic by encrypting it with an encryption key agreed with the second part, and

the second part (SN, MT) is adapted to verify the first part by decrypting the encryption of the service logic with a key corresponding to the agreed key and to execute the service logic only if the verification also succeeds.

11. A system as claimed in claim 9 or 10, **characterized** in that

the telecommunication system is a mobile communication system (IMT-2000) comprising at least one service provider and serving network,

the first part is the service provider (SP), and

the second part is the serving network (SN) comprising at least one network element (SCN).

12. A system as claimed in claim 9 or 10, **characterized** in that

the telecommunication system is a mobile communication system (IMT-2000) comprising at least one service provider (SP) and mobile terminal (MT) which is connected to the service provider through a serving network (SN) and which mobile terminal (MT) comprises in addition to actual mobile equipment (ME) a subscriber identity module (USIM) which is detachably connected to the mobile equipment,

the first part is the service provider (SP), and

the second part is the actual mobile equipment (ME).

13. A network element (SP) generating a telecommunication system service for a user, which produces the service by means of a service logic and which comprises means for identifying the user requesting the service and for checking if the service is subscribed to the user and for loading the service logic into the telecommunication system if the service is subscribed to the user,

characterized in that the network element (SP) comprises means for appending the authentication data to the service logic being loaded so that the user of the service is authenticated before the service logic is executed.

14. A network element (SP) as claimed in claim 13, **characterized** in that it comprises means for signing the service logic before it is loaded into the network.

15. A network element (SP) as claimed in claim 13 or 14, **characterized** in that it comprises a processor arranged to execute software routines, and said means have been implemented as software routines.

16. An apparatus of a telecommunication system, which apparatus comprises service logic executing means for providing a service from a service provider of a telecommunication system to a user of the service,

characterized in that the apparatus (SCN, ME) comprises

separation means for separating the authentication data of a user from a loaded service logic,

authentication means responsive to the separation means for user authentication, and

service logic execution means are adapted to be responsive to the authentication means.

17. An apparatus (SCN, ME) as claimed in claim 16, **characterized** in that

it comprises verification means for service provider verification by means of verification data in the loaded service logic, and

the service logic verification means are adapted to be responsive to the authentication means.

18. An apparatus (SCN, ME) as claimed in claim 16 or 17, **characterized** in that it comprises a processor arranged to execute software routines, and said means are implemented as software routines.

19. An apparatus as claimed in claim 16, 17 or 18, **characterized** in that it is a network element (SCN) of a mobile communication system, which is adapted to function as a service platform.

20. An apparatus as claimed in claim 16, 17 or 18, **characterized** in that it is the mobile equipment (ME) in a mobile communication system.

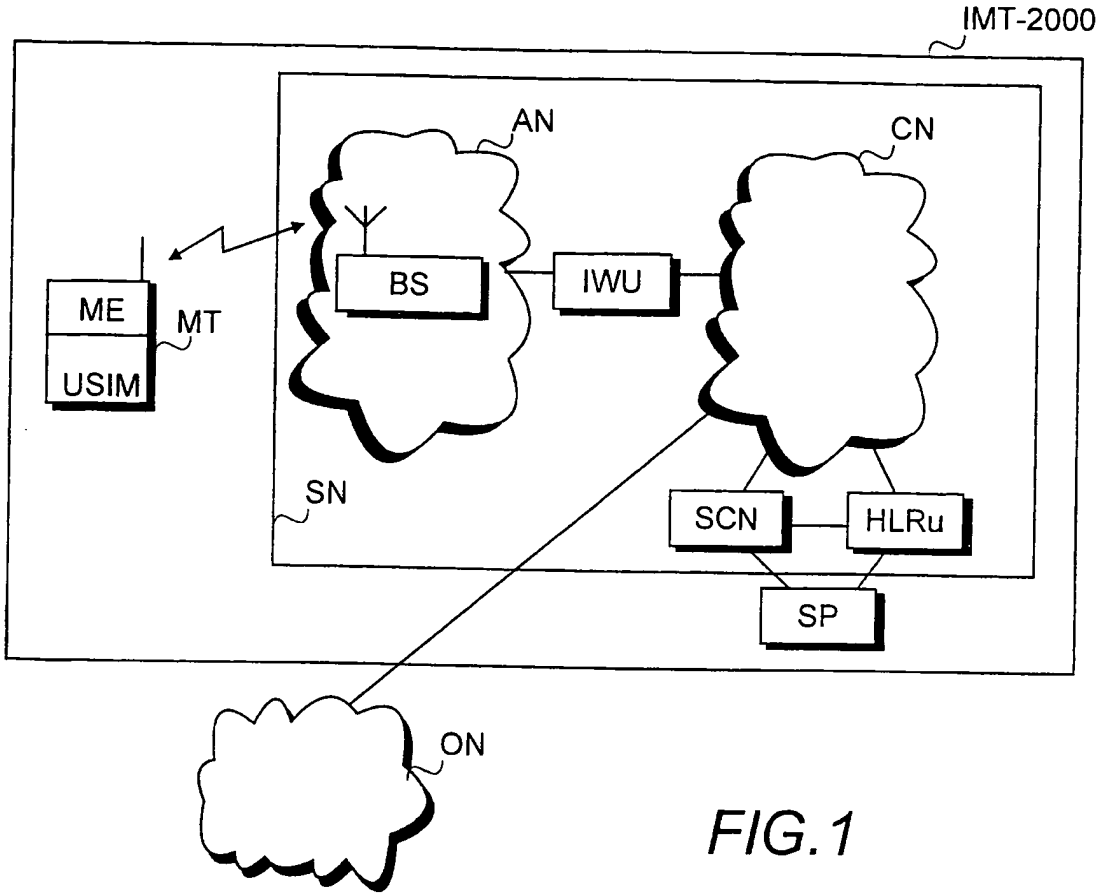


FIG.1

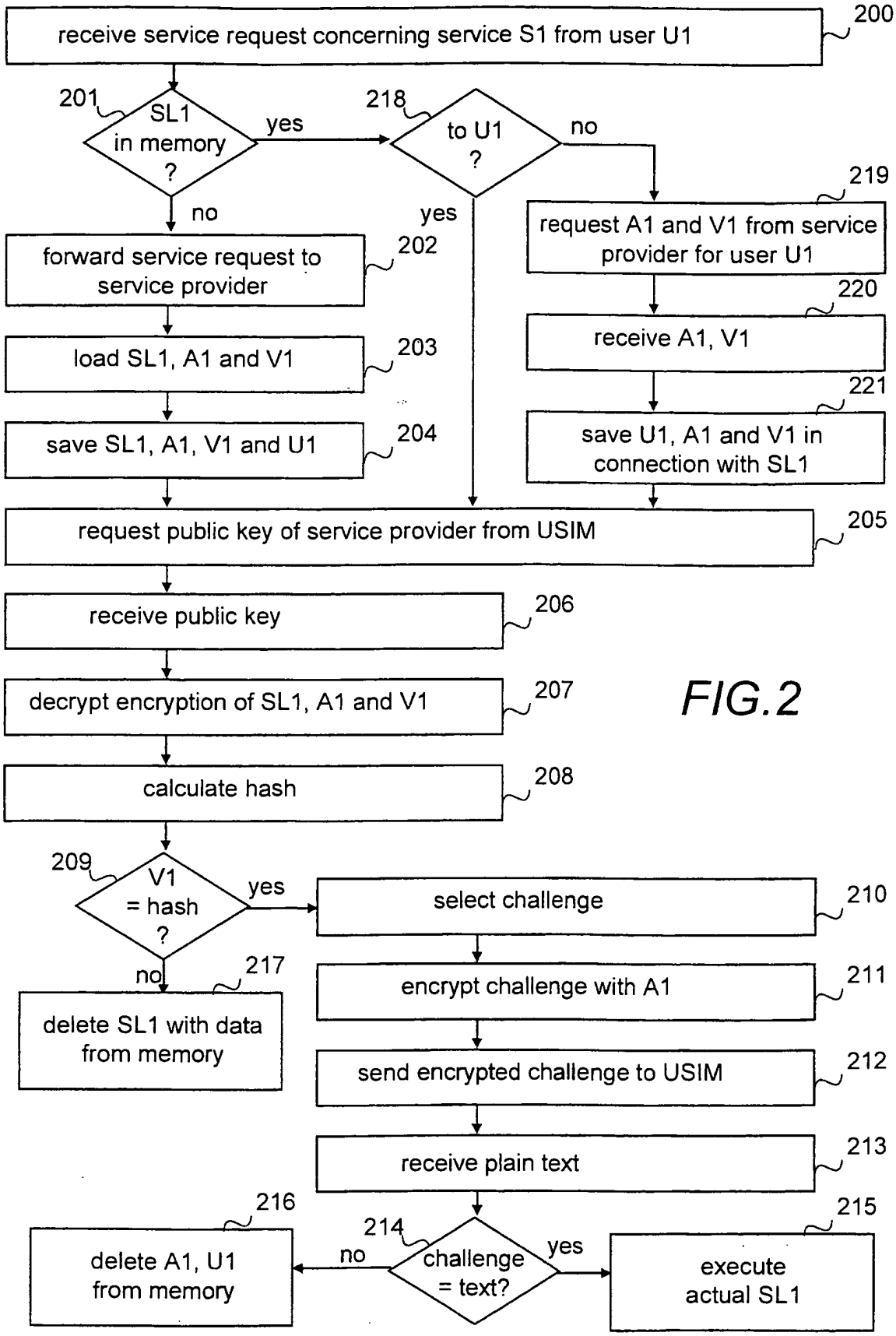

receive service request concerning service S1 from user U1
200
201
SL1 in memory ?
yes
218
to U1 ?
no
219
request A1 and V1 from service provider for user U1
no
yes
forward service request to service provider
202
220
receive A1, V1
load SL1, A1 and V1
203
221
save U1, A1 and V1 in connection with SL1
save SL1, A1, V1 and U1
204
request public key of service provider from USIM
205
receive public key
206
decrypt encryption of SL1, A1 and V1
207
calculate hash
208
209
V1 = hash ?
yes
select challenge
210
no
217
encrypt challenge with A1
211
delete SL1 with data from memory
send encrypted challenge to USIM
212
receive plain text
213
216
214
215
delete A1, U1 from memory
no
challenge = text?
yes
execute actual SL1

FIG.2

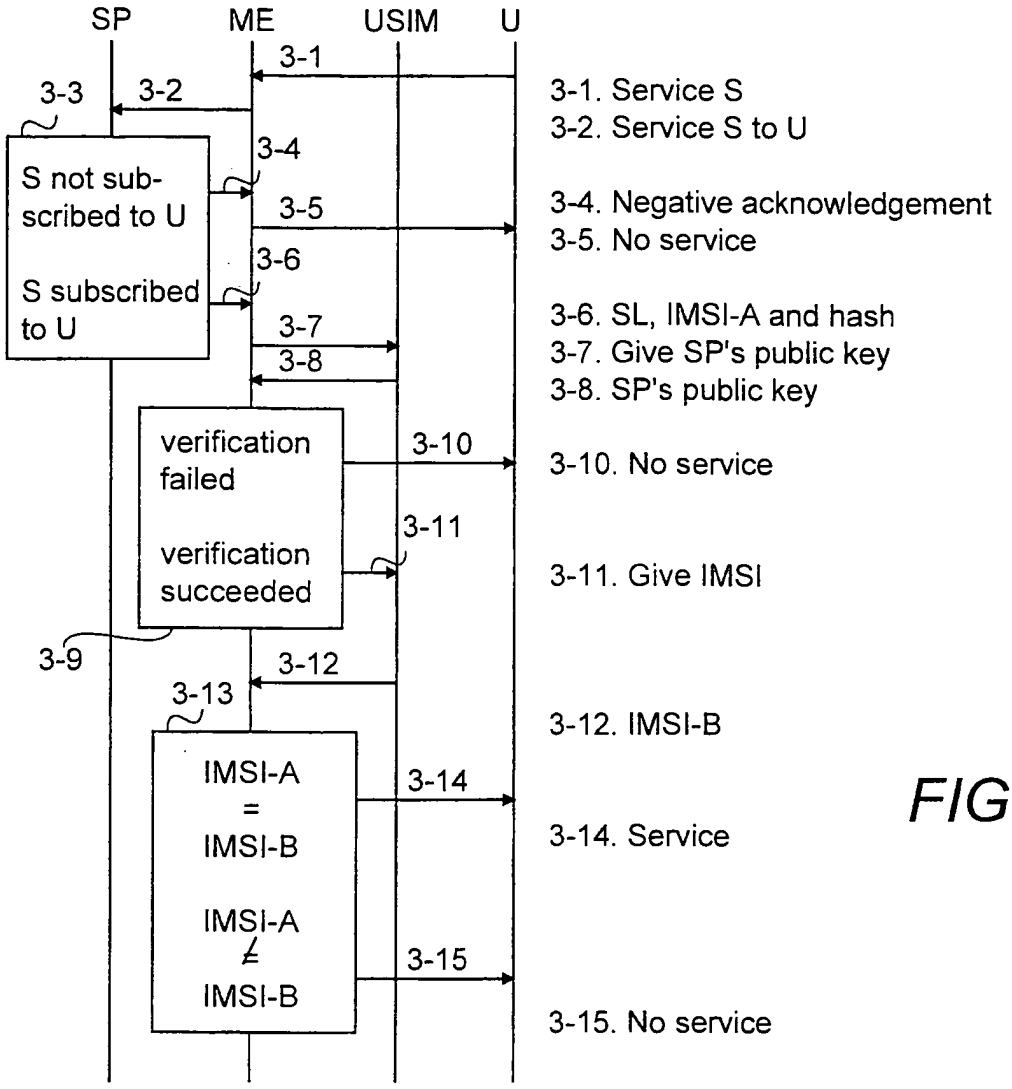

SP
ME
USIM
U
3-1
3-2
3-3
3-4
S not sub-scribed to U
3-5
3-6
S subscribed to U
3-7
3-8
verification failed
3-10
verification succeeded
3-11
3-9
3-12
3-13
IMSI-A = IMSI-B
3-14
IMSI-A ≠ IMSI-B
3-15
3-1. Service S
3-2. Service S to U
3-4. Negative acknowledgement
3-5. No service
3-6. SL, IMSI-A and hash
3-7. Give SP's public key
3-8. SP's public key
3-10. No service
3-11. Give IMSI
3-12. IMSI-B
3-14. Service
3-15. No service

FIG.3

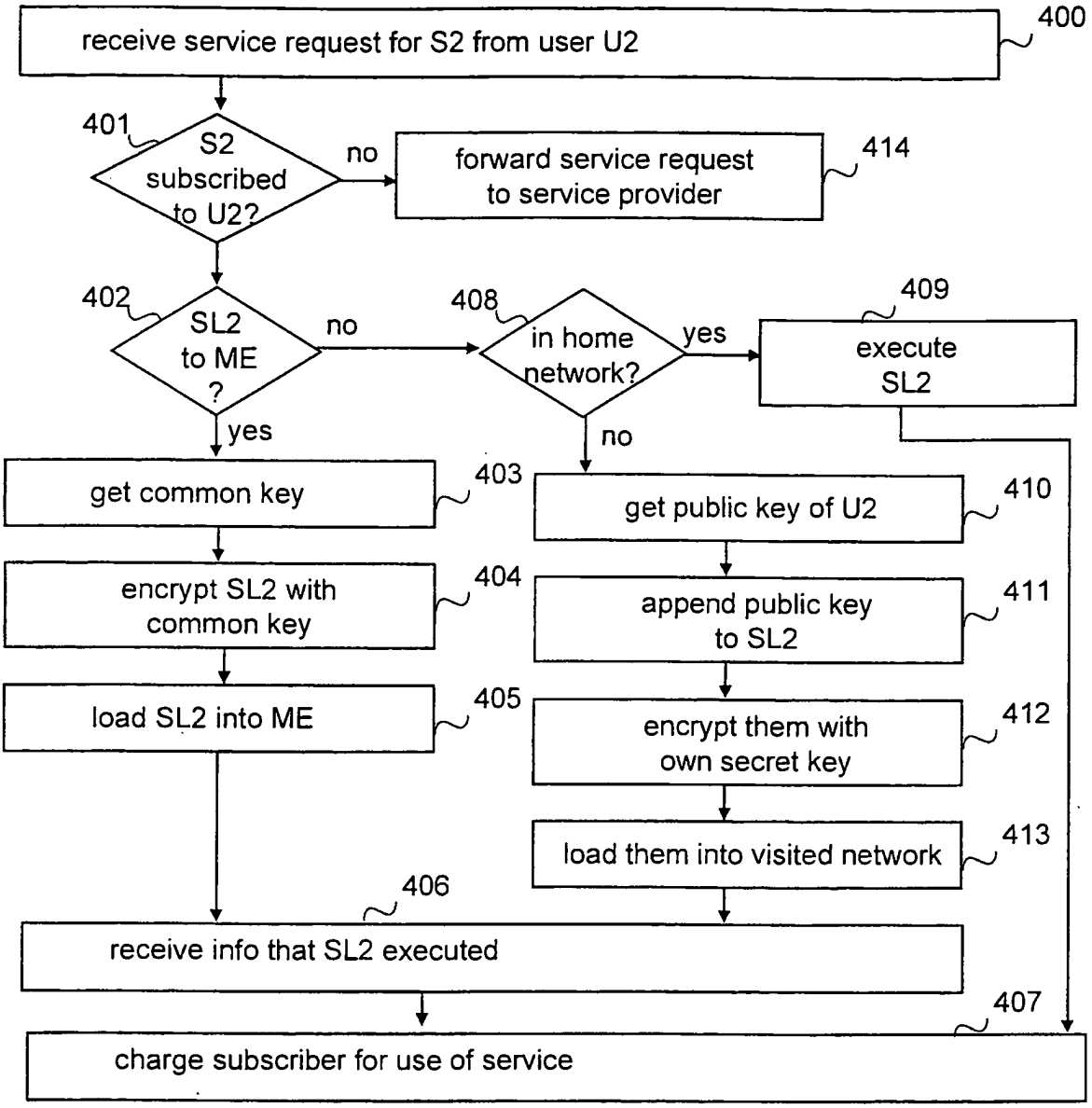

receive service request for S2 from user U2
400
401
S2 subscribed to U2?
no
forward service request to service provider
414
402
SL2 to ME ?
no
408
in home network?
yes
409
execute SL2
yes
no
get common key
403
get public key of U2
410
encrypt SL2 with common key
404
append public key to SL2
411
load SL2 into ME
405
encrypt them with own secret key
412
load them into visited network
413
406
receive info that SL2 executed
407
charge subscriber for use of service

FIG.4

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : H04N 7/30 — A2

(11) International Publication Number: **WO 00/05898**

(43) International Publication Date: 3 February 2000 (03.02.00)

(21) International Application Number: PCT/US99/16638

(22) International Filing Date: 21 July 1999 (21.07.99)

(30) Priority Data:

60/093,860	23 July 1998 (23.07.98)	US
09/169,829	11 October 1998 (11.10.98)	US

(63) Related by Continuation (CON) or Continuation-in-Part (CIP) to Earlier Applications

US	60/093,860 (CIP)
Filed on	23 July 1998 (23.07.98)
US	09/169,829 (CIP)
Filed on	11 October 1998 (11.10.98)

(71) Applicant *(for all designated States except US)*: OPTIVISION, INC. [US/US]; 3450 Hillview Avenue, Palo Alto, CA 94304 (US).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: LI, Weiping [US/US]; 159 California Avenue, J103, Palo Alto, CA 94306 (US).

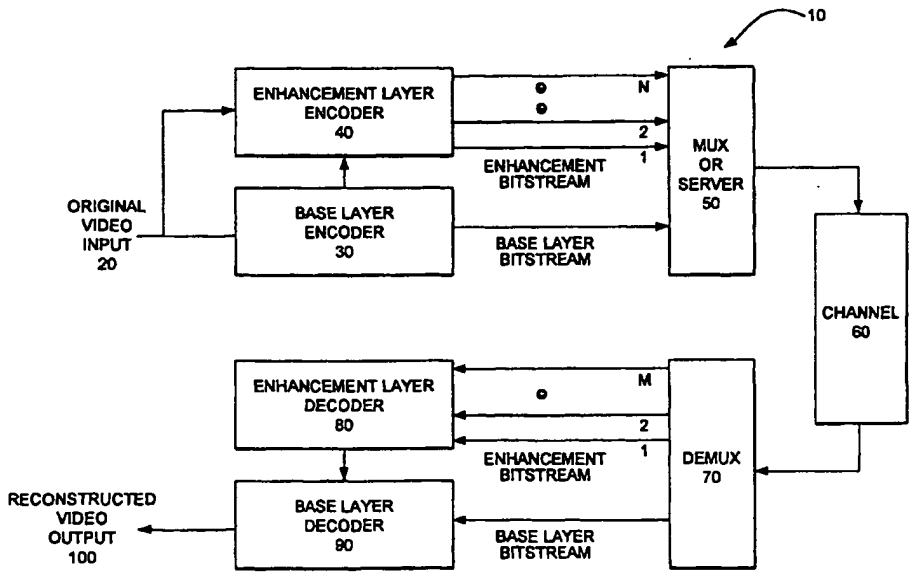
(74) Agent: DAVIS, Paul; Wilson Sonsini Goodrich & Rosati, 650 Page Mill Road, Palo Alto, CA 94304–1050 (US).

(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published

Without international search report and to be republished upon receipt of that report.

(54) Title: SCALABLE VIDEO CODING AND DECODING



(57) Abstract

A video encoding method and apparatus for adapting a video input to a bandwidth of a transmission channel of a network that includes determining the number N enhancement layer bitstreams capable of being adapted to the bandwidth of the transmission channel of a network. A base layer bitstream is encoded from the video input wherein a plurality of enhancement layer bitstreams are encoded from the video input. The enhancement layer bitstreams are based on the base layer bitstream, wherein the plurality of enhancement layer bitstreams complements the base layer bitstream and the base layer bitstream and N enhancement layer bitstreams are transmitted to the network.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

SCALABLE VIDEO CODING AND DECODING

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates to a method and apparatus for the scaling of data signals the bandwidth of the transmission channel; and more particularly to a scalable video method and apparatus for coding video such that the received video is adapted to the bandwidth of the transmission channel.

Description of Related Art

Signal compression in the video arena has long been employed to increase the bandwidth of either the generating, transmitting, or receiving device. MPEG - an acronym for Moving Picture Experts Group - refers to the family of digital video compression standards and file formats developed by the group. For instance, the MPEG-1 video sequence is an ordered stream of bits, with special bit patterns marking the beginning and ending of a logical section.

MPEG achieves high compression rate by storing only the changes from one frame to another, instead of each entire frame. The video information is then encoded using a technique called DCT (Discrete Cosine Transform) which is a technique for representing a waveform data as a weighted sum of cosines. MPEG use a type of lossy compression wherein some data is removed. But the diminishment of data is generally imperceptible to the human eye. It should be noted that the DCT itself does not lose data; rather, data compression technologies that rely on DCT approximate some of the coefficients to reduce the amount of data.

The basic idea behind MPEG video compression is to remove spatial redundancy within a video frame and temporal redundancy between video frames. The DCT-based (Discrete Cosine Transform) compression is used to reduce spatial redundancy and motion compensation is used to exploit temporal redundancy. The images in a video stream usually do not change much within small time intervals.

Thus, the idea of motion-compensation is to encode a video frame based on other video frames temporally close to it.

A video stream is a sequence of video frames, each frame being a still image. A video player displays one frame after another, usually at a rate close to 30 frames per second. Macroblocks are formed, each macroblock consists of four 8 x 8 luminance blocks and two 8 x 8 chrominance blocks. Macroblocks are the units for motion-compensated compression, wherein blocks are basic unit used for DCT compression. Frames can be encoded in three types: intra-frames (I-frames), forward predicted frames (P-frames), and bi-directional predicted frames (B-frames).

An I-frame is encoded as a single image, with no reference to any past or future frames. Each 8 x 8 block is encoded independently, except that the coefficient in the upper left corner of the block, called the DC coefficient, is encoded relative to the DC coefficient of the previous block. The block is first transformed from the spatial domain into a frequency domain using the DCT (Discrete Cosine Transform), which separates the signal into independent frequency bands. Most frequency information is in the upper left corner of the resulting 8 x 8 block. After the DCT coefficients are produced the data is quantized, i.e. divided or separated. Quantization can be thought of as ignoring lower-order bits and is the only lossy part of the whole compression process other than sub-sampling.

The resulting data is then run-length encoded in a zig-zag ordering to optimize compression. The zig-zag ordering produces longer runs of 0's by taking advantage of the fact that there should be little high-frequency information (more 0's as one zig-zags from the upper left corner towards the lower right corner of the 8 x 8 block).

A P-frame is encoded relative to the past reference frame. A reference frame is a P- or I-frame. The past reference frame is the closest preceding reference frame. A P-macroblock is encoded as a 16 x 16 area of the past reference frame, plus an error term.

To specify the 16 x 16 area of the reference frame, a motion vector is included. A motion vector (0, 0) means that the 16 x 16 area is in the same position as the macroblock we are encoding. Other motion vectors are generated are relative to that position. Motion vectors may include half-pixel values, in which case pixels are averaged. The error term is encoded using the DCT, quantization, and run-length

encoding. A macroblock may also be skipped which is equivalent to a (0, 0) vector and an all-zero error term.

A B-frame is encoded relative to the past reference frame, the future reference frame, or both frames.

A pictorial view of the above processes and techniques in application are depicted in prior art Fig. 15, which illustrates the decoding process for a SNR scalability. Scalable video coding means coding video in such a way that the quality of a received video is adapted to the bandwidth of the transmission channel. Such a coding technique is very desirable for transmitting video over a network with a time-varying bandwidth.

SNR scalability defines a mechanism to refine the DCT coefficients encoded in another (lower) layer of a scalable hierarchy. As illustrated in prior art Fig. 15, data from two bitstreams is combined after the inverse quantization processes by adding the DCT coefficients, Until the dat is combined, the decoding processes of the two layers are independent of each other.

The lower layer (base layer) is derived from the first bitstream and can itself be either non-scalable, or require the spatial or temporal scalability decoding process, and hence the decoding of additional bitstream, to be applied. The enhancement layer, derived from the second bitstream, contains mainly coded DCT coefficients and a small overhead.

In the current MPEG-2 video coding standard, there is an SNR scalability extension that allows two levels of scalability. MPEG achieves high compression rate by storing only the changes from one frame to another, instead of each entire frame. There are at least two disadvantages of employing the MPEG-2 standard for encoding video data. One disadvantage is that the scalability granularity is not fine enough, because the MPEG-2 process is an all or none method. Either the receiving device can receive all of the data from the base layer and the enhancement layer or only the data from the base layer bitstream. Therefore, the granularity is not scalable. In a network environment, more than two levels of scalability are usually needed.

Another disadvantage is that the enhancement layer coding in MPEG-2 is not efficient. Too many bits are needed in the enhancement layer in order to have a noticeable increase in video quality.

The present invention overcomes these disadvantages and others by providing, among other advantages, an efficient scalable video coding method with increased granularity.

SUMMARY OF THE INVENTION

The present invention can be characterized as a scalable video coding means and a system for encoding video data, such that quality of the final image is gradually improved as more bits are received. The improved quality and scalability are achieved by a method wherein an enhancement layer is subdivided into layers or levels of bitstream layers. Each bitstream layer is capable of carrying information complementary to the base layer information, in that as each of the enhancement layer bitstreams are added to the corresponding base layer bitstreams the quality of the resulting images are improved.

The number N of enhancement layers is determined or limited by the network that provides the transmission channel to the destination point. While the base layer bitstream is always transmitted to the destination point, the same is not necessarily true for the enhancement layers. Each layer is given a priority coding and transmission is effectuated according to the priority coding. In the event that all of the enhancement layers cannot be transmitted the lower priority coded layers will be omitted. The omission of one or more enhancement layers may be due to a multitude of reasons.

For instance, the server which provides the transmission channel to the destination point may be experiencing large demand on its resources from other users, in order to try and accommodate all of its users the server will prioritize the data and only transmit the higher priority coded packets of information. The transmission channel may be the limiting factor because of the bandwidth of the channel, i.e. Internet access port, Ethernet protocol, LAN, WAN, twisted pair cable, co-axial cable, etc. or the destination device itself, i.e. modem, absence of an enhanced video card, etc. may not be able to receive the additional bandwidth made available to it. In these instances only M number (M is an integer number = 0, 1, 2, . . .) of enhancement layers may be received, wherein N number (N is an integer number = 0, 1, 2, . . .) of enhancement layers were generated at the encoding stage, $M \leq N$.

To achieve these and other advantages and in accordance with the purpose of the present invention, as embodied and broadly described, the scalable video method and apparatus according to one aspect of the invention includes a video encoding method for adapting a video input to a bandwidth of a transmission channel of a network, the method includes determining the number N of enhancement layer bitstreams capable of being adapted to the bandwidth of the transmission channel of the network. Encoding a base layer bitstream from the video input is then performed and encoding N number of enhancement layer bitstreams from the video input based on the base layer bitstream, wherein the plurality of enhancement layer bitstreams complements the base layer bitstream. The base layer bitstream and the N enhancement layer bitstreams are then provided to the network.

According to another aspect of the present invention, a video decoding method for adapting a video input to a bandwidth of a transmission channel of a network includes, determining number M of enhancement layer bitstreams of said video input capable of being received from said transmission channel of said network. Decoding a base layer bitstream from received video input and decoding M number of enhancement layer bitstreams from the received video input based on the base layer bitstream, wherein the M received enhancement layer bitstreams complements the base layer bitstream. Then reconstructing the base layer bitstream and N enhancement layer bitstreams.

According to still another aspect of the present invention, a video decoding method for adapting a video input to a bandwidth of a receiving apparatus, the method includes demultiplexing a base layer bitstream and at least one of a plurality of enhancement layer bitstreams received from a network, decoding the base layer bitstream, decoding at least one of the plurality of enhancement layer bitstreams based on generated base layer bitstream, wherein the at least one of the plurality of enhancement layer bitstreams enhances the base layer bitstream. Then reconstructing a video output.

According to a further aspect of the present invention, a video encoding method for encoding enhancement layers based on a base layer bitstream encoded from a video input, the video encoding method includes, taking a difference between an

original DCT coefficient and a reference point and dividing the difference between the original DCT coefficient and the reference point into N bit-planes.

According to a still further aspect of the present invention, a method of coding motion vectors of a plurality of macroblocks, includes determining an average motion vector from N motion vectors for N macroblocks, utilizing the determined average motion vector as the motion vector for the N macroblocks, and encoding 1/N motion vectors in a base layer bitstream.

Additional features and advantages of the invention will be set forth in the description which follows, and in part will be apparent from the description, or may be learned by practice of the invention. The aspects and other advantages of the invention will be realized and attained by the structure particularly pointed out in the written description and claims hereof as well as the appended drawings.

It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory and are intended to provide further explanation of the invention as claimed.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are included to provide a further understanding of the invention and are incorporated in and constitute a part of this specification, illustrate embodiments of the invention and together with the description serve to explain the principles of the invention. In the drawings:

Fig. 1 illustrates a flow diagram of the scalable video encoding method of the present invention;

Fig. 2A illustrates conventional probability distribution of DCT coefficient values;

Fig. 2B illustrates conventional probability distribution of DCT coefficient residues;

Fig. 3A illustrates the probability distribution of DCT coefficient values of the present invention;

Fig. 3B illustrates the probability distribution of DCT coefficient residues of the present invention;

Figs. 3C and 3D illustrates a method for taking a difference of a DCT coefficient of the present invention;

Fig. 5 illustrates a flow diagram for finding the maximum number of bit-planes in the DCT differences of a frame of the present invention;

Fig. 6 illustrates a flow diagram for generating (RUN, EOP) Symbols of the present invention;

Fig. 7 Illustrates a flow diagram for encoding enhancement layers of the present invention;

Fig. 8 illustrates a flow diagram for encoding (RUN, EOP) symbols and sign_enh values of one DCT block of one bit-plane;

Fig. 9 illustrates a flow diagram for encoding a sign_enh value of the present invention;

Fig. 10 illustrates a flow diagram for adding enhancement difference to a DCT coefficient of the present invention;

Fig. 11 illustrates a flow diagram for converting enhancement difference to a DCT coefficient of the present invention;

Fig. 12 illustrates a flow diagram for decoding enhancement layers of the present invention;

Fig. 13 illustrates a flow diagram for decoding (RUN, EOP) symbols and sign_enh values of one DCT block of one bit-plane;

Fig. 14 illustrates a flow diagram for decoding a sign_enh value; and

Fig. 15 illustrates a prior a conventional SNR scalability flow diagram.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Reference will now be made in detail to the preferred embodiments of the present invention, examples of which are illustrated in the accompanying drawings.

Fig. 1 illustrates the scalable video diagram 10 of an embodiment of the present invention. The original video input 20 is encoded by the base layer encoder 30 in accordance with the method of represent by flow diagram 400 of Fig. 4. A DCT coefficient OC and its corresponding base layer quantized DCT coefficient QC are generated and a difference determined pursuant to steps 420 and 430 of Fig. 4. The

difference information from the base layer encoder 30 is passed to the enhancement layer encoder 40 that encodes the enhancement information.

The encoding of the enhancement layer encoder is performed pursuant to methods 500 - 900 as depicted in Figs. 5 - 10, respectively and will be briefly described. The bitstream from the base layer encoder 30 and the N bitstreams from the enhancement layer encoder 40 are capable of being sent to the transmission channel 60 by at least two methods.

In the first method all bitstreams are multiplexed together by multiplexor 50 with different priority identifiers, e.g., the base layer bitstream is guaranteed, enhancement bitstream layer 1 provided by enhancement layer encoder 40 is given a higher priority than enhancement bitstream layer 2. The prioritization is continued until all N (wherein N is an integer from 0, 1, 2, . . .) of the bitstreams layers are prioritized. Logic in the encoding layers 30 or 40 in negotiation with the network and intermediated devices determine the number N of bitstream layers to be generated.

The number of bitstream layers generated is a function of the total possible bandwidth of the transmission channel 60, i.e. Ethernet, LAN, or WAN connections (this list is not intended to exhaustive but only representation of potential limiting devices and/or equipment), and the network and other intermediate devices. The number of bitstream layers M (wherein M is an integer and $M \leq N$) reaching the destination point 100 can be further limited by not just the physical constraints of the intermediate devices but the congestion on the network, thereby necessitating the dropping of bitstream layers according to their priority.

In a second method the server 50 knows the transmission channel 60 condition, i.e. congestion and other physical constraints, and selectively sends the bitstreams to the channel according to the priority identifiers. In either case, the destination point 100 receives the bitstream for the base layer and M bitstreams for the enhancement layer, where $M \leq N$.

The bitstreams M are sent to the base layer 90 and enhancement layer 80 decoders after being demultiplexed by demultiplexor 70. The decoded enhancement information from the enhancement layer decoder is passed to the base layer decoder to composite the reconstructed video output 100. The decoding of the multiplexed

bitstreams are accomplished pursuant to the methods and algorithms depicted in flow diagrams 1100 - 1400 of Figs. 11 - 14, respectively.

The base layer encoder and decoder are capable of performing logic pursuant to the MPEG-1, MPEG-2, or MPEG-4 (Version-1) standards that are hereby incorporated by reference into this disclosure.

Taking Residue with Probability Distribution Preserved

A detailed description of the probability distribution residue will now be made with reference to Figs 2A - 3B

In the current MPEG-2 signal-to-noise ratio (SNR) scalability extension, a residue or difference is taken between the original DCT coefficient and the quantized DCT coefficient. Fig. 2A illustrates the distribution of a residual signal as a DCT coefficient. In taking the residue small values have higher probabilities and large values have smaller probabilities. The intervals along the horizontal axis represent quantization bins. The dot in the center of each interval represents the quantized DCT coefficient. Taking the residue between the original and the quantized DCT coefficient is equivalent to moving the origin to the quantization point.

Therefore, the probability distribution of the residue becomes that as shown in Figure 2B. The residue from the positive side of Fig. 2A has a higher probability of being negative than positive and the residue taken from the negative side of the Fig. 2A has a higher probability of being positive than negative. The result is that the probability distribution of the residue becomes almost uniform. Thus making coding the residue more difficult.

A vastly superior method is to generate a difference between the original and the lower boundary points of the quantized interval as shown in Fig. 3A and Fig. 3B. In this method, the residue is taken from the positive side of Fig. 2A remains positive and the residue from the negative side of Fig. 2A remains negative. Taking the residue is equivalent to moving the origin to the reference point as illustrated in Fig. 3A. Thus, the probability of the residue becomes as shown in Fig. 3B. This method preserves the shape of the original non-uniform distribution. Although the dynamic range of the residue taken in such a manner seems to be twice of that depicted in Fig. 2B, their is no longer a need to code the sign, i.e. - or +, of the residue. The sign of the residue is

encoded in the base layer bitstream corresponding the enhancement layer, therefore this redundancy is eliminated and bits representing the sign are thus saved. Therefore, there is only a need to code the magnitude that still has a nonuniform distribution.

Bit plane coding of residual DCT coefficients

After taking residues of all the DCT coefficients in an 8 x 8 block, bit plane coding is used to code the residue. In bit-plane coding method the bit-plane coding method considers each residual DCT coefficient as a binary number of several bits instead of as a decimal integer of a certain value as in the run-level coding method. The bit-plane coding method in the present invention only replaces runlevel coding part. Therefore, all the other syntax elements remain the same.

An example of and description of the bit-plane coding method will now be made, wherein 64 residual DCT coefficients for an Inter-block and 63 residual DCT coefficients for an Intra-block (excluding the Intra-DC component that is coded using a separate method) are utilized for the example. The 64 (or 63) residual DCT coefficients are ordered into a one-dimensional array and at least one of the residual coefficients is non-zero. The bit-plane coding method then performs the following steps.

The maximum value of all the residual DCT coefficients in a frame is determined and the minimum number of bits, N, needed to represent the maximum value in the binary format is also determined. N is the number of bitplanes layers for this frame and is coded in the frame header.

Within each 8 x 8 block is represent every one of the 64 (or 63) residual DCT coefficients with N bits in the binary format and there is formed N bit-planes or layers or levels. A bit-plane is defined as an array of 64 (or 63) bits, taken one from each residual DCT coefficient at the same significant position.

The most significant bit-plane is determined with at least one non-zero bit and then the number of all-zero bit-planes between the most significant bit-plane determined and the Nth one is coded. Then starting from the most significant bit plane (MSB plane), 2-D symbols are formed of two components: (a) number of consecutive O's before a I (RUN), (b) whether there are any l's left on this bit plane, i.e. End-Of-Plane (EOP). If a bit-plane after the MSB plane contains all O's, a special symbol

ALL-ZERO is formed to represent an all-zero bit-plane. Note that the MSB plane does not have the all-zero case because any all-zero bit-planes before the MSB plane have been coded in the previous steps.

Four 2-D VLC tables are used, wherein the table VT-C-Table-0 corresponds to the MSB plane; table VLC-Table- I corresponds to the second MSB plane; table VLC-Table-2 corresponds to the third MSB plane; and table VLC-Table-3 corresponds to the fourth MSB and all the lower bit planes. For the ESCAPE cases, RUN is coded with 6 bits, EOP is coded with 1 bit. Escape coding is a method to code very small probability events which are not in the coding tables individually.

An example of the above process will now follow. For illustration purposes, we will assume that the residual values after the zigzag ordering are given as follows and N = 6: The following representation is thereby produced.

10, 0, 6, 0, 0, 3, 0, 2, 2, 0, 0, 2, 0, 0, 1, 0, ... 0, 0

The maximum value in this block is found to be 10 and the minimum number of bits to represent 10 in the binary format (1010) is 4. Therefore, two all-zero bit-planes before the MSB plane are coded with a code for the value 2 and the remaining 4 bit-planes are coded using the (RUN, EOP) codes. Writing every value in the binary format using 4 bits, the 4 bit-planes are formed as follows:

1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,0, 0 (MSB-plane)
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,0, 0 (Second MSB-plane)
1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0,0, 0 (Third MSB-plane)
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,0, 0 (Fourth MSB-plane or LSB-plane)

Converting the bits of each bit-plane into (RUN, EOP) symbols results in the following:

(0, 1)	(MSB-plane)
(2, 1)	(Second MSB-plane)
(0, 0), (1,O), (2,O), (1,O), (0, 0), (2, 1)	(Third MSB-plane)

(5, 0), (8, 1) (Fourth MSB-plane or LSB-plane)

Therefore, there are 10 symbols to be coded using the (RUN, EOP) VLC tables. Based on their locations in the bit-planes, different VLC tables are used for the coding. The enhancement bitstream using all four bitplanes looks as follows:

code leading-all-zero(2)

code msb(O, 1)

code msb-1(2,1)

code-msb-2(0,0), code_msb-2(1,0), code-msb-2(2,O), code-msb-2(1,0), code-msb-2(0,0), code-msb-2(2, 1) code_msb-3(5,0), code_msb-3(8, 1).

In an alternative embodiment, several enhancement bitstreams may be formed from the four bit-planes, in this example from the respective sets comprising one or more of the four bit-planes.

Motion Vector Sharing

In this alternative embodiment of the present invention motion vector sharing is capable of being utilized when the base layer bitstream exceeds a predetermined size or more levels of scalability are needed for the enhancement layer. By lowering the number of bits required for coding the motion vectors in the base layer the bandwidth requirements of the base layer bitstream is reduced. In base layer coding, a macroblock (16 x 16 pixels for the luminance component and W pixels for each chron-luminance components) of the current frame is compared with the previous frame within a search range. The closest match in the previous frame is used as a prediction of the current macroblock. The relative displacement of the prediction to the current macroblock, in the horizontal and vertical directions, is called a motion vector.

The difference between the current macroblock and it's prediction is coded using the DCT coding. In order for the decoder to reconstruct the current macroblock, the motion vector has to be coded in the bitstream. Since there is a fixed number of bits for coding a frame, the more bits spent on coding the motion vectors results in fewer bits for coding the motion compensated differences. Therefore, it is desirable to lower the number of bits for coding the motion vectors and leave more bits for coding the differences between the current macroblock and its prediction.

For each set of 2 x 2 motion vectors, the average motion vector can be determined and used for the four macroblocks. In order to not change the syntax of the base layer coding, four macroblocks are forced to have the identical motion vectors. Since only one out four motion vectors is coded in the bitstream, the amount of bits spent on motion vector coding is reduced, therefore, there are more bits available for coding the differences. The cost for pursuing such a method is that the four macroblocks, which share the same motion vector may, not get the best matched prediction individually and the motion compensated difference may have a larger dynamic range, thus necessitating more bits to code the motion vector.

For a given fixed bitrate, the savings from coding one out of four motion vectors may not compensate the increased number of bits required to code the difference with a larger dynamic range. However, for a time varying bitrate, a wider dynamic range for the enhancement layer provides more flexibility to achieve the best possible usage of the available bandwidth.

Coding Sign Bits

In an alternative embodiment of the present invention, if the base layer quantized DCT coefficient is non-zero, the corresponding enhancement layer difference will have the same sign as the base layer quantized DCT. Therefore, there is no need to code the sign bit in the enhancement layer.

Conversely, if the base layer quantized DCT coefficient is zero and corresponding enhancement layer difference is non-zero, a sign bit is placed into enhancement layer bitstream immediately after the MSB of the difference.
An example of the above method will now follow.

Difference of a DCT block after ordering

- 10, 0, 6, 0, 0, 3, 0, 2, 2, 0, 0, 2, 0, 0, 1, 0, ...0, 0

Sign indications of the DCT block after ordering

- 3, 3, 3, 3, 2, 0, 3, 3, 1, 2, 2, 0, 3, 3, 1, 2, ... 2, 3

- 0: base layer quantized DCT coefficient = 0 and difference >0
- 1: base layer quantized DCT coefficient = 0 and difference <0
- 2: base layer quantized DCT coefficient = 0 and difference =0

- 3: base layer quantized DCT coefficient = 0.

In this example, the sign bits associated with values 10, 6, 2 don't need to be coded and the sign bits associated with 3, 2, 2, 1 are coded in the following way:

Code(All Zero)

code (All Zero)

code(0,1)

code(2,1)

code(0,0),code(1,0),code(2,0),0,code(1,0),code(0,0),1,code(2,1),0

code(5,0),code(8,1),1

For every DCT difference, there is a sign indication associated with it. There are four possible cases. In the above coding 0, 1, 2, and 3 are used to denote the four cases. If the sign indication is 2 or 3, the sign bit does not have to be coded because it is either associated with a zero difference or available from the corresponding base layer data. If the sign indication is 0 or 1 a sign bit code is required once per difference value, i.e. not every bit-plane of the difference value. Therefore, a sign bit is put immediately after the most significant bit of the difference.

Optimal Reconstruction of the DCT Coefficients

In an alternative embodiment of the present invention, even though N enhancement bitstream layers or planes may have been generated, only M, wherein M $\leq$ N enhancement layer bits are available for reconstruction of the DCT coefficients due to the channel capacity, and other constraints such as congestion among others, the decoder 80 of Fig. 1 may receive no enhancement difference or only a partial enhancement difference. In such a case, the optimal reconstruction of the DCT coefficients is capable of proceeding along the following method:

If decoded difference = 0, the reconstruction point is the same as that in base layer, otherwise, the reconstructed difference = decoded difference + ¼ *(1<<decoded_bit_plane) and the reconstruction point = reference point + reconstructed difference * Q_enh +Q_enh/2.

In the present embodiment, referring to Figs. 3C and 3D, the optimal reconstruction point is not the lower boundary of a quantization bin. The above method specifies how to obtain the optimal reconstruction point in cases where the

difference is quantized and received partially, i.e. not all of the enhancement layers generated are either transmitted or received as shown in Fig. 1. wherein $M \leq N$.

What is claimed is:

1. A video encoding method for adapting a video input to a bandwidth of a transmission channel of a network, the method comprising the steps of:

determining number N of enhancement layer bitstreams capable of being adapted to said bandwidth of said transmission channel of said network;

encoding a base layer bitstream from said video input;

encoding N number of enhancement layer bitstreams from said video input based on the base layer bitstream, wherein the N enhancement layer bitstreams complements the base layer bitstream; and

providing the base layer bitstream and N enhancement layer bitstreams to said network.

2. The video encoding method according to claim 1, wherein the determining step includes negotiating with intermediate devices on said network.

3. The video encoding method according to claim 2, wherein negotiating includes determining destination resources.

4. The video encoding method according to claim 1, wherein the step of encoding the base layer bitstreams is performed by a MPEG-1 encoding method.

5. The video encoding method according to claim 1, wherein the step of encoding the base layer bitstreams is performed by a MPEG-2 encoding method.

6. The video encoding method according to claim 1, wherein the step of encoding the base layer bitstreams is performed by a MPEG-4 encoding method.

7. The video encoding method according to claim 1, wherein the step of encoding the base layer bitstreams is performed by a Discrete Cosine Transform (DCT) method.

8. The video encoding method according to claim 7, wherein after encoding the base layer bitstreams by a Discrete Cosine Transform (DCT) method a DCT coefficient is quantized.

9. The video encoding method according to claim 1, wherein the enhancement layer bitstreams are based on the difference of an original base layer DCT coefficient and a corresponding base layer quantized DCT coefficient.

10. The video encoding method according to claim 1, wherein the base layer bitstream and the N enhancement layer provide to the network are multiplexed.

11. A video decoding method for adapting a video input to a bandwidth of a transmission channel of a network, the method comprising the steps of:

determining number M of enhancement layer bitstreams of said video input capable of being received from said transmission channel of said network;

decoding a base layer bitstream from received video input;

decoding M number of enhancement layer bitstreams from the received video input based on the base layer bitstream, wherein the M received enhancement layer bitstreams complements the base layer bitstream; and

reconstructing the base layer bitstream and N enhancement layer bitstreams.

12. The video decoding method according to claim 11, wherein the determining step includes negotiating with intermediate devices on said network.

13. The video decoding method according to claim 12, wherein negotiating includes determining destination resources.

14. The video decoding method according to claim 11, wherein the step of decoding the base layer bitstreams is performed by a MPEG-1 decoding method.

15. The video decoding method according to claim 11, wherein the step of decoding the base layer bitstreams is performed by a MPEG-2 decoding method.

16. The video decoding method according to claim 11, wherein the step of decoding the base layer bitstreams is performed by a MPEG-4 decoding method.

17. The video decoding method according to claim 11, wherein the step of decoding the base layer bitstreams is performed by a Discrete Cosine Transform (DCT) method.

18. The video decoding method according to claim 17, wherein after decoding the base layer bitstreams by a Discrete Cosine Transform (DCT) method a DCT coefficient is unquantized.

19. The video decoding method according to claim 11, wherein coding of the enhancement layer bitstreams are based on the difference of an original base layer DCT coefficient and a corresponding base layer quantized DCT coefficient.

20. The video decoding method according to claim 11, wherein the base layer bitstream and the M enhancement layers to be reconstructed are de-multiplexed.

21. A video decoding method for adapting a video input to a bandwidth of a receiving apparatus, the method comprising the steps of:
demultiplexing a base layer bitstream and at least one of a plurality of enhancement layer bitstreams received from a network;
decoding the base layer bitstream;
decoding at least one of the plurality of enhancement layer bitstreams based on generated base layer bitstream, wherein the at least one of the plurality of enhancement layer bitstreams enhances the base layer bitstream; and
reconstructing a video output.

22. A video encoding method for encoding enhancement layers based on a base layer bitstream encoded from a video input, the video encoding method comprising the steps of:
taking a difference between an original DCT coefficient and a reference point; and
dividing the difference between the original DCT coefficient and the reference point into N bit-planes.

23. The video encoding method according to claim 22, wherein RUN and EOP symbols represents the N bit-planes of a DCT block.

24. The video encoding method according to claim 23, wherein the RUN and EOP symbols are encoded.

25. The video encoding method according to claim 24, wherein a sign bit is encoded if the DCT difference is equal to zero or the sign of the DCT difference is the same as the corresponding base layer bitstream data.

26. A video decoding method for reconstructing DCT coefficients M enhancement layers of N enhancement layers have been received, wherein $M \leq N$, comprising:

means for taking a reconstruction difference as a decoded difference and a portion of a decoded bit-plane;

means for taking a reconstruction point as a reference point and a reconstructed difference; and

determining an optimal reconstruction point.

27. A method of coding motion vectors of a plurality of macroblocks, the method comprising the steps of:

determining an average motion vector from N motion vectors for N macroblocks;

utilizing the determined average motion vector as the motion vector for the N macroblocks; and

encoding 1/N motion vectors in a base layer bitstream.

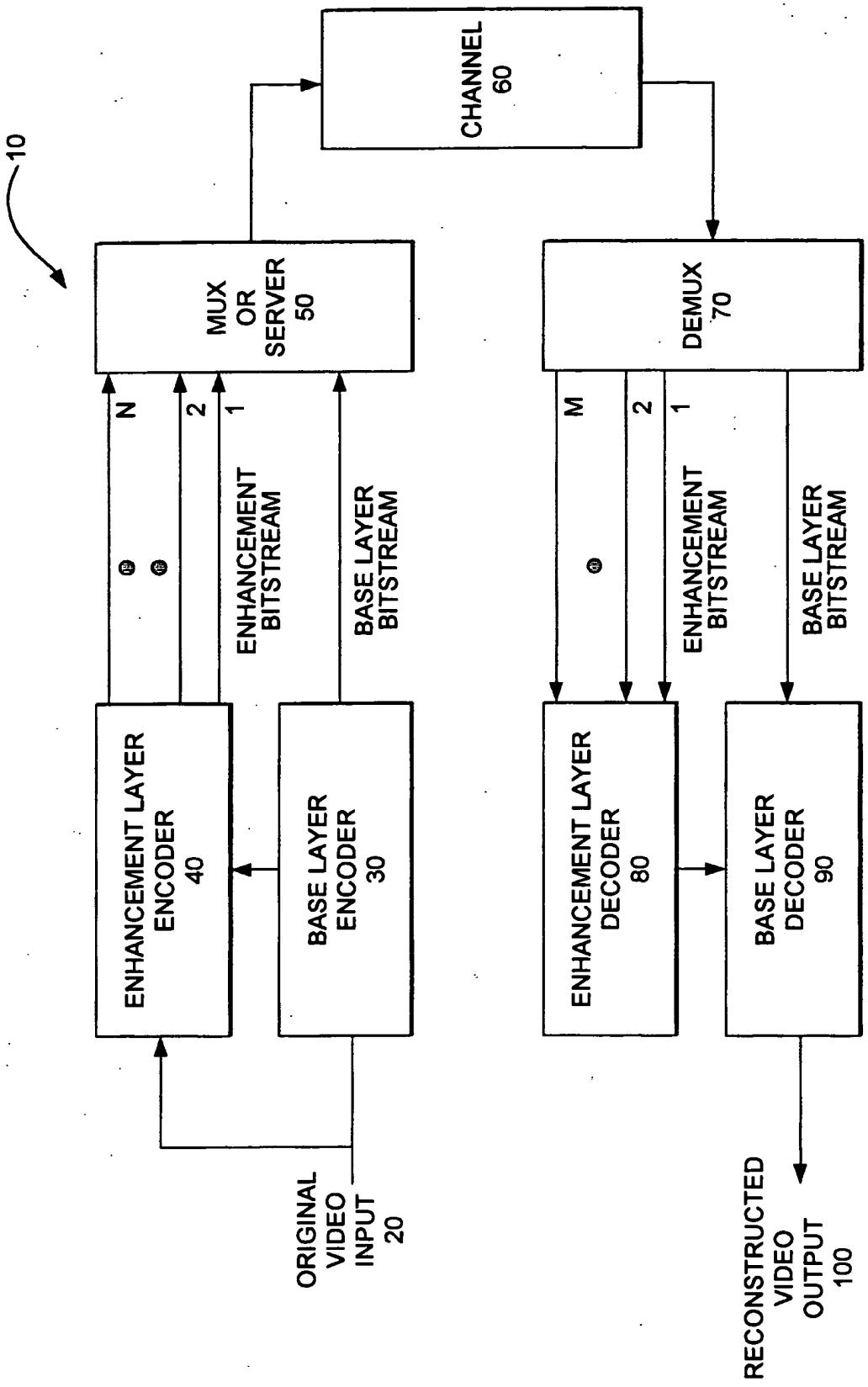


FIG. 1

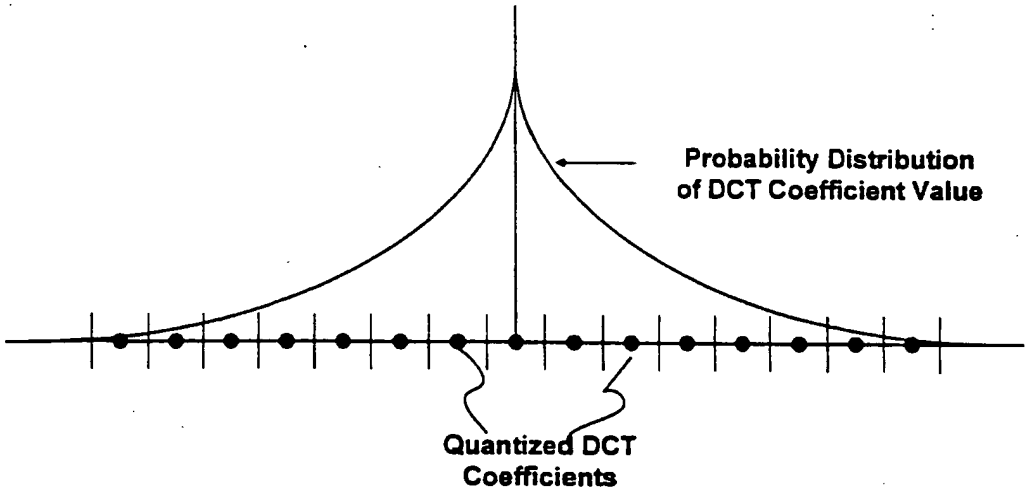


FIG. 2A

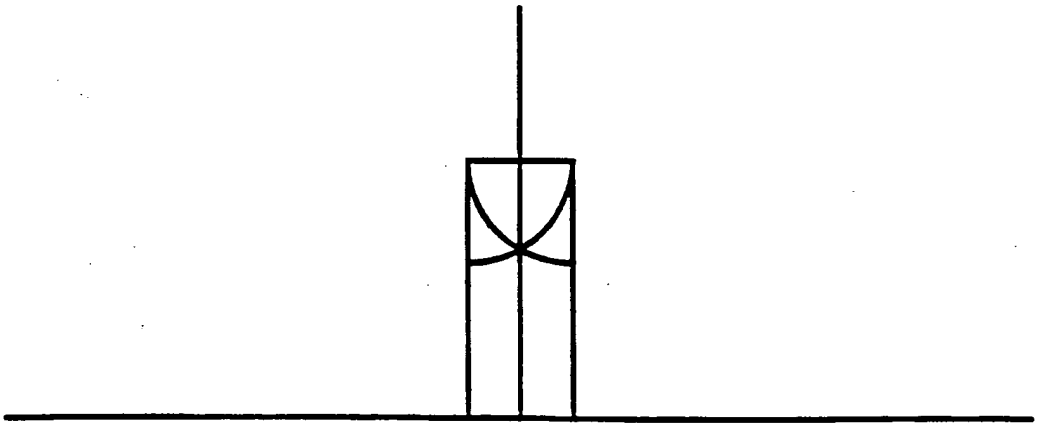


FIG. 2B

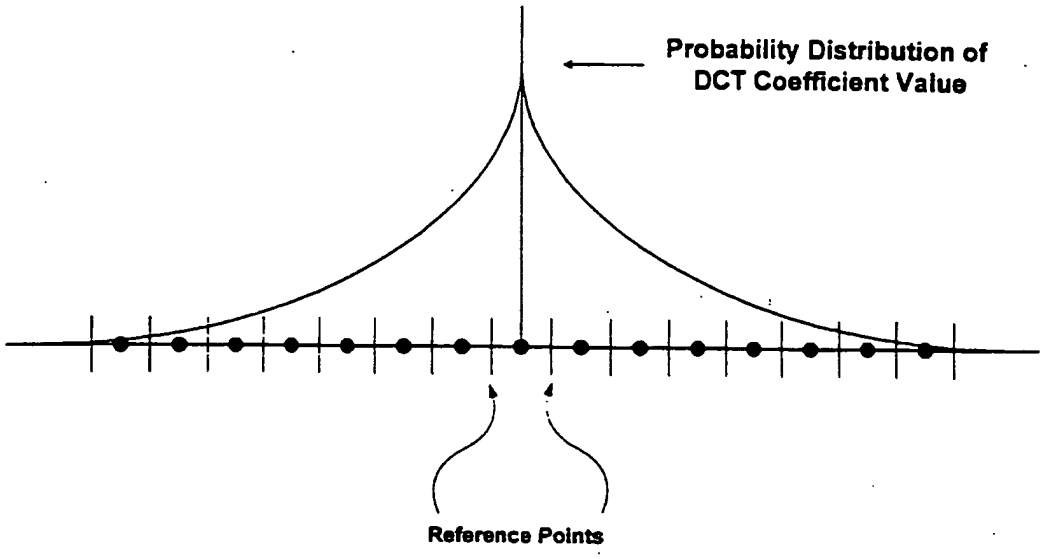


FIG. 3A

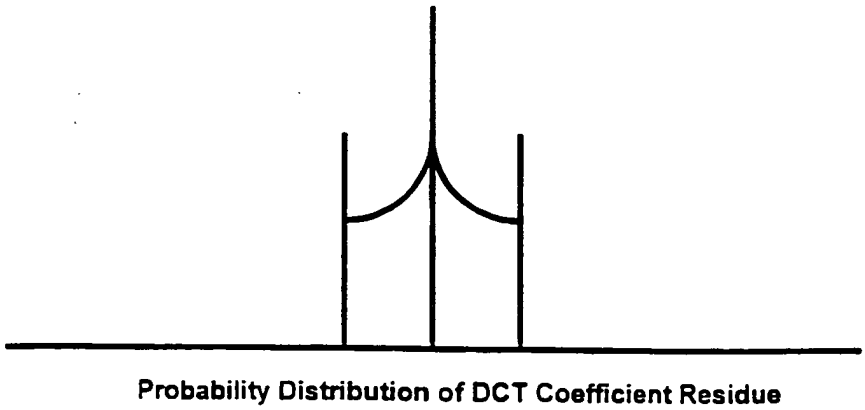


FIG. 3B

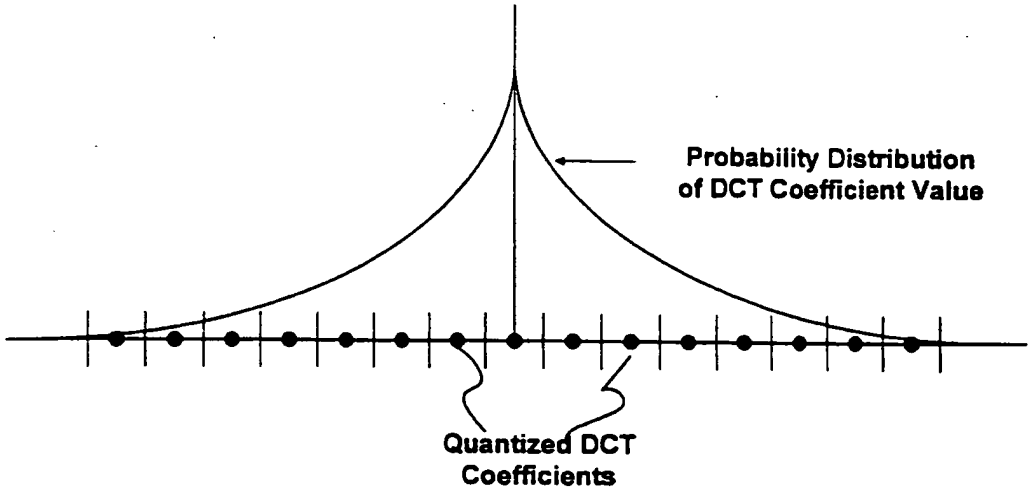


FIG. 3C

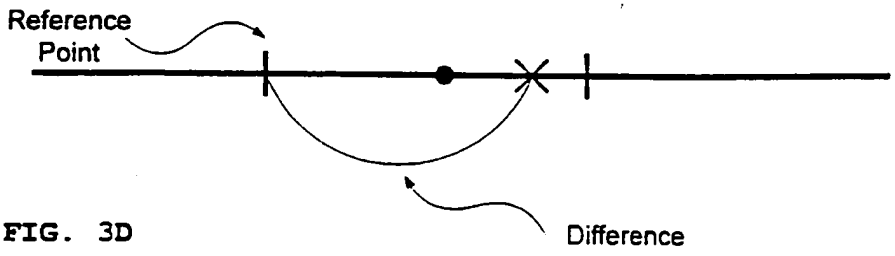


FIG. 3D

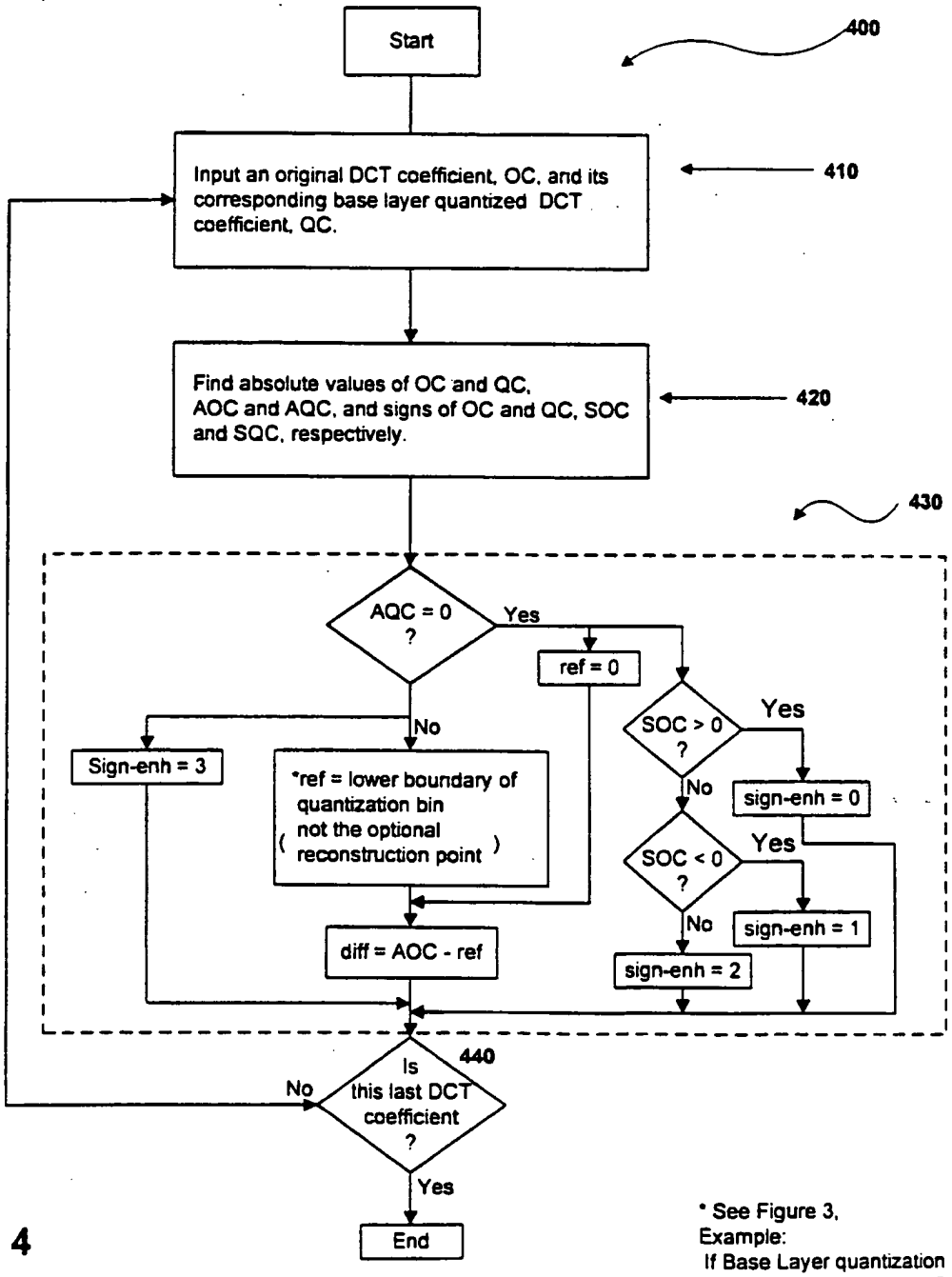


FIG. 4

* See Figure 3,
Example:
If Base Layer quantization is
AQC = AOC / (2*Q)
lower boundary is AQC * (2*Q)
optimal point is AQC*(2*Q) + Q

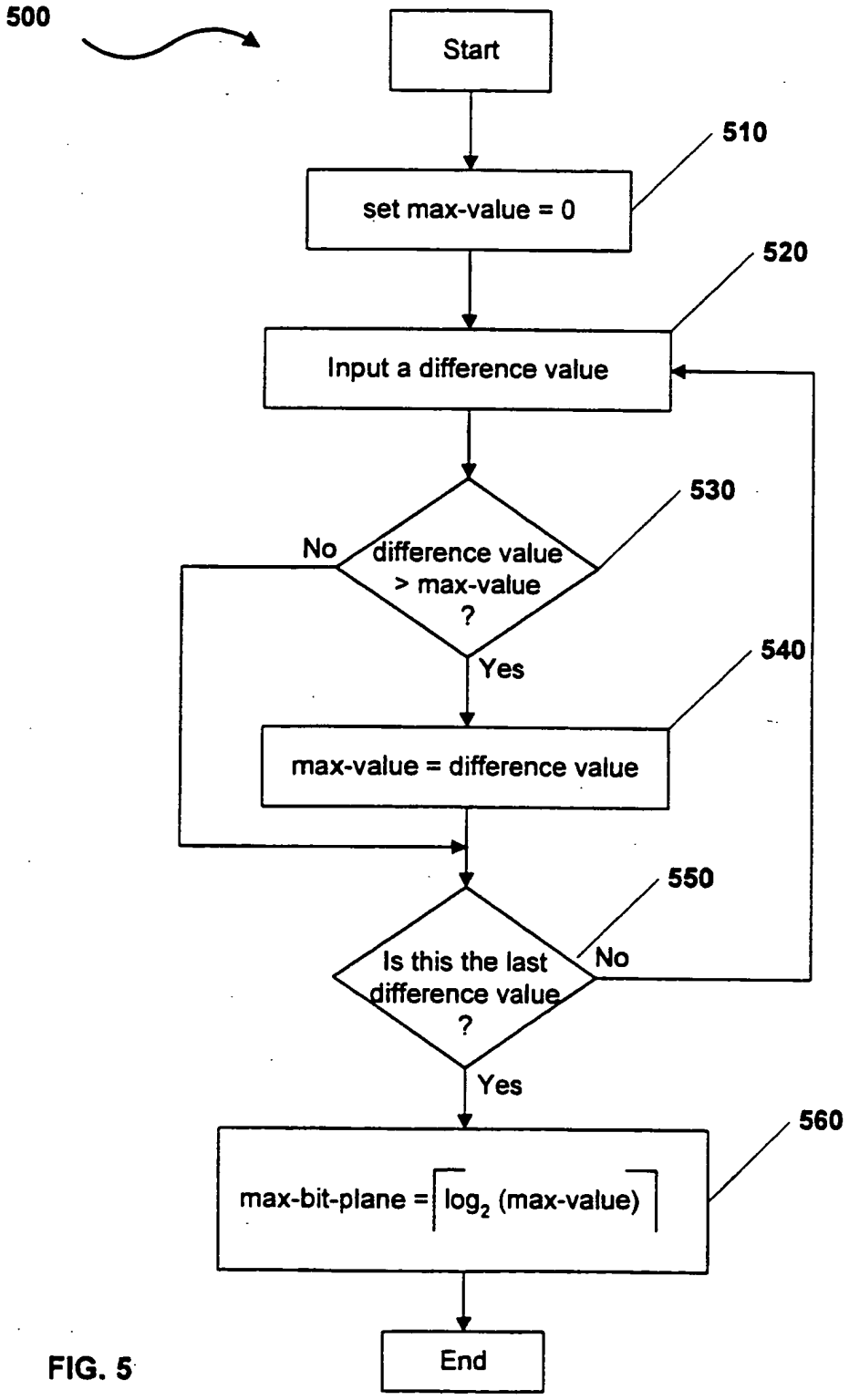

500
Start
510
set max-value = 0
520
Input a difference value
530
difference value > max-value ?
No
Yes
540
max-value = difference value
550
Is this the last difference value ?
No
Yes
560
max-bit-plane = $\lceil \log_2 (\text{max-value}) \rceil$
End

FIG. 5

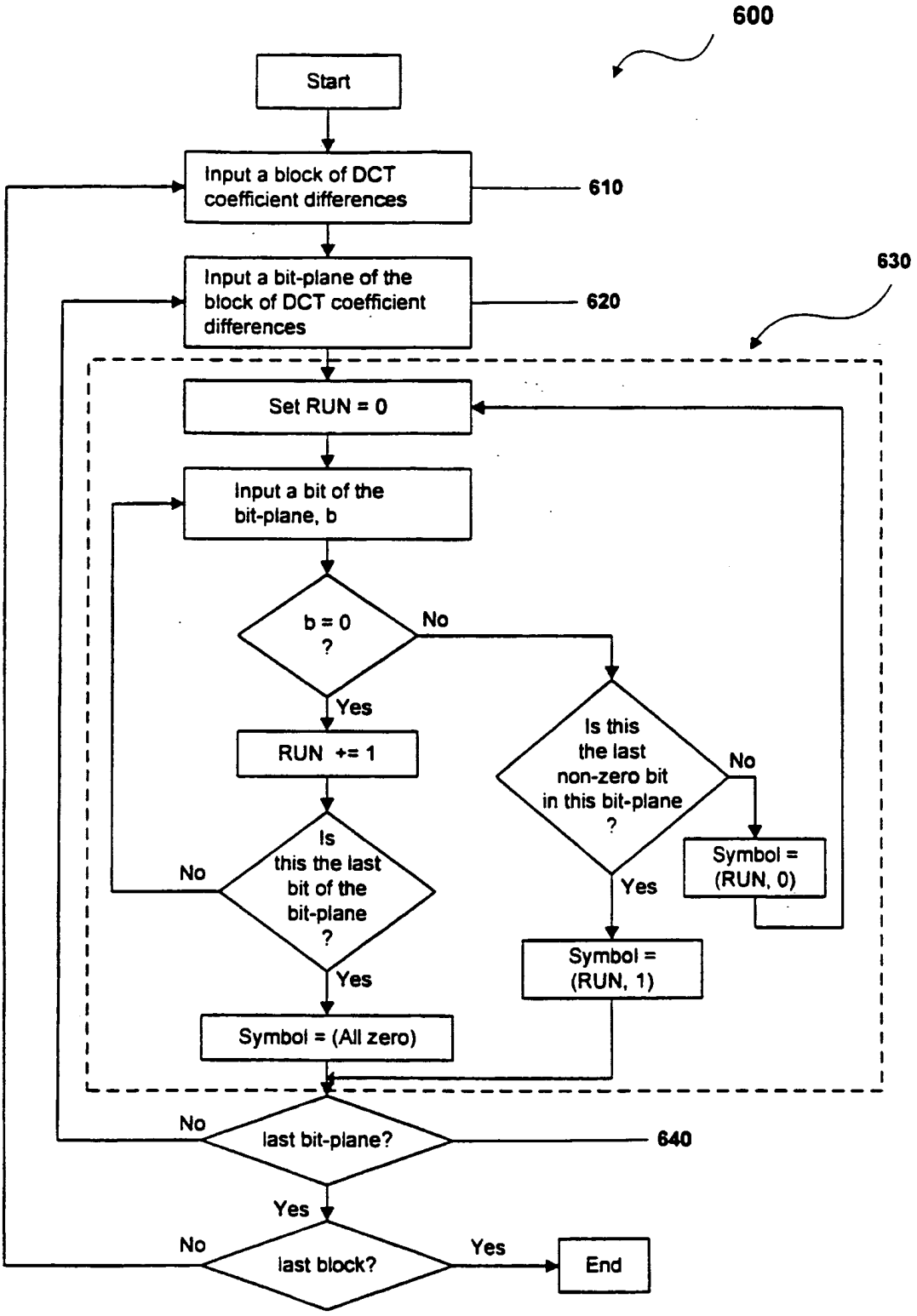

600
Start
Input a block of DCT coefficient differences
610
Input a bit-plane of the block of DCT coefficient differences
620
630
Set RUN = 0
Input a bit of the bit-plane, b
b = 0 ?
No
Yes
RUN += 1
Is this the last bit of the bit-plane ?
No
Yes
Symbol = (All zero)
Is this the last non-zero bit in this bit-plane ?
No
Yes
Symbol = (RUN, 0)
Symbol = (RUN, 1)
No
last bit-plane?
640
Yes
No
last block?
Yes
End

FIG. 6

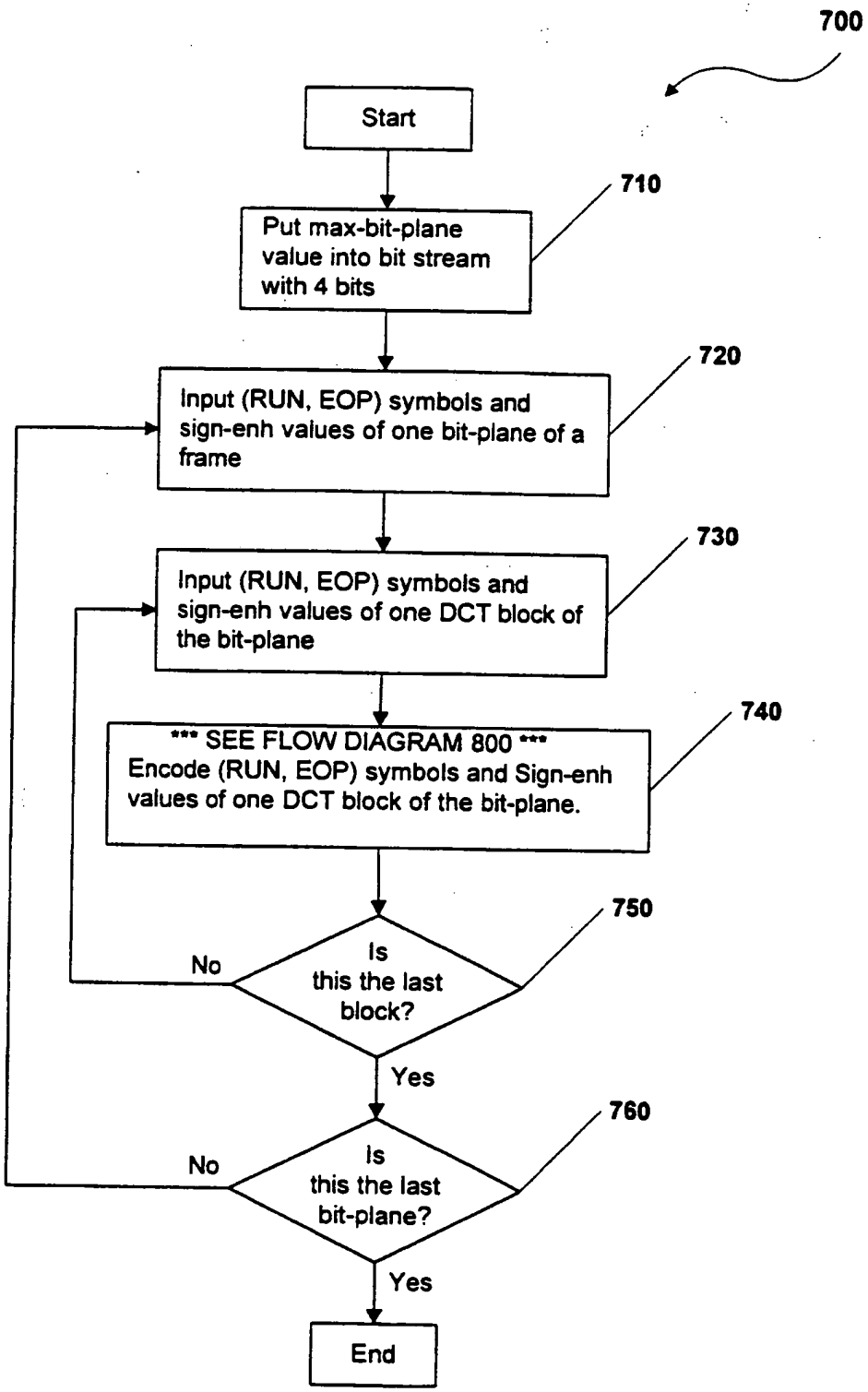

700
Start
710
Put max-bit-plane value into bit stream with 4 bits
720
Input (RUN, EOP) symbols and sign-enh values of one bit-plane of a frame
730
Input (RUN, EOP) symbols and sign-enh values of one DCT block of the bit-plane
740
*** SEE FLOW DIAGRAM 800 ***
Encode (RUN, EOP) symbols and Sign-enh values of one DCT block of the bit-plane.
750
Is this the last block?
No
Yes
760
Is this the last bit-plane?
No
Yes
End

FIG. 7

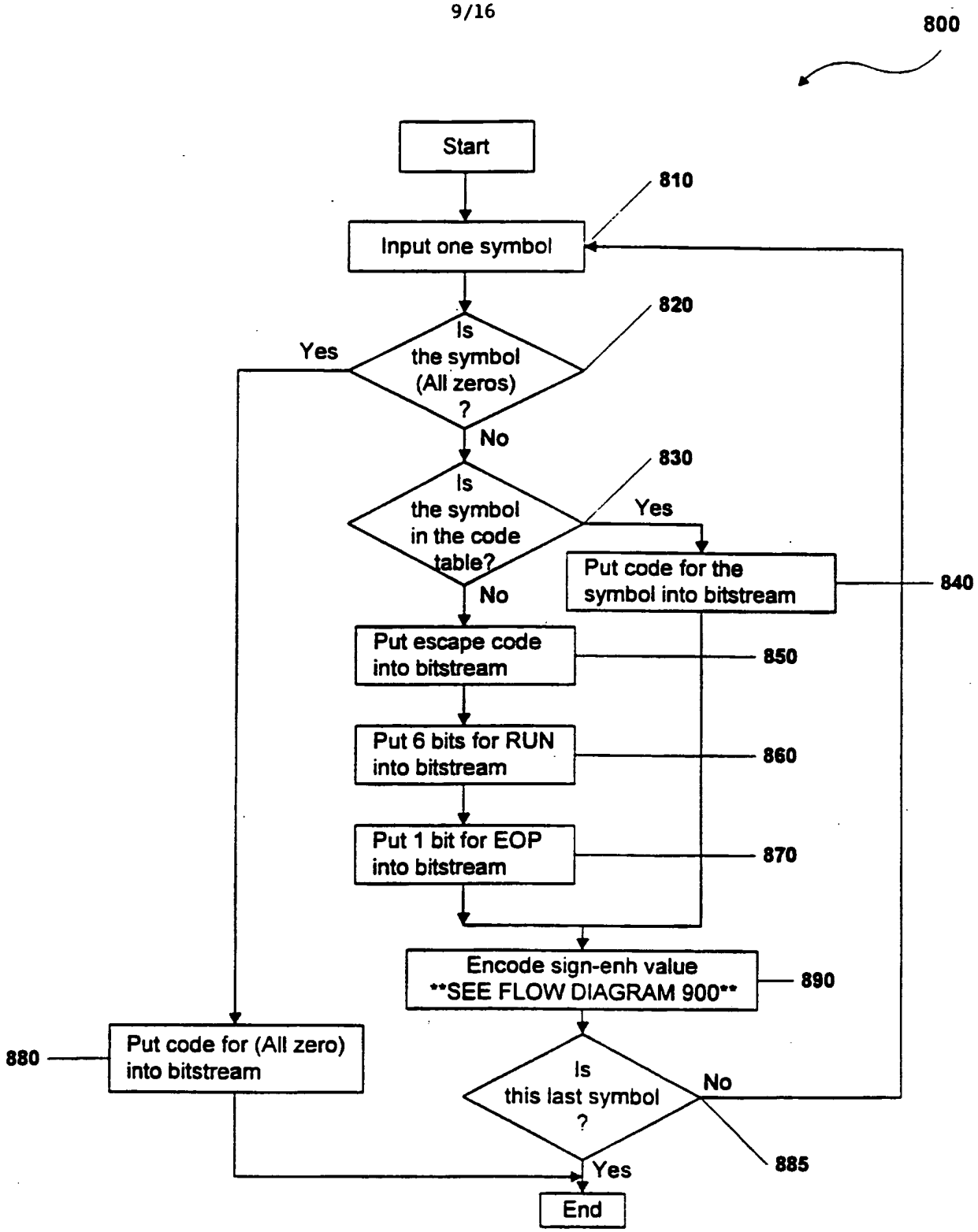

800
Start
810
Input one symbol
820
Is the symbol (All zeros) ?
Yes
No
830
Is the symbol in the code table?
Yes
No
Put code for the symbol into bitstream
840
Put escape code into bitstream
850
Put 6 bits for RUN into bitstream
860
Put 1 bit for EOP into bitstream
870
Encode sign-enh value **SEE FLOW DIAGRAM 900**
890
Put code for (All zero) into bitstream
880
Is this last symbol ?
No
885
Yes
End

FIG. 8

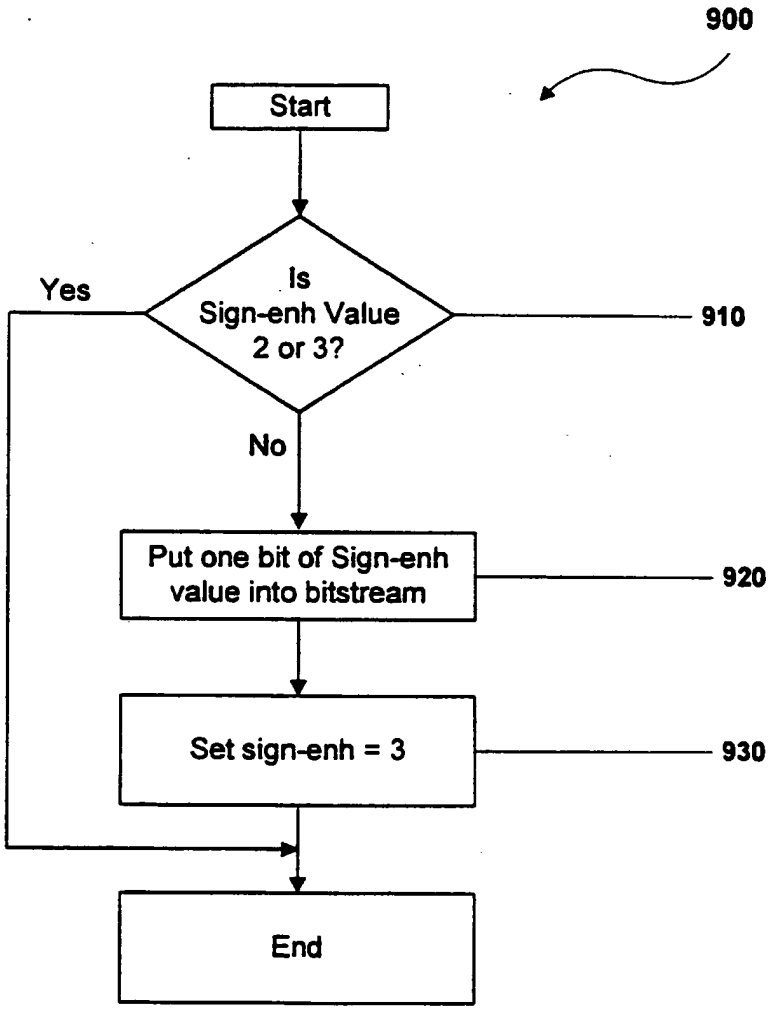

900
Start
Is
Sign-enh Value
2 or 3?
Yes
910
No
Put one bit of Sign-enh
value into bitstream
920
Set sign-enh = 3
930
End

FIG. 9

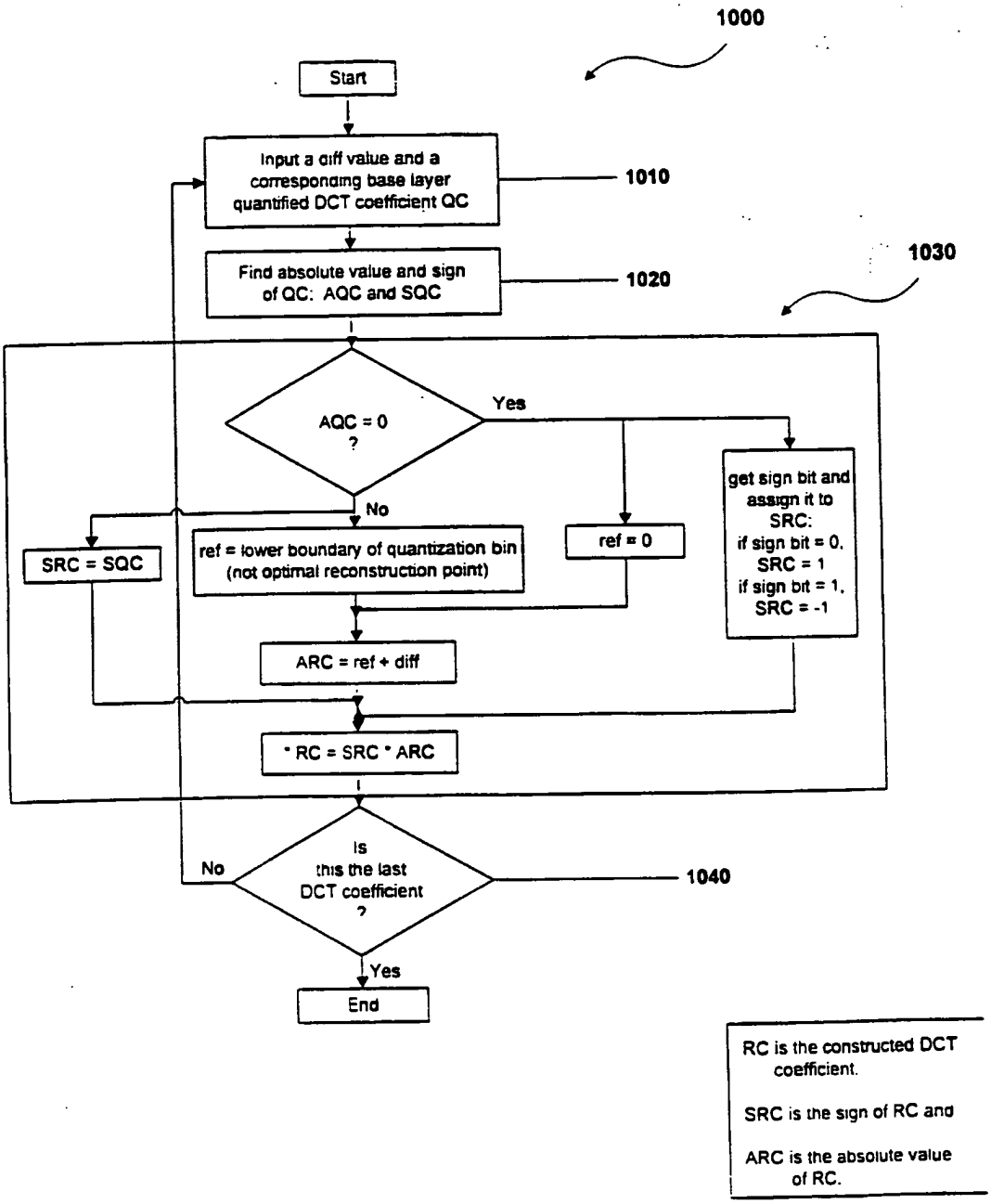

1000
Start
Input a diff value and a corresponding base layer quantified DCT coefficient QC
1010
Find absolute value and sign of QC: AQC and SQC
1020
1030
AQC = 0 ?
Yes
No
SRC = SQC
ref = lower boundary of quantization bin (not optimal reconstruction point)
ref = 0
get sign bit and assign it to SRC: if sign bit = 0, SRC = 1 if sign bit = 1, SRC = -1
ARC = ref + diff
* RC = SRC * ARC
Is this the last DCT coefficient ?
No
1040
Yes
End
RC is the constructed DCT coefficient.
SRC is the sign of RC and
ARC is the absolute value of RC.

FIG. 10

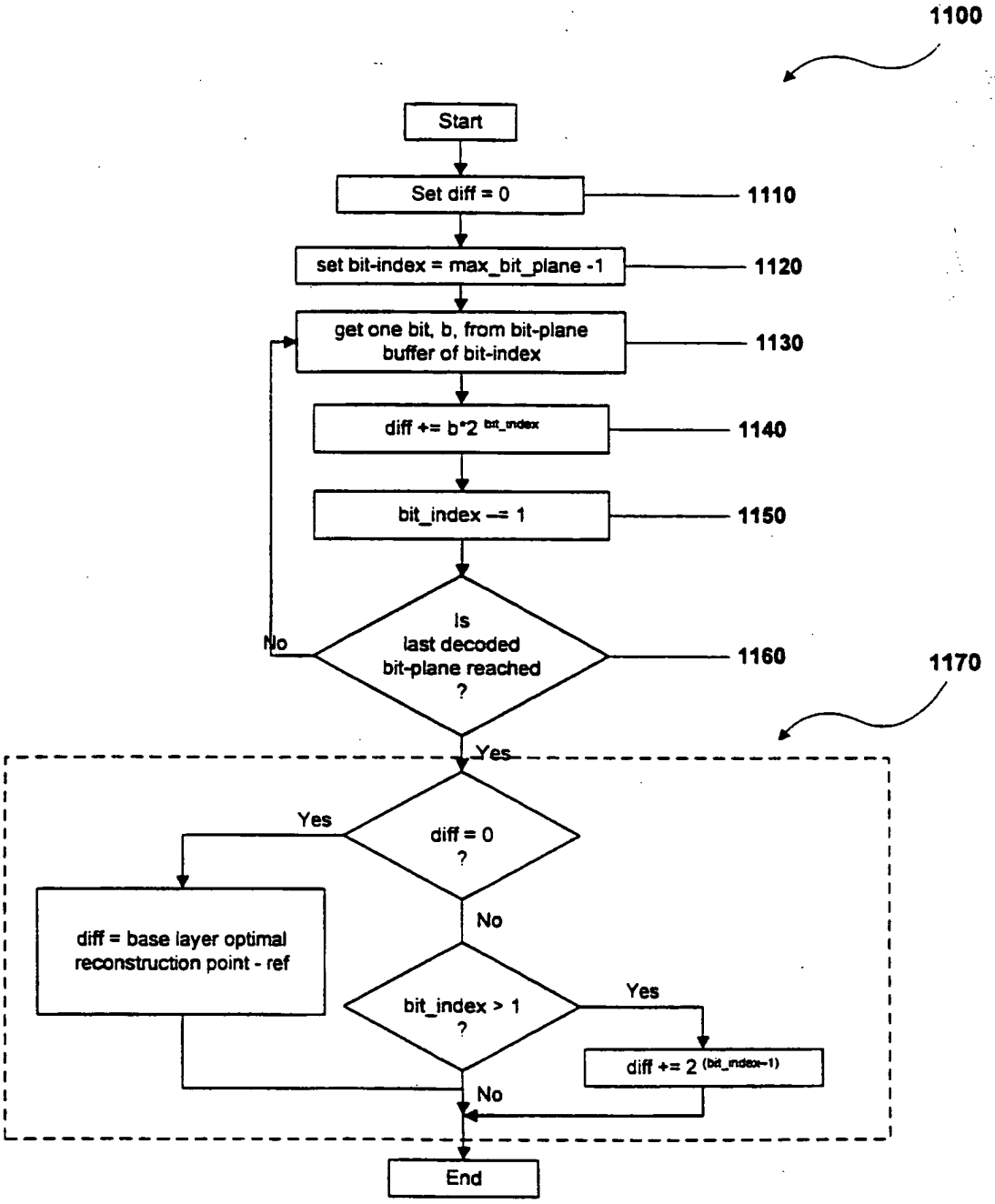

1100
Start
Set diff = 0
1110
set bit-index = max_bit_plane -1
1120
get one bit, b, from bit-plane buffer of bit-index
1130
diff += b*2 bit_index
1140
bit_index -= 1
1150
Is last decoded bit-plane reached ?
1160
No
Yes
1170
diff = 0 ?
Yes
diff = base layer optimal reconstruction point - ref
No
bit_index > 1 ?
Yes
diff += 2 (bit_index-1)
No
End

FIG. 11

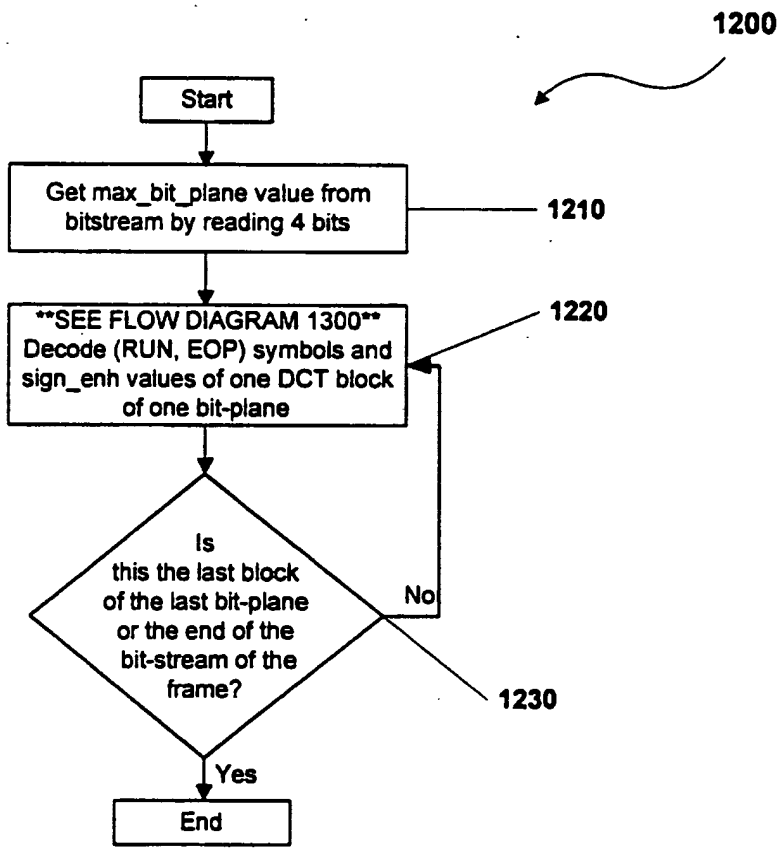

1200
Start
Get max_bit_plane value from bitstream by reading 4 bits
1210
SEE FLOW DIAGRAM 1300 Decode (RUN, EOP) symbols and sign_enh values of one DCT block of one bit-plane
1220
Is this the last block of the last bit-plane or the end of the bit-stream of the frame?
No
1230
Yes
End

FIG. 12

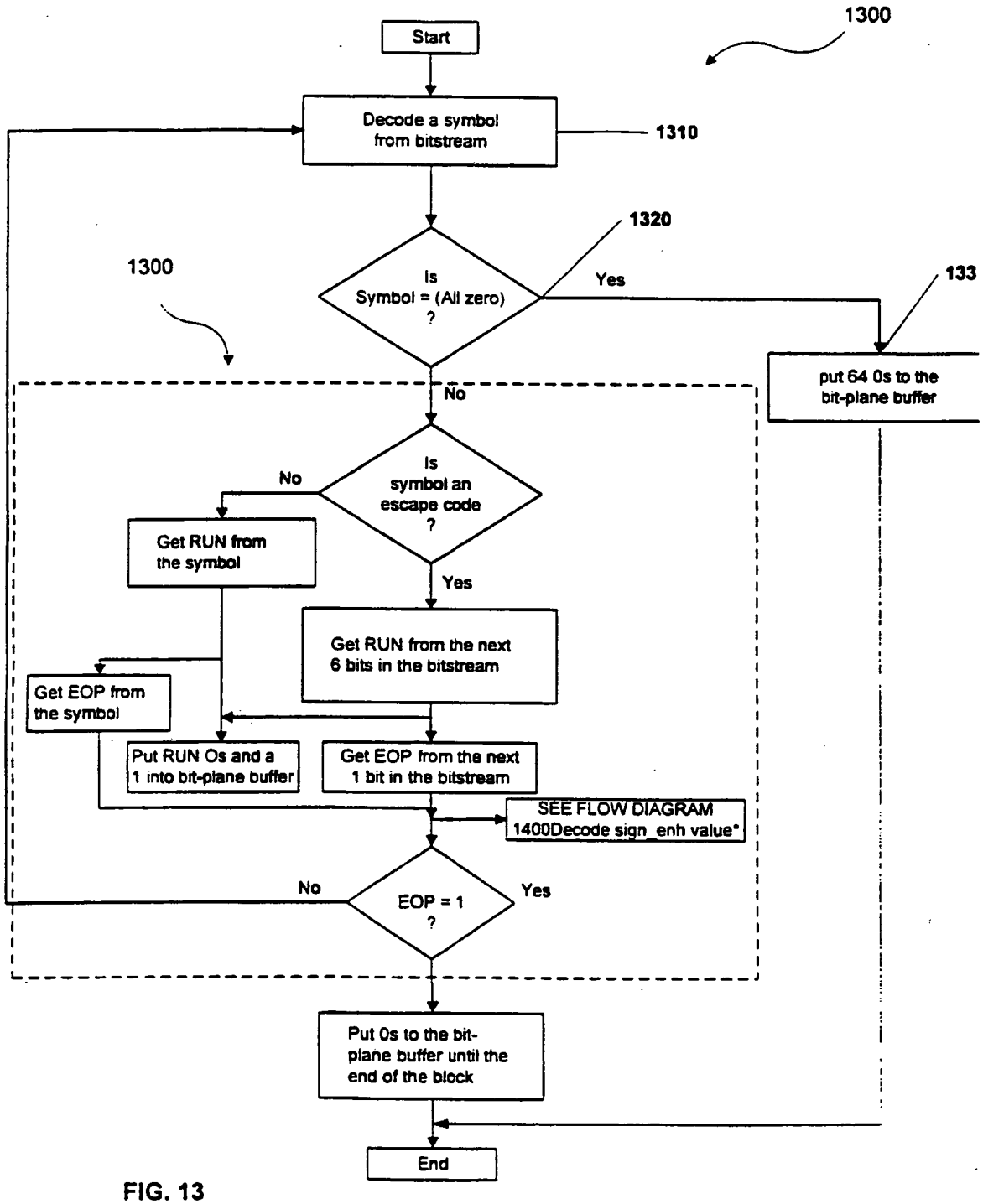

1300
Start
Decode a symbol from bitstream
1310
1320
1300
Is Symbol = (All zero) ?
Yes
133
put 64 0s to the bit-plane buffer
No
Is symbol an escape code ?
No
Get RUN from the symbol
Yes
Get RUN from the next 6 bits in the bitstream
Get EOP from the symbol
Put RUN 0s and a 1 into bit-plane buffer
Get EOP from the next 1 bit in the bitstream
SEE FLOW DIAGRAM 1400Decode sign_enh value*
No
EOP = 1 ?
Yes
Put 0s to the bit-plane buffer until the end of the block
End

FIG. 13

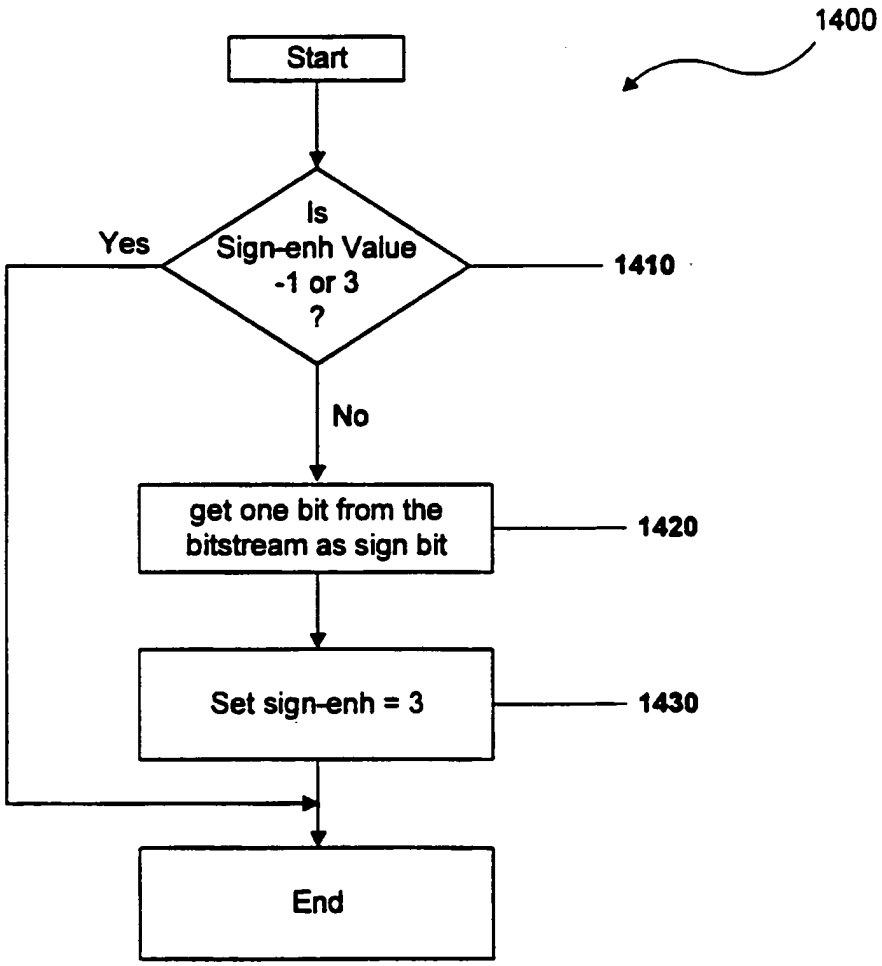

1400
Start
Is
Sign-enh Value
-1 or 3
?
Yes
1410
No
get one bit from the
bitstream as sign bit
1420
Set sign-enh = 3
1430
End

FIG. 14

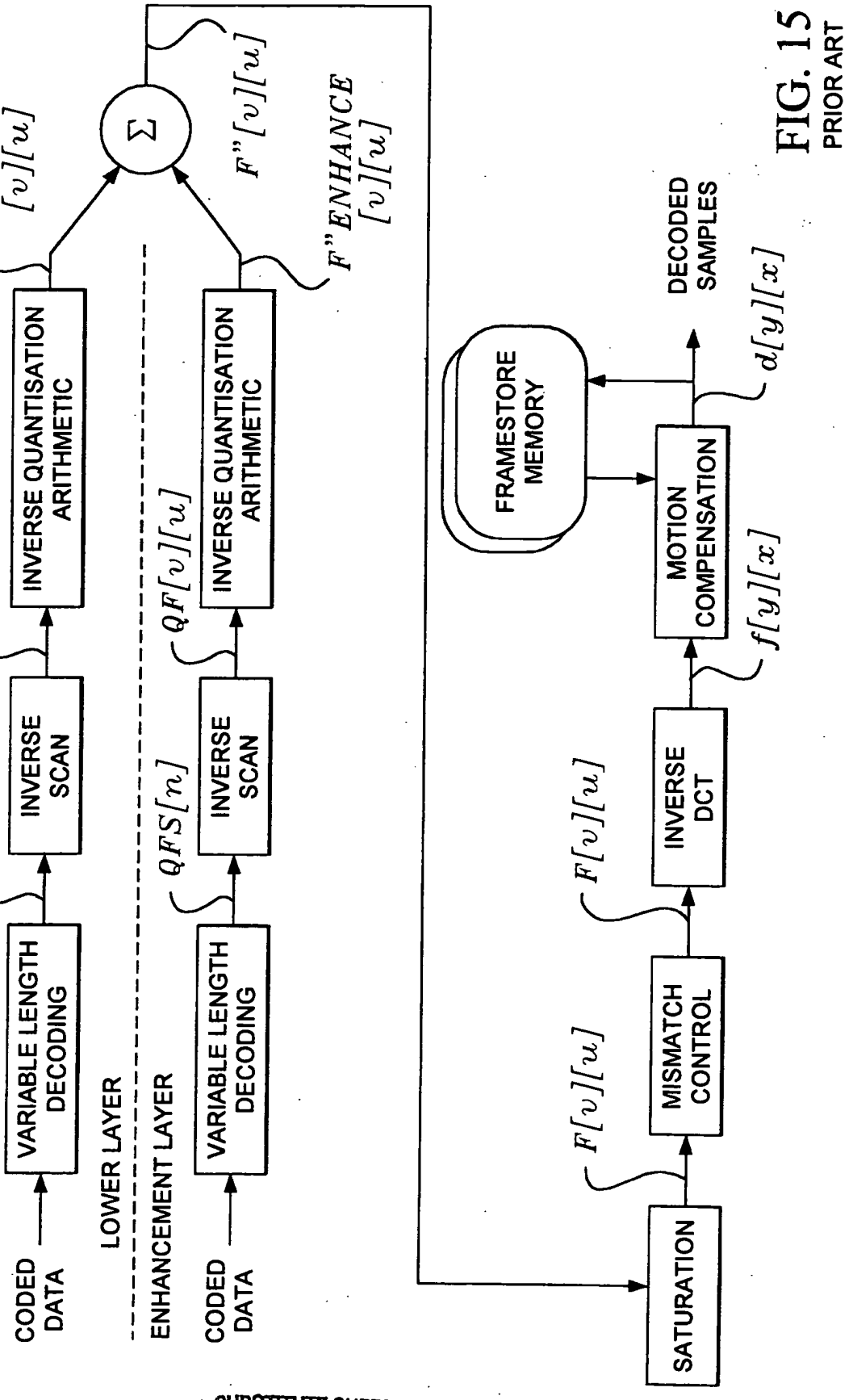


FIG. 15
PRIOR ART

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : H04L 9/00 — A2

(11) International Publication Number: **WO 00/59152**

(43) International Publication Date: 5 October 2000 (05.10.00)

(21) International Application Number: PCT/US00/04983

(22) International Filing Date: 25 February 2000 (25.02.00)

(30) Priority Data:

60/126,614	27 March 1999 (27.03.99)	US
09/290,363	12 April 1999 (12.04.99)	US
09/482,928	13 January 2000 (13.01.00)	US

(71) Applicant: MICROSOFT CORPORATION [US/US]; One Microsoft Way, Redmond, WA 98052 (US).

(72) Inventors: BLINN, Arnold, N.; 9401 NE 27th Street, Bellevue, WA 98004 (US). JONES, Thomas, C.; 23617 NE 6th Street, Redmond, WA 98053-3618 (US).

(74) Agents: ROCCI, Steven, J. et al.; Woodcock Washburn Kurtz Mackiewicz & Norris LLP, 46th floor, One Liberty Place, Philadelphia, PA 19103 (US).

(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

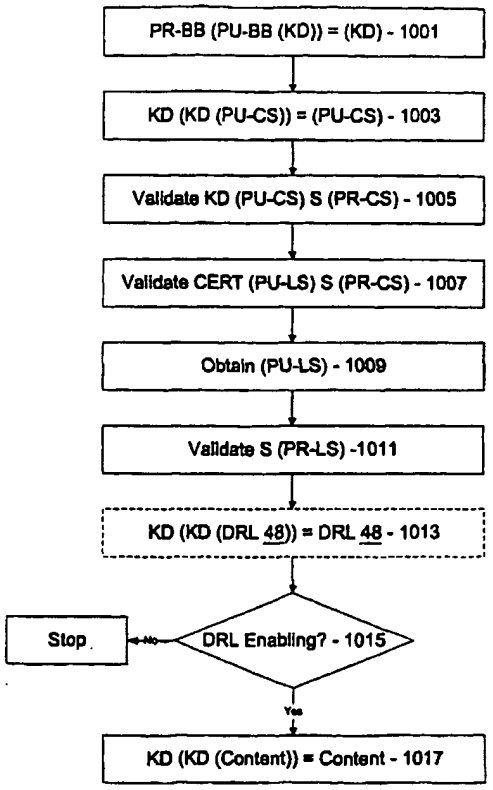
Published

Without international search report and to be republished upon receipt of that report.

(54) Title: METHOD FOR INTERDEPENDENTLY VALIDATING A DIGITAL CONTENT PACKAGE AND A CORRESPONDING DIGITAL LICENSE

(57) Abstract

A method is disclosed for a device to interdependently validate a digital content package having a piece of digital content in an encrypted form, and a corresponding digital license for rendering the digital content. A first key is derived from a source available to the device, and a first digital signature is obtained from the digital content package. The first key is applied to the first digital signature to validate the first digital signature and the digital content package. A second key is derived based on the first digital signature, and a second digital signature is obtained from the license. The second key is applied to the second digital signature to validate the second digital signature and the license.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

METHOD FOR INTERDEPENDENTLY VALIDATING A DIGITAL CONTENT PACKAGE AND A CORRESPONDING DIGITAL LICENSE

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a continuation of U.S. Patent Application No. 09/290,363, filed April 12, 1999 and entitled "ENFORCEMENT ARCHITECTURE AND METHOD FOR DIGITAL RIGHTS MANAGEMENT", and claims the benefit of U.S. Provisional Application No. 60/21,614, filed March 27, 1999 and entitled "ENFORCEMENT ARCHITECTURE AND METHOD FOR DIGITAL RIGHTS MANAGEMENT", both of which are hereby incorporated by reference.

TECHNICAL FIELD

The present invention relates to an architecture for enforcing rights in digital content. More specifically, the present invention relates to such an enforcement architecture that allows access to encrypted digital content only in accordance with parameters specified by license rights acquired by a user of the digital content.

BACKGROUND OF THE INVENTION

Digital rights management and enforcement is highly desirable in connection with digital content such as digital audio, digital video, digital text, digital data, digital multimedia, etc., where such digital content is to be distributed to users. Typical modes of distribution include tangible devices such as a magnetic (floppy) disk, a magnetic tape, an optical (compact) disk (CD), etc., and intangible media such as an electronic bulletin board, an electronic network, the Internet, etc. Upon being received by the user, such user renders or 'plays' the digital content with the aid of an appropriate rendering device such as a media player on a personal computer or the like.

Typically, a content owner or rights-owner, such as an author, a publisher, a broadcaster, etc. (hereinafter "content owner"), wishes to distribute such digital content to a user or recipient in exchange for a license fee or some other consideration. Such content owner, given the choice, would likely wish to restrict what

the user can do with such distributed digital content. For example, the content owner would like to restrict the user from copying and re-distributing such content to a second user, at least in a manner that denies the content owner a license fee from such second user.

In addition, the content owner may wish to provide the user with the flexibility to purchase different types of use licenses at different license fees, while at the same time holding the user to the terms of whatever type of license is in fact purchased. For example, the content owner may wish to allow distributed digital content to be played only a limited number of times, only for a certain total time, only on a certain type of machine, only on a certain type of media player, only by a certain type of user, etc.

However, after distribution has occurred, such content owner has very little if any control over the digital content. This is especially problematic in view of the fact that practically every new or recent personal computer includes the software and hardware necessary to make an exact digital copy of such digital content, and to download such exact digital copy to a write-able magnetic or optical disk, or to send such exact digital copy over a network such as the Internet to any destination.

Of course, as part of the legitimate transaction where the license fee was obtained, the content owner may require the user of the digital content to promise not to re-distribute such digital content. However, such a promise is easily made and easily broken. A content owner may attempt to prevent such re-distribution through any of several known security devices, usually involving encryption and decryption. However, there is likely very little that prevents a mildly determined user from decrypting encrypted digital content, saving such digital content in an un-encrypted form, and then re-distributing same.

A need exists, then, for providing an enforcement architecture and method that allows the controlled rendering or playing of arbitrary forms of digital content, where such control is flexible and definable by the content owner of such digital content. A need also exists for providing a controlled rendering environment

on a computing device such as a personal computer, where the rendering environment includes at least a portion of such enforcement architecture. Such controlled rendering environment allows that the digital content will only be rendered as specified by the content owner, even though the digital content is to be rendered on a computing device which is not under the control of the content owner.

Further, a need exists for a trusted component running on the computing device, where the trusted component enforces the rights of the content owner on such computing device in connection with a piece of digital content, even against attempts by the user of such computing device to access such digital content in ways not permitted by the content owner. As but one example, such a trusted software component prevents a user of the computing device from making a copy of such digital content, except as otherwise allowed for by the content owner thereof.

SUMMARY OF THE INVENTION

The aforementioned needs are satisfied at least in part by an enforcement architecture and method for digital rights management, where the architecture and method enforce rights in protected (secure) digital content available on a medium such as the Internet, an optical disk, etc. For purposes of making content available, the architecture includes a content server from which the digital content is accessible over the Internet or the like in an encrypted form. The content server may also supply the encrypted digital content for recording on an optical disk or the like, wherein the encrypted digital content may be distributed on the optical disk itself. At the content server, the digital content is encrypted using an encryption key, and public / private key techniques are employed to bind the digital content with a digital license at the user's computing device or client machine.

When a user attempts to render the digital content on a computing device, the rendering application invokes a Digital Rights Management (DRM) system on such user's computing device. If the user is attempting to render the digital content for the first time, the DRM system either directs the user to a license server to obtain a license to render such digital content in the manner sought, or transparently obtains

such license from such license server without any action necessary on the part of the user. The license includes:

- a decryption key (KD) that decrypts the encrypted digital content;
- a description of the rights (play, copy, etc.) conferred by the license and related conditions (begin date, expiration date, number of plays, etc.), where such description is in a digitally readable form; and
- a digital signature that ensures the integrity of the license.

The user cannot decrypt and render the encrypted digital content without obtaining such a license from the license server. The obtained license is stored in a license store in the user's computing device.

Importantly, the license server only issues a license to a DRM system that is 'trusted' (i.e., that can authenticate itself). To implement 'trust', the DRM system is equipped with a 'black box' that performs decryption and encryption functions for such DRM system. The black box includes a public / private key pair, a version number and a unique signature, all as provided by an approved certifying authority. The public key is made available to the license server for purposes of encrypting portions of the issued license, thereby binding such license to such black box. The private key is available to the black box only, and not to the user or anyone else, for purposes of decrypting information encrypted with the corresponding public key. The DRM system is initially provided with a black box with a public / private key pair, and the user is prompted to download from a black box server an updated secure black box when the user first requests a license. The black box server provides the updated black box, along with a unique public/private key pair. Such updated black box is written in unique executable code that will run only on the user's computing device, and is re-updated on a regular basis. When a user requests a license, the client machine sends the black box public key, version number, and signature to the license server, and such license server issues a license only if the version number is current and the signature is valid. A license request also includes an identification of the digital content for which a license is requested and a key ID that identifies the

decryption key associated with the requested digital content. The license server uses the black box public key to encrypt the decryption key, and the decryption key to encrypt the license terms, then downloads the encrypted decryption key and encrypted license terms to the user's computing device along with a license signature.

Once the downloaded license has been stored in the DRM system license store, the user can render the digital content according to the rights conferred by the license and specified in the license terms. When a request is made to render the digital content, the black box is caused to decrypt the decryption key and license terms, and a DRM system license evaluator evaluates such license terms. The black box decrypts the encrypted digital content only if the license evaluation results in a decision that the requestor is allowed to play such content. The decrypted content is provided to the rendering application for rendering.

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing summary, as well as the following detailed description of the embodiments of the present invention, will be better understood when read in conjunction with the appended drawings. For the purpose of illustrating the invention, there are shown in the drawings embodiments which are presently preferred. As should be understood, however, the invention is not limited to the precise arrangements and instrumentalities shown. In the drawings:

Fig. 1 is a block diagram showing an enforcement architecture in accordance with one embodiment of the present invention;

Fig. 2 is a block diagram of the authoring tool of the architecture of Fig. 1 in accordance with one embodiment of the present invention;

Fig. 3 is a block diagram of a digital content package having digital content for use in connection with the architecture of Fig. 1 in accordance with one embodiment of the present invention;

Fig. 4 is a block diagram of the user's computing device of Fig. 1 in accordance with one embodiment of the present invention;

Figs. 5A and 5B are flow diagrams showing the steps performed in connection with the Digital Rights Management (DRM) system of the computing device of Fig. 4 to render content in accordance with one embodiment of the present invention;

Fig. 6 is a flow diagram showing the steps performed in connection with the DRM system of Fig. 4 to determine whether any valid, enabling licenses are present in accordance with one embodiment of the present invention;

Fig. 7 is a flow diagram showing the steps performed in connection with the DRM system of Fig. 4 to obtain a license in accordance with one embodiment of the present invention;

Fig. 8 is a block diagram of a digital license for use in connection with the architecture of Fig. 1 in accordance with one embodiment of the present invention;

Fig. 9 is a flow diagram showing the steps performed in connection with the DRM system of Fig. 4 to obtain a new black box in accordance with one embodiment of the present invention;

Fig. 10 is a flow diagram showing the key transaction steps performed in connection with the DRM system of Fig. 4 to validate a license and a piece of digital content and render the content in accordance with one embodiment of the present invention;

Fig. 11 is a block diagram showing the license evaluator of Fig. 4 along with a Digital Rights License (DRL) of a license and a language engine for interpreting the DRL in accordance with one embodiment of the present invention; and

Fig. 12 is a block diagram representing a general purpose computer system in which aspects of the present invention and/or portions thereof may be incorporated.

Detailed Description of the Invention

Referring to the drawings in details, wherein like numerals are used to indicate like elements throughout, there is shown in Fig. 1 an enforcement architecture 10 in accordance with one embodiment of the present invention. Overall, the enforcement architecture 10 allows an owner of digital content 12 to specify license rules that must be satisfied before such digital content 12 is allowed to be rendered on a user's computing device 14. Such license rules are embodied within a digital license 16 that the user / user's computing device 14 (hereinafter, such terms are interchangeable unless circumstances require otherwise) must obtain from the content owner or an agent thereof. The digital content 12 is distributed in an encrypted form, and may be distributed freely and widely. Preferably, the decrypting key (KD) for decrypting the digital content 12 is included with the license 16.

COMPUTER ENVIRONMENT

Fig. 12 and the following discussion are intended to provide a brief general description of a suitable computing environment in which the present invention and/or portions thereof may be implemented. Although not required, the invention is described in the general context of computer-executable instructions, such as program modules, being executed by a computer, such as a client workstation or a server. Generally, program modules include routines, programs, objects, components, data structures and the like that perform particular tasks or implement particular abstract data types. Moreover, it should be appreciated that the invention and/or portions thereof may be practiced with other computer system configurations, including hand-held devices, multi-processor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers and the like. The invention may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote memory storage devices.

As shown in Fig. 12, an exemplary general purpose computing system

includes a conventional personal computer 120 or the like, including a processing unit 121, a system memory 122, and a system bus 18 that couples various system components including the system memory to the processing unit 121. The system bus 18 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. The system memory includes read-only memory (ROM) 19 and random access memory (RAM) 20. A basic input/output system 21 (BIOS), containing the basic routines that help to transfer information between elements within the personal computer 120, such as during start-up, is stored in ROM 19.

The personal computer 120 may further include a hard disk drive 22 for reading from and writing to a hard disk (not shown), a magnetic disk drive 128 for reading from or writing to a removable magnetic disk 129, and an optical disk drive 25 for reading from or writing to a removable optical disk 131 such as a CD-ROM or other optical media. The hard disk drive 22, magnetic disk drive 128, and optical disk drive 25 are connected to the system bus 18 by a hard disk drive interface 27, a magnetic disk drive interface 28, and an optical drive interface 29, respectively. The drives and their associated computer-readable media provide non-volatile storage of computer readable instructions, data structures, program modules and other data for the personal computer 20.

Although the exemplary environment described herein employs a hard disk, a removable magnetic disk 129, and a removable optical disk 131, it should be appreciated that other types of computer readable media which can store data that is accessible by a computer may also be used in the exemplary operating environment. Such other types of media include a magnetic cassette, a flash memory card, a digital video disk, a Bernoulli cartridge, a random access memory (RAM), a read-only memory (ROM), and the like.

A number of program modules may be stored on the hard disk, magnetic disk 129, optical disk 131, ROM 19 or RAM 20, including an operating system 30, one or more application programs 136, other program modules 137 and

program data 138. A user may enter commands and information into the personal computer 120 through input devices such as a keyboard 35 and pointing device 142. Other input devices (not shown) may include a microphone, joystick, game pad, satellite disk, scanner, or the like. These and other input devices are often connected to the processing unit 121 through a serial port interface 41 that is coupled to the system bus, but may be connected by other interfaces, such as a parallel port, game port, or universal serial bus (USB). A monitor 42 or other type of display device is also connected to the system bus 18 via an interface, such as a video adapter 148. In addition to the monitor 42, a personal computer typically includes other peripheral output devices (not shown), such as speakers and printers. The exemplary system of Fig. 12 also includes a host adapter 50, a Small Computer System Interface (SCSI) bus 156, and an external storage device 162 connected to the SCSI bus 156.

The personal computer 120 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 149. The remote computer 149 may be another personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the personal computer 120, although only a memory storage device 150 has been illustrated in Fig. 12. The logical connections depicted in Fig. 12 include a local area network (LAN) 46 and a wide area network (WAN) 47. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets, and the Internet.

When used in a LAN networking environment, the personal computer 120 is connected to the LAN 46 through a network interface or adapter 48. When used in a WAN networking environment, the personal computer 120 typically includes a modem 49 or other means for establishing communications over the wide area network 47, such as the Internet. The modem 49, which may be internal or external, is connected to the system bus 18 via the serial port interface 41. In a networked environment, program modules depicted relative to the personal computer 120, or portions thereof, may be stored in the remote memory storage device. It will be

appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

ARCHITECTURE

Referring again to Fig. 1, in one embodiment of the present invention, the architecture 10 includes an authoring tool 18, a content-key database 20, a content server 22, a license server 24, and a black box server 26, as well as the aforementioned user's computing device 14.

ARCHITECTURE - Authoring Tool 18

The authoring tool 18 is employed by a content owner to package a piece of digital content 12 into a form that is amenable for use in connection with the architecture 10 of the present invention. In particular, the content owner provides the authoring tool 18 with the digital content 12, instructions and/or rules that are to accompany the digital content 12, and instructions and/or rules as to how the digital content 12 is to be packaged. The authoring tool 18 then produces a digital content package 12p having the digital content 12 encrypted according to an encryption / decryption key, and the instructions and/or rules that accompany the digital content 12.

In one embodiment of the present invention, the authoring tool 18 is instructed to serially produce several different digital content 12 packages 12p, each having the same digital content 12 encrypted according to a different encryption / decryption key. As should be understood, having several different packages 12p with the same digital content 12 may be useful for tracking the distribution of such packages 12p / content 12 (hereinafter simply "digital content 12", unless circumstances require otherwise). Such distribution tracking is not ordinarily necessary, but may be used by an investigative authority in cases where the digital content 12 has been illegally sold or broadcast.

In one embodiment of the present invention, the encryption / decryption key that encrypts the digital content 12 is a symmetric key, in that the encryption key is also the decryption key (KD). As will be discussed below in more detail, such decryption key (KD) is delivered to a user's computing device 14 in a hidden form as

part of a license 16 for such digital content 12. Preferably, each piece of digital content 12 is provided with a content ID (or each package 12p is provided with a package ID), each decryption key (KD) has a key ID, and the authoring tool 18 causes the decryption key (KD), key ID, and content ID (or package ID) for each piece of digital content 12 (or each package 12p) to be stored in the content-key database 20. In addition, license data regarding the types of licenses 16 to be issued for the digital content 12 and the terms and conditions for each type of license 16 may be stored in the content-key database 20, or else in another database (not shown). Preferably, the license data can be modified by the content owner at a later time as circumstances and market conditions may require.

In use, the authoring tool 18 is supplied with information including, among other things:

- the digital content 12 to be packaged;
- the type and parameters of watermarking and/or fingerprinting to be employed, if any;
- the type and parameters of data compression to be employed, if any;
- the type and parameters of encryption to be employed;
- the type and parameters of serialization to be employed, if any; and
- the instructions and/or rules that are to accompany the digital content 12.

As is known, a watermark is a hidden, computer-readable signal that is added to the digital content 12 as an identifier. A fingerprint is a watermark that is different for each instance. As should be understood, an instance is a version of the digital content 12 that is unique. Multiple copies of any instance may be made, and any copy is of a particular instance. When a specific instance of digital content 12 is illegally sold or broadcast, an investigative authority can perhaps identify suspects according to the watermark / fingerprint added to such digital content 12.

Data compression may be performed according to any appropriate compression algorithm without departing from the spirit and scope of the present

invention. For example, the .mp3 or .wav compression algorithm may be employed. Of course, the digital content 12 may already be in a compressed state, in which case no additional compression is necessary.

The instructions and/or rules that are to accompany the digital content 12 may include practically any appropriate instructions, rules, or other information without departing from the spirit and scope of the present invention. As will be discussed below, such accompanying instructions / rules / information are primarily employed by the user and the user's computing device 14 to obtain a license 16 to render the digital content 12. Accordingly, such accompanying instructions / rules / information may include an appropriately formatted license acquisition script or the like, as will be described in more detail below. In addition, or in the alternative, such accompanying instructions / rules / information may include 'preview' information designed to provide a user with a preview of the digital content 12.

With the supplied information, the authoring tool 18 then produces one or more packages 12p corresponding to the digital content 12. Each package 12p may then be stored on the content server 22 for distribution to the world.

In one embodiment of the present invention, and referring now to Fig. 2, the authoring tool 18 is a dynamic authoring tool 18 that receives input parameters which can be specified and operated on. Accordingly, such authoring tool 18 can rapidly produce multiple variations of package 12p for multiple pieces of digital content 12. Preferably, the input parameters are embodied in the form of a dictionary 28, as shown, where the dictionary 28 includes such parameters as:

- the name of the input file 29a having the digital content 12;
- the type of encoding that is to take place
- the encryption / decryption key (KD) to be employed,
- the accompanying instructions / rules / information ('header information') to be packaged with the digital content 12 in the package 12p.
- the type of muxing that is to occur: and

- the name of the output file 29b to which the package 12p based on the digital content 12 is to be written.

As should be understood, such dictionary 28 is easily and quickly modifiable by an operator of the authoring tool 18 (human or machine), and therefore the type of authoring performed by the authoring tool 18 is likewise easily and quickly modifiable in a dynamic manner. In one embodiment of the present invention, the authoring tool 18 includes an operator interface (not shown) displayable on a computer screen to a human operator. Accordingly, such operator may modify the dictionary 28 by way of the interface, and further may be appropriately aided and/or restricted in modifying the dictionary 28 by way of the interface.

In the authoring tool 18, and as seen in Fig. 2, a source filter 18a receives the name of the input file 29a having the digital content 12 from the dictionary 28, and retrieves such digital content 12 from such input file and places the digital content 12 into a memory 29c such as a RAM or the like. An encoding filter 18b then performs encoding on the digital content 12 in the memory 29c to transfer the file from the input format to the output format according to the type of encoding specified in the dictionary 28 (i.e., .wav to .asp, .mp3 to .asp, etc.). and places the encoded digital content 12 in the memory 29c. As shown, the digital content 12 to be packaged (music, e.g.) is received in a compressed format such as the .wav or .mp3 format, and is transformed into a format such as the .asp (active streaming protocol) format. Of course, other input and output formats may be employed without departing from the spirit and scope of the present invention.

Thereafter, an encryption filter 18c encrypts the encoded digital content 12 in the memory 29c according to the encryption / decryption key (KD) specified in the dictionary 28, and places the encrypted digital content 12 in the memory 29c. A header filter 18d then adds the header information specified in the dictionary 28 to the encrypted digital content 12 in the memory 29c.

As should be understood, depending on the situation, the package 12p may include multiple streams of temporally aligned digital content 12 (one stream

being shown in Fig. 2), where such multiple streams are multiplexed (i.e., 'muxed'). Accordingly, a mux filter 18e performs muxing on the header information and encrypted digital content 12 in the memory 29c according to the type of muxing specified in the dictionary 28, and places the result in the memory 29c. A file writer filter 18f then retrieves the result from the memory 29c and writes such result to the output file 29b specified in the dictionary 28 as the package 12p.

It should be noted that in certain circumstances, the type of encoding to be performed will not normally change. Since the type of muxing typically is based on the type of encoding, it is likewise the case that the type of muxing will not normally change, either. If this is in fact the case, the dictionary 28 need not include parameters on the type of encoding and/or the type of muxing. Instead, it is only necessary that the type of encoding be 'hardwired' into the encoding filter and/or that the type of muxing be 'hardwired' into the mux filter. Of course, as circumstance require, the authoring tool 18 may not include all of the aforementioned filters, or may include other filters, and any included filter may be hardwired or may perform its function according to parameters specified in the dictionary 28, all without departing from the spirit and scope of the present invention.

Preferably, the authoring tool 18 is implemented on an appropriate computer, processor, or other computing machine by way of appropriate software. The structure and operation of such machine and such software should be apparent based on the disclosure herein and therefore do not require any detailed discussion in the present disclosure.

ARCHITECTURE - Content Server 22

Referring again to Fig. 1, in one embodiment of the present invention, the content server 22 distributes or otherwise makes available for retrieval the packages 12p produced by the authoring tool 18. Such packages 12p may be distributed as requested by the content server 22 by way of any appropriate distribution channel without departing from the spirit and scope of the present invention. For example, such distribution channel may be the Internet or another network, an electronic bulletin

board, electronic mail, or the like. In addition, the content server 22 may be employed to copy the packages 12p onto magnetic or optical disks or other storage devices, and such storage devices may then be distributed.

It will be appreciated that the content server 22 distributes packages 12p without regard to any trust or security issues. As discussed below, such issues are dealt with in connection with the license server 24 and the relationship between such license server 24 and the user's computing device 14. In one embodiment of the present invention, the content server 22 freely releases and distributes packages 12p having digital content 12 to any distributee requesting same. However, the content server 22 may also release and distribute such packages 12p in a restricted manner without departing from the spirit and scope of the present invention. For example, the content server 22 may first require payment of a pre-determined distribution fee prior to distribution, or may require that a distributee identify itself, or may indeed make a determination of whether distribution is to occur based on an identification of the distributee.

In addition, the content server 22 may be employed to perform inventory management by controlling the authoring tool 18 to generate a number of different packages 12p in advance to meet an anticipated demand. For example, the server could generate 100 packages 12p based on the same digital content 12, and serve each package 12p 10 times. As supplies of packages 12p dwindle to 20, for example, the content server 22 may then direct the authoring tool 18 to generate 80 additional packages 12p, again for example.

Preferably, the content server 22 in the architecture 10 has a unique public / private key pair (PU-CS, PR-CS) that is employed as part of the process of evaluating a license 16 and obtaining a decryption key (KD) for decrypting corresponding digital content 12, as will be explained in more detail below. As is known, a public / private key pair is an asymmetric key, in that what is encrypted in one of the keys in the key pair can only be decrypted by the other of the keys in the key pair. In a public / private key pair encryption system, the public key may be made

known to the world, but the private key should always be held in confidence by the owner of such private key. Accordingly, if the content server 22 encrypts data with its private key (PR-CS), it can send the encrypted data out into the world with its public key (PU-CS) for decryption purposes. Correspondingly, if an external device wants to send data to the content server 22 so that only such content server 22 can decrypt such data, such external device must first obtain the public key of the content server 22 (PU-CS) and then must encrypt the data with such public key. Accordingly, the content server 22 (and only the content server 22) can then employ its private key (PR-CS) to decrypt such encrypted data.

As with the authoring tool 18, the content server 22 is implemented on an appropriate computer, processor, or other computing machine by way of appropriate software. The structure and operation of such machine and such software should be apparent based on the disclosure herein and therefore do not require any detailed discussion in the present disclosure. Moreover, in one embodiment of the present invention, the authoring tool 18 and the content server 22 may reside on a single computer, processor, or other computing machine, each in a separate work space. It should be recognized, moreover, that the content server 22 may in certain circumstances include the authoring tool 18 and/or perform the functions of the authoring tool 18, as discussed above.

Structure of Digital Content Package 12p

Referring now to Fig. 3, in one embodiment of the present invention, the digital content package 12p as distributed by the content server 22 includes:

- the digital content 12 encrypted with the encryption / decryption key (KD), as was discussed above (i.e., (KD(CONTENT)));
- the content ID (or package ID) of such digital content 12 (or package 12p);
- the key ID of the decryption key (KD);
- license acquisition information, preferably in an un-encrypted form; and

- the key KD encrypting the content server 22 public key (PU-CS), signed by the content server 22 private key (PR-CS) (i.e., (KD (PU-CS) S (PR-CS))).

With regard to (KD (PU-CS) S (PR-CS)), it is to be understood that such item is to be used in connection with validating the digital content 12 and/or package 12p, as will be explained below. Unlike a certificate with a digital signature (see below), the key (PU-CS) is not necessary to get at (KD (PU-CS)). Instead, the key (PU-CS) is obtained merely by applying the decryption key (KD). Once so obtained, such key (PU-CS) may be employed to test the validity of the signature (S (PR-CS)).

It should also be understood that for such package 12p to be constructed by the authoring tool 18, such authoring tool 18 must already possess the license acquisition information and (KD (PU-CS) S (PR-CS)), presumably as header information supplied by the dictionary 28. Moreover, the authoring tool 18 and the content server 22 must presumably interact to construct (KD (PU-CS) S (PR-CS)). Such interaction may for example include the steps of:

- the content server 22 sending (PU-CS) to the authoring tool 18;
- the authoring tool 18 encrypting (PU-CS) with (KD) to produce (KD (PU-CS));
- the authoring tool 18 sending (KD (PU-CS)) to the content server 22;
- the content server 22 signing (KD (PU-CS)) with (PR-CS) to produce (KD (PU-CS) S (PR-CS)); and
- the content server 22 sending (KD (PU-CS) S (PR-CS)) to the authoring tool 18.

ARCHITECTURE - License Server 24

Referring again to Fig. 1, in one embodiment of the present invention, the license server 24 performs the functions of receiving a request for a license 16 from a user's computing device 14 in connection with a piece of digital content 12,

determining whether the user's computing device 14 can be trusted to honor an issued license 16, negotiating such a license 16, constructing such license 16, and transmitting such license 16 to the user's computing device 14. Preferably, such transmitted license 16 includes the decryption key (KD) for decrypting the digital content 12. Such license server 24 and such functions will be explained in more detail below. Preferably, and like the content server 22, the license server 24 in the architecture 10 has a unique public / private key pair (PU-LS, PR-LS) that is employed as part of the process of evaluating a license 16 and obtaining a decryption key (KD) for decrypting corresponding digital content 12, as will be explained in more detail below.

As with the authoring tool 18 and the content server 22, the license server 24 is implemented on an appropriate computer, processor, or other computing machine by way of appropriate software. The structure and operation of such machine and such software should be apparent based on the disclosure herein and therefore do not require any detailed discussion in the present disclosure. Moreover, in one embodiment of the present invention the authoring tool 18 and/or the content server 22 may reside on a single computer, processor, or other computing machine together with the license server 24, each in a separate work space.

In one embodiment of the present invention, prior to issuance of a license 16, the license server 24 and the content server 22 enter into an agency agreement or the like, wherein the license server 24 in effect agrees to be the licensing authority for at least a portion of the digital content 12 distributed by the content server 22. As should be understood, one content server 22 may enter into an agency agreement or the like with several license servers 24, and/or one license server 24 may enter into an agency agreement or the like with several content servers 22, all without departing from the spirit and scope of the present invention.

Preferably, the license server 24 can show to the world that it does in fact have the authority to issue a license 16 for digital content 12 distributed by the content server 22. To do so, it is preferable that the license server 24 send to the content server 22 the license server 24 public key (PU-LS), and that the content server

22 then send to the license server 24 a digital certificate containing PU-LS as the contents signed by the content server 22 private key (CERT (PU-LS) S (PR-CS)). As should be understood, the contents (PU-LS) in such certificate can only be accessed with the content server 22 public key (PU-CS). As should also be understood, in general, a digital signature of underlying data is an encrypted form of such data, and will not match such data when decrypted if such data has been adulterated or otherwise modified.

As a licensing authority in connection with a piece of digital content 12, and as part of the licensing function, the license server 24 must have access to the decryption key (KD) for such digital content 12. Accordingly, it is preferable that license server 24 have access to the content-key database 20 that has the decryption key (KD), key ID, and content ID (or package ID) for such digital content 12 (or package 12p).

ARCHITECTURE - Black Box Server 26

Still referring to Fig. 1, in one embodiment of the present invention, the black box server 26 performs the functions of installing and/or upgrading a new black box 30 in a user's computing device 14. As will be explained in more detail below, the black box 30 performs encryption and decryption functions for the user's computing device 14. As will also be explained in more detail below, the black box 30 is intended to be secure and protected from attack. Such security and protection is provided, at least in part, by upgrading the black box 30 to a new version as necessary by way of the black box server 26, as will be explained in more detail below.

As with the authoring tool 18, the content server 22, and the license server 24, the black box server 26 is implemented on an appropriate computer, processor, or other computing machine by way of appropriate software. The structure and operation of such machine and such software should be apparent based on the disclosure herein and therefore do not require any detailed discussion in the present disclosure. Moreover, in one embodiment of the present invention the license server 24, the authoring tool 18, and/or the content server 22 may reside on a single computer.

processor, or other computing machine together with the black box server 26, each in a separate work space. Note, though, that for security purposes. it may be wise to have the black box server 26 on a separate machine.

ARCHITECTURE - User's Computing Device 14

Referring now to Fig. 4, in one embodiment of the present invention, the user's computing device 14 is a personal computer or the like. having elements including a keyboard, a mouse, a screen, a processor, RAM. ROM, a hard drive, a floppy drive, a CD player, and/or the like. However, the user's computing device 14 may also be a dedicated viewing device such as a television or monitor, a dedicated audio device such as a stereo or other music player, a dedicated printer, or the like, among other things, all without departing from the spirit and scope of the present invention.

The content owner for a piece of digital content 12 must trust that the user's computing device 14 will abide by the rules specified by such content owner, i.e. that the digital content 12 will not be rendered unless the user obtains a license 16 that permits the rendering in the manner sought. Preferably, then, the user's computing device 14 must provide a trusted component or mechanism 32 that can satisfy to the content owner that such computing device 14 will not render the digital content 12 except according to the license rules embodied in the license 16 associated with the digital content 12 and obtained by the user.

Here, the trusted mechanism 32 is a Digital Rights Management (DRM) system 32 that is enabled when a user requests that a piece of digital content 12 be rendered, that determines whether the user has a license 16 to render the digital content 12 in the manner sought, that effectuates obtaining such a license 16 if necessary, that determines whether the user has the right to play the digital content 12 according to the license 16, and that decrypts the digital content 12 for rendering purposes if in fact the user has such right according to such license 16. The contents and function of the DRM system 32 on the user's computing device 14 and in connection with the architecture 10 are described below.

DRM SYSTEM 32

The DRM system 32 performs four main functions with the architecture 10 disclosed herein: (1) content acquisition, (2) license acquisition, (3) content rendering, and (4) black box 30 installation / update. Preferably, any of the functions can be performed at any time, although it is recognized that some of the functions already require that digital content 12 be acquired.

DRM SYSTEM 32 - Content Acquisition

Acquisition of digital content 12 by a user and/or the user's computing device 14 is typically a relatively straight-forward matter and generally involves placing a file having encrypted digital content 12 on the user's computing device 14. Of course, to work with the architecture 10 and the DRM system 32 disclosed herein, it is necessary that the encrypted digital content 12 be in a form that is amenable to such architecture 10 and DRM system 32, such as the digital package 12p as will be described below.

As should be understood, the digital content 12 may be obtained in any manner from a content server 22, either directly or indirectly, without departing from the spirit and scope of the present invention. For example, such digital content 12 may be downloaded from a network such as the Internet, located on an obtained optical or magnetic disk or the like, received as part of an E-mail message or the like, or downloaded from an electronic bulletin board or the like.

Such digital content 12, once obtained, is preferably stored in a manner such that the obtained digital content 12 is accessible by a rendering application 34 (to be described below) running on the computing device 14, and by the DRM system 32. For example, the digital content 12 may be placed as a file on a hard drive (not shown) of the user's computing device 14, or on a network server (not shown) accessible to the computing device 14. In the case where the digital content 12 is obtained on an optical or magnetic disk or the like, it may only be necessary that such disk be present in an appropriate drive (not shown) coupled to the user's computing device 14.

In the present invention, it is not envisioned that any special tools are

necessary to acquire digital content 12, either from the content server 22 as a direct distribution source or from some intermediary as an indirect distribution source. That is, it is preferable that digital content 12 be as easily acquired as any other data file. However, the DRM system 32 and/or the rendering application 34 may include an interface (not shown) designed to assist the user in obtaining digital content 12 . For example, the interface may include a web browser especially designed to search for digital content 12, links to pre-defined Internet web sites that are known to be sources of digital content 12, and the like.

DRM SYSTEM 32 - Content Rendering, Part 1

Referring now to Fig. 5A, in one embodiment of the present invention, assuming the encrypted digital content 12 has been distributed to and received by a user and placed by the user on the computing device 14 in the form of a stored file, the user will attempt to render the digital content 12 by executing some variation on a render command (step 501). For example, such render command may be embodied as a request to 'play' or 'open' the digital content 12. In some computing environments, such as for example the "MICROSOFT WINDOWS" operating system, distributed by MICROSOFT Corporation of Redmond, Washington, such play or open command may be as simple as 'clicking' on an icon representative of the digital content 12. Of course, other embodiments of such render command may be employed without departing from the spirit and scope of the present invention. In general, such render command may be considered to be executed whenever a user directs that a file having digital content 12 be opened, run, executed, and/or the like.

Importantly, and in addition, such render command may be embodied as a request to copy the digital content 12 to another form, such as to a printed form, a visual form, an audio form, etc. As should be understood, the same digital content 12 may be rendered in one form, such as on a computer screen, and then in another form, such as a printed document. In the present invention, each type of rendering is performed only if the user has the right to do so, as will be explained below.

In one embodiment of the present invention, the digital content 12 is in

the form of a digital file having a file name ending with an extension, and the computing device 14 can determine based on such extension to start a particular kind of rendering application 34. For example, if the file name extension indicates that the digital content 12 is a text file, the rendering application 34 is some form of word processor such as the "MICROSOFT WORD", distributed by MICROSOFT Corporation of Redmond, Washington. Likewise, if the file name extension indicates that the digital content 12 is an audio, video, and/or multimedia file, the rendering application 34 is some form of multimedia player, such as "MICROSOFT MEDIA PLAYER", also distributed by MICROSOFT Corporation of Redmond, Washington.

Of course, other methods of determining a rendering application may be employed without departing from the spirit and scope of the present invention. As but one example, the digital content 12 may contain meta-data in an un-encrypted form (i.e., the aforementioned header information). where the meta-data includes information on the type of rendering application 34 necessary to render such digital content 12.

Preferably, such rendering application 34 examines the digital content 12 associated with the file name and determines whether such digital content 12 is encrypted in a rights-protected form (steps 503. 505). If not protected. the digital content 12 may be rendered without further ado (step 507). If protected. the rendering application 34 determines from the encrypted digital content 12 that the DRM system 32 is necessary to play such digital content 12. Accordingly, such rendering application 34 directs the user's computing device 14 to run the DRM system 32 thereon (step 509). Such rendering application 34 then calls such DRM system 32 to decrypt the digital content 12 (step 511). As will be discussed in more detail below, the DRM system 32 in fact decrypts the digital content 12 only if the user has a valid license 16 for such digital content 12 and the right to play the digital content 12 according to the license rules in the valid license 16. Preferably. once the DRM system 32 has been called by the rendering application 34, such DRM system 32 assumes control from the rendering application 34. at least for purposes of determining whether

the user has a right to play such digital content 12 (step 513).

DRM System 32 Components

In one embodiment of the present invention, and referring again to Fig. 4, the DRM system 32 includes a license evaluator 36, the black box 30, a license store 38, and a state store 40.

DRM System 32 Components - License Evaluator 36

The license evaluator 36 locates one or more licenses 16 that correspond to the requested digital content 12, determines whether such licenses 16 are valid, reviews the license rules in such valid licenses 16, and determines based on the reviewed license rules whether the requesting user has the right to render the requested digital content 12 in the manner sought, among other things. As should be understood, the license evaluator 36 is a trusted component in the DRM system 32. In the present disclosure, to be 'trusted' means that the license server 24 (or any other trusting element) is satisfied that the trusted element will carry out the wishes of the owner of the digital content 12 according to the rights description in the license 16, and that a user cannot easily alter such trusted element for any purpose, nefarious or otherwise.

The license evaluator 36 has to be trusted in order to ensure that such license evaluator 36 will in fact evaluate a license 16 properly, and to ensure that such license evaluator 36 has not been adulterated or otherwise modified by a user for the purpose of bypassing actual evaluation of a license 16. Accordingly, the license evaluator 36 is run in a protected or shrouded environment such that the user is denied access to such license evaluator 36. Other protective measures may of course be employed in connection with the license evaluator 36 without departing from the spirit and scope of the present invention.

DRM System 32 Components - Black Box 30

Primarily, and as was discussed above, the black box 30 performs encryption and decryption functions in the DRM system 32. In particular, the black box 30 works in conjunction with the license evaluator 36 to decrypt and encrypt certain information as part of the license evaluation function. In addition, once the

license evaluator 36 determines that a user does in fact have the right to render the requested digital content 12 in the manner sought, the black box 30 is provided with a decryption key (KD) for such digital content 12, and performs the function of decrypting such digital content 12 based on such decryption key (KD).

The black box 30 is also a trusted component in the DRM system 32. In particular, the license server 24 must trust that the black box 30 will perform the decryption function only in accordance with the license rules in the license 16, and also trust that such black box 30 will not operate should it become adulterated or otherwise modified by a user for the nefarious purpose of bypassing actual evaluation of a license 16. Accordingly, the black box 30 is also run in a protected or shrouded environment such that the user is denied access to such black box 30. Again, other protective measures may be employed in connection with the black box 30 without departing from the spirit and scope of the present invention. Preferably, and like the content server 22 and license server 24, the black box 30 in the DRM system 32 has a unique public / private key pair (PU-BB, PR-BB) that is employed as part of the process of evaluating the license 16 and obtaining a decryption key (KD) for decrypting the digital content 12, as will be described in more detail below.

DRM System 32 Components - License Store 38

The license store 38 stores licenses 16 received by the DRM system 32 for corresponding digital content 12. The license store 38 itself need not be trusted since the license store 38 merely stores licenses 16, each of which already has trust components built thereinto, as will be described below. In one embodiment of the present invention, the license store 38 is merely a sub-directory of a drive such as a hard disk drive or a network drive. However, the license store 38 may be embodied in any other form without departing from the spirit and scope of the present invention, so long as such license store 38 performs the function of storing licenses 16 in a location relatively convenient to the DRM system 32.

DRM System 32 Components - State Store 40

The state store 40 performs the function of maintaining state

information corresponding to licenses 16 presently or formerly in the license store 38. Such state information is created by the DRM system 32 and stored in the state store 40 as necessary. For example, if a particular license 16 only allows a pre-determined number of renderings of a piece of corresponding digital content 12, the state store 40 maintains state information on how many renderings have in fact taken place in connection with such license 16. The state store 40 continues to maintain state information on licenses 16 that are no longer in the license store 38 to avoid the situation where it would otherwise be advantageous to delete a license 16 from the license store 38 and then obtain an identical license 16 in an attempt to delete the corresponding state information from the state store 40.

The state store 40 also has to be trusted in order to ensure that the information stored therein is not reset to a state more favorable to a user. Accordingly, the state store 40 is likewise run in a protected or shrouded environment such that the user is denied access to such state store 40. Once again, other protective measures may of course be employed in connection with the state store 40 without departing from the spirit and scope of the present invention. For example, the state store 40 may be stored by the DRM system 32 on the computing device 14 in an encrypted form.

DRM SYSTEM 32 - Content Rendering, Part 2

Referring again to Fig. 5A, and again discussing content rendering in one embodiment of the present invention, once the DRM system 32 has assumed control from the calling rendering application 34, such DRM system 32 then begins the process of determining whether the user has a right to render the requested digital content 12 in the manner sought. In particular, the DRM system 32 either locates a valid, enabling license 16 in the license store (steps 515, 517) or attempts to acquire a valid, enabling license 16 from the license server 24 (i.e. performs the license acquisition function as discussed below and as shown in Fig. 7).

As a first step, and referring now to Fig. 6, the license evaluator 36 of such DRM system 32 checks the license store 38 for the presence of one or more received licenses 16 that correspond to the digital content 12 (step 601). Typically, the

license 16 is in the form of a digital file, as will be discussed below, although it will be recognized that the license 16 may also be in other forms without departing from the spirit and scope of the present invention. Typically, the user will receive the digital content 12 without such license 16, although it will likewise be recognized that the digital content 12 may be received with a corresponding license 16 without departing from the spirit and scope of the present invention.

As was discussed above in connection with Fig. 3, each piece of digital content 12 is in a package 12p with a content ID (or package ID) identifying such digital content 12 (or package 12p), and a key ID identifying the decryption key (KD) that will decrypt the encrypted digital content 12. Preferably, the content ID (or package ID) and the key ID are in an un-encrypted form. Accordingly, and in particular, based on the content ID of the digital content 12, the license evaluator 36 looks for any license 16 in the license store 38 that contains an identification of applicability to such content ID. Note that multiple such licenses 16 may be found, especially if the owner of the digital content 12 has specified several different kinds of licenses 16 for such digital content 12, and the user has obtained multiple ones of such licenses 16. If in fact the license evaluator 36 does not find in the license store 38 any license 16 corresponding to the requested digital content 12, the DRM system 32 may then perform the function of license acquisition (step 519 of Fig. 5), to be described below.

Assume now that the DRM system 32 has been requested to render a piece of digital content 12, and one or more licenses 16 corresponding thereto are present in the license store 38. In one embodiment of the present invention, then, the license evaluator 36 of the DRM system 32 proceeds to determine for each such license 16 whether such license 16 itself is valid (steps 603 and 605 of Fig. 6). Preferably, and in particular, each license 16 includes a digital signature 26 based on the content 28 of the license 16. As should be understood, the digital signature 26 will not match the license 16 if the content 28 has been adulterated or otherwise modified. Thus, the license evaluator 36 can determine based on the digital signature 26 whether the

content 28 is in the form that it was received from the license server 24 (i.e., is valid). If no valid license 16 is found in the license store 38, the DRM system 32 may then perform the license acquisition function described below to obtain such a valid license 16.

Assuming that one or more valid licenses 16 are found, for each valid license 16, the license evaluator 36 of the DRM system 32 next determines whether such valid license 16 gives the user the right to render the corresponding digital content 12 in the manner desired (i.e., is enabling) (steps 607 and 609). In particular, the license evaluator 36 determines whether the requesting user has the right to play the requested digital content 12 based on the rights description in each license 16 and based on what the user is attempting to do with the digital content 12. For example, such rights description may allow the user to render the digital content 12 into a sound, but not into a decrypted digital copy.

As should be understood, the rights description in each license 16 specifies whether the user has rights to play the digital content 12 based on any of several factors, including who the user is, where the user is located, what type of computing device 14 the user is using, what rendering application 34 is calling the DRM system 32, the date, the time, etc. In addition, the rights description may limit the license 16 to a pre-determined number of plays, or pre-determined play time, for example. In such case, the DRM system 32 must refer to any state information with regard to the license 16, (i.e., how many times the digital content 12 has been rendered, the total amount of time the digital content 12 has been rendered, etc.), where such state information is stored in the state store 40 of the DRM system 32 on the user's computing device 14.

Accordingly, the license evaluator 36 of the DRM system 32 reviews the rights description of each valid license 16 to determine whether such valid license 16 confers the rights sought to the user. In doing so, the license evaluator 36 may have to refer to other data local to the user's computing device 14 to perform a determination of whether the user has the rights sought. As seen in Fig. 4, such data

may include an identification 42 of the user's computing device (machine) 14 and particular aspects thereof, an identification 44 of the user and particular aspects thereof, an identification of the rendering application 34 and particular aspects thereof, a system clock 46, and the like. If no valid license 16 is found that provides the user with the right to render the digital content 12 in the manner sought, the DRM system 32 may then perform the license acquisition function described below to obtain such a license 16, if in fact such a license 16 is obtainable.

Of course, in some instances the user cannot obtain the right to render the digital content 12 in the manner requested, because the content owner of such digital content 12 has in effect directed that such right not be granted. For example, the content owner of such digital content 12 may have directed that no license 16 be granted to allow a user to print a text document, or to copy a multimedia presentation into an un-encrypted form. In one embodiment of the present invention, the digital content 12 includes data on what rights are available upon purchase of a license 16, and types of licenses 16 available. However, it will be recognized that the content owner of a piece of digital content 12 may at any time change the rights currently available for such digital content 12 by changing the licenses 16 available for such digital content 12.

DRM SYSTEM 32 - License Acquisition

Referring now to Fig. 7, if in fact the license evaluator 36 does not find in the license store 38 any valid, enabling license 16 corresponding to the requested digital content 12, the DRM system 32 may then perform the function of license acquisition. As shown in Fig. 3, each piece of digital content 12 is packaged with information in an un-encrypted form regarding how to obtain a license 16 for rendering such digital content 12 (i.e., license acquisition information).

In one embodiment of the present invention, such license acquisition information may include (among other things) types of licenses 16 available, and one or more Internet web sites or other site information at which one or more appropriate license servers 24 may be accessed, where each such license server 24 is in fact capable

of issuing a license 16 corresponding to the digital content 12. Of course, the license 16 may be obtained in other manners without departing from the spirit and scope of the present invention. For example, the license 16 may be obtained from a license server 24 at an electronic bulletin board, or even in person or via regular mail in the form of a file on a magnetic or optical disk or the like.

Assuming that the location for obtaining a license 16 is in fact a license server 24 on a network, the license evaluator 36 then establishes a network connection to such license server 24 based on the web site or other site information, and then sends a request for a license 16 from such connected license server 24 (steps 701, 703). In particular, once the DRM system 32 has contacted the license server 24, such DRM system 32 transmits appropriate license request information 36 to such license server 24. In one embodiment of the present invention, such license 16 request information 36 may include:

- the public key of the black box 30 of the DRM system 32 (PU-BB);
- the version number of the black box 30 of the DRM system 32;
- a certificate with a digital signature from a certifying authority certifying the black box 30 (where the certificate may in fact include the aforementioned public key and version number of the black box 30);
- the content ID (or package ID) that identifies the digital content 12 (or package 12p);
- the key ID that identifies the decryption key (KD) for decrypting the digital content 12;
- the type of license 16 requested (if in fact multiple types are available);
- the type of rendering application 34 that requested rendering of the digital content 12;

and/or the like, among other things. Of course, greater or lessor amounts of license 16 request information 36 may be transmitted to the license server 24 by the DRM system

32 without departing from the spirit and scope of the present invention. For example, information on the type of rendering application 34 may not be necessary, while additional information about the user and/or the user's computing device 14 may be necessary.

Once the license server 24 has received the license 16 request information 36 from the DRM system 32, the license server 24 may then perform several checks for trust / authentication and for other purposes. In one embodiment of the present invention, such license server 24 checks the certificate with the digital signature of the certifying authority to determine whether such has been adulterated or otherwise modified (steps 705, 707). If so, the license server 24 refuses to grant any license 16 based on the request information 36. The license server 24 may also maintain a list of known 'bad' users and/or user's computing devices 14, and may refuse to grant any license 16 based on a request from any such bad user and/or bad user's computing device 14 on the list. Such 'bad' list may be compiled in any appropriate manner without departing from the spirit and scope of the present invention.

Based on the received request and the information associated therewith, and particularly based on the content ID (or package ID) in the license request information, the license server 24 can interrogate the content-key database 20 (Fig. 1) and locate a record corresponding to the digital content 12 (or package 12p) that is the basis of the request. As was discussed above, such record contains the decryption key (KD), key ID, and content ID for such digital content 12. In addition, such record may contain license data regarding the types of licenses 16 to be issued for the digital content 12 and the terms and conditions for each type of license 16. Alternatively, such record may include a pointer, link, or reference to a location having such additional information.

As mentioned above, multiple types of licenses 16 may be available. For example, for a relatively small license fee, a license 16 allowing a limited number of renderings may be available. For a relatively greater license fee, a license 16

allowing unlimited renderings until an expiration date may be available. For a still greater license fee, a license 16 allowing unlimited renderings without any expiration date may be available. Practically any type of license 16 having any kind of license terms may be devised and issued by the license server 24 without departing from the spirit and scope of the present invention.

In one embodiment of the present invention, the request for a license 16 is accomplished with the aid of a web page or the like as transmitted from the license server 24 to the user's computing device 14. Preferably, such web page includes information on all types of licenses 16 available from the license server 24 for the digital content 12 that is the basis of the license 16 request.

In one embodiment of the present invention, prior to issuing a license 16, the license server 24 checks the version number of the black box 30 to determine whether such black box 30 is relatively current (steps 709, 711). As should be understood, the black box 30 is intended to be secure and protected from attacks from a user with nefarious purposes (i.e., to improperly render digital content 12 without a license 16, or outside the terms of a corresponding license 16). However, it is to be recognized that no system and no software device is in fact totally secure from such an attack.

As should also be understood, if the black box 30 is relatively current, i.e., has been obtained or updated relatively recently, it is less likely that such black box 30 has been successfully attacked by such a nefarious user. Preferably, and as a matter of trust, if the license server 24 receives a license request with request information 36 including a black box 30 version number that is not relatively current, such license server 24 refuses to issue the requested license 16 until the corresponding black box 30 is upgraded to a current version, as will be described below. Put simply, the license server 24 will not trust such black box 30 unless such black box 30 is relatively current.

In the context of the black box 30 of the present invention, the term 'current' or 'relatively current' may have any appropriate meaning without departing

from the spirit and scope of the present invention. consistent with the function of providing trust in the black box 30 based on the age or use thereof. For example, 'current' may be defined according to age (i.e., less than one month old). As an alternative example, 'current' may be defined based on a number of times that the black box 30 has decrypted digital content 12 (i.e., less than 200 instances of decryption). Moreover, 'current' may be based on policy as set by each license server 24, where one license server 24 may define 'current' differently from another license server 24, and a license server 24 may further define 'current' differently depending on the digital content 12 for which a license 16 is requested. or depending on the type of license 16 requested, among other things.

Assuming that the license server 24 is satisfied from the version number of a black box 30 or other indicia thereof that such black box 30 is current, the license server 24 then proceeds to negotiate terms and conditions for the license 16 with the user (step 713). Alternatively, the license server 24 negotiates the license 16 with the user, then satisfies itself from the version number of the black box 30 that such black box 30 is current (i.e., performs step 713, then step 711). Of course, the amount of negotiation varies depending on the type of license 16 to be issued, and other factors. For example, if the license server 24 is merely issuing a paid-up unlimited use license 16, very little need be negotiated. On the other hand, if the license 16 is to be based on such items as varying values, sliding scales, break points, and other details, such items and details may need to be worked out between the license server 24 and the user before the license 16 can be issued.

As should be understood, depending on the circumstances, the license negotiation may require that the user provide further information to the license server 24 (for example, information on the user, the user's computing device 14, etc.). Importantly, the license negotiation may also require that the user and the license server 24 determine a mutually acceptable payment instrument (a credit account, a debit account, a mailed check, etc.) and/or payment method (paid-up immediately, spread over a period of time, etc.), among other things.

Once all the terms of the license 16 have been negotiated and agreed to by both the license server 24 and user (step 715). a digital license 16 is generated by the license server 24 (step 719), where such generated license 16 is based at least in part on the license request, the black box 30 public key (PU-BB), and the decryption key (KD) for the digital content 12 that is the basis of the request as obtained from the content-key database 20. In one embodiment of the present invention, and as seen in Fig. 8, the generated license 16 includes:

- the content ID of the digital content 12 to which the license 16 applies;
- a Digital Rights License (DRL) 48 (i.e., the rights description or actual terms and conditions of the license 16 written in a predetermined form that the license evaluator 36 can interrogate), perhaps encrypted with the decryption key (KD) (i.e., KD (DRL));
- the decryption key (KD) for the digital content 12 encrypted with the black box 30 public key (PU-BB) as receive in the license request (i.e.,(PU-BB (KD));
- a digital signature from the license server 24 (without any attached certificate) based on (KD (DRL)) and (PU-BB (KD)) and encrypted with the license server 24 private key (i.e., (S (PR-LS))); and
- the certificate that the license server 24 obtained previously from the content server 22, such certificate indicating that the license server 24 has the authority from the content server 22 to issue the license 16 (i.e., (CERT (PU-LS) S (PR-CS))).

As should be understood, the aforementioned elements and perhaps others are packaged into a digital file or some other appropriate form. As should also be understood, if the DRL 48 or (PU-BB (KD)) in the license 16 should become adulterated or otherwise modified, the digital signature (S (PR-LS)) in the license 16 will not match and therefore will not validate such license 16. For this reason, the DRL 48 need not necessarily be in an encrypted form (i.e., (KD(DRL)) as mentioned

above), although such encrypted form may in some instances be desirable and therefore may be employed without departing from the spirit and scope of the present invention.

Once the digital license 16 has been prepared. such license 16 is then issued to the requestor (i.e., the DRM system 32 on the user's computing device 14) (step 719 of Fig. 7). Preferably, the license 16 is transmitted over the same path through which the request therefor was made (i.e., the Internet or another network), although another path may be employed without departing from the spirit and scope of the present invention. Upon receipt, the requesting DRM system 32 preferably automatically places the received digital license 16 in the license store 38 (step 721).

It is to be understood that a user's computing device 14 may on occasion malfunction, and licenses 16 stored in the license store 38 of the DRM system 32 on such user's computing device 14 may become irretrievably lost. Accordingly, it is preferable that the license server 24 maintain a database 50 of issued licenses 16 (Fig. 1), and that such license server 24 provide a user with a copy or re-issue (hereinafter 're-issue') of an issued license 16 if the user is in fact entitled to such re-issue. In the aforementioned case where licenses 16 are irretrievably lost, it is also likely the case that state information stored in the state store 40 and corresponding to such licenses 16 is also lost. Such lost state information should be taken into account when re-issuing a license 16. For example, a fixed number of renderings license 16 might legitimately be re-issued in a pro-rated form after a relatively short period of time, and not re-issued at all after a relatively longer period of time.

DRM SYSTEM 32 - Installation/Upgrade of Black Box 30

As was discussed above, as part of the function of acquiring a license 16, the license server 24 may deny a request for a license 16 from a user if the user's computing device 14 has a DRM system 32 with a black box 30 that is not relatively current, i.e., has a relatively old version number. In such case, it is preferable that the black box 30 of such DRM system 32 be upgraded so that the license acquisition function can then proceed. Of course, the black box 30 may be upgraded at other times

without departing from the spirit and scope of the present invention.

Preferably, as part of the process of installing the DRM system 32 on a user's computing device 14, a non-unique 'lite' version of a black box 30 is provided. Such 'lite' black box 30 is then upgraded to a unique regular version prior to rendering a piece of digital content 12. As should be understood, if each black box 30 in each DRM system 32 is unique, a security breach into one black box 30 cannot easily be replicated with any other black box 30.

Referring now to Fig. 9, the DRM system 32 obtains the unique black box 30 by requesting same from a black box server 26 or the like (as was discussed above and as shown in Fig. 1) (step 901). Typically, such request is made by way of the Internet, although other means of access may be employed without departing from the spirit and scope of the present invention. For example, the connection to a black box server 26 may be a direct connection, either locally or remotely. An upgrade from one unique non-lite black box 30 to another unique non-lite black box 30 may also be requested by the DRM system 32 at any time, such as for example a time when a license server 24 deems the black box 30 not current, as was discussed above.

Thereafter, the black box server 26 generates a new unique black box 30 (step 903). As seen in Fig. 3, each new black box 30 is provided with a version number and a certificate with a digital signature from a certifying authority. As was discussed above in connection with the license acquisition function, the version number of the black box 30 indicates the relative age and/or use thereof. The certificate with the digital signature from the certifying authority, also discussed above in connection with the license acquisition function, is a proffer or vouching mechanism from the certifying authority that a license server 24 should trust the black box 30. Of course, the license server 24 must trust the certifying authority to issue such a certificate for a black box 30 that is in fact trustworthy. It may be the case, in fact, that the license server 24 does not trust a particular certifying authority, and refuses to honor any certificate issued by such certifying authority. Trust may not occur, for example, if a particular certifying authority is found to be engaging in a pattern of

improperly issuing certificates.

Preferably, and as was discussed above, the black box server 26 includes a new unique public / private key pair (PU-BB, PR-BB) with the newly generated unique black box 30 (step 903 of Fig. 9). Preferably, the private key for the black box 30 (PR-BB) is accessible only to such black box 30, and is hidden from and inaccessible by the remainder of the world, including the computing device 14 having the DRM system 32 with such black box 30, and the user thereof.

Most any hiding scheme may be employed without departing from the spirit and scope of the present invention, so long as such hiding scheme in fact performs the function of hiding the private key (PR-BB) from the world. As but one example, the private key (PR-BB) may be split into several sub-components, and each sub-component may be encrypted uniquely and stored in a different location. In such a situation, it is preferable that such sub-components are never assembled in full to produce the entire private key (PR-BB).

In one embodiment of the present invention, such private key (PR-BB) is encrypted according to code-based encryption techniques. In particular, in such embodiment, the actual software code of the black box 30 (or other software code) is employed as encrypting key(s). Accordingly, if the code of the black box 30 (or the other software code) becomes adulterated or otherwise modified, for example by a user with nefarious purposes, such private key (PR-BB) cannot be decrypted.

Although each new black box 30 is delivered with a new public / private key pair (PU-BB, PR-BB), such new black box 30 is also preferably given access to old public / private key pairs from old black boxes 30 previously delivered to the DRM system 32 on the user's computing device 14 (step 905). Accordingly, the upgraded black box 30 can still employ the old key pairs to access older digital content 12 and older corresponding licenses 16 that were generated according to such old key pairs, as will be discussed in more detail below.

Preferably, the upgraded black box 30 delivered by the black box server 26 is tightly tied to or associated with the user's computing device 14. Accordingly,

the upgraded black box 30 cannot be operably transferred among multiple computing devices 14 for nefarious purposes or otherwise. In one embodiment of the present invention, as part of the request for the black box 30 (step 901) the DRM system 32 provides hardware information unique to such DRM system 32 and/or unique to the user's computing device 14 to the black box server 26, and the black box server 26 generates a black box 30 for the DRM system 32 based in part on such provided hardware information. Such generated upgraded black box 30 is then delivered to and installed in the DRM system 32 on the user's computing device 14 (steps 907, 909). If the upgraded black box 30 is then somehow transferred to another computing device 14, the transferred black box 30 recognizes that it is not intended for such other computing device 14, and does not allow any requested rendering to proceed on such other computing device 14.

Once the new black box 30 is installed in the DRM system 32, such DRM system 32 can proceed with a license acquisition function or with any other function.

DRM SYSTEM 32 - Content Rendering, Part 3

Referring now to Fig. 5B, and assuming, now, that the license evaluator 36 has found at least one valid license 16 and that at least one of such valid licenses 16 provides the user with the rights necessary to render the corresponding digital content 12 in the manner sought (i.e., is enabling), the license evaluator 36 then selects one of such licenses 16 for further use (step 519). Specifically, to render the requested digital content 12, the license evaluator 36 and the black box 30 in combination obtain the decryption key (KD) from such license 16, and the black box 30 employs such decryption key (KD) to decrypt the digital content 12. In one embodiment of the present invention, and as was discussed above, the decryption key (KD) as obtained from the license 16 is encrypted with the black box 30 public key (PU-BB(KD)), and the black box 30 decrypts such encrypted decryption key with its private key (PR-BB) to produce the decryption key (KD) (steps 521, 523). However, other methods of obtaining the decryption key (KD) for the digital content 12 may be employed without

departing from the spirit and scope of the present invention.

Once the black box 30 has the decryption key (KD) for the digital content 12 and permission from the license evaluator 36 to render the digital content 12, control may be returned to the rendering application 34 (steps 525, 527). In one embodiment of the present invention, the rendering application 34 then calls the DRM system 32 / black box 30 and directs at least a portion of the encrypted digital content 12 to the black box 30 for decryption according to the decryption key (KD) (step 529). The black box 30 decrypts the digital content 12 based upon the decryption key (KD) for the digital content 12, and then the black box 30 returns the decrypted digital content 12 to the rendering application 34 for actual rendering (steps 533, 535). The rendering application 34 may either send a portion of the encrypted digital content 12 or the entire digital content 12 to the black box 30 for decryption based on the decryption key (KD) for such digital content 12 without departing from the spirit and scope of the present invention.

Preferably, when the rendering application 34 sends digital content 12 to the black box 30 for decryption, the black box 30 and/or the DRM system 32 authenticates such rendering application 34 to ensure that it is in fact the same rendering application 34 that initially requested the DRM system 32 to run (step 531). Otherwise, the potential exists that rendering approval may be obtained improperly by basing the rendering request on one type of rendering application 34 and in fact rendering with another type of rendering application 34. Assuming the authentication is successful and the digital content 12 is decrypted by the black box 30, the rendering application 34 may then render the decrypted digital content 12 (steps 533, 535).

Sequence of Key Transactions

Referring now to Fig. 10, in one embodiment of the present invention, a sequence of key transactions is performed to obtain the decryption key (KD) and evaluate a license 16 for a requested piece of digital content 12 (i.e., to perform steps 515-523 of Figs. 5A and 5B). Mainly, in such sequence, the DRM system 32 obtains the decryption key (KD) from the license 16, uses information obtained from the

license 16 and the digital content 12 to authenticate or ensure the validity of both, and then determines whether the license 16 in fact provides the right to render the digital content 12 in the manner sought. If so, the digital content 12 may be rendered.

Bearing in mind that each license 16 for the digital content 12, as seen in Fig. 8, includes:

- the content ID of the digital content 12 to which the license 16 applies;
- the Digital Rights License (DRL) 48, perhaps encrypted with the decryption key (KD) (i.e., KD (DRL));
- the decryption key (KD) for the digital content 12 encrypted with the black box 30 public key (PU-BB) (i.e.,(PU-BB (KD));
- the digital signature from the license server 24 based on (KD (DRL)) and (PU-BB (KD)) and encrypted with the license server 24 private key (i.e., (S (PR-LS))); and
- the certificate that the license server 24 obtained previously from the content server 22 (i.e., (CERT (PU-LS) S (PR-CS))),

and also bearing in mind that the package 12p having the digital content 12, as seen in Fig. 3, includes:

- the content ID of such digital content 12;
- the digital content 12 encrypted by KD (i.e., (KD(CONTENT)));
- a license acquisition script that is not encrypted; and
- the key KD encrypting the content server 22 public key (PU-CS), signed by the content server 22 private key (PR-CS) (i.e., (KD (PU-CS) S (PR-CS))),

in one embodiment of the present invention, the specific sequence of key transactions that are performed with regard to a specific one of the licenses 16 for the digital content 12 is as follows:

1. Based on (PU-BB (KD)) from the license 16, the black box 30 of the DRM system 32 on the user's computing device 14 applies its private key (PR-

BB) to obtain (KD) (step 1001). (PR-BB (PU-BB (KD)) = (KD)). Note, importantly, that the black box 30 could then proceed to employ KD to decrypt the digital content 12 without any further ado. However, and also importantly, the license server 24 trusts the black box 30 not to do so. Such trust was established at the time such license server 24 issued the license 16 based on the certificate from the certifying authority vouching for the trustworthiness of such black box 30. Accordingly, despite the black box 30 obtaining the decryption key (KD) as an initial step rather than a final step, the DRM system 32 continues to perform all license 16 validation and evaluation functions, as described below.

2. Based on (KD (PU-CS) S (PR-CS)) from the digital content 12, the black box 30 applies the newly obtained decryption key (KD) to obtain (PU-CS) (step 1003). (KD (KD (PU-CS)) = (PU-CS)). Additionally, the black box 30 can apply (PU-CS) as against the signature (S (PR-CS)) to satisfy itself that such signature and such digital content 12 / package 12p is valid (step 1005). If not valid, the process is halted and access to the digital content 12 is denied.

3. Based on (CERT (PU-LS) S (PR-CS)) from the license 16, the black box 30 applies the newly obtained content server 22 public key (PU-CS) to satisfy itself that the certificate is valid (step 1007), signifying that the license server 24 that issued the license 16 had the authority from the content server 22 to do so, and then examines the certificate contents to obtain (PU-LS) (step 1009). If not valid, the process is halted and access to the digital content 12 based on the license 16 is denied.

4. Based on (S (PR-LS)) from the license 16, the black box 30 applies the newly obtained license server 24 public key (PU-LS) to satisfy itself that the license 16 is valid (step 1011). If not valid, the process is halted and access to the digital content 12 based on the license 16 is denied.

5. Assuming all validation steps are successful, and that the DRL 48 in the license 16 is in fact encrypted with the decryption key (KD), the license evaluator 36 then applies the already-obtained decryption key (KD) to (KD(DRL)) as obtained from the license 16 to obtain the license terms from the license 16 (i.e., the

DRL 48) (step 1013). Of course, if the DRL 48 in the license 16 is not in fact encrypted with the decryption key (KD), step 1013 may be omitted. The license evaluator 36 then evaluates / interrogates the DRL 48 and determines whether the user's computing device 14 has the right based on the DRL 48 in the license 16 to render the corresponding digital content 12 in the manner sought (i.e., whether the DRL 48 is enabling) (step 1015). If the license evaluator 36 determines that such right does not exist, the process is halted and access to the digital content 12 based on the license 16 is denied.

6. Finally, assuming evaluation of the license 16 results in a positive determination that the user's computing device 14 has the right based on the DRL 48 terms to render the corresponding digital content 12 in the manner sought, the license evaluator 36 informs the black box 30 that such black box 30 can render the corresponding digital content 12 according to the decryption key (KD). The black box 30 thereafter applies the decryption key (KD) to decrypt the digital content 12 from the package 12p (i.e., (KD(KD(CONTENT)) = (CONTENT)) (step 1017).

It is important to note that the above-specified series of steps represents an alternating or 'ping-ponging' between the license 16 and the digital content 12. Such ping-ponging ensures that the digital content 12 is tightly bound to the license 16, in that the validation and evaluation process can only occur if both the digital content 12 and license 16 are present in a properly issued and valid form. In addition, since the same decryption key (KD) is needed to get the content server 22 public key (PU-CS) from the license 16 and the digital content 12 from the package 12p in a decrypted form (and perhaps the license terms (DRL 48) from the license 16 in a decrypted form), such items are also tightly bound. Signature validation also ensures that the digital content 12 and the license 16 are in the same form as issued from the content server 22 and the license server 24, respectively. Accordingly, it is difficult if not impossible to decrypt the digital content 12 by bypassing the license server 24, and also difficult if not impossible to alter and then decrypt the digital content 12 or the license 16.

In one embodiment of the present invention, signature verification, and especially signature verification of the license 16, is alternately performed as follows. Rather than having a signature encrypted by the private key of the license server 16 (PR-LS), as is seen in Fig. 8, each license 16 has a signature encrypted by a private root key (PR-R) (not shown), where the black box 30 of each DRM system 32 includes a public root key (PU-R) (also not shown) corresponding to the private root key (PR-R). The private root key (PR-R) is known only to a root entity, and a license server 24 can only issue licenses 16 if such license server 24 has arranged with the root entity to issue licenses 16.

In particular, in such embodiment:

1. the license server 24 provides its public key (PU-LS) to the root entity;
2. the root entity returns the license server public key (PU-LS) to such license server 24 encrypted with the private root key (PR-R) (i.e., (CERT (PU-LS) S (PR-R))); and
3. the license server 24 then issues a license 16 with a signature encrypted with the license server private key (S (PR-LS)), and also attaches to the license the certificate from the root entity (CERT (PU-LS) S (PR-R)).

For a DRM system 18 to validate such issued license 16, then, the DRM system 18:

1. applies the public root key (PU-R) to the attached certificate (CERT (PU-LS) S (PR-R)) to obtain the license server public key (PU-LS); and
2. applies the obtained license server public key (PU-LS) to the signature of the license 16 (S (PR-LS).

Importantly, it should be recognized that just as the root entity gave the license server 24 permission to issue licenses 16 by providing the certificate (CERT (PU-LS) S (PR-R)) to such license server 24, such license server 24 can provide a

similar certificate to a second license server 24 (i.e.. (CERT (PU-LS2) S (PR-LS1)), thereby allowing the second license server to also issue licenses 16. As should now be evident, a license 16 issued by the second license server would include a first certificate (CERT (PU-LS1) S (PR-R)) and a second certificate (CERT (PU-LS2) S (PR-LS1)). Likewise, such license 16 is validated by following the chain through the first and second certificates. Of course, additional links in the chain may be added and traversed.

One advantage of the aforementioned signature verification process is that the root entity may periodically change the private root key (PR-R), thereby likewise periodically requiring each license server 24 to obtain a new certificate (CERT (PU-LS) S (PR-R)). Importantly, as a requirement for obtaining such new certificate, each license server may be required to upgrade itself. As with the black box 30, if a license server 24 is relatively current, i.e., has been upgraded relatively recently, it is less likely that license server 24 has been successfully attacked. Accordingly, as a matter of trust, each license server 24 is preferably required to be upgraded periodically via an appropriate upgrade trigger mechanism such as the signature verification process. Of course, other upgrade mechanisms may be employed without departing from the spirit and scope of the present invention.

Of course, if the private root key (PR-R) is changed, then the public root key (PU-R) in each DRM system 18 must also be changed. Such change may for example take place during a normal black box 30 upgrade, or in fact may require that a black box 30 upgrade take place. Although a changed public root key (PU-R) may potentially interfere with signature validation for an older license 16 issued based on an older private root key (PR-R), such interference may be minimized by requiring that an upgraded black box 30 remember all old public root keys (PU-R). Alternatively, such interference may be minimized by requiring signature verification for a license 16 only once, for example the first time such license 16 is evaluated by the license evaluator 36 of a DRM system 18. In such case, state information on whether signature verification has taken place should be compiled, and such state information

should be stored in the state store 40 of the DRM system 18.

Digital Rights License 48

In the present invention, the license evaluator 36 evaluates a Digital Rights License (DRL) 48 as the rights description or terms of a license 16 to determine if such DRL 48 allows rendering of a corresponding piece of digital content 12 in the manner sought. In one embodiment of the present invention, the DRL 48 may be written by a licensor (i.e., the content owner) in any DRL language.

As should be understood, there are a multitude of ways to specify a DRL 48. Accordingly, a high degree of flexibility must be allowed for in any DRL language. However, it is impractical to specify all aspects of a DRL 48 in a particular license language, and it is highly unlikely that the author of such a language can appreciate all possible licensing aspects that a particular digital licensor may desire. Moreover, a highly sophisticated license language may be unnecessary and even a hindrance for a licensor providing a relatively simple DRL 48. Nevertheless, a licensor should not be unnecessarily restricted in how to specify a DRL 48. At the same time, the license evaluator 36 should always be able to get answers from a DRL 48 regarding a number of specific license questions.

In the present invention, and referring now to Fig. 11, a DRL 48 can be specified in any license language, but includes a language identifier or tag 54. The license evaluator 36 evaluating the license 16, then, performs the preliminary step of reviewing the language tag 54 to identify such language, and then selects an appropriate license language engine 52 for accessing the license 16 in such identified language. As should be understood, such license language engine 52 must be present and accessible to the license evaluator 36. If not present, the language tag 54 and/or the DRL 48 preferably includes a location 56 (typically a web site) for obtaining such language engine 52.

Typically, the language engine 52 is in the form of an executable file or set of files that reside in a memory of the user's computing device 14, such as a hard drive. The language engine 52 assists the license evaluator 36 to directly interrogate

the DRL 48, the license evaluator 36 interrogates the DRL 48 indirectly via the language engine 48 acting as an intermediary, or the like. When executed, the language engine 52 runs in a work space in a memory of the user's computing device 14, such as RAM. However, any other form of language engine 52 may be employed without departing from the spirit and scope of the present invention.

Preferably, any language engine 52 and any DRL language supports at least a number of specific license questions that the license evaluator 36 expects to be answered by any DRL 48, as will be discussed below. Accordingly, the license evaluator 36 is not tied to any particular DRL language; a DRL 48 may be written in any appropriate DRL language; and a DRL 48 specified in a new license language can be employed by an existing license evaluator 36 by having such license evaluator 36 obtain a corresponding new language engine 52.

DRL Languages

Two examples of DRL languages, as embodied in respective DRLs 48, are provided below. The first, 'simple' DRL 48 is written in a DRL language that specifies license attributes, while the second 'script' DRL 48 is written in a DRL language that can perform functions according to the script specified in the DRL 48. While written in a DRL language, the meaning of each line of code should be apparent based on the linguistics thereof and/or on the attribute description chart that follows:

Simple DRL 48:

```
<LICENSE>
    <DATA>
        <NAME>Beastie Boy's Play</NAME>
        <ID>39384</ID>
        <DESCRIPTION>Play the song 3 times</DESCRIPTION>
        <TERMS></TERMS>
        <VALIDITY>
            <NOTBEFORE>19980102 23:20:14Z</NOTBEFORE>
            <NOTAFTER>19980102 23:20:14Z</NOTAFTER>
        </VALIDITY>
        <ISSUEDDATE>19980102 23:20:14Z</ISSUEDDATE>
        <LICENSORSITE>http://www.foo.com</LICENSORSITE>
```

```
<CONTENT>
        <NAME>Beastie Boy's</NAME>
        <ID>392</ID>
        <KEYID>39292</KEYID>
        <TYPE>MS Encrypted ASF 2.0</TTYPE>
</CONTENT>
<OWNER>
        <ID>939KDKD393KD</ID>
        <NAME>Universal</NAME>
        <PUBLICKEY></PUBLICKEY>
</OWNER>
<LICENSEE>
        <NAME>Arnold</NAME>
        <ID>939KDKD393KD</ID>
        <PUBLICKEY></PUBLICKEY>
</LICENSEE>
<PRINCIPAL TYPE='AND'>
        <PRINCIPAL TYPE='OR'>
                <PRINCIPAL>
                        <TYPE>x86Computer</TYPE>
                        <ID>3939292939d9e939</ID>
                        <NAME>Personal Computer</NAME>
                        <AUTHTYPE>Intel Authenticated Boot PC
                        SHA-1 DSA512</AUTHTYPE>
                        <AUTHDATA>29293939</AUTHDATA>
                </PRINCIPAL>
                <PRINCIPAL>
                        <TYPE>Application</TYPE>
                        <ID>2939495939292</ID>
                        <NAME>Window's Media Player</NAME>
                        <AUTHTYPE>Authenticode SHA-
                        1</AUTHTYPE>
                        <AUTHDATA>93939</AUTHDATA>
                </PRINCIPAL>
        </PRINCIPAL>
        <PRINCIPAL>
                <TYPE>Person</TYPE>
                <ID>39299482010</ID>
                <NAME>Arnold Blinn</NAME>
                <AUTHTYPE>Authenticate user</AUTHTYPE>
                <AUTHDATA>\\redmond\arnoldb</AUTHDATA>
        </PRINCIPAL>
</PRINCIPAL>
```

```
                <DRLTYPE>Simple</DRLTYPE>  [the language tag 54]
                <DRLDATA>
                        <START>19980102 23:20:14Z</START>
                        <END>19980102 23:20:14Z</END>
                        <COUNT>3</COUNT>
                        <ACTION>PLAY</ACTION>
                </DRLDATA>
                <ENABLINGBITS>aaaabbbbccccdddd</ENABLINGBITS>
        </DATA>
        <SIGNATURE>
        <SIGNERNAME>Universal</SIGNERNAME>
                <SIGNERID>9382ABK3939DKD</SIGNERID>
                <HASHALGORITHMID>MD5</HASHALGORITHMID>
                <SIGNALGORITHMID>RSA 128</SIGNALGORITHMID>
                <SIGNATURE>xxxyyyxxxyyyxxxyyy</SIGNATURE>
                <SIGNERPUBLICKEY></SIGNERPUBLICKEY>
                <CONTENTSIGNEDSIGNERPUBLICKEY></CONTENTSIGNEDSI
                GNERPUBLICKEY>
        </SIGNATURE>
</LICENSE>
```

Script DRL 48:

```
<LICENSE>
        <DATA>
                <NAME>Beastie Boy's Play</NAME>
                <ID>39384</ID>
                <DESCRIPTION>Play the song unlimited</DESCRIPTION>
                <TERMS></TERMS>
                <VALIDITY>
                        <NOTBEFORE>19980102 23:20:14Z</NOTBEFORE>
                        <NOTAFTER>19980102 23:20:14Z</NOTAFTER>
                </VALIDITY>
                <ISSUEDDATE>19980102 23:20:14Z</ISSUEDDATE>
                <LICENSORSITE>http://www.foo.com</LICENSORSITE>
                <CONTENT>
                        <NAME>Beastie Boy's</NAME
                        <ID>392</ID>
                        <KEYID>39292</KEYID>
                        <TYPE>MS Encrypted ASF 2.0</TTYPE>
                </CONTENT>
                <OWNER>
                        <ID>939KDKD393KD</ID>
```

```
                <NAME>Universal</NAME>
                <PUBLICKEY></PUBLICKEY>
            </OWNER>
            <LICENSEE>
                <NAME>Arnold</NAME>
                <ID>939KDKD393KD</ID>
                <PUBLICKEY></PUBLICKEY>
            </LICENSEE>
            <DRLTYPE>Script</DRLTYPE>    [the language tag 54]
            <DRLDATA>
                function on_enable(action. args) as boolean
                    result = False
                    if action = "PLAY" then
                        result = True
                    end if
                    on_action = False
                end function
                . . .
            </DRLDATA>
        </DATA>
        <SIGNATURE>
            <SIGNERNAME>Universal</SIGNERNAME>
            <SIGNERID>9382</SIGNERID>
            <SIGNERPUBLICKEY></SIGNERPUBLICKEY>
            <HASHID>MD5</HASHID>
            <SIGNID>RSA 128</SIGNID>
            <SIGNATURE>xxxyyyxxxyyyxxxyyy</SIGNATURE>
            <CONTENTSIGNEDSIGNERPUBLICKEY></CONTENTSIGNEDSI
            GNERPUBLICKEY>
        </SIGNATURE>
    </LICENSE>
```

In the two DRLs 48 specified above, the attributes listed have the following descriptions and data types:

Attribute	Description	Data Type
Id	ID of the license	GUID
Name	Name of the license	String
Content Id	ID of the content	GUID
Content Key Id	ID for the encryption key of the content	GUID
Content Name	Name of the content	String
Content Type	Type of the content	String

Owner Id	ID of the owner of the content	GUID
Owner Name	Name of the owner of the content	String
Owner Public Key	Public key for owner of content. This is a base-64 encoded public key for the owner of the content.	String
Licensee Id	Id of the person getting license. It may be null.	GUID
Licensee Name	Name of the person getting license. It may be null.	String
Licensee Public Key	Public key of the licensee. This is the base-64 encoded public key of the licensee. It may be null.	String
Description	Simple human readable description of the license	String
Terms	Legal terms of the license. This may be a pointer to a web page containing legal prose.	String
Validity Not After	Validity period of license expiration	Date
Validity Not Before	Validity period of license start	Date
Issued Date	Date the license was issued	Date
DRL Type	Type of the DRL. Example include "SIMPLE" or "SCRIPT"	String
DRL Data	Data specific to the DRL	String
Enabling Bits	These are the bits that enable access to the actual content. The interpretation of these bits is up to the application, but typically this will be the private key for decryption of the content. This data will be base-64 encoded. Note that these bits are encrypted using the public key of the individual machine.	String
Signer Id	ID of person signing license	GUID
Signer Name	Name of person signing license	String
Signer Public Key	Public key for person signing license. This is the base-64 encode public key for the signer.	String
Content Signed Signer Public Key	Public key for person signing the license that has been signed by the content server private key. The public key to verify this signature will be encrypted in the content. This is base-64 encoded.	String

Hash Alg Id	Algorithm used to generate hash. This is a string, such as "MD5".	String
Signature Alg Id	Algorithm used to generate signature. This is a string, such as "RSA 128".	String
Signature	Signature of the data. This is base-64 encoded data.	String

Methods

As was discussed above, it is preferable that any language engine 52 and any DRL language support at least a number of specific license questions that the digital license evaluator 36 expects to be answered by any DRL 48. Recognizing such supported questions may include any questions without departing from the spirit and scope of the present invention, and consistent with the terminology employed in the two DRL 48 examples above, in one embodiment of the present invention, such supported questions or 'methods' include 'access methods', 'DRL methods', and 'enabling use methods', as follows:

Access Methods

Access methods are used to query a DRL 48 for top-level attributes.

VARIANT QueryAttribute (BSTR key)

Valid keys include License.Name, License.Id, Content.Name, Content.Id, Content.Type, Owner.Name, Owner.Id, Owner.PublicKey, Licensee.Name, Licensee.Id, Licensee.PublicKey, Description, and Terms, each returning a BSTR variant; and Issued, Validity.Start and Validity.End, each returning a Date Variant.

DRL Methods

The implementation of the following DRL methods varies from DRL 48 to DRL 48. Many of the DRL methods contain a variant parameter labeled 'data' which is intended for communicating more advanced information with a DRL 48. It

is present largely for future expandability.

Boolean IsActivated(Variant data)

This method returns a Boolean indicating whether the DRL 48 / license 16 is activated. An example of an activated license 16 is a limited operation license 16 that upon first play is active for only 48 hours.

Activate(Variant data)

This method is used to activate a license 16. Once a license 16 is activated, it cannot be deactivated.

Variant QueryDRL(Variant data)

This method is used to communicate with a more advanced DRL 48. It is largely about future expandability of the DRL 48 feature set.

Variant GetExpires(BSTR action, Variant data)

This method returns the expiration date of a license 16 with regard to the passed-in action. If the return value is NULL, the license 16 is assumed to never expire or does not yet have an expiration date because it hasn't been activated. or the like.

Variant GetCount(BSTR action, Variant data)

This method returns the number of operations of the passed-in action that are left. If NULL is returned, the operation can be performed an unlimited number of times.

Boolean IsEnabled(BSTR action, Variant data)

This method indicates whether the license 16 supports the requested action at the present time.

Boolean IsSunk(BSTR action, Variant data)

This method indicates whether the license 16 has been paid for. A license 16 that is paid for up front would return TRUE, while a license 16 that is not paid for up front, such as a license 16 that collects payments as it is used, would return FALSE.

Enabling Use Methods.

These methods are employed to enable a license 16 for use in decrypting content.

Boolean Validate (BSTR key)

This method is used to validate a license 16. The passed-in key is the black box 30 public key (PU-BB) encrypted by the decryption key (KD) for the corresponding digital content 12 (i.e.,(KD(PU-BB))) for use in validation of the signature of the license 16. A return value of TRUE indicates that the license 16 is valid. A return value of FALSE indicates invalid.

int OpenLicense 16(BSTR action, BSTR key. Variant data)

This method is used to get ready to access the decrypted enabling bits. The passed-in key is (KD(PU-BB)) as described above. A return value of 0 indicates success. Other return values can be defined.

BSTR GetDecryptedEnablingBits (BSTR action. Variant data)

Variant GetDecryptedEnablingBitsAsBinary (BSTR action. Variant Data)

These methods are used to access the enabling bits in decrypted form. If this is not successful for any of a number of reasons, a null string or null variant is returned.

void CloseLicense 16 (BSTR action, Variant data)

This method is used to unlock access to the enabling bits for performing the passed-in action. If this is not successful for any of a number of reasons. a null string is returned.

Heuristics

As was discussed above, if multiple licenses 16 are present for the same piece of digital content 12, one of the licenses 16 must be chosen for further use. Using the above methods, the following heuristics could be implemented to make such choice. In particular, to perform an action (say "PLAY") on a piece of digital content 12, the following steps could be performed:

1. Get all licenses 16 that apply to the particular piece of digital content 12.
2. Eliminate each license 16 that does not enable the action by calling the IsEnabled function on such license 16.
3. Eliminate each license 16 that is not active by calling IsActivated on such license 16.
4. Eliminate each license 16 that is not paid for up front by calling IsSunk on such license 16.
5. If any license 16 is left, use it. Use an unlimited-number-of-plays license 16 before using a limited-number-of-plays license 16, especially if the unlimited-number-of-plays license 16 has an expiration date. At any time, the user should be allowed to select a specific license 16 that has already been acquired, even if the choice is not cost-effective. Accordingly, the user can select a license 16 based on criteria that are perhaps not apparent to the DRM system 32.
6. If there are no licenses 16 left, return status so indicating. The user would then be given the option of:

   using a license 16 that is not paid for up front, if available;

   activating a license 16, if available; and/or

   performing license acquisition from a license server 24.

CONCLUSION

The programming necessary to effectuate the processes performed in connection with the present invention is relatively straight-forward and should be

apparent to the relevant programming public. Accordingly, such programming is not attached hereto. Any particular programming, then, may be employed to effectuate the present invention without departing from the spirit and scope thereof.

In the foregoing description, it can be seen that the present invention comprises a new and useful enforcement architecture 10 that allows the controlled rendering or playing of arbitrary forms of digital content 12, where such control is flexible and definable by the content owner of such digital content 12. Also, the present invention comprises a new useful controlled rendering environment that renders digital content 12 only as specified by the content owner, even though the digital content 12 is to be rendered on a computing device 14 which is not under the control of the content owner. Further, the present invention comprises a trusted component that enforces the rights of the content owner on such computing device 14 in connection with a piece of digital content 12, even against attempts by the user of such computing device 14 to access such digital content 12 in ways not permitted by the content owner.

It should be appreciated that changes could be made to the embodiments described above without departing from the inventive concepts thereof. It should be understood, therefore, that this invention is not limited to the particular embodiments disclosed, but it is intended to cover modifications within the spirit and scope of the present invention as defined by the appended claims.

CLAIMS

1. A method for a device to interdependently validate:

a digital content package having a piece of digital content in an encrypted form; and

a corresponding digital license for rendering the digital content,

the method comprising:

deriving a first key from a source available to the device;

obtaining a first digital signature from the digital content package;

applying the first key to the first digital signature to validate the first digital signature and the digital content package;

deriving a second key based on the first digital signature;

obtaining a second digital signature from the license; and

applying the second key to the second digital signature to validate the second digital signature and the license.

2. The method of claim 1 wherein deriving the first key comprises:

obtaining a first encrypted key from the license;

applying a key available to the device to the first encrypted key to decrypt the first encrypted key;

obtaining a second encrypted key from the digital content; and

applying the decrypted first encrypted key to the second encrypted key to produce the first key.

3. The method of claim 2 wherein the encrypted digital content is decryptable according to a decryption key (KD), and wherein the first encrypted key is the decryption key (KD) encrypted with the device public key (PU-D) (i.e.,(PU-D (KD))).

4. The method of claim 2 wherein the device has a public key (PU-D) and a private key (PR-D), and wherein the key available to the device is (PR-D).

5. The method of claim 2 wherein the encrypted digital content is decryptable according to a decryption key (KD), wherein the digital content package is provided by a content provider having a public key (PU-C) and a private key (PR-C), and wherein the second encrypted key is the content provider public key (PU-C) encrypted with the decryption key (KD) (i.e., KD (PU-C)).

6. The method of claim 2 wherein the second encrypted key is the basis for the first digital signature.

7. The method of claim 1 wherein deriving the second key comprises:

obtaining a signed certificate from the license. the signed certificate having contents therein; and

applying the first key to the signature of the signed certificate to produce the contents of the certificate and also to validate the signature.

8. The method of claim 7 wherein the digital license is provided by a license provider having a public key (PU-L) and a private key (PR-L), and wherein the contents of the certificate is (PU-L).

9. The method of claim 8 wherein the digital content package is provided by a content provider having a public key (PU-C) and a private key (PR-C), and wherein the signed certificate is a certificate containing the license provider public key (PU-L) and signed by the content provider private key (PR-C) (i.e., (CERT (PU-L) S (PR-C))).

10. The method of claim 8 wherein the digital content package is provided by a content provider authorized by a root source to provide the package, wherein the root source has a public key (PU-R) and a private key (PR-R) and wherein the signed certificate is a certificate containing the license provider public key (PU-L) and signed by the root source private key (PR-R) (i.e., (CERT (PU-L) S (PR-R))).

11. The method of claim 1 wherein the digital content package is provided by a content provider having a public key (PU-C) and a private key (PR-C), and wherein the first key is (PU-C).

12. The method of claim 11 wherein the encrypted digital content is decryptable according to a decryption key (KD), and wherein the first digital signature is based on the content provider public key (PU-C) encrypted with the decryption key (KD) and

is signed by the content provider private key (PR-C) (i.e.. (KD (PU-C) S (PR-C))).

13. The method of claim 12 wherein deriving (PU-C) comprises:

deriving (KD) from a source available to the device;

applying (KD) to (KD (PU-C) S (PR-C)) to produce (PU-C).

14. The method of claim 13 wherein the device has a public key (PU-D) and a private key (PR-D), wherein the license has the decryption key (KD) encrypted with the device public key (PU-D) (i.e.,(PU-D (KD))). and wherein deriving (KD) comprises:

obtaining (PU-D (KD)) from the license;

applying (PR-D) to (PU-D (KD)) to produce (KD).

15 The method of claim 14 wherein the license has a license rights description specifying terms and conditions that must be satisfied before the digital content may be rendered, the license rights description being encrypted with the decryption key (KD) (i.e., (KD (DRL))), the method further comprising applying (KD) to (KD(DRL)) to obtain the license terms and conditions.

16. The method of claim 14 wherein the license has a license rights description specifying terms and conditions that must be satisfied before the digital content may be rendered, the method further comprising:

evaluating the license terms and conditions to determine whether the digital content is permitted to be rendered in the manner sought;

if so, applying (KD) to the encrypted digital content to decrypt such encrypted digital content; and

rendering the decrypted digital content.

17. The method of claim 11 wherein the encrypted digital content package is provided by a content provider authorized by a root source to provide the package, wherein the root source has a public key (PU-R) and a private key (PR-R) and wherein the first digital signature is a signed certificate containing the content provider public key (PU-C) and signed by the root source private key (PR-R) (i.e., (CERT (PU-C) S (PR-R))).

18. The method of claim 1 wherein the digital license is provided by a license provider having a public key (PU-L) and a private key (PR-L). and wherein the second key is (PU-L).

19. The method of claim 18 wherein the second digital signature is a digital signature encrypted with the license provider private key (i.e.. (S (PR-L))).

20. The method of claim 19 wherein the digital content package is provided by a content provider having a public key (PU-C) and a private key (PR-C), wherein the

license has a certificate containing the license provider public key (PU-L) and signed by the content provider private key (PR-C) (i.e., (CERT (PU-L) S (PR-C))), and wherein deriving (PU-L) comprises:

deriving (PU-C) from a source available to the device;

obtaining (CERT (PU-L) S (PR-C)) from the license; and

applying (PU-C) to (CERT (PU-L) S (PR-C)) to validate (CERT (PU-L) S (PR-C)), to produce (PU-L) and also to validate the content provider.

21. The method of claim 20 wherein the encrypted digital content is decryptable according to a decryption key (KD), wherein the first digital signature is based on the content provider public key (PU-C) encrypted with the decryption key (KD) and is signed by the content provider private key (PR-C) (i.e., (KD (PU-C) S (PR-C))), and wherein deriving (PU-C) comprises:

deriving (KD) from a source available to the device;

applying (KD) to (KD (PU-C) S (PR-C)) to produce (PU-C).

22. The method of claim 21 wherein the device has a public key (PU-D) and a private key (PR-D), wherein the license has the decryption key (KD) encrypted with the device public key (PU-D) (i.e.,(PU-D (KD))), and wherein deriving (KD) comprises:

obtaining (PU-D (KD)) from the license;

applying (PR-D) to (PU-D (KD)) to produce (KD).

23. The method of claim 22 wherein the license has a license rights description specifying terms and conditions that must be satisfied before the digital content may be rendered, the license rights description being encrypted with the decryption key (KD) (i.e., (KD (DRL))), the method further comprising applying (KD) to (KD(DRL)) to obtain the license terms and conditions.

24. The method of claim 22 wherein the license has a license rights description specifying terms and conditions that must be satisfied before the digital content may be rendered, the method further comprising:

evaluating the license terms and conditions to determine whether the digital content is permitted to be rendered in the manner sought;

if so, applying (KD) to the encrypted digital content to decrypt such encrypted digital content; and

rendering the decrypted digital content.

25. A method for a device to interdependently validate a piece of digital content and a corresponding digital license for rendering the digital content. the digital content being encrypted, the encrypted digital content being decryptable according to a decryption key (KD) and being packaged in a digital content package. the digital content package being provided by a content provider having a public key (PU-C) and a private key (PR-C), the digital license being provided by a license provider having

a public key (PU-L) and a private key (PR-L). the device having a public key (PU-D) and a private key (PR-D), the digital content package comprising:

the encrypted digital content; and

the content provider public key (PU-C) encrypted with the decryption key (KD) and signed by the content provider private key (PR-C) (i.e., (KD (PU-C) S (PR-C)));

the digital license comprising:

the decryption key (KD) encrypted with the device public key (PU-D) (i.e.,(PU-D (KD)));

a digital signature from the license provider (without any attached certificate) based on (KD (DRL)) and (PU-D (KD)) and encrypted with the license provider private key (i.e., (S (PR-L))); and

a certificate containing the license provider public key (PU-L) and signed by the content provider private key (PR-C) (i.e., (CERT (PU-L) S (PR-C)));

the method comprising:

obtaining (PU-D (KD)) from the license;

applying (PR-D) to (PU-D (KD)) to produce (KD);

obtaining (KD (PU-C) S (PR-C)) from the digital content package;

applying (KD) to (KD (PU-C) S (PR-C)) to produce (PU-C);

applying (PU-C) to (S (PR-C)) to validate (KD (PU-C) S (PR-C)), thereby validating the digital content package;

obtaining (CERT (PU-L) S (PR-C)) from the license;

applying (PU-C) to (CERT (PU-L) S (PR-C)) to validate (CERT (PU-L) S (PR-C)), thereby validating the content provider, and also to obtain (PU-L);

obtaining (S (PR-L)) from the license; and

applying (PU-L) to (S (PR-L)), thereby validating the license.

26. The method of claim 25 wherein the digital content package further comprises a content / package ID identifying one of the digital content and the digital content package, and wherein the license further comprises the content / package ID of the corresponding digital content / digital content package, the method further comprising ensuring that the content / package ID of the license in fact corresponds to the content / package ID of the digital content / digital content package.

27. The method of claim 25 wherein the license further comprises a license rights description (DRL) specifying terms and conditions that must be satisfied before the digital content may be rendered, the method further comprising;

evaluating the license terms and conditions to determine whether the digital content is permitted to be rendered in the manner sought;

if so, applying (KD) to the encrypted digital content to decrypt such encrypted digital content; and

rendering the decrypted digital content.

28. The method of claim 27 wherein the license rights description is encrypted with the decryption key (KD) (i.e., (KD (DRL))), the method further comprising applying (KD) to (KD (DRL)) to obtain the license terms and conditions.

29. A computer-readable medium having computer-executable instructions for performing a method for a device to interdependently validate:

a digital content package having a piece of digital content in an encrypted form; and

a corresponding digital license for rendering the digital content, the method comprising:

deriving a first key from a source available to the device;

obtaining a first digital signature from the digital content package;

applying the first key to the first digital signature to validate the first digital signature and the digital content package;

deriving a second key based on the first digital signature;

obtaining a second digital signature from the license; and

applying the second key to the second digital signature to validate the second digital signature and the license.

30. The method of claim 28 wherein deriving the first key comprises:

obtaining a first encrypted key from the license;

applying a key available to the device to the first encrypted key to

decrypt the first encrypted key;

obtaining a second encrypted key from the digital content; and

applying the decrypted first encrypted key to the second encrypted key to produce the first key.

31. The method of claim 30 wherein the encrypted digital content is decryptable according to a decryption key (KD), and wherein the first encrypted key is the decryption key (KD) encrypted with the device public key (PU-D) (i.e.,(PU-D (KD))).

32. The method of claim 30 wherein the device has a public key (PU-D) and a private key (PR-D), and wherein the key available to the device is (PR-D).

33. The method of claim 30 wherein the encrypted digital content is decryptable according to a decryption key (KD), wherein the digital content package is provided by a content provider having a public key (PU-C) and a private key (PR-C), and wherein the second encrypted key is the content provider public key (PU-C) encrypted with the decryption key (KD) (i.e., KD (PU-C)).

34. The method of claim 30 wherein the second encrypted key is the basis for the first digital signature.

35. The method of claim 29 wherein deriving the second key comprises:

obtaining a signed certificate from the license. the signed certificate having contents therein; and

applying the first key to the signature of the signed certificate to produce the contents of the certificate and also to validate the signature.

36. The method of claim 35 wherein the digital license is provided by a license provider having a public key (PU-L) and a private key (PR-L), and wherein the contents of the certificate is (PU-L).

37. The method of claim 36 wherein the digital content package is provided by a content provider having a public key (PU-C) and a private key (PR-C), and wherein the signed certificate is a certificate containing the license provider public key (PU-L) and signed by the content provider private key (PR-C) (i.e.. (CERT (PU-L) S (PR-C))).

38. The method of claim 36 wherein the digital content package is provided by a content provider authorized by a root source to provide the package. wherein the root source has a public key (PU-R) and a private key (PR-R) and wherein the signed certificate is a certificate containing the license provider public key (PU-L) and signed by the root source private key (PR-R) (i.e.. (CERT (PU-L) S (PR-R))).

39. The method of claim 29 wherein the digital content package is provided by a content provider having a public key (PU-C) and a private key (PR-C), and wherein

the first key is (PU-C).

40. The method of claim 39 wherein the encrypted digital content is decryptable according to a decryption key (KD), and wherein the first digital signature is based on the content provider public key (PU-C) encrypted with the decryption key (KD) and is signed by the content provider private key (PR-C) (i.e., (KD (PU-C) S (PR-C))).

41. The method of claim 40 wherein deriving (PU-C) comprises:

deriving (KD) from a source available to the device;

applying (KD) to (KD (PU-C) S (PR-C)) to produce (PU-C).

42. The method of claim 41 wherein the device has a public key (PU-D) and a private key (PR-D), wherein the license has the decryption key (KD) encrypted with the device public key (PU-D) (i.e.,(PU-D (KD))). and wherein deriving (KD) comprises:

obtaining (PU-D (KD)) from the license;

applying (PR-D) to (PU-D (KD)) to produce (KD).

43. The method of claim 42 wherein the license has a license rights description specifying terms and conditions that must be satisfied before the digital content may be rendered, the license rights description being encrypted with the decryption key (KD) (i.e., (KD (DRL))), the method further comprising applying (KD) to (KD(DRL))

to obtain the license terms and conditions.

44. The method of claim 42 wherein the license has a license rights description specifying terms and conditions that must be satisfied before the digital content may be rendered, the method further comprising:

evaluating the license terms and conditions to determine whether the digital content is permitted to be rendered in the manner sought;

if so, applying (KD) to the encrypted digital content to decrypt such encrypted digital content; and

rendering the decrypted digital content.

45. The method of claim 39 wherein the encrypted digital content package is provided by a content provider authorized by a root source to provide the package, wherein the root source has a public key (PU-R) and a private key (PR-R) and wherein the first digital signature is a signed certificate containing the content provider public key (PU-C) and signed by the root source private key (PR-R) (i.e., (CERT (PU-C) S (PR-R))).

46. The method of claim 29 wherein the digital license is provided by a license provider having a public key (PU-L) and a private key (PR-L), and wherein the second key is (PU-L).

47. The method of claim 46 wherein the second digital signature is a digital signature encrypted with the license provider private key (i.e., (S (PR-L))).

48. The method of claim 47 wherein the digital content package is provided by a content provider having a public key (PU-C) and a private key (PR-C), wherein the license has a certificate containing the license provider public key (PU-L) and signed by the content provider private key (PR-C) (i.e., (CERT (PU-L) S (PR-C))), and wherein deriving (PU-L) comprises:

deriving (PU-C) from a source available to the device;

obtaining (CERT (PU-L) S (PR-C)) from the license; and

applying (PU-C) to (CERT (PU-L) S (PR-C)) to validate (CERT (PU-L) S (PR-C)), to produce (PU-L) and also to validate the content provider.

49. The method of claim 48 wherein the encrypted digital content is decryptable according to a decryption key (KD), wherein the first digital signature is based on the content provider public key (PU-C) encrypted with the decryption key (KD) and is signed by the content provider private key (PR-C) (i.e., (KD (PU-C) S (PR-C))), and wherein deriving (PU-C) comprises:

deriving (KD) from a source available to the device:

applying (KD) to (KD (PU-C) S (PR-C)) to produce (PU-C).

50. The method of claim 49 wherein the device has a public key (PU-D) and a

private key (PR-D), wherein the license has the decryption key (KD) encrypted with the device public key (PU-D) (i.e.,(PU-D (KD))), and wherein deriving (KD) comprises:

obtaining (PU-D (KD)) from the license;

applying (PR-D) to (PU-D (KD)) to produce (KD).

51. The method of claim 50 wherein the license has a license rights description specifying terms and conditions that must be satisfied before the digital content may be rendered, the license rights description being encrypted with the decryption key (KD) (i.e., (KD (DRL))), the method further comprising applying (KD) to (KD(DRL)) to obtain the license terms and conditions.

52. The method of claim 50 wherein the license has a license rights description specifying terms and conditions that must be satisfied before the digital content may be rendered, the method further comprising:

evaluating the license terms and conditions to determine whether the digital content is permitted to be rendered in the manner sought;

if so, applying (KD) to the encrypted digital content to decrypt such encrypted digital content; and

rendering the decrypted digital content.

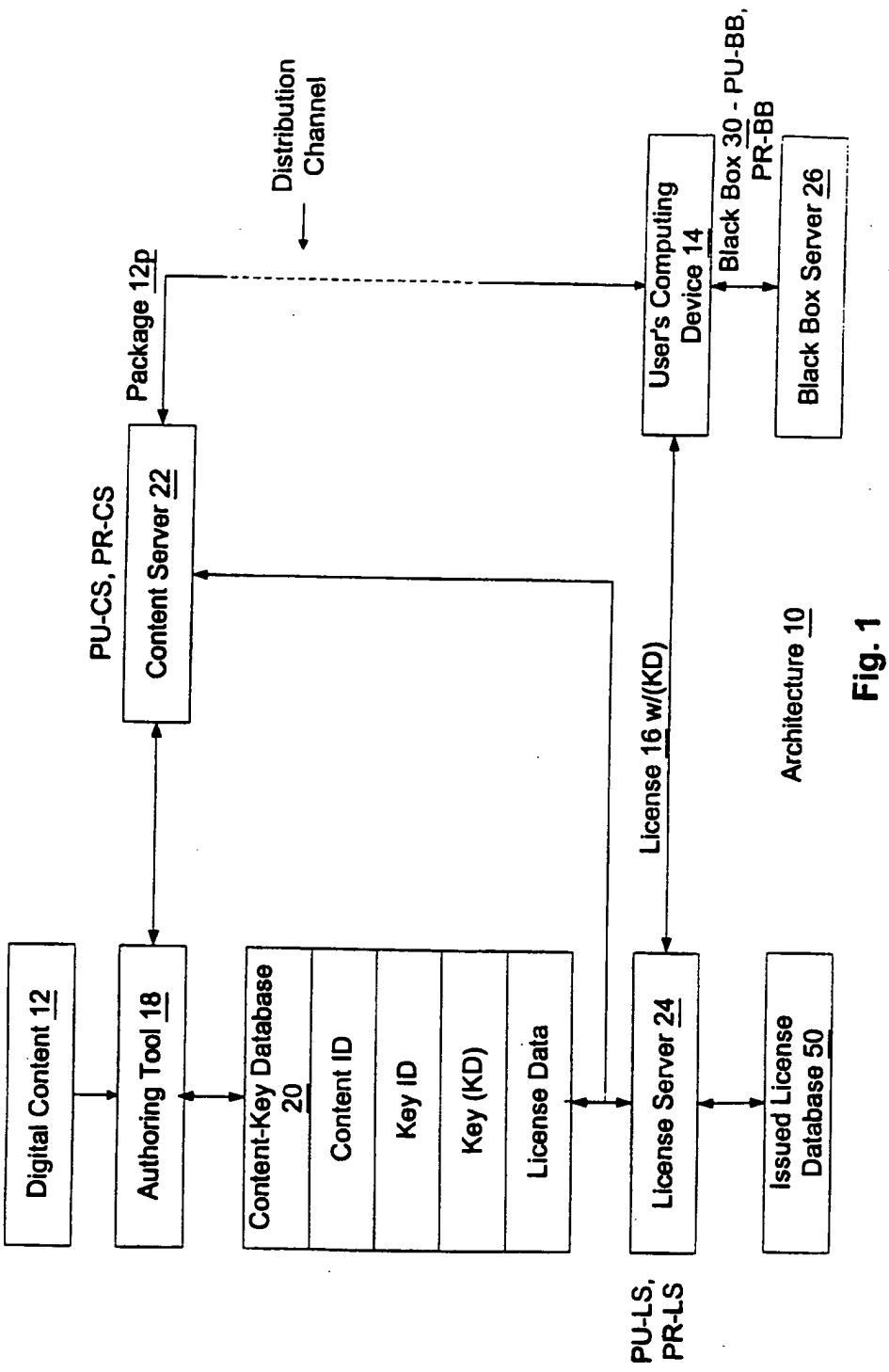

Distribution Channel
Package 12p
User's Computing Device 14
Black Box 30 - PU-BB, PR-BB
Black Box Server 26
PU-CS, PR-CS
Content Server 22
License 16 w/(KD)
Digital Content 12
Authoring Tool 18
Content-Key Database 20
Content ID
Key ID
Key (KD)
License Data
License Server 24
Issued License Database 50
PU-LS, PR-LS
Architecture 10

Fig. 1

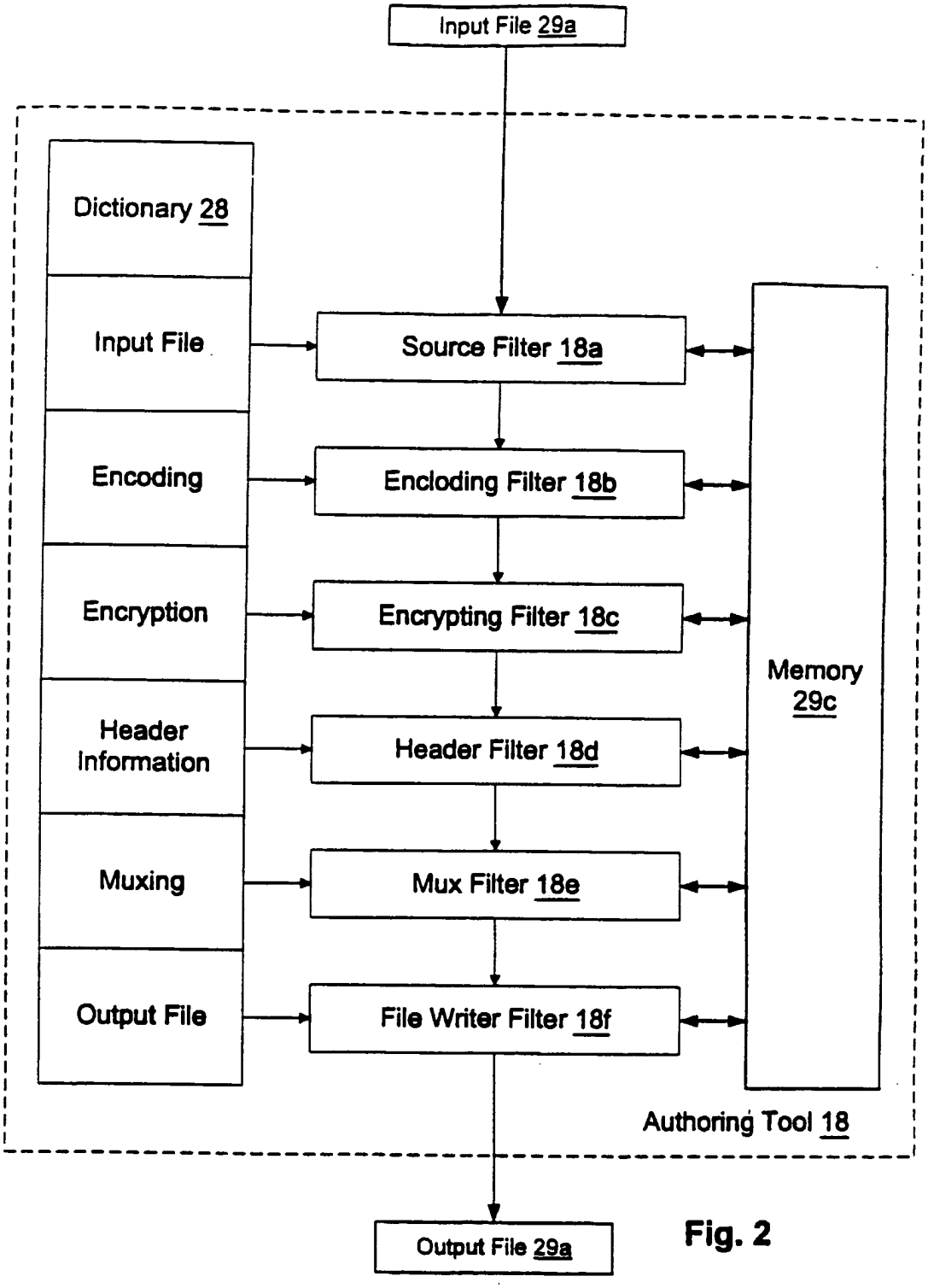

Input File 29a
Dictionary 28
Input File
Encoding
Encryption
Header Information
Muxing
Output File
Source Filter 18a
Encloding Filter 18b
Encrypting Filter 18c
Header Filter 18d
Mux Filter 18e
File Writer Filter 18f
Memory 29c
Authoring Tool 18
Output File 29a

Fig. 2

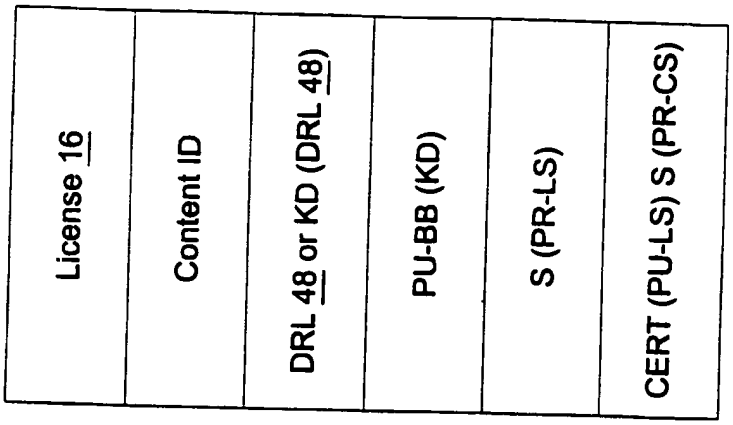

License 16
Content ID
DRL 48 or KD (DRL 48)
PU-BB (KD)
S (PR-LS)
CERT (PU-LS) S (PR-CS)

Fig. 8

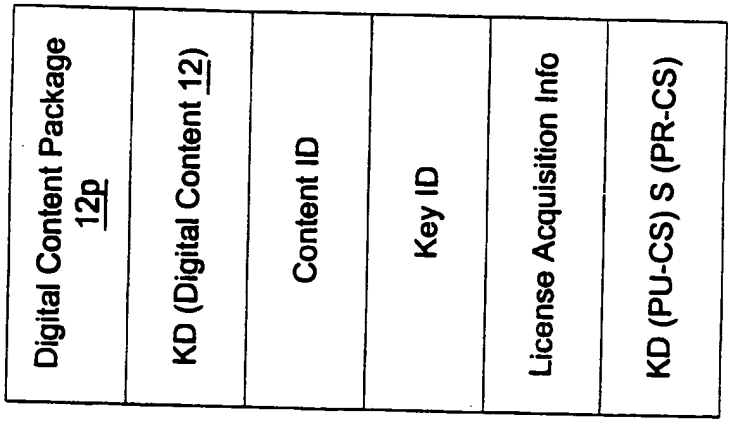

Digital Content Package 12p
KD (Digital Content 12)
Content ID
Key ID
License Acquisition Info
KD (PU-CS) S (PR-CS)

Fig. 3

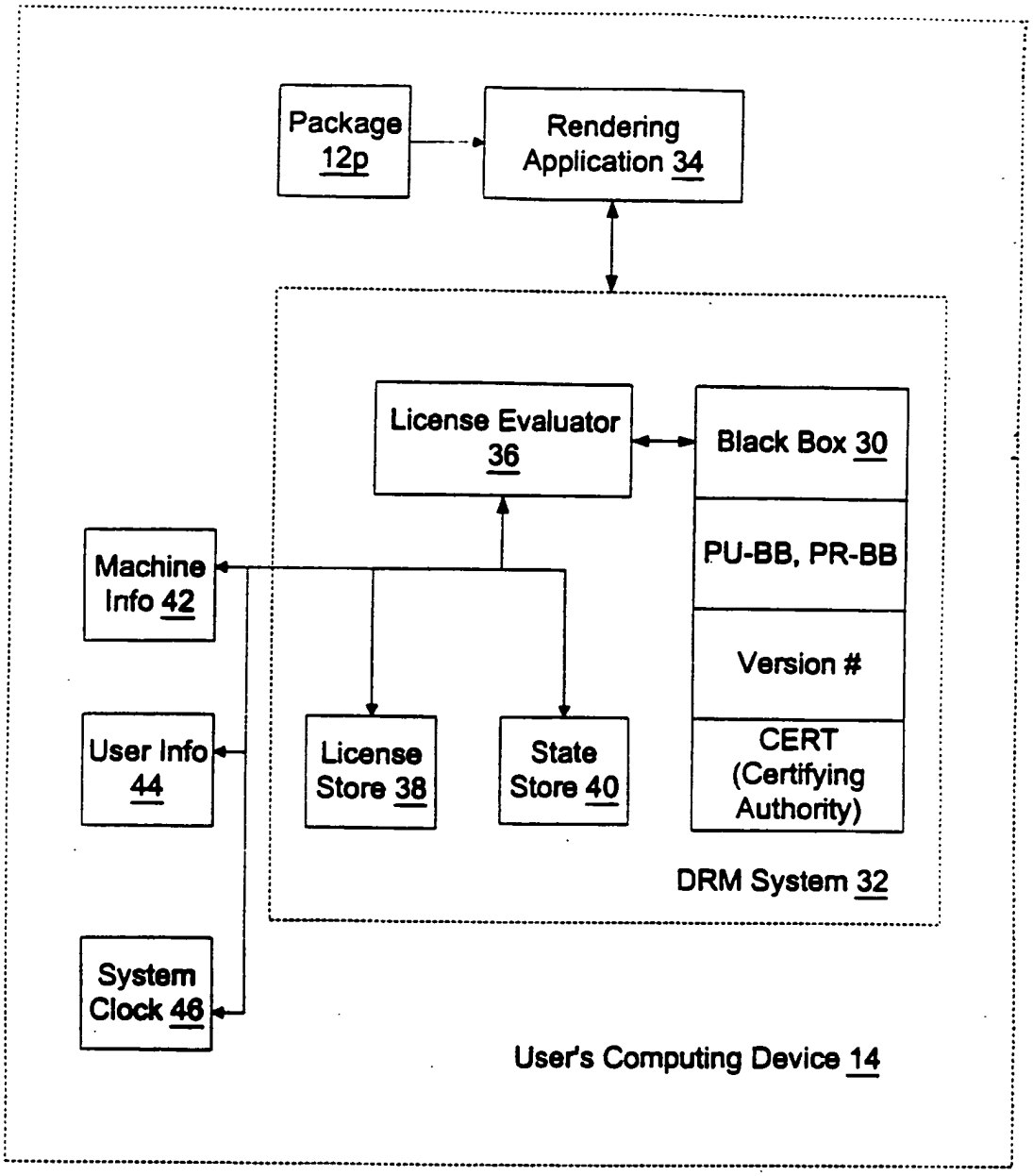

Package 12p
Rendering Application 34
License Evaluator 36
Black Box 30
PU-BB, PR-BB
Version #
CERT (Certifying Authority)
Machine Info 42
User Info 44
License Store 38
State Store 40
DRM System 32
System Clock 46
User's Computing Device 14

Fig. 4

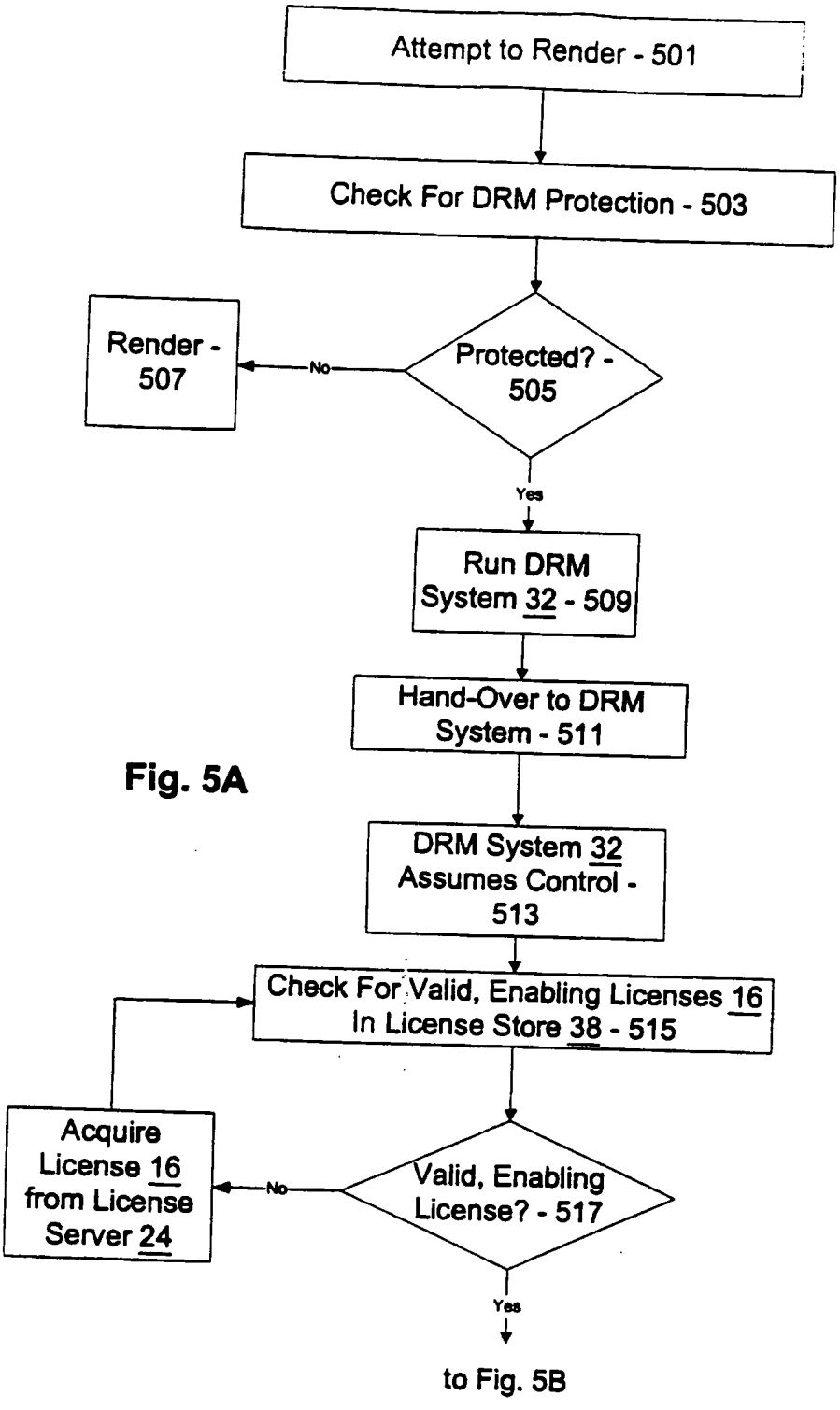

Attempt to Render - 501
Check For DRM Protection - 503
Protected? - 505
No
Render - 507
Yes
Run DRM System 32 - 509
Hand-Over to DRM System - 511
DRM System 32 Assumes Control - 513
Check For Valid, Enabling Licenses 16 In License Store 38 - 515
Valid, Enabling License? - 517
No
Acquire License 16 from License Server 24
Yes
to Fig. 5B

Fig. 5A

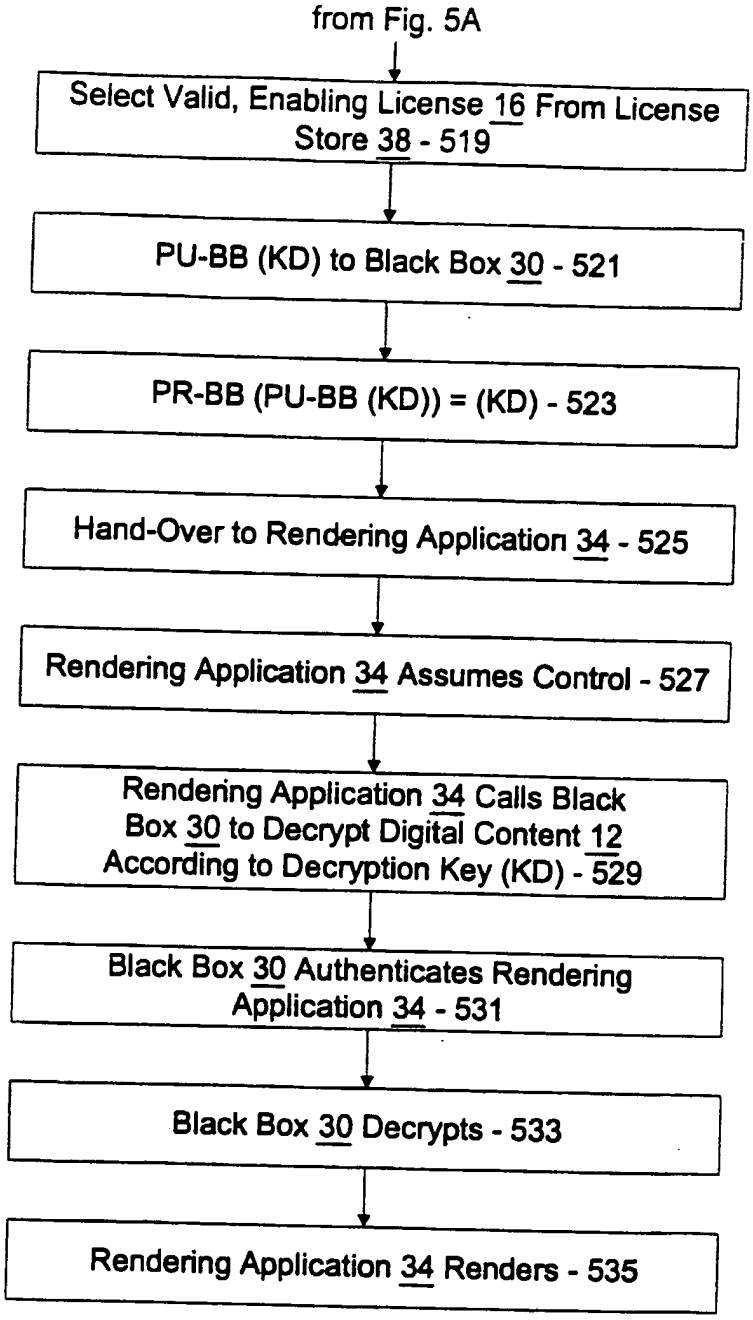

from Fig. 5A
Select Valid, Enabling License 16 From License Store 38 - 519
PU-BB (KD) to Black Box 30 - 521
PR-BB (PU-BB (KD)) = (KD) - 523
Hand-Over to Rendering Application 34 - 525
Rendering Application 34 Assumes Control - 527
Rendering Application 34 Calls Black Box 30 to Decrypt Digital Content 12 According to Decryption Key (KD) - 529
Black Box 30 Authenticates Rendering Application 34 - 531
Black Box 30 Decrypts - 533
Rendering Application 34 Renders - 535

Fig. 5B

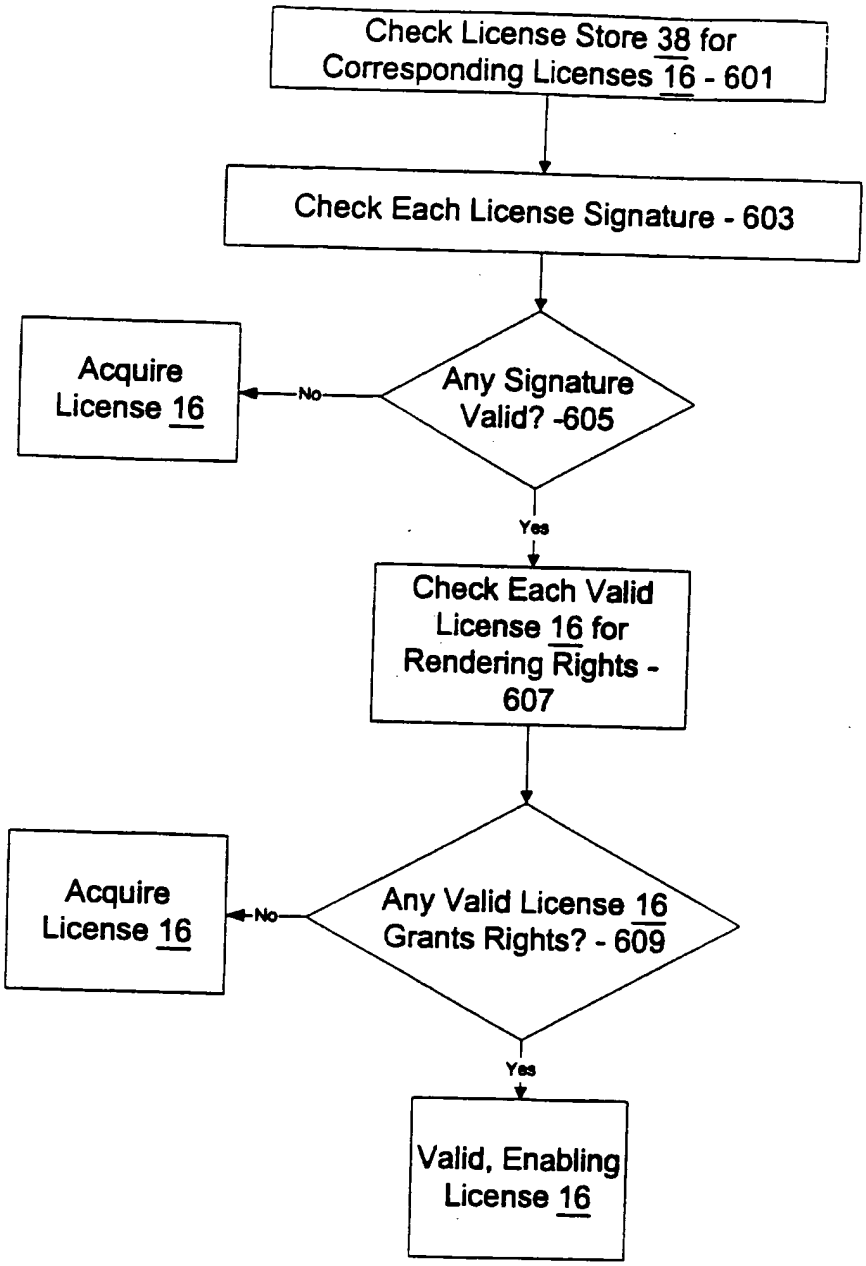

Check License Store 38 for Corresponding Licenses 16 - 601
Check Each License Signature - 603
Any Signature Valid? -605
No
Acquire License 16
Yes
Check Each Valid License 16 for Rendering Rights - 607
Any Valid License 16 Grants Rights? - 609
No
Acquire License 16
Yes
Valid, Enabling License 16

Fig. 6

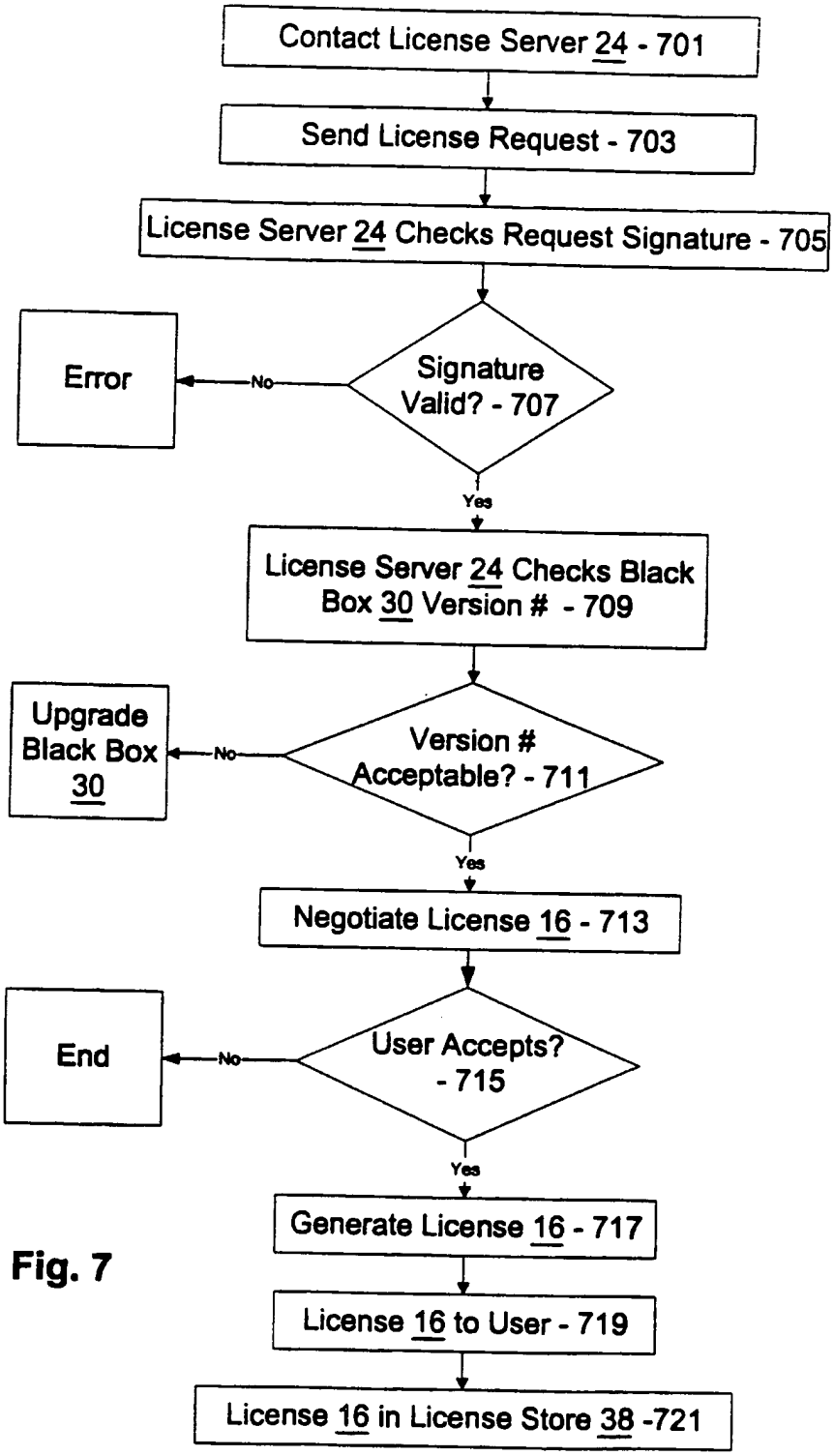

Contact License Server 24 - 701
Send License Request - 703
License Server 24 Checks Request Signature - 705
Signature Valid? - 707
No
Error
Yes
License Server 24 Checks Black Box 30 Version # - 709
Version # Acceptable? - 711
No
Upgrade Black Box 30
Yes
Negotiate License 16 - 713
User Accepts? - 715
No
End
Yes
Generate License 16 - 717
License 16 to User - 719
License 16 in License Store 38 -721

Fig. 7

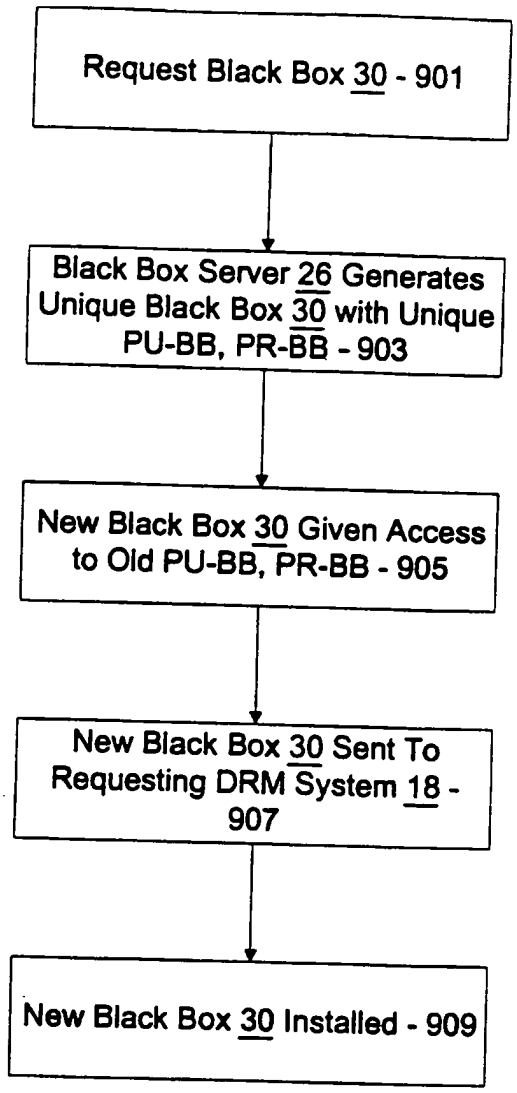

Request Black Box 30 - 901
Black Box Server 26 Generates Unique Black Box 30 with Unique PU-BB, PR-BB - 903
New Black Box 30 Given Access to Old PU-BB, PR-BB - 905
New Black Box 30 Sent To Requesting DRM System 18 - 907
New Black Box 30 Installed - 909

Fig. 9

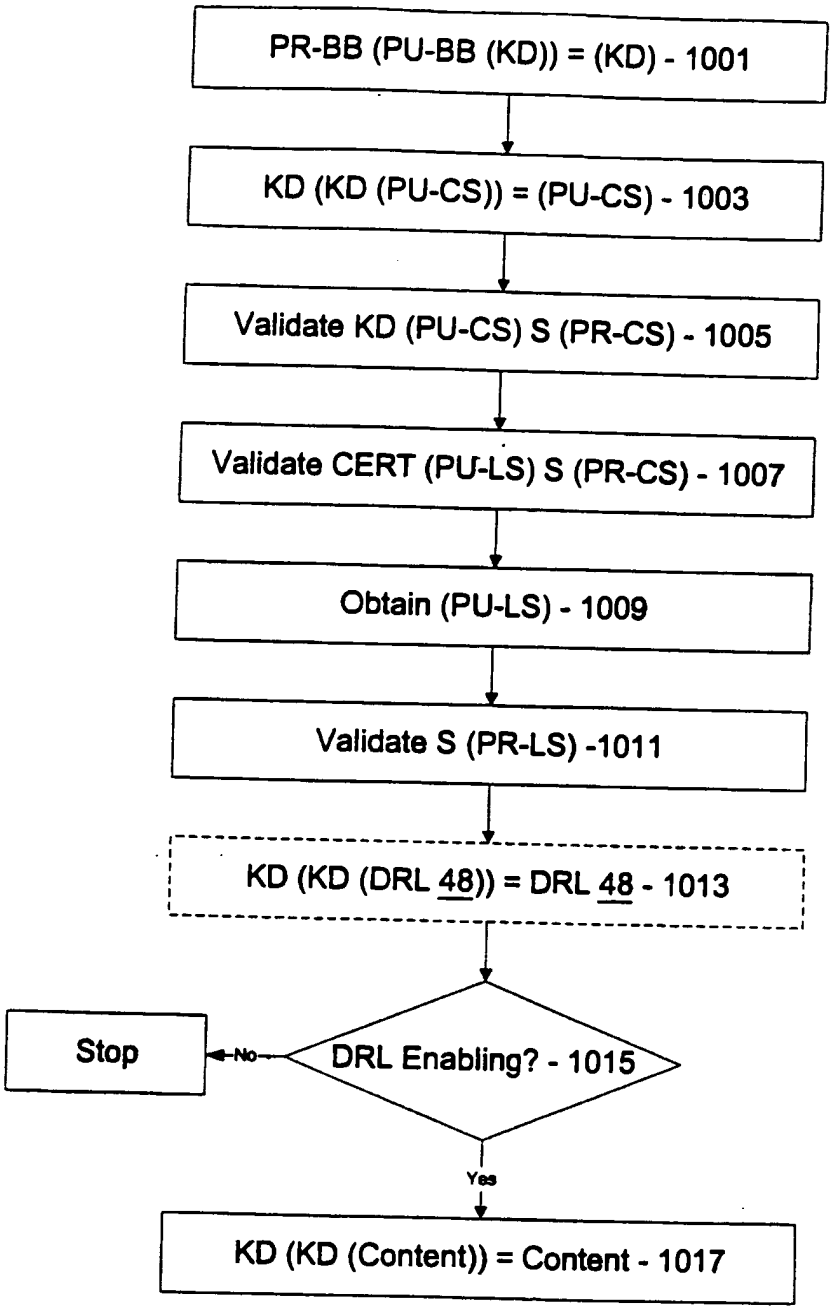

PR-BB (PU-BB (KD)) = (KD) - 1001
KD (KD (PU-CS)) = (PU-CS) - 1003
Validate KD (PU-CS) S (PR-CS) - 1005
Validate CERT (PU-LS) S (PR-CS) - 1007
Obtain (PU-LS) - 1009
Validate S (PR-LS) -1011
KD (KD (DRL 48)) = DRL 48 - 1013
Stop
No
DRL Enabling? - 1015
Yes
KD (KD (Content)) = Content - 1017

Fig. 10

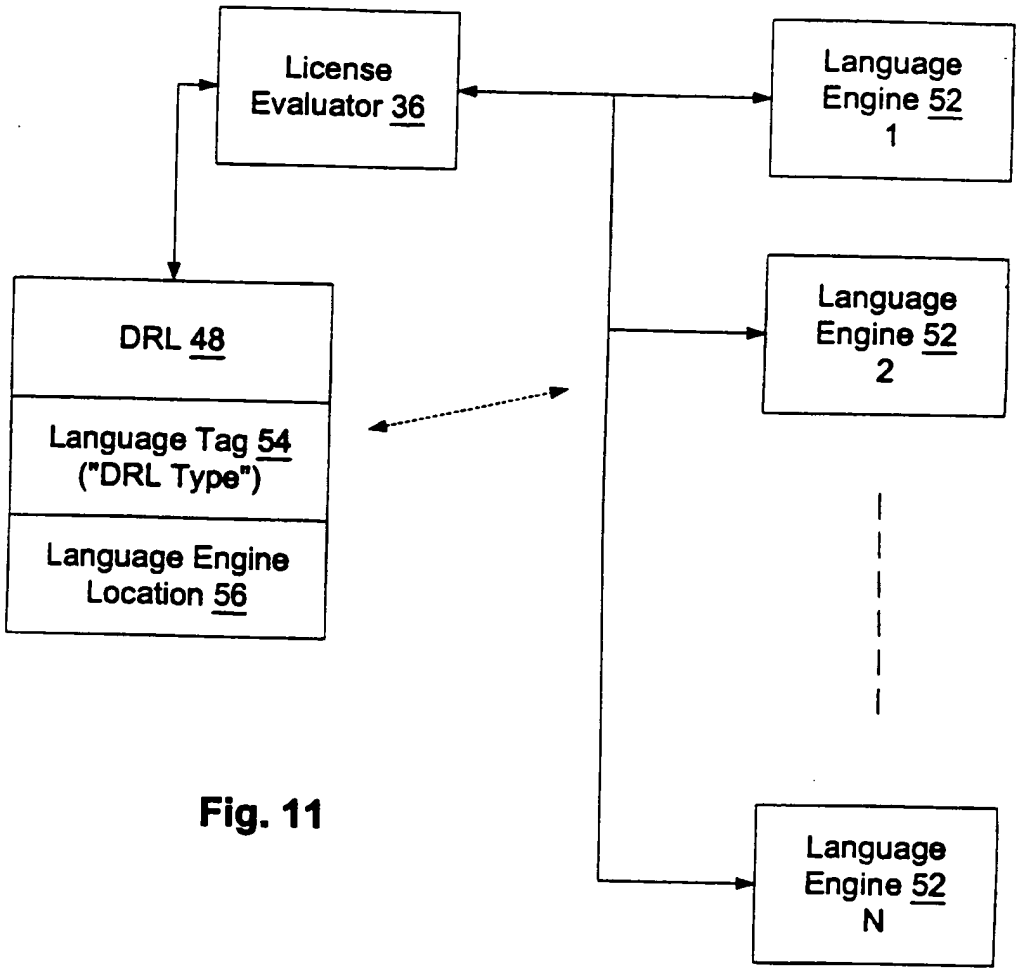

License Evaluator 36
Language Engine 52 1
Language Engine 52 2
Language Engine 52 N
DRL 48
Language Tag 54 ("DRL Type")
Language Engine Location 56

Fig. 11

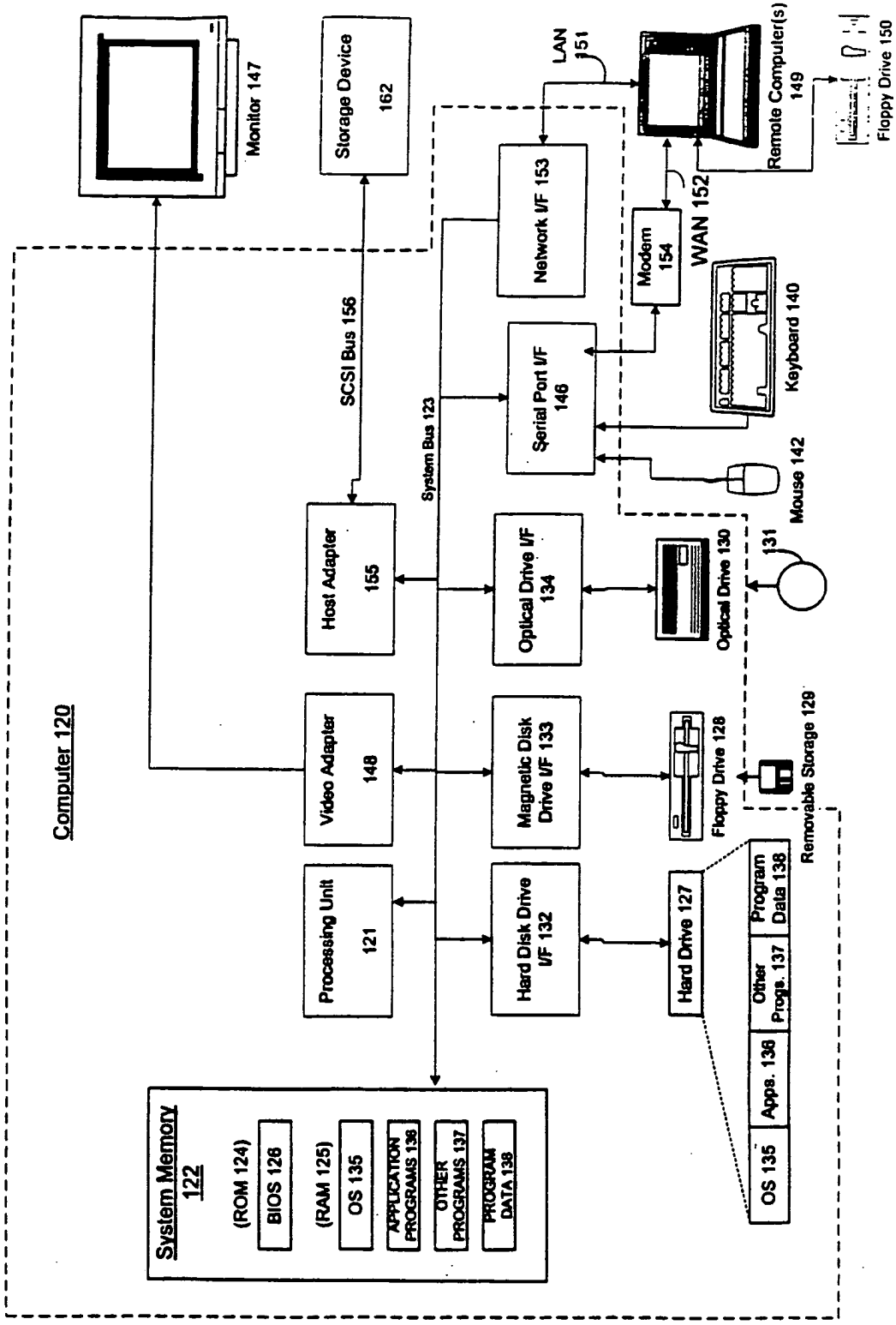

Computer 120
System Memory 122
(ROM 124)
BIOS 126
(RAM 125)
OS 135
APPLICATION PROGRAMS 136
OTHER PROGRAMS 137
PROGRAM DATA 138
Processing Unit 121
Video Adapter 148
Host Adapter 155
Hard Disk Drive I/F 132
Magnetic Disk Drive I/F 133
Optical Drive I/F 134
Serial Port I/F 146
Network I/F 153
Modem 154
WAN 152
LAN 151
Remote Computer(s) 149
Floppy Drive 150
Monitor 147
Storage Device 162
SCSI Bus 156
System Bus 123
Keyboard 140
Mouse 142
131
Optical Drive 130
Floppy Drive 128
Removable Storage 129
Hard Drive 127
OS 135
Appa. 136
Other Progs. 137
Program Data 138

Fig. 12

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
30 November 2000 (30.11.2000)

PCT

(10) International Publication Number
WO 00/72118 A1

(51) International Patent Classification[7]: G06F 1/00

(21) International Application Number: PCT/US00/10213

(22) International Filing Date: 13 April 2000 (13.04.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/316,717 21 May 1999 (21.05.1999) US

(71) Applicant: COMPAQ COMPUTERS INC. [US/US]; 10435 N. Tantau Avenue, Loc 200-16, Cupertino, CA 95014-3548 (US).

(72) Inventors: GLASSMAN, Steven, C.; 615 Palo Alto Avenue, Mountain View, CA 94041 (US). MANASSE, Mark, S.; 1270 Monterey Boulevard, San Francisco, CA 94127 (US).

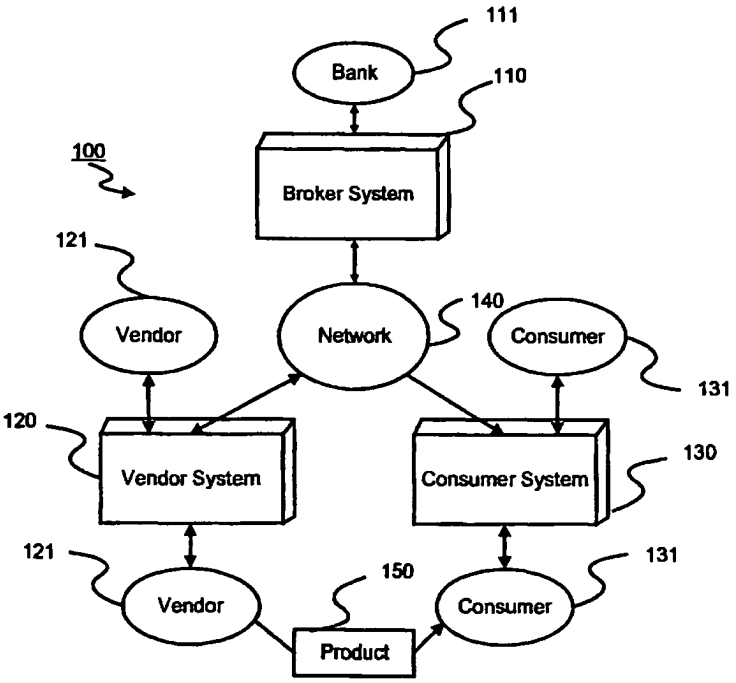
(74) Agents: GRANATELLI, Lawrence; Fenwick & West LLP, Two Palo Alto Square, Palo Alto, CA 94306 et al. (US).

(81) Designated States *(national)*: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU,

[Continued on next page]

(54) Title: METHOD AND SYSTEM FOR ENFORCING LICENSES ON AN OPEN NETWORK



(57) Abstract: An electronic commerce system and method enforces a license agreement for content on an open network (140) by restricting the number of consumers (131) that can concurrently access the content. A consumer (131) initially acquires vendor scrip, either from a broker or the vendor (121) itself. The consumer (131) presents the vendor scrip to the vendor (121) along with a request to access the content. In response, the vendor (121) gathers information about the consumer (131) to determine whether the consumer (131) belongs to the class allowed to access the content. The information may be gathered from the scrip or from other sources. If the consumer (131) belongs to the class, then the vendor (121) determines if a license to access the content is available. Generally, a license is available if the number of other consumers (131) having licenses to access the content is less than the maximum specified in the license agreement. If no licenses are available, the vendor (121) provides the consumer (131) with an estimate of when a license will be available. If a license is available, the vendor (121) directs the consumer (131) to obtain license scrip which allows the consumer (131) to access the content. The license scrip expires after a relatively brief period of time. When the consumer (131) uses the license scrip to access the content, the vendor (121) provides the consumer (131) with new license scrip having a later expiration time.

WO 00/72118 A1

MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

— *With international search report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

METHOD AND SYSTEM FOR ENFORCING LICENSES ON AN OPEN NETWORK

BACKGROUND

FIELD OF THE INVENTION

This invention relates generally to an electronic commerce system and more particularly to a commerce system supporting restricted use of a resource, and even more particularly to a commerce system supporting N-user license agreements.

BACKGROUND OF THE INVENTION

It is common for a library, corporation, or other organization to purchase content that will be made available to members of the organization. Often, the content is subject to a license restriction limiting distribution of the content. For example, a corporation may license or purchase a magazine and then distribute the magazine to interested employees. Typically, the corporation is restricted by the licensing agreement or copyright law from photocopying the magazine. Accordingly, the corporation must either obtain multiple copies of the magazine or circulate the single copy through the organization.

Similarly, the content licensed or purchased by the organization may be in electronic form. For example, the corporation may license a CD-ROM holding an electronic version of the magazine. While the CD-ROM can be loaded onto a server accessible to employees of the corporation via a computer network, the content may be restricted by an N-user license that forbids the corporation from allowing more than N users to simultaneously access the CD-ROM. To implement the restriction, software executing on the server tracks the number of people currently accessing the CD-ROM and blocks usage that exceeds the scope of the license.

In existing systems, the license control is performed by a combination of a specialized lock server and a client program. The lock server validates users' requests for access to the content and maintains the status of active users. The client program interacts with the lock server to acquire a lock and to provide access to the content.

There are many existing implementations of lock servers. However, they all are subject to one or more of the following undesirable restrictions:

each content source has its own, separate, and proprietary lock server;

the user's system already has the content (protected from direct access) and the client program gets the lock to access the content;

acquiring a lock is a complicated action; and/or

the set of valid users is limited.

For these reasons, existing lock servers are undesirable on an open network.

A lock server providing an N-user license on an open network should also support the following requirements:

an unrestricted set of potential users;

no single administrative domain covers all users;

the users do not need to have a separate user application for each source of content;

access to the content can be easily restricted; and

the content exists on the server and not with the user.

Accordingly, there is a need for a way to provide restricted access to electronic content that works with a wide variety of possible access schemes. Preferably, the solution will allow enforcement of an N-user license for content located on an open network like the Internet.

SUMMARY OF THE INVENTION

The above needs are met by a method and system for electronic commerce that uses special scrip - called "license scrip" - to provide temporary licenses to consumers accessing content. Scrip is primarily used as a form of electronic currency, however it can be more generally considered as a one-time token representing a general value. When scrip is used as an electronic currency, its value is monetary. When scrip is used as a temporary license, its value is the permission to access specific content. This permission may be unlimited or it may be for only a relatively brief period of time, say a few minutes to a few hours.

Accessing content with license scrip is very much like buying regular content with monetary scrip. Instead of having a price specified in monetary terms. Each page of content has a price (which may be zero) given in terms of license scrip. A consumer obtains license scrip from the vendor, preferably exchanging regular vendor scrip for the license scrip.

The vendor uses the license scrip to enforce an N-user license agreement - granting up to N people simultaneous access to the content. The vendor tracks the number and identity of consumers currently having licenses to access the content (i.e., consumers currently possessing valid license scrip).

A consumer initially lacks the license scrip needed to access the content. Upon receiving an access request from the consumer, the vendor determines whether a license is available. If a license is not available, the vendor tells the consumer to try again later and, optionally, provides the consumer with an estimate of when a license will be available.

If a license is available, then the vendor directs the consumer to obtain license scrip. Normally, the consumer obtains license scrip by requesting it from the vendor, but the consumer may get the license by any acceptable means. After receiving a license scrip

request. the vendor verifies that the consumer belongs to a class entitled to have a license. For example, if licenses are available to residents of only a certain state, the vendor ensures that the consumer resides in the state before granting the consumer a license.

If a license is available, then the vendor provides the consumer with the license scrip and remembers the granted license. The license scrip is preferably set to expire after a brief time period, but the duration of the license may vary depending upon business or legal concerns. To access content covered by the license, the consumer provides the license scrip when requesting content from the vendor. Each time the consumer accesses the content, the vendor returns replacement license scrip having the same or a later expiration time. Accordingly, the consumer can access the content as long as their license remains valid. When the consumer has not accessed the content for a while, the license scrip expires and the consumer can no longer access the content without obtaining new license scrip.

BRIEF DESCRIPTION OF THE DRAWINGS

FIGURE 1 is a top-level block diagram illustrating a computerized system for conducting electronic commerce;

FIGURE 2 is a block diagram illustrating a computer system used in the system of FIG. 1;

FIGURE 3 is a flow diagram illustrating the operations of the system of FIG. 1;

FIGURE 4 is a block diagram illustrating the data fields of a piece of scrip used in the system of FIG. 1;

FIGURE 5 is a diagram illustrating transactions between a consumer and a vendor utilizing license scrip to enforce an N-user license agreement according to the present invention; and

FIGURE 6 is a flow chart illustrating steps for determining whether to grant a license to a consumer.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

A preferred embodiment of the present invention restricts access to electronic content through the use of an electronic commerce system. Accordingly, it is useful to describe the electronic commerce system before detailing how the system is utilized according to the present invention.

FIG. 1 shows a computerized system 100 for conducting electronic commerce. The system 100 includes a broker system 110, a vendor system 120, and a consumer system 130 interconnected by a communications network 140.

For clarity, the system 100 depicted in FIG. 1 shows only single broker, vendor, and consumer systems. In actual practice, any number of broker, vendor, and consumer systems can be interconnected by the network 140. The network 140 can be public or private, such as, for example, the Internet, an organization's intranet, a switched telephone system, a satellite linked network, or another form of network. The broker 111 using the broker system 110 can be a bank, a credit provider, an Internet service provider, a telephone company, or any institution the consumer trusts to sell electronic currency called "scrip."

The vendor system 120 is operated by a vendor 121. The vendor 121 provides products and/or content 150 of any type to consumers and, in one embodiment, provides content which is available by subscription. Each subscription page (i.e., page of data that is available for "purchase") has a price of zero but requires a special type of scrip, called "subscription scrip," before it can be accessed. Since the price of a page is zero, the consumer 131 can "purchase" an unlimited number of pages once the consumer 131 has the

proper subscription scrip 330. The subscription expires when the subscription scrip 330 expires.

A consumer 131 can use the consumer computer system 130 to electronically acquire the products or content 150 of the vendor 121. As used herein, "consumer" refers to an organization such as a library or corporation, a member of the organization, such as a librarian or an employee, or an individual, such as a person visiting a library or a home computer user. Of course, actions attributed to the organization are usually performed by a member of the organization.

A computer system 200 suitable for use as the broker, vendor, and consumer systems is shown in FIG. 2. The computer system 200 includes a central processing unit (CPU) 210, a memory 220, and an input/output interface 230 connected to each other by a communications bus 240. The CPU 210, at the direction of users 250, e.g. brokers, vendors, and/or consumers, executes software programs, or modules, for manipulating data. The programs and data can be stored in the memory 220 as a database (DB) 221. The DB 221 storing programs and data on the consumer computer system 130 is referred to as a "wallet." In a preferred embodiment of the present invention described herein, many of the operations attributed to the consumer are, in fact, performed automatically by the wallet 221.

The memory 220 can include volatile semiconductor memory as well as persistent storage media, such as disks. The I/O interface 230 is for communicating data with the network 140, the users 250, and other computer system peripheral equipment, such as printers, tapes, etc.

The computer system 200 is scaled in size to function as the broker, vendor, or consumer systems. For example, when scaled as the consumer computer system 130, the computer system 200 can be a small personal computer (PC), fixed or portable. The

configurations of the computer system 200 suitable for use by the broker 111 and the vendor 121 may include multiple processors and large database equipped with "fail-safe" features. The fail-safe features ensure that the database 221 is securely maintained for long periods of time.

FIG. 3 shows an operation of the electronic commerce system 100. The consumer 131 uses currency to purchase electronic broker scrip 320 generated by the broker 111. Here, purchasing means that upon a validation of the authenticity of the consumer 131 and the consumer's currency 310, the broker system 110 generates signals, in the form of data records. The signals are communicated, via the network 140, to the consumer system 130 for storage in the wallet 221 of the memory 220 of the consumer system 130.

The scrip is stamped by the generator of the scrip to carry information that is verifiable by the originator, and any other system that has an explicit agreement with the originator. In addition, each scrip is uniquely identifiable and valid at only a single recipient. After a single use, the recipient of the scrip can invalidate it, meaning that the signals of the data record are no longer accepted for processing by the recipient computer system.

In one embodiment, the consumer 131 exchanges the broker scrip 320 with the broker 111 for vendor scrip 330. To complete this transaction, the broker system 110 executes licensed software programs which generate scrip 330 for consumers as needed. Alternatively, the broker 111, in a similar transaction 303, exchanges currency 310 for bulk vendor scrip 330 which is then sold to consumers.

In another embodiment, the consumer 131 exchanges currency with the vendor 121 for regular vendor. In this latter embodiment, there is no need for a broker 111. In addition, the vendor scrip may be free, meaning that the consumer 131 does not need to exchange currency for the scrip.

The consumer 131, in a transaction 304, provides the scrip 330 to the vendor 121. The vendor 121 checks the stamp of the scrip 330 to verify its authenticity, and also checks to make sure the value of the scrip covers the requested content and has not expired. Approval of the transaction results in the delivery of the desired content 150 to the consumer 131. The vendor 121 can also return 304 modified scrip 330 to the consumer 131 as change.

FIG. 4 is a block diagram illustrating the data fields of a single piece of scrip 400. The scrip 400 is logically separated into seven data fields. The Vendor field 410 identifies the vendor for the scrip 400. The Value field 412 gives the value of the scrip 400. The scrip ID field 414 is the unique identifier of the scrip. The Customer ID field 416 is used by the broker 111 and vendor 121 to verify that the consumer has the right to spend the scrip. The Expires field 418 gives the expiration time for the scrip 400. The Props field 420 holds consumer properties, such as the consumer's age, state of residence, employer, etc. Finally, the Stamp field 422 holds a digital stamp and is used to detect tampering with the scrip 400.

The present invention uses "license" scrip, which can be thought of as special purpose scrip having a short period of validity. A consumer with license scrip has a license to view the content covered by the license until the scrip expires.

FIG. 5 is a diagram illustrating transactions between a consumer 510 and a vendor 512 utilizing license scrip to enforce an N-user license agreement according to the present invention. In the transactions of FIG. 5, the vendor 512, for example, can be a library located at a state university. Assume the library purchases a four user license for a CD-ROM and makes the CD-ROM available to other terminals in the library via a local area network and residents of the state via the Internet. To conform with the license, the library must ensure that no more than four consumers are simultaneously accessing the CD-ROM. In this