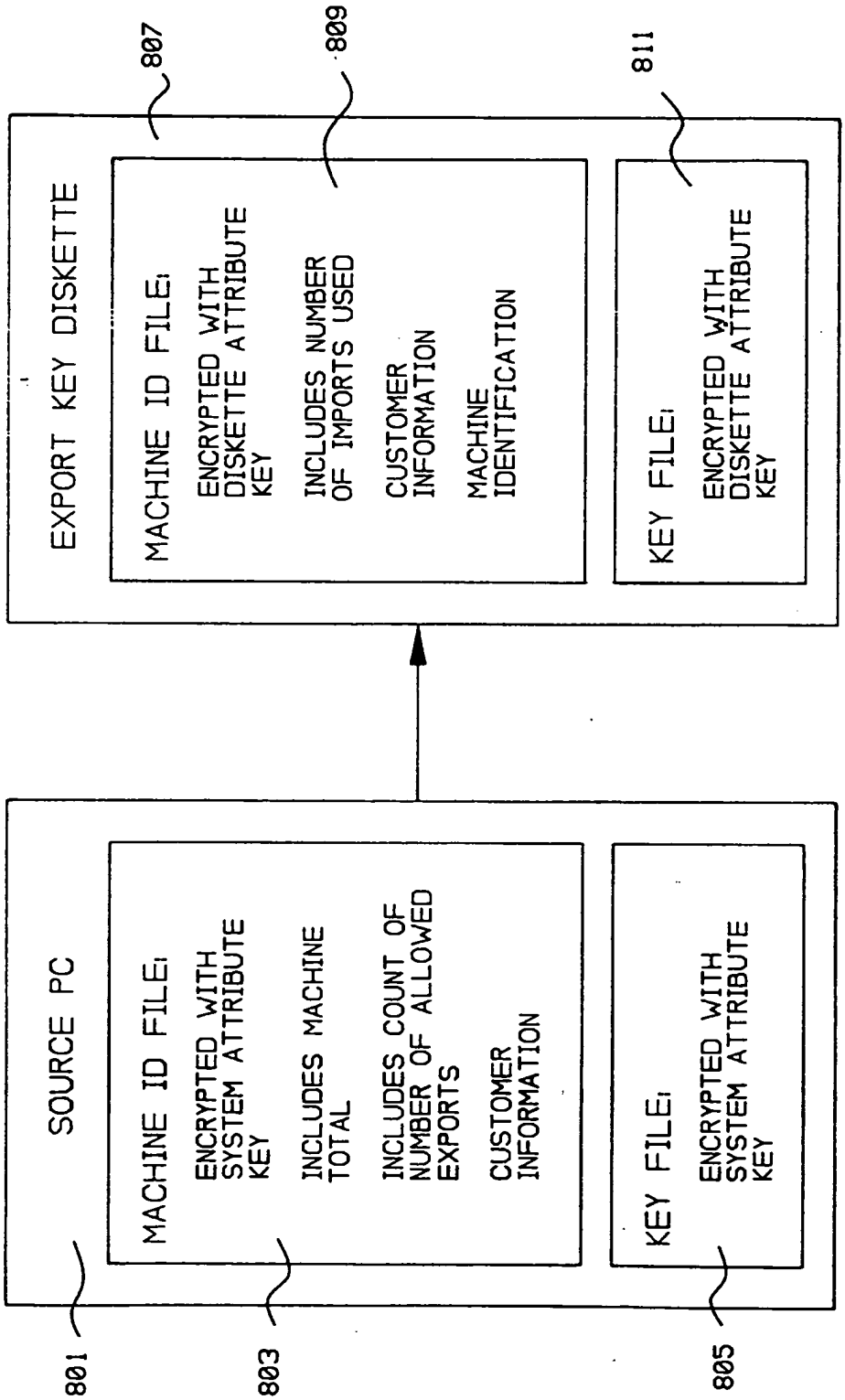


FIG. 32

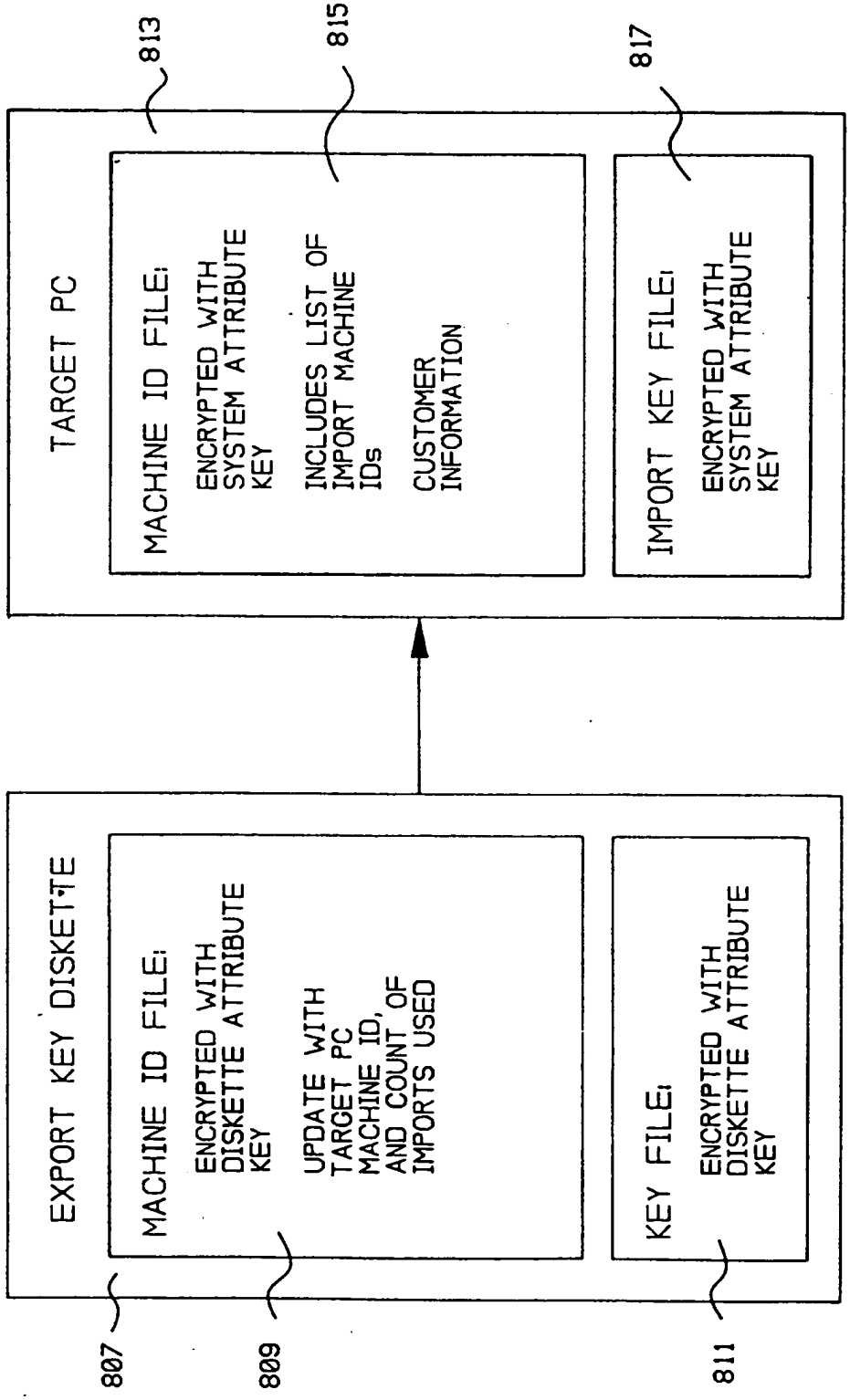
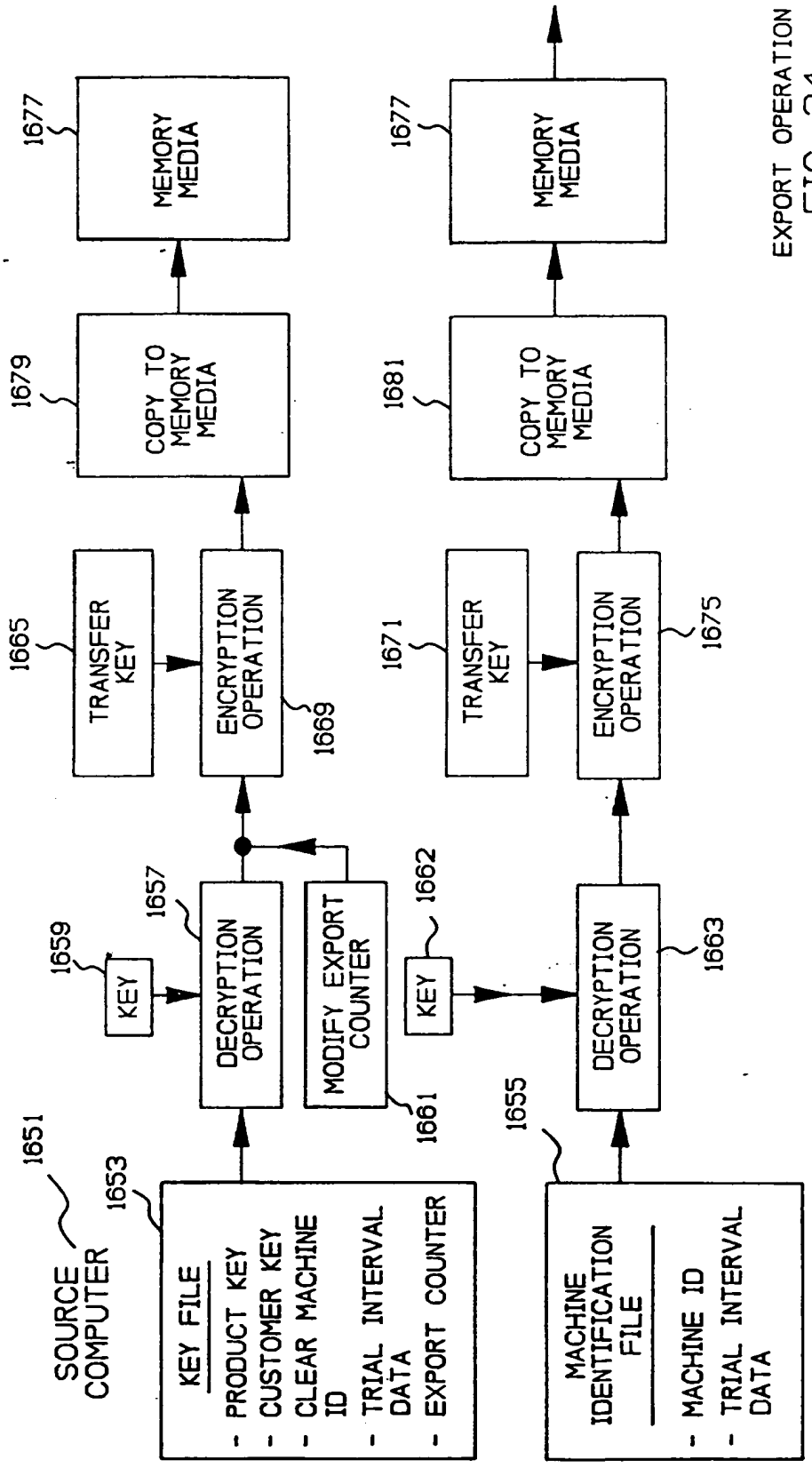


FIG. 33



EXPORT OPERATION
FIG. 34

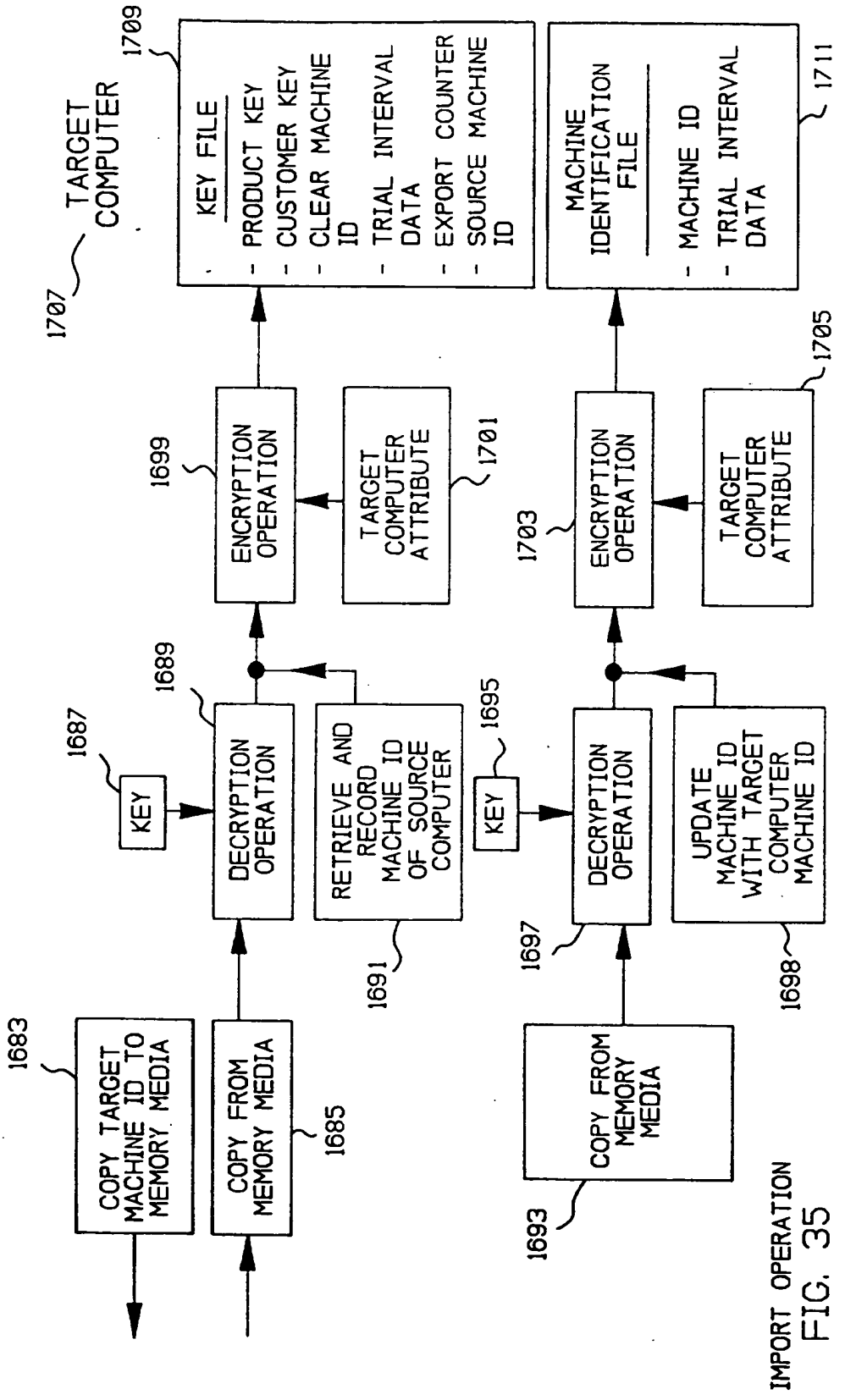


FIG. 35

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 95 10 5400

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
X	WO-A-94 07204 (UNILOC) * abstract; figures 41,2,8 * * page 6, line 11 - page 9, line 5 * * page 10, line 3 - line 10 * * page 12, line 7 - page 17, line 13 *	1,9	G06F1/00 G06F12/14
Y		2,4-8,10	
Y	GB-A-2 136 175 (ATALLA) * the whole document *	2	
Y	EP-A-0 268 139 (IBM) * column 1, line 1 - column 3, line 1 * * column 6, line 7 - column 7, line 50 * * column 9, line 20 - line 29 * * column 19, line 9 - line 50 * * column 21, line 6 - line 18 * * claims 2,9 *	4-8,10	
A		3	
A	EP-A-0 561 685 (FUJITSU) * the whole document *	9	

TECHNICAL FIELDS SEARCHED (Int.Cl.6)

G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	25 July 1995	Powell, D

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 715 243 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
05.06.1996 Bulletin 1996/23

(51) Int Cl.[6]: **G06F 1/00**, G06F 17/60

(21) Application number: **95308414.2**

(22) Date of filing: **23.11.1995**

(84) Designated Contracting States:
DE FR GB

(30) Priority: **23.11.1994 US 344773**

(71) Applicant: **XEROX CORPORATION**
Rochester New York 14644 (US)

(72) Inventors:
- **Stefik, Mark J.**
  Woodside, California 94062 (US)
- **Pirolli, Peter L.T.**
  El Cerrito, California 94530 (US)
- **Merkle, Ralph C.**
  Sunnyvale, California 94087 (US)

(74) Representative: **Goode, Ian Roy et al**
Rank Xerox Ltd
Patent Department
Parkway
Marlow Buckinghamshire SL7 1YL (GB)

(54) System for controlling the distribution and use of digital works having a fee reporting mechanism

(57) A fee accounting mechanism for reporting fees associated with the distribution and use of digital works. Usage rights and fees are attached to digital works. The usage rights define how the digital work may be used or further distributed. Usage fees are specified as part of a usage right. The digital works and their usage rights and fees are stored in repositories (201). The repositories control access to the digital works. Upon determination that the exercise of a usage right requires a fee, the repository generates a fee reporting transaction (302). Fee reporting is done to a credit server (301). The credit server collects the fee information and periodically transmits it to a billing clearinghouse (303).

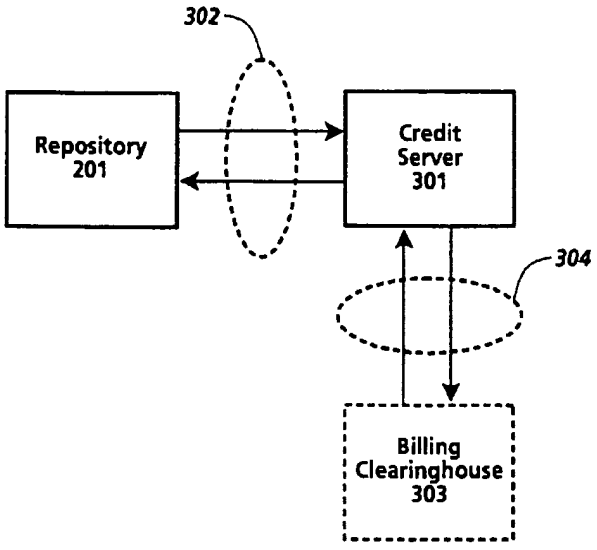


Fig. 3

EP 0 715 243 A1

Printed by Jouve, 75001 PARIS (FR)

Description

The present invention relates to the field of distribution and usage rights enforcement for digitally encoded works.

A fundamental issue facing the publishing and information industries as they consider electronic publishing is how to prevent the unauthorized and unaccounted distribution or usage of electronically published materials. Electronically published materials are typically distributed in a digital form and recreated on a computer based system having the capability to recreate the materials. Audio and video recordings, software, books and multimedia works are all being electronically published. Companies in these industries receive royalties for each accounted for delivery of the materials, e.g. the sale of an audio CD at a retail outlet. Any unaccounted distribution of a work results in an unpaid royalty (e.g. copying the audio recording CD to another digital medium.)

The ease in which electronically published works can be "perfectly" reproduced and distributed is a major concern. The transmission of digital works over networks is commonplace. One such widely used network is the Internet. The Internet is a widespread network facility by which computer users in many universities, corporations and government entities communicate and trade ideas and information. Computer bulletin boards found on the Internet and commercial networks such as CompuServ and Prodigy allow for the posting and retrieving of digital information. Information services such as Dialog and LEXIS/NEXIS provide databases of current information on a wide variety of topics. Another factor which will exacerbate the situation is the development and expansion of the National Information Infrastructure (the NII). It is anticipated that, as the NII grows, the transmission of digital works over networks will increase many times over. It would be desirable to utilize the NII for distribution of digital works without the fear of widespread unauthorized copying.

The most straightforward way to curb unaccounted distribution is to prevent unauthorized copying and transmission. For existing materials that are distributed in digital form, various safeguards are used. In the case of software, copy protection schemes which limit the number of copies that can be made or which corrupt the output when copying is detected have been employed. Another scheme causes software to become disabled after a predetermined period of time has lapsed. A technique used for workstation based software is to require that a special hardware device must be present on the workstation in order for the software to run, e.g., see US-A-4,932,054 entitled "Method and Apparatus for Protecting Computer Software Utilizing Coded Filter Network in Conjunction with an Active Coded Hardware Device." Such devices are provided with the software and are commonly referred to as dongles.

Yet another scheme is to distribute software, but which requires a "key" to enable its use. This is employed in distribution schemes where "demos" of the software are provided on a medium along with the entire product. The demos can be freely used, but in order to use the actual product, the key must be purchased. These schemes do not hinder copying of the software once the key is initially purchased.

It is an object of the present invention to provide an improved system and method for controlling the use and distribution of digital works.

The invention accordingly provides a system and method as claimed in the accompanying claims.

In a system for the control of distribution and use of digital works, a fee reporting mechanism for reporting fees associated with such distribution and use is disclosed. The system includes a means for attaching usage rights to a digital work. The usage rights define how the digital work may be used or further distributed by a possessor of the digital work. Usage fees are specified as part of a usage right. The ability to report usage fees may be a condition to the exercise of a usage right. Further, different fees may be assigned to different usage rights.

The present invention enables various usage fee scenarios to be used. Fees may be assessed on a per use basis, on a metered basis or based on a predetermined schedule. Fees may also be discounted on a predetermined schedule, or they can be marked-up a predetermined percentage (e.g. as a distributor fee). Fee reporting may also be deferred to a later time, to accommodate special deals, rebates or some other external information not yet available.

The present invention supports usage fees in an additive fashion. Usage fees may be reported for a composite digital work, i.e. a digital work comprised of a plurality of discrete digital works each having their own usage rights, and for distributors of digital works. Accordingly, fees to multiple revenue owners can be reported.

Usage fee reporting is done to a credit server. The credit server collects the fee information and periodically transmits it to a billing clearinghouse. Alternatively, the credit server may have a pre-allocated credit which is decremented as fees are incurred. In this alternative embodiment, the credit server would have to be periodically reallocated with credits to enable further use.

A system and method in accordance with the invention will now be described, by way of example, with reference to the accompanying drawings, in which:-

Figure 1 is a flowchart illustrating a simple instantiation of the operation of the currently preferred embodiment of the present invention.

Figure 2 is a block diagram illustrating the various repository types and the repository transaction flow between them in the currently preferred embodiment of the present invention.

Figure 3 is a block diagram of a repository coupled with a credit server in the currently preferred embodiment of

the present invention.

Figures 4a and 4b are examples of rendering systems as may be utilized in the currently preferred embodiment of the present invention.

Figure 5 illustrates a contents file layout for a digital work as may be utilized in the currently preferred embodiment of the present invention.

Figure 6 illustrates a contents file layout for an individual digital work of the digital work of Figure 5 as may be utilized in the currently preferred embodiment of the present invention.

Figure 7 illustrates the components of a description block of the currently preferred embodiment of the present invention.

Figure 8 illustrates a description tree for the contents file layout of the digital work illustrated in Figure 5.

Figure 9 illustrates a portion of a description tree corresponding to the individual digital work illustrated in Figure 6.

Figure 10 illustrates a layout for the rights portion of a description block as may be utilized in the currently preferred embodiment of the present invention.

Figure 11 is a description tree wherein certain d-blocks have PRINT usage rights and is used to illustrate "strict" and "lenient" rules for resolving usage rights conflicts.

Figure 12 is a block diagram of the hardware components of a repository as are utilized in the currently preferred embodiment of the present invention.

Figure 13 is a block diagram of the functional (logical) components of a repository as are utilized in the currently preferred embodiment of the present invention.

Figure 14 is diagram illustrating the basic components of a usage right in the currently preferred embodiment of the present invention.

Figure 15 lists the usage rights grammar of the currently preferred embodiment of the present invention.

Figure 16 is a flowchart illustrating the steps of certificate delivery, hotlist checking and performance testing as performed in a registration transaction as may be performed in the currently preferred embodiment of the present invention.

Figure 17 is a flowchart illustrating the steps of session information exchange and clock synchronization as may be performed in the currently preferred embodiment of the present invention, after each repository in the registration transaction has successfully completed the steps described in Figure 16.

Figure 18 is a flowchart illustrating the basic flow for a usage transaction, including the common opening and closing step, as may be performed in the currently preferred embodiment of the present invention.

Figure 19 is a state diagram of server and client repositories in accordance with a transport protocol followed when moving a digital work from the server to the client repositories, as may be performed in the currently preferred embodiment of the present invention.

OVERVIEW

A system for controlling use and distribution of digital works is disclosed. The present invention is directed to supporting commercial transactions involving digital works.

Herein the terms "digital work", "work" and "content" refer to any work that has been reduced to a digital representation. This would include any audio, video, text, or multimedia work and any accompanying interpreter (e.g. software) that may be required for recreating the work. The term composite work refers to a digital work comprised of a collection of other digital works. The term "usage rights" or "rights" is a term which refers to rights granted to a recipient of a digital work. Generally, these rights define how a digital work can be used and if it can be further distributed. Each usage right may have one or more specified conditions which must be satisfied before the right may be exercised.

Figure 1 is a high level flowchart omitting various details but which demonstrates the basic operation of the present invention. Referring to Figure 1, a creator creates a digital work, step 101. The creator will then determine appropriate usage rights and fees, attach them to the digital work, and store them in Repository 1, step 102. The determination of appropriate usage rights and fees will depend on various economic factors. The digital work remains securely in Repository 1 until a request for access is received. The request for access begins with a session initiation by another repository. Here a Repository 2 initiates a session with Repository 1, step 103. As will be described in greater detail below, this session initiation includes steps which helps to insure that the respective repositories are trustworthy. Assuming that a session can be established, Repository 2 may then request access to the Digital Work for a stated purpose, step 104. The purpose may be, for example, to print the digital work or to obtain a copy of the digital work. The purpose will correspond to a specific usage right. In any event, Repository 1 checks the usage rights associated with the digital work to determine if the access to the digital work may be granted, step 105. The check of the usage rights essentially involves a determination of whether a right associated with the access request has been attached to the digital work and if all conditions associated with the right are satisfied. If the access is denied, repository 1 terminates the session with an error message, step 106. If access is granted, repository 1 transmits the digital work to repository

2, step 107. Once the digital work has been transmitted to repository 2, repository 1 and 2 each generate billing information for the access which is transmitted to a credit server, step 108. Such double billing reporting is done to insure against attempts to circumvent the billing process.

Figure 2 illustrates the basic interactions between repository types in the present invention. As will become apparent from Figure 2, the various repository types will serve different functions. It is fundamental that repositories will share a core set of functionality which will enable secure and trusted communications. Referring to Figure 2, a repository 201 represents the general instance of a repository. The repository 201 has two modes of operation; a server mode and a requester mode. When in the server mode, the repository will be receiving and processing access requests to digital works. When in the requester mode, the repository will be initiating requests to access digital works. Repository 201 is general in the sense that its primary purpose is as an exchange medium for digital works. During the course of operation, the repository 201 may communicate with a plurality of other repositories, namely authorization repository 202, rendering repository 203 and master repository 204. Communication between repositories occurs utilizing a repository transaction protocol 205.

Communication with an authorization repository 202 may occur when a digital work being accessed has a condition requiring an authorization. Conceptually, an authorization is a digital certificate such that possession of the certificate is required to gain access to the digital work. An authorization is itself a digital work that can be moved between repositories and subjected to fees and usage rights conditions. An authorization may be required by both repositories involved in an access to a digital work.

Communication with a rendering repository 203 occurs in connection with the rendering of a digital work. As will be described in greater detail below, a rendering repository is coupled with a rendering device (e.g. a printer device) to comprise a rendering system.

Communication with a master repository 205 occurs in connection with obtaining an identification certificate. Identification certificates are the means by which a repository is identified as "trustworthy". The use of identification certificates is described below with respect to the registration transaction.

Figure 3 illustrates the repository 201 coupled to a credit server 301. The credit server 301 is a device which accumulates billing information for the repository 201. The credit server 301 communicates with repository 201 via billing transactions 302 to record billing transactions. Billing transactions are reported to a billing clearinghouse 303 by the credit server 301 on a periodic basis. The credit server 301 communicates to the billing clearinghouse 303 via clearinghouse transactions 304. The clearinghouse transactions 304 enable a secure and encrypted transmission of information to the billing clearinghouse 303.

RENDERING SYSTEMS

A rendering system is generally defined as a system comprising a repository and a rendering device which can render a digital work into its desired form. Examples of a rendering system may be a computer system, a digital audio system, or a printer. A rendering system has the same security features as a repository. The coupling of a rendering repository with the rendering device may occur in a manner suitable for the type of rendering device.

Figure 4a illustrates a printer as an example of a rendering system. Referring to Figure 4, printer system 401 has contained therein a printer repository 402 and a print device 403. It should be noted that the the dashed line defining printer system 401 defines a secure system boundary. Communications within the boundary are assumed to be secure. Depending on the security level, the boundary also represents a barrier intended to provide physical integrity. The printer repository 402 is an instantiation of the rendering repository 205 of Figure 2. The printer repository 402 will in some instances contain an ephemeral copy of a digital work which remains until it is printed out by the print engine 403. In other instances, the printer repository 402 may contain digital works such as fonts, which will remain and can be billed based on use. This design assures that all communication lines between printers and printing devices are encrypted, unless they are within a physically secure boundary. This design feature eliminates a potential "fault" point through which the digital work could be improperly obtained. The printer device 403 represents the printer components used to create the printed output.

Also illustrated in Figure 4a is the repository 404. The repository 404 is coupled to the printer repository 402. The repository 404 represents an external repository which contains digital works.

Figure 4b is an example of a computer system as a rendering system. A computer system may constitute a "multifunction" device since it may execute digital works (e.g. software programs) and display digital works (e.g. a digitized photograph). Logically, each rendering device can be viewed as having its own repository, although only one physical repository is needed. Referring to Figure 4b, a computer system 410 has contained therein a display/execution repository 411. The display/execution repository 411 is coupled to display device, 412 and execution device 413. The dashed box surrounding the computer system 410 represents a security boundary within which communications are assumed to be secure. The display/execution repository 411 is further coupled to a credit server 414 to report any fees to be billed for access to a digital work and a repository 415 for accessing digital works stored therein.

STRUCTURE OF DIGITAL WORKS

Usage rights are attached directly to digital works. Thus, it is important to understand the structure of a digital work. The structure of a digital work, in particular composite digital works, may be naturally organized into an acyclic structure such as a hierarchy. For example, a magazine has various articles and photographs which may have been created and are owned by different persons. Each of the articles and photographs may represent a node in a hierarchical structure. Consequently, controls, i.e. usage rights, may be placed on each node by the creator. By enabling control and fee billing to be associated with each node, a creator of a work can be assured that the rights and fees are not circumvented.

In the currently preferred embodiment, the file information for a digital work is divided into two files: a "contents" file and a "description tree" file. From the perspective of a repository, the "contents" file is a stream of addressable bytes whose format depends completely on the interpreter used to play, display or print the digital work. The description tree file makes it possible to examine the rights and fees for a work without reference to the content of the digital work. It should be noted that the term description tree as used herein refers to any type of acyclic structure used to represent the relationship between the various components of a digital work.

Figure 5 illustrates the layout of a contents file. Referring to Figure 5, a digital work is comprised of story A 510, advertisement 511, story B 512 and story C 513. It is assumed that the digital work is stored starting at a relative address of 0. Each of the parts of the digital work are stored linearly so that story A 510 is stored at approximately addresses 0-30,000, advertisement 511 at addresses 30,001-40,000, story B 512 at addresses 40,001-60,000 and story C 513 at addresses 60,001-85K. The detail of story A 510 is illustrated in Figure 6. Referring to Figure 6, the story A 510 is further broken down to show text 614 stored at address 0-1500, soldier photo 615 at addresses 1501-10,000, graphics 616 stored at addresses 10,001-25,000 and sidebar 617 stored address 25,001-30,000. Note that the data in the contents file may be compressed (for saving storage) or encrypted (for security).

From Figures 5 and 6 it is readily observed that a digital work can be represented by its component parts as a hierarchy. The description tree for a digital work is comprised of a set of related descriptor blocks (d-blocks). The contents of each d-block is described with respect to Figure 7. Referring to Figure 7, a d-block 700 includes an identifier 701 which is a unique identifier for the work in the repository, a starting address 702 providing the start address of the first byte of the work, a length 703 giving the number of bytes in the work, a rights portion 704 wherein the granted usage rights and their status data are maintained, a parent pointer 705 for pointing to a parent d-block and child pointers 706 for pointing to the child d-blocks. In the currently preferred embodiment, the identifier 701 has two parts. The first part is a unique number assigned to the repository upon manufacture. The second part is a unique number assigned to the work upon creation. The rights portion 704 will contain a data structure, such as a look-up table, wherein the various information associated with a right is maintained. The information required by the respective usage rights is described in more detail below. D-blocks form a strict hierarchy. The top d-block of a work has no parent; all other d-blocks have one parent. The relationship of usage rights between parent and child d-blocks and how conflicts are resolved is described below.

A special type of d-block is a "shell" d-block. A shell d-block adds no new content beyond the content of its parts. A shell d-block is used to add rights and fee information, typically by distributors of digital works.

Figure 8 illustrates a description tree for the digital work of Figure 5. Referring to Figure 8, a top d-block 820 for the digital work points to the various stories and advertisements contained therein. Here, the top d-block 820 points to d-block 821 (representing story A 510), d-block 822 (representing the advertisement 511), d-block 823 (representing story B 512) and and d-block 824 (representing story C 513).

The portion of the description tree for Story A 510 is illustrated in Figure 9. D-block 925 represents text 614, d-block 926 represents photo 615, d-block 927 represents graphics 616 by and d-block 928 represents sidebar 617.

The rights portion 704 of a descriptor block is further illustrated in Figure 10. Figure 10 illustrates a structure which is repeated in the rights portion 704 for each right. Referring to Figure 10, each right will have a right code field 1050 and status information field 1052. The right code field 1050 will contain a unique code assigned to a right. The status information field 1052 will contain information relating to the state of a right and the digital work. Such information is indicated below in Table 1. The rights as stored in the rights portion 704 may typically be in numerical order based on the right code.

TABLE 1

DIGITAL WORK STATE INFORMATION		
Property	**Value**	**Use**
Copies-in-Use	Number	A counter of the number of copies of a work that are in use. Incremented when another copy is used; decremented when use is completed.
Loan-Period	Time-Units	Indicator of the maximum number of time-units that a document can be loaned out
Loaner-Copy	Boolean	Indicator that the current work is a loaned out copy of an authorized digital work.
Remaining-Time	Time-Units	Indicator of the remaining time of use on a metered document right.
Document-Descr	String	A string containing various identifying information about a document. The exact format of this is not specified, but it can include information such as a publisher name, author name, ISBN number, and so on.
Revenue-Owner	RO-Descr	A handle identifying a revenue owner for a digital work. This is used reporting usage fees.
Publication-Date	Date-Descr	The date that the digital work was published.
History-list	History-Rec	A list of events recording the repostories and dates for operations that copy, transfer, backup, or restore a digital work.

The approach for representing digtal works by separating description data from content assumes that parts of a file are contiguous but takes no position on the actual representation of content. In particular, it is neutral to the question of whether content representation may take an object oriented approach. It would be natural to represent content as objects. In principle, it may be convenient to have content objects that include the billing structure and rights information that is represented in the d-blocks. Such variations in the design of the representation are possible and are viable alternatives but may introduce processing overhead, e.g. the interpretation of the objects.

Digital works are stored in a repository as part of a hierarchical file system. Folders (also termed directories and sub-directories) contain the digital works as well as other folders. Digital works and folders in a folder are ordered in alphabetical order. The digital works are typed to reflect how the files are used. Usage rights can be attached to folders so that the folder itself is treated as a digital work. Access to the folder would then be handled in the same fashion as any other digital work As will be described in more detail below, the contents of the folder are subject to their own rights. Moreover, file management rights may be attached to the folder which define how folder contents can be managed.

ATTACHING USAGE RIGHTS TO A DIGITAL WORK

It is fundamental to the present invention that the usage rights are treated as part of the digital work. As the digital work is distributed, the scope of the granted usage rights will remain the same or may be narrowed. For example, when a digital work is transferred from a document server to a repository, the usage rights may include the right to loan a copy for a predetermined period of time (called the original rights). When the repository loans out a copy of the digital work, the usage rights in the loaner copy (called the next set of rights) could be set to prohibit any further rights to loan out the copy. The basic idea is that one cannot grant more rights than they have.

The attachment of usage rights into a digital work may occur in a variety of ways. If the usage rights will be the same for an entire digital work, they could be attached when the digital work is processed for deposit in the digital work server. In the case of a digital work having different usage rights for the various components, this can be done as the digital work is being created. An authoring tool or digital work assembling tool could be utilized which provides for an automated process of attaching the usage rights.

As will be described below, when a digital work is copied, transferred or loaned, a "next set of rights" can be specified. The"next set of rights" will be attached to the digital work as it is transported.

Resolving Conflicting Rights

Because each part of a digital work may have its own usage rights, there will be instances where the rights of a "contained part" are different from its parent or container part. As a result, conflict rules must be established to dictate when and how a right may be exercised. The hierarchical structure of a digital work facilitates the enforcement of such rules. A "strict" rule would be as follows: a right for a part in a digital work is sanctioned if and only if it is sanctioned

for the part, for ancestor d-blocks containing the part and for all descendent d-blocks. By sanctioned, it is meant that (1) each of the respective parts must have the right, and (2) any conditions for exercising the right are satisfied.

It also possible to implement the present invention using a more lenient rule. In the more lenient rule, access to the part may be enabled to the descendent parts which have the right, but access is denied to the descendents which do not.

An example of applying both the strict rule and lenient is illustrated with reference to Figure 11. Referring to Figure 11, a root d-block 1101 has child d-blocks 1102-1105. In this case, root d-block represents a magazine, and each of the child d-blocks 1102-1105 represent articles in the magazine. Suppose that a request is made to PRINT the digital work represented by root d-block 1101 wherein the strict rule is followed. The rights for the root d-block 1101 and child d-blocks 1102-1105 are then examined. Root d-block 1101 and child d-blocks 1102 and 1105 have been granted PRINT rights. Child d-block 1103 has not been granted PRINT rights and child d-block 1104 has PRINT rights conditioned on payment of a usage fee.

Under the strict rule the PRINT right cannot be exercised because the child d-block does not have the PRINT right. Under the lenient rule, the result would be different. The digital works represented by child d-blocks 1102 and 1105 could be printed and the digital work represented by d-block 1104 could be printed so long as the usage fee is paid. Only the digital work represented by d-block 1103 could not be printed. This same result would be accomplished under the strict rule if the requests were directed to each of the individual digital works.

The present invention supports various combinations of allowing and disallowing access. Moreover, as will be described below, the usage rights grammar permits the owner of a digital work to specify if constraints may be imposed on the work by a container part. The manner in which digital works may be sanctioned because of usage rights conflicts would be implementation specific and would depend on the nature of the digital works.

REPOSITORIES

In the description of Figure 2, it was indicated that repositories come in various forms. All repositories provide a core set of services for the transmission of digital works. The manner in which digital works are exchanged is the basis for all transaction between repositories. The various repository types differ in the ultimate functions that they perform. Repositories may be devices themselves, or they may be incorporated into other systems. An example is the rendering repository 203 of Figure 2.

A repository will have associated with it a repository identifier. Typically, the repository identifier would be a unique number assigned to the repository at the time of manufacture. Each repository will also be classified as being in a particular security class. Certain communications and transactions may be conditioned on a repository being in a particular security class. The various security classes are described in greater detail below.

As a prerequisite to operation, a repository will require possession of an identification certificate. Identification certificates are encrypted to prevent forgery and are issued by a Master repository. A master repository plays the role of an authorization agent to enable repositories to receive digital works. Identification certificates must be updated on a periodic basis. Identification certificates are described in greater detail below with respect to the registration transaction.

A repository has both a hardware and functional embodiment. The functional embodiment is typically software executing on the hardware embodiment. Alternatively, the functional embodiment may be embedded in the hardware embodiment such as an Application Specific Integrated Circuit (ASIC) chip.

The hardware embodiment of a repository will be enclosed in a secure housing which if compromised, may cause the repository to be disabled. The basic components of the hardware embodiment of a repository are described with reference to Figure 12. Referring to Figure 12, a repository is comprised of a processing means 1200, storage system 1207, clock 1205 and external interface 1206. The processing means 1200 is comprised of a processor element 1201 and processor memory 1202. The processing means 1201 provides controller, repository transaction and usage rights transaction functions for the repository. Various functions in the operation of the repository such as decryption and/or decompression of digital works and transaction messages are also performed by the processing means 1200. The processor element 1201 may be a microprocessor or other suitable computing component. The processor memory 1202 would typically be further comprised of Read Only Memories (ROM) and Random Access Memories (RAM). Such memories would contain the software instructions utilized by the processor element 1201 in performing the functions of the repository.

The storage system 1207 is further comprised of descriptor storage 1203 and content storage 1204. The description tree storage 1203 will store the description tree for the digital work and the content storage will store the associated content. The description tree storage 1203 and content storage 1204 need not be of the same type of storage medium, nor are they necessarily on the same physical device. So for example, the descriptor storage 1203 may be stored on a solid state storage (for rapid retrieval of the description tree information), while the content storage 1204 may be on a high capacity storage such as an optical disk.

The clock 1205 is used to time-stamp various time based conditions for usage rights or for metering usage fees which may be associated with the digital works. The clock 1205 will have an uninterruptable power supply, e.g. a battery, in order to maintain the integrity of the time-stamps. The external interface means 1206 provides for the signal connection to other repositories and to a credit server. The external interface means 1206 provides for the exchange of signals via such standard interfaces such as RS-232 or Personal Computer Manufacturers Card Industry Association (PCMCIA) standards, or FDDI. The external interface means 1206 may also provide network connectivity.

The functional embodiment of a repository is described with reference to Figure 13. Referring to Figure 13, the functional embodiment is comprised of an operating system 1301, core repository services 1302, usage transaction handlers 1303, repository specific functions, 1304 and a user interface 1305. The operating system 1301 is specific to the repository and would typically depend on the type of processor being used. The operating system 1301 would also provide the basic services for controlling and interfacing between the basic components of the repository.

The core repository services 1302 comprise a set of functions required by each and every repository. The core repository services 1302 include the session initiation transactions which are defined in greater detail below. This set of services also includes a generic ticket agent which is used to "punch" a digital ticket and a generic authorization server for processing authorization specifications. Digital tickets and authorizations are specific mechanisms for controlling the distribution and use of digital works and are described in more detail below. Note that coupled to the core repository services are a plurality of identification certificates 1306. The identification certificates 1306 are required to enable the use of the repository.

The usage transactions handlers 1303 comprise functionality for processing access requests to digital works and for billing fees based on access. The usage transactions supported will be different for each repository type. For example, it may not be necessary for some repositories to handle access requests for digital works.

The repository specific functionality 1304 comprises functionality that is unique to a repository. For example, the master repository has special functionality for issuing digital certificates and maintaining encryption keys. The repository specific functionality 1304 would include the user interface implementation for the repository.

Repository Security Classes

For some digital works the losses caused by any individual instance of unauthorized copying is insignificant and the chief economic concern lies in assuring the convenience of access and low-overhead billing. In such cases, simple and inexpensive handheld repositories and network-based workstations may be suitable repositories, even though the measures and guarantees of security are modest.

At the other extreme, some digital works such as a digital copy of a first run movie or a bearer bond or stock certificate would be of very high value so that it is prudent to employ caution and fairly elaborate security measures to ensure that they are not copied or forged. A repository suitable for holding such a digital work could have elaborate measures for ensuring physical integrity and for verifying authorization before use.

By arranging a universal protocol, all kinds of repositories can communicate with each other in principle. However, creators of some works will want to specify that their works will only be transferred to repositories whose level of security is high enough. For this reason, document repositories have a ranking system for classes and levels of security. The security classes in the currently preferred embodiment are described in Table 2.

TABLE 2

REPOSITORY SECURITY LEVELS	
Level	Description of Security
0	Open system.Document transmission is unencrypted. No digital certificate is required for identification. The security of the system depends mostly on user honesty, since only modest knowledge may be needed to circumvent the security measures. The repositery has no provisions for preventing unauthorized programs from running and accessing or copying files. The system does not prevent the use of removable storage and does not encrypt stored files.
1	Minimal security. Like the previous class except that stored files are minimally encrypted, including ones on removable storage.
2	Basic security. Like the previous class except that special tools and knowledge are required to compromise the programming, the contents of the repository, or the state of the clock. All digital communications are encrypted. A digital certificate is provided as identification. Medium level encryption is used. Repository identification number is unforgeable.

Continuation of the Table on the next page

TABLE 2 (continued)

REPOSITORY SECURITY LEVELS	
Level	Description of Security
3	General security. Like the previous class plus the requirement of special tools are needed to compromise the physical integrity of the repository and that modest encryption is used on all transmissions. Password protection is required to use the local user interface. The digital clock system cannot be reset without authorization. No works would be stored on removable storage. When executing works as programs, it runs them in their own address space and does not give them direct access to any file storage or other memory containing system code or works. They can access works only through the transmission transaction protocol.
4	Like the previous class except that high level encryption is used on all communications. Sensors are used to record attempts at physical and electronic tampering. After such tampering, the repository will not perform other transactions until it has reported such tampering to a designated server.
5	Like the previous class except that if the physical or digital attempts at tampering exceed some preset thresholds that threaten the physical integrity of the repository or the integrity of digital and cryptographic barriers, then the repository will save only document description records of history but will erase or destroy any digital identifiers that could be misused if released to an unscrupulous. It also modifies any certificates of authenticity to indicate that the physical system has been compromised. It also erases the contents of designated documents.
6	Like the previous class except that the repository will attempt wireless communication to report tampering and will employ noisy alarms.
10	This would correspond to a very high level of security. This server would maintain constant communications to remote security systems reporting transactions, sensor readings, and attempts to circumvent security.

The characterization of security levels described in Table 2 is not intended to be fixed. More important is the idea of having different security levels for different repositories. It is anticipated that new security classes and requirements will evolve according to social situations and changes in technology.

Repository User Interface

A user interface is broadly defined as the mechanism by which a user interacts with a repository in order to invoke transactions to gain access to a digital work, or exercise usage rights. As described above, a repository may be embodied in various forms. The user interface for a repository will differ depending on the particular embodiment. The user interface may be a graphical user interface having icons representing the digital works and the various transactions that may be performed. The user interface may be a generated dialog in which a user is prompted for information.

The user interface itself need not be part of the repository. As a repository may be embedded in some other device, the user interface may merely be a part of the device in which the repository is embedded. For example, the repository could be embedded in a "card" that is inserted into an available slot in a computer system. The user interface may be a combination of a display, keyboard, cursor control device and software executing on the computer system.

At a minimum, the user interface must permit a user to input information such as access requests and alpha numeric data and provide feedback as to transaction status. The user interface will then cause the repository to initiate the suitable transactions to service the request. Other facets of a particular user interface will depend on the functionality that a repository will provide.

CREDIT SERVERS

In the present invention, fees may be associated with the exercise of a right. The requirement for payment of fees is described with each version of a usage right in the usage rights language. The recording and reporting of such fees is performed by the credit server. One of the capabilities enabled by associating fees with rights is the possibility of supporting a wide range of charging models. The simplest model, used by conventional software, is that there is a single fee at the time of purchase, after which the purchaser obtains unlimited rights to use the work as often and for as long as he or she wants. Alternative models, include metered use and variable fees. A single work can have different fees for different uses. For example, viewing a photograph on a display could have different fees than making a hardcopy

or including it in a newly created work. A key to these alternative charging models is to have a low overhead means of establishing fees and accounting for credit on these transactions.

A credit server is a computational system that reliably authorizes and records these transactions so that fees are billed and paid. The credit server reports fees to a billing clearinghouse. The billing clearinghouse manages the financial transactions as they occur. As a result, bills may be generated and accounts reconciled. Preferably, the credit server would store the fee transactions and periodically communicate via a network with the billing clearinghouse for reconciliation. In such an embodiment, communications with the billing clearinghouse would be encrypted for integrity and security reasons. In another embodiment, the credit server acts as a "debit card" where transactions occur in "real-time" against a user account.

A credit server is comprised of memory, a processing means, a clock, and interface means for coupling to a repository and a financial institution (e.g. a modem). The credit server will also need to have security and authentication functionality. These elements are essentially the same elements as those of a repository. Thus, a single device can be both a repository and a credit server, provided that it has the appropriate processing elements for carrying out the corresponding functions and protocols. Typically, however, a credit server would be a cardsized system in the possession of the owner of the credit. The credit server is coupled to a repository and would interact via financial transactions as described below. Interactions with a financial institution may occur via protocols established by the financial institutions themselves.

In the currently preferred embodiment credit servers associated with both the server and the repository report the financial transaction to the billing clearinghouse. For example, when a digital work is copied by one repository to another for a fee, credit servers coupled to each of the repositories will report the transaction to the billing clearinghouse. This is desirable in that it insures that a transaction will be accounted for in the event of some break in the communication between a credit server and the billing clearinghouse. However, some implementations may embody only a single credit server reporting the transaction to minimize transaction processing at the risk of losing some transactions.

USAGE RIGHTS LANGUAGE

The present invention uses statements in a high level "usage rights language" to define rights associated with digital works and their parts. Usage rights statements are interpreted by repositories and are used to determine what transactions can be successfully carried out for a digital work and also to determine parameters for those transactions. For example, sentences in the language determine whether a given digital work can be copied, when and how it can be used, and what fees (if any) are to be charged for that use. Once the usage rights statements are generated, they are encoded in a suitable form for accessing during the processing of transactions.

Defining usage rights in terms of a language in combination with the hierarchical representation of a digital work enables the support of a wide variety of distribution and fee schemes. An example is the ability to attach multiple versions of a right to a work. So a creator may attach a PRINT right to make 5 copies for $10.00 and a PRINT right to make unlimited copies for $100.00. A purchaser may then choose which option best fits his needs. Another example is that rights and fees are additive. So in the case of a composite work, the rights and fees of each of the components works is used in determining the rights and fees for the work as a whole.

The basic contents of a right are illustrated in Figure 14. Referring to Figure 14, a right 1450 has a transactional component 1451 and a specifications component 1452. A right 1450 has a label (e.g. COPY or PRINT) which indicates the use or distribution privileges that are embodied by the right. The transactional component 1451 corresponds to a particular way in which a digital work may be used or distributed. The transactional component 1451 is typically embodied in software instructions in a repository which implement the use or distribution privileges for the right. The specifications components 1452 are used to specify conditions which must be satisfied prior to the right being exercised or to designate various transaction related parameters. In the currently preferred embodiment, these specifications include copy count 1453, Fees and Incentives 1454, Time 1455, Access and Security 1456 and Control 1457. Each of these specifications will be described in greater detail below with respect to the language grammar elements.

The usage rights language is based on the grammar described below. A grammar is a convenient means for defining valid sequence of symbols for a language. In describing the grammar the notation "[a|b|c]" is used to indicate distinct choices among alternatives. In this example, a sentence can have either an "a", "b" or "c". It must include exactly one of them. The braces {} are used to indicate optional items. Note that brackets, bars and braces are used to describe the language of usage rights sentences but do not appear in actual sentences in the language.

In contrast, parentheses are part of the usage rights language. Parentheses are used to group items together in lists. The notation (x^*) is used to indicate a variable length list, that is, a list containing one or more items of type x. The notation $(x)^*$ is used to indicate a variable number of lists containing x.

Keywords in the grammar are words followed by colons. Keywords are a common and very special case in the language. They are often used to indicate a single value, typically an identifier. In many cases, the keyword and the parameter are entirely optional. When a keyword is given, it often takes a single identifier as its value. In some cases,

the keyword takes a list of identifiers.

In the usage rights language, time is specified in an hours:minutes:seconds (or hh:mm:ss) representation. Time zone indicators, e.g. PDT for Pacific Daylight Time, may also be specified. Dates are represented as year/ month/day (or YYYY/MMM/DD). Note that these time and date representations may specify moments in time or units of time Money units are specified in terms of dollars.

Finally, in the usage rights language, various "things" will need to interact with each other. For example, an instance of a usage right may specify a bank account, a digital ticket, etc.. Such things need to be identified and are specified herein using the suffix "-ID."

The Usage Rights Grammar is listed in its entirety in Figure 15 and is described below.

Grammar element 1501 **"Digital Work Rights: = (Rights*)"** define the digital work rights as a set of rights. The set of rights attached to a digital work define how that digital work may be transferred, used, performed or played. A set of rights will attach to the entire digital work and in the case of compound digital works, each of the components of the digital work. The usage rights of components of a digital may be different.

Grammar element 1502 **"Right : = (Right-Code {Copy-Count} {Control-Spec} {Time-Spec} {Access-Spec} {Fee-Spec})"** enumerates the content of a right. Each usage right must specify a right code. Each right may also optionally specify conditions which must be satisfied before the right can be exercised. These conditions are copy count, control, time, access and fee conditions. In the currently preferred embodiment, for the optional elements, the following defaults apply: copy count equals 1, no time limit on the use of the right, no access tests or a security level required to use the right and no fee is required. These conditions will each be described in greater detail below.

It is important to note that a digital work may have multiple versions of a right, each having the same right code. The multiple version would provide alternative conditions and fees for accessing the digital work.

Grammar element 1503 **"Right-Code : = Render-Code I Transport-Code I File-Management-CodeI Derivative-Works- Code Configuration-Code"** distinguishes each of the specific rights into a particular right type (although each right is identified by distinct right codes). In this way, the grammar provides a catalog of possible rights that can be associated with parts of digital works. In the following, rights are divided into categories for convenience in describing them.

Grammar element 1504 **"Render-Code : = [Play : {Player: Player-ID} I Print: {Printer: Printer-ID}]"** lists a category of rights all involving the making of ephemeral, transitory, or non-digital copies of the digital work. After use the copies are erased.

- Play — A process of rendering or performing a digital work on some processor. This includes such things as playing digital movies, playing digital music, playing a video game, running a computer program, or displaying a document on a display.
- Print — To render the work in a medium that is not further protected by usage rights, such as printing on paper.

Grammar element 1505 **"Transport-Code : = [Copy I Transfer I Loan (Remaining-Rights: Next-Set-of-Rights)] {(Next-Copy-Rights: Next-Set of Rights)}"** lists a category of rights involving the making of persistent, usable copies of the digital work on other repositories. The optional Next-Copy-Rights determine the rights on the work after it is transported. If this is not specified, then the rights on the transported copy are the same as on the original. The optional Remaining-Rights specify the rights that remain with a digital work when it is loaned out. If this is not specified, then the default is that no rights can be exercised when it is loaned out.

- Copy — Make a new copy of a work
- Transfer — Moving a work from one repository to another.
- Loan — Temporarily loaning a copy to another repository for a specified period of time.

Grammar element 1506 **"File-Management-Code : = Backup {Back-Up-Copy-Rights: Next-Set -of Rights}I Restore I Delete I Folder I Directory {Name:Hide-Local I Hide - Remote}{Parts:Hide-Local I Hide-Remote}"** lists a category of rights involving operations for file management, such as the making of backup copies to protect the copy owner against catastrophic equipment failure.

Many software licenses and also copyright law give a copy owner the right to make backup copies to protect against catastrophic failure of equipment. However, the making of uncontrolled backup copies is inherently at odds with the ability to control usage, since an uncontrolled backup copy can be kept and then restored even after the authorized copy was sold.

The File management rights enable the making and restoring of backup copies in a way that respects usage rights, honoring the requirements of both the copy owner and the rights grantor and revenue owner. Backup copies of work descriptions (including usage rights and fee data) can be sent under appropriate protocol and usage rights control to other document repositories of sufficiently high security. Further rights permit organization of digital works into folders

which themselves are treated as digital works and whose contents may be "hidden" from a party seeking to determine the contents of a repository.

- Backup To make a backup copy of a digital work as protection against media failure.
- Restore To restore a backup copy of a digital work.
- Delete To delete or erase a copy of a digital work.
- Folder To create and name folders, and to move files and folders between folders.
- Directory To hide a folder or its contents.

Grammar element 1507 **"Derivative-Works-Code : [Extract I Embed I Edit {Process: Process-ID}] {Next-Copy-Rights : Next-Set-of Rights}"** lists a category of rights involving the use of a digital work to create new works.

- Extract To remove a portion of a work, for the purposes of creating a new work.
- Embed To include a work in an existing work.
- Edit To alter a digital work by copying, selecting and modifying portions of an existing digital work.

Grammar element 1508 **"Configuration-Code: = Install I Uninstall"** lists a category of rights for installing and uninstalling software on a repository (typically a rendering repository.) This would typically occur for the installation of a new type of player within the rendering repository.

- Install: To install new software on a repository.
- Uninstall: To remove existing software from a repository.

Grammar element 1509 **"Next-Set-of-Rights : = {(Add: Set-Of-Rights)} {(Delete: Set-Of-Rights)} {(Replace: Set-Of-Rights)} {(Keep: Set-Of-Rights)}"** defines how rights are carried forward for a copy of a digital work. If the Next-Copy-Rights is not specified, the rights for the next copy are the same as those of the current copy. Otherwise, the set of rights for the next copy can be specified. Versions of rights after Add: are added to the current set of rights. Rights after Delete: are deleted from the current set of rights. If only right codes are listed after Delete:, then all versions of rights with those codes are deleted. Versions of rights after Replace: subsume all versions of rights of the same type in the current set of rights.

If Remaining-Rights is not specified, then there are no rights for the original after all Loan copies are loaned out. If Remaining-Rights is specified, then the Keep: token can be used to simplify the expression of what rights to keep behind. A list of right codes following keep means that all of the versions of those listed rights are kept in the remaining copy. This specification can be overridden by subsequent Delete: or Replace: specifications.

Copy Count Specification

For various transactions, it may be desirable to provide some limit as to the number of "copies" of the work which may be exercised simultaneously for the right. For example, it may be desirable to limit the number of copies of a digital work that may be loaned out at a time or viewed at a time.

Grammar element 1510 **"Copy-Count : = (Copies: positive-integer I 0 I unlimited)"** provides a condition which defines the number of "copies" of a work subject to the right . A copy count can be 0, a fixed number, or unlimited. The copy-count is associated with each right, as opposed to there being just a single copy-count for the digital work. The Copy-Count for a right is decremented each time that a right is exercised. When the Copy-Count equals zero, the right can no longer be exercised. If the Copy-Count is not specified, the default is one.

Control Specification

Rights and fees depend in general on rights granted by the creator as well as further restrictions imposed by later distributors. Control specifications deal with interactions between the creators and their distributors governing the imposition of further restrictions and fees. For example, a distributor of a digital work may not want an end consumer of a digital work to add fees or otherwise profit by commercially exploiting the purchased digital work.

Grammar element 1511 **"Control-Spec : = (Control: {Restrictable I Unrestrictable} {Unchargeable I Chargeable})"** provides a condition to specify the effect of usage rights and fees of parents on the exercise of the right. A digital work is restrictable if higher level d-blocks can impose further restrictions (time specifications and access specifications) on the right. It is unrestrictable if no further restrictions can be imposed. The default setting is restrictable. A right is unchargeable if no more fees can be imposed on the use of the right. It is chargeable if more fees can be imposed. The default is chargeable.

Time Specification

It is often desirable to assign a start date or specify some duration as to when a right may be exercised. Grammar element 1512 **"Time-Spec : = ({Fixed-Interval | Sliding-Interval | Meter-Time} Until: Expiration-Date)"** provides for specification of time conditions on the exercise of a right. Rights may be granted for a specified time. Different kinds of time specifications are appropriate for different kinds of rights. Some rights may be exercised during a fixed and predetermined duration. Some rights may be exercised for an interval that starts the first time that the right is invoked by some transaction. Some rights may be exercised or are charged according to some kind of metered time, which may be split into separate intervals. For example, a right to view a picture for an hour might be split into six ten minute viewings or four fifteen minute viewings or twenty three minute viewings.

The terms "time" and "date" are used synonymously to refer to a moment in time. There are several kinds of time specifications. Each specification represents some limitation on the times over which the usage right applies. The Expiration-Date specifies the moment at which the usage right ends. For example, if the Expiration-Date is "Jan 1, 1995," then the right ends at the first moment of 1995. If the Expiration-Date is specified as "forever", then the rights are interpreted as continuing without end. If only an expiration date is given, then the right can be exercised as often as desired until the expiration date.

Grammar element 1513 **"Fixed-Interval : = From: Start-Time"** is used to define a predetermined interval that runs from the start time to the expiration date.

Grammar element 1514 **"Sliding-Interval : = Interval: Use-Duration"** is used to define an indeterminate (or "open") start time. It sets limits on a continuous period of time over which the contents are accessible. The period starts on the first access and ends after the duration has passed or the expiration date is reached, whichever comes first. For example, if the right gives 10 hours of continuous access, the use-duration would begin when the first access was made and end 10 hours later.

Grammar element 1515 **"Meter-Time: = Time-Remaining: Remaining-Use"** is used to define a "meter time," that is, a measure of the time that the right is actually exercised. It differs from the Sliding-Interval specification in that the time that the digital work is in use need not be continuous. For example, if the rights guarantee three days of access, those days could be spread out over a month. With this specification, the rights can be exercised until the meter time is exhausted or the expiration date is reached, whichever comes first.

Remaining-Use: = Time-Unit
Start-Time: = Time-Unit
Use-Duration: = Time-Unit
All of the time specifications include time-unit specifications in their ultimate instantiation.

Security Class and Authorization Specification

The present invention provides for various security mechanisms to be introduced into a distribution or use scheme. Grammar element 1516 **"Access-Spec : = ({SC: Security-Class} {Authorization: Authorization-ID*} {Other-Authorization: Authorization-ID*} {Ticket: Ticket-ID})"** provides a means for restricting access and transmission. Access specifications can specify a required security class for a repository to exercise a right or a required authorization test that must be satisfied.

The keyword **"SC:"** is used to specify a minimum security level for the repositories involved in the access. If "SC: " is not specified, the lowest security level is acceptable.

The optional **"Authorization:"** keyword is used to specify required authorizations on the same repository as the work. The optional **"Other-Authorization:"** keyword is used to specify required authorizations on the other repository in the transaction.

The optional **"Ticket:"** keyword specifies the identity of a ticket required for the transaction. A transaction involving digital tickets must locate an appropriate digital ticket agent who can "punch" or otherwise validate the ticket before the transaction can proceed. Tickets are described in greater detail below.

In a transaction involving a repository and a document server, some usage rights may require that the repository have a particular authorization, that the server have some authorization, or that both repositories have (possibly different) authorizations. Authorizations themselves are digital works (hereinafter referred to as an authorization object) that can be moved between repositories in the same manner as other digital works. Their copying and transferring is subject to the same rights and fees as other digital works. A repository is said to have an authorization if that authorization object is contained within the repository.

In some cases, an authorization may be required from a source other than the document server and repository. An authorization object referenced by an Authorization-ID can contain digital address information to be used to set up a communications link between a repository and the authorization source. These are analogous to phone numbers. For such access tests, the communication would need to be established and authorization obtained before the right

could be exercised.

For one-time usage rights, a variant on this scheme is to have a digital ticket. A ticket is presented to a digital ticket agent, whose type is specified on the ticket. In the simplest case, a certified generic ticket agent, available on all repositories, is available to "punch" the ticket. In other cases, the ticket may contain addressing information for locating a "special" ticket agent. Once a ticket has been punched, it cannot be used again for the same kind of transaction (unless it is unpunched or refreshed in the manner described below.) Punching includes marking the ticket with a timestamp of the date and time it was used. Tickets are digital works and can be copied or transferred between repositories according to their usage rights.

In the currently preferred embodiment, a "punched" ticket becomes "unpunched" or "refreshed" when it is copied or extracted. The Copy and Extract operations save the date and time as a property of the digital ticket. When a ticket agent is given a ticket, it can simply check whether the digital copy was made after the last time that it was punched. Of course, the digital ticket must have the copy or extract usage rights attached thereto.

The capability to unpunch a ticket is inportant in the following cases:

- A digital work is circulated at low cost with a limitation that it can be used only once.
- A digital work is circulated with a ticket that can be used once to give discounts on purchases of other works.
- A digital work is circulated with a ticket (included in the purchase price and possibly embedded in the work) that can be used for a future upgrade.

In each of these cases, if a paid copy is made of the digital work (including the ticket) the new owner would expect to get a fresh (unpunched) ticket, whether the copy seller has used the work or not. In contrast, loaning a work or simply transferring it to another repository should not revitalize the ticket.

Usage Fees and Incentives Specification

The billing for use of a digital work is fundamental to a commercial distribution system. Grammar Element 1517 **"Fee-Spec: = {Scheduled-Discount} Regular-Fee-Spec I Scheduled-Fee-Spec I Markup-Spec"** provides a range of options for billing for the use of digital works.

A key feature of this approach is the development of low-overhead billing for transactions in potentially small amounts. Thus, it becomes feasible to collect fees of only a few cents each for thousands of transactions.

The grammar differentiates between uses where the charge is per use from those where it is metered by the time unit. Transactions can support fees that the user pays for using a digital work as well as incentives paid by the right grantor to users to induce them to use or distribute the digital work.

The optional scheduled discount refers to the rest of the fee specification--discounting it by a percentage over time. If it is not specified, then there is no scheduled discount. Regular fee specifications are constant over time. Scheduled fee specifications give a schedule of dates over which the fee specifications change. Markup specifications are used in d-blocks for adding a percentage to the fees already being charged.

Grammar Element 1518 **"Scheduled-Discount: = (Scheduled-Discount: (Time-Spec Percentage)*)"** A Scheduled-Discount is a essentially a scheduled modifier of any other fee specification for this version of the right of the digital work. (It does not refer to children or parent digital works or to other versions of rights.). It is a list of pairs of times and percentages. The most recent time in the list that has not yet passed at the time of the transaction is the one in effect. The percentage gives the discount percentage. For example, the number 10 refers to a 10% discount.

Grammar Element 1519 **"Regular-Fee-Spec : = ({Fee: I Incentive: } [Per-Use-Spec I Metered-Rate-Spec I Best-Price-Spec I Call-For-Price-Spec] {Min: Money-Unit Per: Time-Spec}{Max: Money-Unit Per: Time-Spec} To: Account-ID)"** provides for several kinds of fee specifications.

Fees are paid by the copy-owner/user to the revenue-owner if **Fee**: is specified. Incentives are paid by the revenue-owner to the user if Incentive: is specified. If the **Min**: specification is given, then there is a minimum fee to be charged per time-spec unit for its use. If the **Max**: specification is given, then there is a maximum fee to be charged per time-spec for its use. When **Fee**: is specified, **Account-ID** identifies the account to which the fee is to be paid. When Incentive: is specified, **Account-ID** identifies the account from which the fee is to be paid.

Grammar element 1520 **"Per-Use-Spec: = Per-Use: Money-unit"** defines a simple fee to be paid every time the right is exercised, regardless of how much time the transaction takes.

Grammar element 1521 **"Metered-Rate-Spec := Metered: Money-Unit Per: Time-Spec"** defines a metered-rate fee paid according to how long the right is exercised. Thus, the time it takes to complete the transaction determines the fee.

Grammar element 1522 **"Best-Price-Spec := Best-Price: Money-unit Max: Money-unit"** is used to specify a best-price that is determined when the account is settled. This specification is to accommodate special deals, rebates, and pricing that depends on information that is not available to the repository. All fee specifications can be combined

with tickets or authorizations that could indicate that the consumer is a wholesaler or that he is a preferred customer, or that the seller be authorized in some way. The amount of money in the **Max**: field is the maximum amount that the use will cost. This is the amount that is tentatively debited from the credit server. However, when the transaction is ultimately reconciled, any excess amount will be returned to the consumer in a separate transaction.

Grammar element 1523 **"Call-For-Price-Spec : = Call-For-Price "** is similar to a **"Best-Price-Spec"** in that it is intended to accommodate cases where prices are dynamic. **A Call-For-Price Spec** requires a communication with a dealer to determine the price. This option cannot be exercised if the repository cannot communicate with a dealer at the time that the right is exercised. It is based on a secure transaction whereby the dealer names a price to exercise the right and passes along a deal certificate which is referenced or included in the billing process.

Grammar element 1524 **"Scheduled-Fee-Spec: = (Schedule: (Time-Spec Regular-Fee-Spec)*)"** is used to provide a schedule of dates over which the fee specifications change. The fee specification with the most recent date not in the future is the one that is in effect. This is similar to but more general than the scheduled discount. It is more general, because it provides a means to vary the fee agreement for each time period.

Grammar element 1525 **"Markup-Spec: = Markup: percentage To: Account-ID"** is provided for adding a percentage to the fees already being charged. For example, a 5% markup means that a fee of 5% of cumulative fee so far will be allocated to the distributor. A markup specification can be applied to all of the other kinds of fee specifications. It is typically used in a shell provided by a distributor. It refers to fees associated with d-blocks that are parts of the current d-block. This might be a convenient specification for use in taxes, or in distributor overhead.

REPOSITORY TRANSACTIONS

When a user requests access to a digital work, the repository will initiate various transactions. The combination of transactions invoked will depend on the specifications assigned for a usage right. There are three basic types of transactions, Session Initiation Transactions, Financial Transactions and Usage Transactions. Generally, session initiation transactions are initiated first to establish a valid session. When a valid session is established, transactions corresponding to the various usage rights are invoked. Finally, request specific transactions are performed.

Transactions occur between two repositories (one acting as a server), between a repository and a document playback platform (e.g. for executing or viewing), between a repository and a credit server or between a repository and an authorization server. When transactions occur between more than one repository, it is assumed that there is a reliable communication channel between the repositories. For example, this could be a TCP/IP channel or any other commercially available channel that has built-in capabilities for detecting and correcting transmission errors. However, it is not assumed that the communication channel is secure. Provisions for security and privacy are part of the requirements for specifying and implementing repositories and thus form the need for various transactions.

Message Transmission

Transactions require that there be some communication between repositories. Communication between repositories occurs in units termed as messages. Because the communication line is assumed to be unsecure, all communications with repositories that are above the lowest security class are encrypted utilizing a public key encryption technique. Public key encryption is a well known technique in the encryption arts. The term key refers to a numeric code that is used with encryption and decryption algorithms. Keys come in pairs, where "writing keys" are used to encrypt data and "checking keys" are used to decrypt data. Both writing and checking keys may be public or private. Public keys are those that are distributed to others Private keys are maintained in confidence.

Key management and security is instrumental in the success of a public key encryption system. In the currently preferred embodiment, one or more master repositories maintain the keys and create the identification certificates used by the repositories.

When a sending repository transmits a message to a receiving repository, the sending repository encrypts all of its data using the public writing key of the receiving repository. The sending repository includes its name, the name of the receiving repository, a session identifier such as a nonce (described below), and a message counter in each message.

In this way, the communication can only be read (to a high probability) by the receiving repository, which holds the private checking key for decryption. The auxiliary data is used to guard against various replay attacks to security. If messages ever arrive with the wrong counter or an old nonce, the repositories can assume that someone is interfering with communication and the transaction terminated.

The respective public keys for the repositories to be used for encryption are obtained in the registration transaction described below.

Session Initiation Transactions

A usage transaction is carried out in a session between repositories. For usage transactions involving more than one repository, or for financial transactions between a repository and a credit server, a registration transaction is performed. A second transaction termed a login transaction, may also be needed to initiate the session. The goal of the registration transaction is to establish a secure channel between two repositories who know each others identities. As it is assumed that the communication channel between the repositories is reliable but not secure, there is a risk that a non-repository may mimic the protocol in order to gain illegitimate access to a repository.

The registration transaction between two repositories is described with respect to Figures 16 and 17. The steps described are from the perspective of a "repository-1" registering its identity with a "repository-2". The registration must be symmetrical so the same set of steps will be repeated for repository-2 registering its identity with repository-1. Referring to Figure 16, repository-1 first generates an encrypted registration identifier, step 1601 and then generates a registration message, step 1602. A registration message is comprised of an identifier of a master repository, the identification certificate for the repository-1 and an encrypted random registration identifier. The identification certificate is encrypted by the master repository in its private key and attests to the fact that the repository (here repository-1) is a bona fide repository. The identification certificate also contains a public key for the repository, the repository security level and a timestamp (indicating a time after which the certificate is no longer valid.) The registration identifier is a number generated by the repository for this registration. The registration identifier is unique to the session and is encrypted in repository-1's private key. The registration identifier is used to improve security of authentication by detecting certain kinds of communications based attacks. Repository-1 then transmits the registration message to repository-2, step 1603.

Upon receiving the registration message, repository-2 determines if it has the needed public key for the master repository, step 1604. If repository-2 does not have the needed public key to decrypt the identification certificate, the registration transaction terminates in an error, step 1618.

Assuming that repository-2 has the proper public key the identification certificate is decrypted, step 1605. Repository-2 saves the encrypted registration identifier, step 1606, and extracts the repository identifier, step 1607. The extracted repository identifier is checked against a "hotlist" of compromised document repositories, step 1608. In the currently preferred embodiment, each repository will contain "hotlists" of compromised repositories. If the repository is on the "hotlist', the registration transaction terminates in an error per step 1618. Repositories can be removed from the hotlist when their certificates expire, so that the list does not need to grow without bound. Also, by keeping a short list of hotlist certificates that it has previously received, a repository can avoid the work of actually going through the list. These lists would be encrypted by a master repository. A minor variation on the approach to improve efficiency would have the repositories first exchange lists of names of hotlist certificates, ultimately exchanging only those lists that they had not previously received. The "hotlists" are maintained and distributed by Master repositories.

Note that rather than terminating in error, the transaction could request that another registration message be sent based on an identification certificate created by another master repository. This may be repeated until a satisfactory identification certificate is found, or it is determined that trust cannot be established.

Assuming that the repository is not on the hotlist, the repository identification needs to be verified. In other words, repository-2 needs to validate that the repository on the other end is really repository-1. This is termed performance testing and is performed in order to avoid invalid access to the repository via a counterfeit repository replaying a recording of a prior session initiation between repository-1 and repository-2. Performance testing is initiated by repository-2 generating a performance message, step 1609. The performance message consists of a nonce, the names of the respective repositories, the time and the registration identifier received from repository-1. A nonce is a generated message based on some random and variable information (e.g. the time or the temperature.) The nonce is used to check whether repository-1 can actually exhibit correct encrypting of a message using the private keys it claims to have, on a message that it has never seen before. The performance message is encrypted using the public key specified in the registration message of repository-1. The performance message is transmitted to repository-1, step 1610, where it is decrypted by repository-1 using its private key, step 1611. Repository-1 then checks to make sure that the names of the two repositories are correct, step 1612, that the time is accurate, step 1613 and that the registration identifier corresponds to the one it sent, step 1614. If any of these tests fails, the transaction is terminated per step 1616. Assuming that the tests are passed, repository-1 transmits the nonce to repository-2 in the clear, step 1615. Repository-2 then compares the received nonce to the original nonce, step 1617. If they are not identical, the registration transaction terminates in an error per step 1618. If they are the same, the registration transaction has successfully completed.

At this point, assuming that the transaction has not terminated, the repositories exchange messages containing session keys to be used in all communications during the session and synchronize their clocks. Figure 17 illustrates the session information exchange and clock synchronization steps (again from the perspective of repository-1.) Referring to Figure 17, repository-1 creates a session key pair, step 1701. A first key is kept private and is used by repository-1 to encrypt messages. The second key is a public key used by repository-2 to decrypt messages. The

second key is encrypted using the public key of repository-2, step 1702 and is sent to repository-2, step 1703. Upon receipt, repository-2 decrypts the second key, step 1704. The second key is used to decrypt messages in subsequent communications. When each repository has completed this step, they are both convinced that the other repository is bona fide and that they are communicating with the original. Each repository has given the other a key to be used in decrypting further communications during the session. Since that key is itself transmitted in the public key of the receiving repository only it will be able to decrypt the key which is used to decrypt subsequent messages.

After the session information is exchanged, the repositories must synchronize their clocks Clock synchronization is used by the repositories to establish an agreed upon time base for the financial records of their mutual transactions. Referring back to Figure 17, repository-2 initiates clock synchronization by generating a time stamp exchange message, step 1705, and transmits it to repository- 1, step 1706. Upon receipt, repository-1 generates its own time stamp message, step 1707 and transmits it back to repository-2, step 1708. Repository-2 notes the current time, step 1709 and stores the time received from repository-1, step 1710. The current time is compared to the time received from repository-1, step 1711. The difference is then checked to see if it exceeds a predetermined tolerance (e.g. one minute), step 1712. If it does, repository-2 terminates the transaction as this may indicate tampering with the repository, step 1713. If not repository-2 computes an adjusted time delta, step 1714. The adjusted time delta is the difference between the clock time of repository-2 and the average of the times from repository-1 and repository-2.

To achieve greater accuracy, repository-2 can request the time again up to a fixed number of times (e.g. five times), repeat the clock synchronization steps, and average the results.

A second session initiation transaction is a Login transaction. The Login transaction is used to check the authenticity of a user requesting a transaction. A Login transaction is particularly prudent for the authorization of financial transactions that will be charged to a credit server. The Login transaction involves an interaction between the user at a user interface and the credit server associated with a repository. The information exchanged here is a login string supplied by the repository/credit server to identify itself to the user, and a Personal Identification Number (PIN) provided by the user to identify himself to the credit server. In the event that the user is accessing a credit server on a repository different from the one on which the user interface resides, exchange of the information would be encrypted using the public and private keys of the respective repositories.

Billing Transactions

Billing Transactions are concerned with monetary transactions with a credit server. Billing Transactions are carried out when all other conditions are satisfied and a usage fee is required for granting the request. For the most part, billing transactions are well understood in the state of the art. These transactions are between a repository and a credit server, or between a credit server and a billing clearinghouse. Briefly, the required transactions include the following:

- Registration and LOG IN transactions by which the repository and user establish their bona fides to a credit server. These transactions would be entirely internal in cases where the repository and credit server are implemented as a single system.
- Registration and LOG IN transactions, by which a credit server establishes its bona fides to a billing clearinghouse.
- An Assign-fee transaction to assign a charge. The information in this transaction would include a transaction identifier, the identities of the repositories in the transaction, and a list of charges from the parts of the digital work. If there has been any unusual event in the transaction such as an interruption of communications, that information is included as well.
- A Begin-charges transaction to assign a charge. This transaction is much the same as an assign-fee transaction except that it is used for metered use. It includes the same information as the assign-fee transaction as well as the usage fee information. The credit-server is then responsible for running a clock.
- An End-charges transaction to end a charge for metered use. (In a variation on this approach, the repositories would exchange periodic charge information for each block of time.)
- A report-charges transaction between a personal credit server and a billing clearinghouse. This transaction is invoked at least once per billing period. It is used to pass along information about charges. On debit and credit cards, this transaction would also be used to update balance information and credit limits as needed.

All billing transactions are given a transaction ID and are reported to the credit severs by both the server and the client. This reduces possible loss of billing information if one of the parties to a transaction loses a banking card and provides a check against tampering with the system.

Usage Transactions

After the session initiation transactions have been completed, the usage request may then be processed. To sim-

plify the description of the steps carried out in processing a usage request, the term requester is used to refer to a repository in the requester mode which is initiating a request, and the term server is used to refer to a repository in the server mode and which contains the desired digital work. In many cases such as requests to print or view a work, the requester and server may be the same device and the transactions described in the following would be entirely internal. In such instances, certain transaction steps, such as the registration transaction, need not be performed.

There are some common steps that are part of the semantics of all of the usage rights transactions. These steps are referred to as the common transaction steps. There are two sets --the "opening" steps and the "closing" steps. For simplicity, these are listed here rather than repeating them in the descriptions of all of the usage rights transactions.

Transactions can refer to a part of a digital work, a complete digital work, or a Digital work containing other digital works. Although not described in detail herein, a transaction may even refer to a folder comprised of a plurality of digital works. The term "work" is used to refer to what ever portion or set of digital works is being accessed.

Many of the steps here involve determining if certain conditions are satisfied. Recall that each usage right may have one or more conditions which must be satisfied before the right can be exercised. Digital works have parts and parts have parts. Different parts can have different rights and fees. Thus, it is necessary to verify that the requirements are met for ALL of the parts that are involved in a transaction For brevity, when reference is made to checking whether the rights exist and conditions for exercising are satisfied, it is meant that all such checking takes place for each of the relevant parts of the work.

Figure 18 illustrates the initial common opening and closing steps for a transaction. At this point it is assumed that registration has occurred and that a "trusted" session is in place. General tests are tests on usage rights associated with the folder containing the work or some containing folder higher in the file system hierarchy. These tests correspond to requirements imposed on the work as a consequence of its being on the particular repository, as opposed to being attached to the work itself. Referring to Figure 18, prior to initiating a usage transaction, the requester performs any general tests that are required before the right associated with the transaction can be exercised, step, 1801. For example, install, uninstall and delete rights may be implemented to require that a requester have an authorization certificate before the right can be exercised. Another example is the requirement that a digital ticket be present and punched before a digital work may be copied to a requester. If any of the general tests fail, the transaction is not initiated, step, 1802. Assuming that such required tests are passed, upon receiving the usage request, the server generates a transaction identifier that is used in records or reports of the transaction, step 1803. The server then checks whether the digital work has been granted the right corresponding to the requested transaction, step 1804. If the digital work has not been granted the right corresponding to the request, the transaction terminates, step 1805. If the digital work has been granted the requested right, the server then determines if the various conditions for exercising the right are satisfied. Time based conditions are examined, step 1806. These conditions are checked by examining the time specification for the the version of the right. If any of the conditions are not satisfied, the transaction terminates per step 1805.

Assuming that the time based conditions are satisfied, the server checks security and access conditions, step 1807. Such security and access conditions are satisfied if: 1) the requester is at the specified security class, or a higher security class, 2) the server satisfies any specified authorization test and 3) the requester satisfies any specified authorization tests and has any required digital tickets. If any of the conditions are not satisfied, the transaction terminates per step 1805.

Assuming that the security and access conditions are all satisfied, the server checks the copy count condition, step 1808. If the copy count equals zero, then the transaction cannot be completed and the transaction terminates per step 1805.

Assuming that the copy count does not equal zero, the server checks if the copies in use for the requested right is greater than or equal to any copy count for the requested right (or relevant parts), step 1809. If the copies in use is greater than or equal to the copy count, this indicates that usage rights for the version of the transaction have been exhausted. Accordingly, the server terminates the transaction, step 1805. If the copy count is less than the copies in use for the transaction the transaction can continue, and the copies in use would be incremented by the number of digital works requested in the transaction, step 1810.

The server then checks if the digital work has a "Loan" access right, step 1811. The "Loan" access right is a special case since remaining rights may be present even though all copies are loaned out. If the digital work has the "Loan" access right, a check is made to see if all copies have been loaned out, step 1812. The number of copies that could be loaned is the sum of the Copy-Counts for all of the versions of the loan right of the digital work. For a composite work, the relevant figure is the minimal such sum of each of the components of the composite work. If all copies have been loaned out, the remaining rights are determined, step 1813. The remaining-rights is determined from the remaining rights specifications from the versions of the Loan right. If there is only one version of the Loan right, then the determination is simple. The remaining rights are the ones specified in that version of the Loan right, or none if Remaining-Rights: is not specified. If there are multiple versions of the Loan right and all copies of all of the versions are loaned out, then the remaining rights is taken as the minimum set (intersection) of remaining rights across all of the versions of the loan right. The server then determines if the requested right is in the set of remaining rights, step 1814. If the

requested right is not in the set of remaining rights, the server terminates the transaction, step 1805.

If Loan is not a usage right for the digital work or if all copies have not been loaned out or the requested right is in the set of remaining rights, fee conditions for the right are then checked, step 1815. This will initiate various financial transactions between the repository and associated credit server. Further, any metering of usage of a digital work will commence. If any financial transaction fails, the transaction terminates per step 1805.

It should be noted that the order in which the conditions are checked need not follow the order of steps 1806-1815.

At this point, right specific steps are now performed and are represented here as step 1816. The right speci fi c steps are described in greater detail below.

The common closing transaction steps are now performed. Each of the closing transaction steps are performed by the server after a successful completion of a transaction. Referring back to Figure 18, the copies in use value for the requested right is decremented by the number of copies involved in the transaction, step 1817. Next, if the right had a metered usage fee specification, the server subtracts the elapsed time from the Remaining-Use-Time associated with the right for every part involved in the transaction, step 1818. Finally, if there are fee specifications associated with the right, the server initiates End-Charge financial transaction to confirm billing, step 1819.

Transmission Protocol

An important area to consider is the transmission of the digital work from the server to the requester. The transmission protocol described herein refers to events occurring after a valid session has been created. The transmission protocol must handle the case of disruption in the communications between the repositories. It is assumed that interference such as injecting noise on the communication channel can be detected by the integrity checks (e.g., parity, checksum, etc.) that are built into the transport protocol and are not discussed in detail herein.

The underlying goal in the transmission protocol is to preclude certain failure modes, such as malicious or accidental interference on the communications channel. Suppose, for example, that a user pulls a card with the credit server at a specific time near the end of a transaction. There should not be a vulnerable time at which "pulling the card" causes the repositories to fail to correctly account for the number of copies of the work that have been created. Restated, there should be no time at which a party can break a connection as a means to avoid payment after using a digital work.

If a transaction is interrupted (and fails), both repositories restore the digital works and accounts to their state prior to the failure, modulo records of the failure itself.

Figure 19 is a state diagram showing steps in the process of transmitting information during a transaction. Each box represents a state of a repository in either the server mode (above the central dotted line 1901) or in the requester mode (below the dotted line 1901). Solid arrows stand for transitions between states. Dashed arrows stand for message communications between the repositories. A dashed message arrow pointing to a solid transition arrow is interpreted as meaning that the transition takes place when the message is received. Unlabeled transition arrows take place unconditionally. Other labels on state transition arrows describe conditions that trigger the transition.

Referring now to Figure 19, the server is initially in a state 1902 where a new transaction is initiated via start message 1903. This message includes transaction information including a transaction identifier and a count of the blocks of data to be transferred. The requester, initially in a wait state 1904 then enters a data wait state 1905.

The server enters a data transmit state 1906 and transmits a block of data 1907 and then enters a wait for acknowledgement state 1908. As the data is received, the requester enters a data receive state 1909 and when the data blocks are completely received it enters an acknowledgement state 1910 and transmits an Acknowledgement message 1911 to the server.

If there are more blocks to send, the server waits until receiving an Acknowledgement message from the requester. When an Acknowledgement message is received it sends the next block to the requester and again waits for acknowledgement. The requester also repeats the same cycle of states.

If the server detects a communications failure before sending the last block, it enters a cancellation state 1912 wherein the transaction is cancelled. Similarly, if the requester detects a communications failure before receiving the last block it enters a cancellation state 1913.

If there are no more blocks to send, the server commits to the transaction and waits for the final Acknowledgement in state 1914. If there is a communications failure before the server receives the final Acknowledgement message, it still commits to the transaction but includes a report about the event to its credit server in state 1915. This report serves two purposes. It will help legitimize any claims by a user of having been billed for receiving digital works that were not completely received. Also it helps to identify repositories and communications lines that have suspicious patterns of use and interruption. The server then enters its completion state 1916.

On the requester side, when there are no more blocks to receive, the requester commits to the transaction in state 1917. If the requester detects a communications failure at this state, it reports the failure to its credit server in state 1918, but still commits to the transaction. When it has committed, it sends an acknowledgement message to the server. The server then enters its completion state 1919.

The key property is that both the server and the requester cancel a transaction if it is interrupted before all of the data blocks are delivered, and commits to it if all of the data blocks have been delivered.

There is a possibility that the server will have sent all of the data blocks (and committed) but the requester will not have received all of them and will cancel the transaction. In this case, both repositories will presumably detect a communications failure and report it to their credit server. This case will probably be rare since it depends on very precise timing of the communications failure. The only consequence will be that the user at the requester repository may want to request a refund from the credit services -- and the case for that refund will be documented by reports by both repositories.

To prevent loss of data, the server should not delete any transferred digital work until receiving the final acknowledgement from the requester. But it also should not use the file. A well known way to deal with this situation is called "two-phase commit" or 2PC.

Two-phase commit works as follows. The first phase works the same as the method described above. The server sends all of the data to the requester. Both repositories mark the transaction (and appropriate files) as uncommitted. The server sends a ready-to-commit message to the requester. The requester sends back an acknowledgement. The server then commits and sends the requester a commit message. When the requester receives the commit message, it commits the file.

If there is a communication failure or other crash, the requester must check back with the server to determine the status of the transaction. The server has the last word on this. The requester may have received all of the data, but if it did not get the final message, it has not committed. The server can go ahead and delete files (except for transaction records) once it commits, since the files are known to have been fully transmitted before starting the 2PC cycle.

There are variations known in the art which can be used to achieve the same effect. For example, the server could use an additional level of encryption when transmitting a work to a client. Only after the client sends a message acknowledging receipt does it send the key. The client then agrees to pay for the digital work. The point of this variation is that it provides a clear audit trail that the client received the work. For trusted systems, however, this variation adds a level of encryption for no real gain in accountability.

The transaction for specific usage rights are now discussed.

The Copy Transaction

A Copy transaction is a request to make one or more independent copies of the work with the same or lesser usage rights. Copy differs from the extraction right discussed later in that it refers to entire digital works or entire folders containing digital works. A copy operation cannot be used to remove a portion of a digital work.

- The requester sends the server a message to initiate the Copy Transaction. This message indicates the work to be copied, the version of the copy right to be used for the transaction, the destination address information (location in a folder) for placing the work, the file data for the work (including its size), and the number of copies requested.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the client according to the transmission protocol. If a Next-Set-Of-Rights has been provided in the version of the right, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted. In any event, the Copy-Count field for the copy of the digital work being sent right is set to the number-of-copies requested.
- The requester records the work contents, data, and usage rights and stores the work. It records the date and time that the copy was made in the properties of the digital work.
- The repositories perform the common closing transaction steps.

The Transfer Transaction

A Transfer transaction is a request to move copies of the work with the same or lesser usage rights to another repository. In contrast with a copy transaction, this results in removing the work copies from the server.

- The requester sends the server a message to initiate the Transfer Transaction. This message indicates the work to be transferred, the version of the transfer right to be used in the transaction, the destination address information for placing the work, the file data for the work, and the number of copies involved.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted. In either case, the Copy-Count field for the transmitted rights are set to the number-of-copies requested.

- The requester records the work contents, data, and usage rights and stores the work.
- The server decrements its copy count by the number of copies involved in the transaction.
- The repositories perform the common closing transaction steps.
- If the number of copies remaining in the server is now zero, it erases the digital work from its memory.

The Loan Transaction

A loan transaction is a mechanism for loaning copies of a digital work. The maximum duration of the loan is determined by an internal parameter of the digital work. Works are automatically returned after a predetermined time period.

- The requester sends the server a message to initiate the Transfer Transaction. This message indicates the work to be loaned, the version of the loan right to be used in the transaction, the destination address information for placing the work, the number of copies involved, the file data for the work, and the period of the loan.
- The server checks the validity of the requested loan period, and ends with an error if the period is not valid. Loans for a loaned copy cannot extend beyond the period of the original loan to the server.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted, as modified to reflect the loan period.
- The requester records the digital work contents, data, usage rights, and loan period and stores the work.
- The server updates the usage rights information in the digital work to reflect the number of copies loaned out.
- The repositories perform the common closing transaction steps.
- The server updates the usage rights data for the digital work. This may preclude use of the work until it is returned from the loan. The user on the requester platform can now use the transferred copies of the digital work. A user accessing the original repository cannot use the digital work , unless there are copies remaining. What happens next depends on the order of events in time.

Case 1. If the time of the loan period is not yet exhausted and the requester sends the repository a Return message.

- The return message includes the requester identification, and the transaction ID.
- The server decrements the copies-in-use field by the number of copies that were returned. (If the number of digital works returned is greater than the number actually borrowed, this is treated as an error.) This step may now make the work available at the server for other users.
- The requester deactivates its copies and removes the contents from its memory.

Case 2. If the time of the loan period is exhausted and the requester has not yet sent a Return message.

- The server decrements the copies-in-use field by the number digital works that were borrowed.
- The requester automatically deactivates its copies of the digital work. It terminates all current uses and erases the digital work copies from memory. One question is why a requester would ever return a work earlier than the period of the loan, since it would be returned automatically anyway. One reason for early return is that there may be a metered fee which determines the cost of the loan. Returning early may reduce that fee.

The Play Transaction

A play transaction is a request to use the contents of a work. Typically, to "play" a work is to send the digital work through some kind of transducer, such as a speaker or a display device. The request implies the intention that the contents will not be communicated digitally to any other system. For example, they will not be sent to a printer, recorded on any digital medium, retained after the transaction or sent to another repository.

This term "play" is natural for examples like playing music, playing a movie, or playing a video game. The general form of play means that a "player" is used to use the digital work. However, the term play covers all media and kinds of recordings. Thus one would "play" a digital work, meaning, to render it for reading, or play a computer program, meaning to execute it. For a digital ticket the player would be a digital ticket agent.

- The requester sends the server a message to initiate the play transaction. This message indicates the work to be played, the version of the play right to be used in the transaction, the identity of the player being used, and the file data for the work.

- The server checks the validity of the player identification and the compatibility of the player identification with the player specification in the right. It ends with an error if these are not satisfactory.
- The repositories perform the common opening transaction steps.
- The server and requester read and write the blocks of data as requested by the player according to the transmission protocol. The requester plays the work contents, using the player.
- When the player is finished, the player and the requester remove the contents from their memory.
- The repositories perform the common closing transaction steps.

The Print Transaction

A Print transaction is a request to obtain the contents of a work for the purpose of rendering them on a "printer." We use the term "printer" to include the common case of writing with ink on paper. However, the key aspect of "printing" in our use of the term is that it makes a copy of the digital work in a place outside of the protection of usage rights. As with all rights, this may require particular authorization certificates.

Once a digital work is printed, the publisher and user are bound by whatever copyright laws are in effect. However, printing moves the contents outside the control of repositories. For example, absent any other enforcement mechanisms, once a digital work is printed on paper, it can be copied on ordinary photocopying machines without intervention by a repository to collect usage fees. If the printer to a digital disk is permitted, then that digital copy is outside of the control of usage rights. Both the creator and the user know this, although the creator does not necessarily give tacit consent to such copying, which may violate copyright laws.

- The requester sends the server a message to initiate a Print transaction. This message indicates the work to be played, the identity of the printer being used, the file data for the work, and the number of copies in the request.
- The server checks the validity of the printer identification and the compatibility of the printer identification with the printer specification in the right. It ends with an error if these are not satisfactory.
- The repositories perform the common opening transaction steps.
- The server transmits blocks of data according to the transmission protocol.
- The requester prints the work contents, using the printer.
- When the printer is finished, the printer and the requester remove the contents from their memory.
- The repositories perform the common closing transaction steps.

The Backup Transaction

A Backup transaction is a request to make a backup copy of a digital work, as a protection against media failure. In the context of repositories, secure backup copies differ from other copies in three ways: (1) they are made under the control of a Backup transaction rather than a Copy transaction, (2) they do not count as regular copies, and (3) they are not usable as regular copies. Generally, backup copies are encrypted.

Although backup copies may be transferred or copied, depending on their assigned rights, the only way to make them useful for playing, printing or embedding is to restore them.

The output of a Backup operation is both an encrypted data file that contains the contents and description of a work, and a restoration file with an encryption key for restoring the encrypted contents. In many cases, the encrypted data file would have rights for "printing" it to a disk outside of the protection system, relying just on its encryption for security. Such files could be stored anywhere that was physically safe and convenient. The restoration file would be held in the repository. This file is necessary for the restoration of a backup copy. It may have rights for transfer between repositories.

- The requester sends the server a message to initiate a backup transaction. This message indicates the work to be backed up, the version of the backup right to be used in the transaction, the destination address information for placing the backup copy, the file data for the work.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, a set of default rights for backup files of the original are transmitted by the server.
- The requester records the work contents, data, and usage rights. It then creates a one-time key and encrypts the contents file. It saves the key information in a restoration file.
- The repositories perform the common closing transaction steps.

In some cases, it is convenient to be able to archive the large, encrypted contents file to secure offline storage,

such as a magneto-optical storage system or magnetic tape. This creation of a non-repository archive file is as secure as the encryption process. Such non-repository archive storage is considered a form of "printing" and is controlled by a print right with a specified "archive-printer." An archive-printer device is programmed to save the encrypted contents file (but not the description file) offline in such a way that it can be retrieved.

The Restore Transaction

A Restore transaction is a request to convert an encrypted backup copy of a digital work into a usable copy. A restore operation is intended to be used to compensate for catastrophic media failure. Like all usage rights, restoration rights can include fees and access tests including authorization checks.

- The requester sends the server a message to initiate a Restore transaction. This message indicates the work to be restored, the version of the restore right for the transaction, the destination address information for placing the work, and the file data for the work.
- The server verifies that the contents file is available (i.e. a digital work corresponding to the request has been backed-up.) If it is not, it ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The server retrieves the key from the restoration file. It decrypts the work contents, data, and usage rights.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, a set of default rights for backup files of the original are transmitted by the server.
- The requester stores the digital work.
- The repositories perform the common closing transaction steps.

The Delete Transaction

A Delete transaction deletes a digital work or a number of copies of a digital work from a repository. Practically all digital works would have delete rights.

- The requester sends the server a message to initiate a delete transaction. This message indicates the work to be deleted, the version of the delete right for the transaction.
- The repositories perform the common opening transaction steps.
- The server deletes the file, erasing it from the file system.
- The repositories perform the common closing transaction steps.

The Directory Transaction

A Directory transaction is a request for information about folders, digital works, and their parts. This amounts to roughly the same idea as protection codes in a conventional file system like TENEX, except that it is generalized to the full power of the access specifications of the usage rights language.

The Directory transaction has the important role of passing along descriptions of the rights and fees associated with a digital work. When a user wants to exercise a right, the user interface of his repository implicitly makes a directory request to determine the versions of the right that are available. Typically these are presented to the user -- such as with different choices of billing for exercising a right. Thus, many directory transactions are invisible to the user and are exercised as part of the normal process of exercising all rights.

- The requester sends the server a message to initiate a Directory transaction. This message indicates the file or folder that is the root of the directory request and the version of the directory right used for the transaction.
- The server verifies that the information is accessible to the requester. In particular, it does not return the names of any files that have a HIDE-NAME status in their directory specifications, and it does not return the parts of any folders or files that have HIDE-PARTS in their specification. If the information is not accessible, the server ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The server sends the requested data to the requester according to the transmission protocol.
- The requester records the data.
- The repositories perform the common closing transaction steps.

The Folder Transaction

A Folder transaction is a request to create or rename a folder, or to move a work between folders. Together with Directory rights, Folder rights control the degree to which organization of a repository can be accessed or modified from another repository.

- The requester sends the server a message to initiate a Folder transaction. This message indicates the folder that is the root of the folder request, the version of the folder right for the transaction, an operation, and data. The operation can be one of create, rename, and move file. The data are the specifications required for the operation, such as a specification of a folder or digital work and a name.
- The repositories perform the common opening transaction steps.
- The server performs the requested operation -- creating a folder, renaming a folder, or moving a work between folders.
- The repositories perform the common closing transaction steps.

The Extract Transaction

A extract transaction is a request to copy a part of a digital work and to create a new work containing it. The extraction operation differs from copying in that it can be used to separate a part of a digital work from d-blocks or shells that place additional restrictions or fees on it. The extraction operation differs from the edit operation in that it does not change the contents of a work, only its embedding in d-blocks. Extraction creates a new digital work.

- The requester sends the server a message to initiate an Extract transaction. This message indicates the part of the work to be extracted, the version of the extract right to be used in the transaction, the destination address information for placing the part as a new work, the file data for the work, and the number of copies involved.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the new work. Otherwise, the rights of the original are transmitted. The Copy-Count field for this right is set to the number-of-copies requested.
- The requester records the contents, data, and usage rights and stores the work. It records the date and time that new work was made in the properties of the work.
- The repositories perform the common closing transaction steps.

The Embed Transaction

An embed transaction is a request to make a digital work become a part of another digital work or to add a shell d-block to enable the adding of fees by a distributor of the work.

- The requester sends the server a message to initiate an Embed transaction. This message indicates the work to be embedded, the version of the embed right to be used in the transaction, the destination address information for placing the part as a a work, the file data for the work, and the number of copies involved.
- The server checks the control specifications for all of the rights in the part and the destination. If they are incompatible, the server ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the new work. Otherwise, the rights of the original are transmitted. The Copy-Count field for this right is set to the number-of-copies requested.
- The requester records the contents, data, and usage rights and embeds the work in the destination file.
- The repositories perform the common closing transaction steps.

The Edit Transaction

An Edit transaction is a request to make a new digital work by copying, selecting and modifying portions of an existing digital work. This operation can actually change the contents of a digital work. The kinds of changes that are permitted depend on the process being used. Like the extraction operation, edit operates on portions of a digital work. In contrast with the extract operation, edit does not affect the rights or location of the work. It only changes the contents. The kinds of changes permitted are determined by the type specification of the processor specified in the rights. In the currently preferred embodiment, an edit transaction changes the work itself and does not make a new work. However,

it would be a reasonable variation to cause a new copy of the work to be made.

- The requester sends the server a message to initiate an Edit transaction. This message indicates the work to be edited, the version of the edit right to be used in the transaction, the file data for the work (including its size), the process-ID for the process, and the number of copies involved.
- The server checks the compatibility of the process-ID to be used by the requester against any process-ID specification in the right. If they are incompatible, it ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The requester uses the process to change the contents of the digital work as desired. (For example, it can select and duplicate parts of it; combine it with other information; or compute functions based on the information. This can amount to editing text, music, or pictures or taking whatever other steps are useful in creating a derivative work.)
- The repositories perform the common closing transaction steps.

The edit transaction is used to cover a wide range of kinds of works. The category describes a process that takes as its input any portion of a digital work and then modifies the input in some way. For example, for text, a process for editing the text would require edit rights. A process for "summarizing" or counting words in the text would also be considered editing. For a music file, processing could involve changing the pitch or tempo, or adding reverberations, or any other audio effect. For digital video works, anything which alters the image would require edit rights. Examples would be colorizing, scaling, extracting still photos, selecting and combining frames into story boards, sharpening with signal processing, and so on.

Some creators may want to protect the authenticity of their works by limiting the kinds of processes that can be performed on them. If there are no edit rights, then no processing is allowed at all. A processor identifier can be included to specify what kind of process is allowed. If no process identifier is specified, then arbitrary processors can be used. For an example of a specific process, a photographer may want to allow use of his photograph but may not want it to be colorized. A musician may want to allow extraction of portions of his work but not changing of the tonality.

Authorization Transactions

There are many ways that authorization transactions can be defined. In the following, our preferred way is to simply define them in terms of other transactions that we already need for repositories. Thus, it is convenient sometimes to speak of "authorization transactions," but they are actually made up of other transactions that repositories already have.

A usage right can specify an authorization-ID, which identifies an authorization object (a digital work in a file of a standard format) that the repository must have and which it must process. The authorization is given to the generic authorization (or ticket) server of the repository which begins to interpret the authorization.

As described earlier, the authorization contains a server identifier, which may just be the generic authorization server or it may be another server. When a remote authorization server is required, it must contain a digital address. It may also contain a digital certificate.

If a remote authorization server is required, then the authorization process first performs the following steps:

- The generic authorization server attempts to set up the communications channel. (If the channel cannot be set up, then authorization fails with an error.)
- When the channel is set up, it performs a registration process with the remote repository. (If registration fails, then the authorization fails with an error.)
- When registration is complete, the generic authorization server invokes a "Play" transaction with the remote repository, supplying the authorization document as the digital work to be played, and the remote authorization server (a program) as the "player." (If the player cannot be found or has some other error, then the authorization fails with an error.)
- The authorization server then "plays" the authorization. This involves decrypting it using either the public key of the master repository that issued the certificate or the session key from the repository that transmitted it. The authorization server then performs various tests. These tests vary according to the authorization server. They include such steps as checking issue and validity dates of the authorization and checking any hot-lists of known invalid authorizations. The authorization server may require carrying out any other transactions on the repository as well, such as checking directories, getting some person to supply a password, or playing some other digital work. It may also invoke some special process for checking information about locations or recent events. The "script" for such steps is contained within the authorization server.
- If all of the required steps are completed satisfactorily, the authorization server completes the transaction normally, signaling that authorization is granted.

The Install Transaction

An Install transaction is a request to install a digital work as runnable software on a repository. In a typical case, the requester repository is a rendering repository and the software would be a new kind or new version of a player. Also in a typical case, the software would be copied to file system of the requester repository before it is installed.

- The requester sends the server an Install message. This message indicates the work to be installed, the version of the Install right being invoked, and the file data for the work (including its size).
- The repositories perform the common opening transaction steps.
- The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error.
- The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step certifies the software.)
- The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.)
- The requester retrieves the instructions in the compatibility-checking script and follows them. If the software is not compatible with the repository, the installation transaction ends with an error. (This step checks platform compatibility.)
- The requester retrieves the instructions in the installation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error. Note that the installation process puts the runnable software in a place in the repository where it is no longer accessible as a work for exercising any usage rights other than the execution of the software as part of repository operations in carrying out other transactions.
- The repositories perform the common closing transaction steps.

The Uninstall Transaction

An Uninstall transaction is a request to remove software from a repository. Since uncontrolled or incorrect removal of software from a repository could compromise its behavioral integrity, this step is controlled.

- The requester sends the server an Uninstall message. This message indicates the work to be uninstalled, the version of the Uninstall right being invoked, and the file data for the work (including its size).
- The repositories perform the common opening transaction steps.
- The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error.
- The requester checks whether the software is installed. If the software is not installed, the transaction ends with an error.
- The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step authenticates the certification of the software, including the script for uninstalling it.)
- The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.)
- The requester retrieves the instructions in the uninstallation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error.
- The repositories perform the common closing transaction steps.

Claims

1. A system for controlling the distribution and use of digital works having a mechanism for reporting fees based on the distribution and use of digital works, said system comprising:

   means for attaching usage rights to a digital work, each of said usage rights specifying how a digital work may be used or distributed, each of said usage rights specifying usage fee information, said usage fee information

comprising a fee type and fee parameters which define a fee to be paid in connection with the exercise of said usage right;
a communication medium for coupling repositories to enable communication between repositories; and
a plurality of repositories, each of said repositories comprising:
an external interface for removably coupling to said communications medium;
storage means for storing digital works having attached usage rights and fees;
requesting means for generating a request to access a digital work stored in another of said plurality of repositories, said request indicating a particular usage right; and
processing means for processing requests to access digital works stored in said storage means and for generating fee transactions when a request indicates a usage right that is attached to a digital work and said usage right specifies usage fee information;
each of said plurality of repositories being removably coupled to a credit server, said credit server being arranged for recording fee transactions from said repository and subsequently reporting said fee transactions to a billing clearinghouse.

2. The fee reporting system as recited in Claim 1 wherein said fee type of said fee information is a metered use fee, a per use fee, a best price fee, a scheduled fee,or a mark-up fee.

3. A method for reporting fees associated with the distribution and use of digital works in a system for controlling the distribution and use of digital works, said method comprising the steps of:

a) attaching one or more usage rights to a digital work, each of said one or more usage rights comprising an indicator of how said digital work may be distributed or used and a usage fee to be paid upon exercise of said right;
b) storing said digital work and attached one or more usage rights in a server repository, said server repository controlling access to said digital work;
c) said server repository receiving a request to access said digital work from a requesting repository;
d) said server repository identifying a usage right associated with said access request;
e) said server repository determining if said identified usage right is the same as one of said one or more usage rights attached to said digital work;
f) if said identified usage right is not the same as any one of said one or more usage rights attached to said digital work, said server repository denying access to said digital work;
g) if said usage right is included with said digital work, said server repository determining if a usage fee is associated with the exercise of said usage right;
h) if a usage fee is associated with usage right, said server repository calculating said usage fee;
i) said server repository transmitting a first assign fee transaction identifying said requesting repository as a payer for said usage fee to a first credit server;
j) said requesting repository transmitting a second assign fee transaction identifying said requesting repository as a payer for said usage fee to a second credit server;
k) said server repository transmitting said digital work to said requesting repository;
l) said server repository transmitting a first confirm fee transaction to said first credit server; and
m) said requesting repository transmitting a second confirm fee transaction to said second credit server.

4. The method as recited in Claim 3 wherein said digital work is comprised of a plurality of independent digital works and said step of said server calculating said usage fee is further comprised of the step of reporting the usage fees for each of the plurality of independent digital works.

5. A method for reporting fees associated with the distribution and use of digital works in a system for controlling the distribution and use of digital works, said method comprising the steps of:

a) attaching one or more usage rights to a digital work, each of said one or more usage rights comprising an indicator of how said digital work may be distributed or used and a usage fee to be paid for exercise of said right;
b) storing said digital work and said attached one or more usage rights in a server repository, said server repository controlling access to said digital work;
c) said server repository receiving a request to access said digital work from a requesting repository;
d) said server repository identifying a usage right associated with said access request;
e) said server repository determining if said digital work has attached thereto said identified usage right;
f) if said identified usage right is not attached to said digital work, said server repository denying access to

said digital work;
g) if said usage right is attached to said digital work, said server repository determining if a usage fee is associated with the exercise of said usage right;
h) if a usage fee is associated with said usage right, said server repository determining a fee type;
i) said server repository transmitting a first fee transaction identifying said requesting repository as a payee for said usage fee to a credit server, said first fee transaction being dependent on said determined fee type; and
k) said server repository transmitting said digital work to said requesting repository.

6. A system for controlling the distribution and utilization of digital works having a mechanism for reporting usage fees, said system comprising:

digital works comprising a first part for storing the digitally encoded data corresponding to a digital work and a second part for storing usage rights and fees for said digital work, said usage rights specifying how a digital work may be used or distributed and said usage fees specifying a fee to be paid in connection with the exercise of a corresponding usage right;
a plurality of repositories, each of said repositories comprising:
communication means for communicating with another of said plurality of repositories;
storage means for storing digital works;
requesting means for generating a request to access a digital work stored in another of said plurality of repositories, said request indicating a particular usage right;
processing means for processing requests to access digital works stored in said storage means and granting access when said particular usage right corresponds to a stored usage right stored in said digital work, said processing means generating fee transactions when said access is granted and said stored usage right specifies a fee;
each of said plurality of repositories being removably coupled to a credit server, said credit server being arranged for recording fee transactions from said repository and subsequently reporting said fee transactions to a billing clearinghouse.

7. The system as recited in Claim 6 wherein said storage means is further comprised of a first storage device for storing said first part of said digital work and a second storage device for storing said second part of said digital work.

8. A method for reporting fees associated with use of rendering digital works by a rendering device in a system for controlling the rendering of digital works by a rendering system, said rendering system comprised of a rendering repository and a rendering device, said rendering device utilizing a rendering digital work for rendering a digital work, said method comprising the steps of:

a) storing a first digital work in a server repository, said digital work specifying a first usage fee to be reported for a use of said first digital work;
b) storing a rendering digital work in said rendering repository, said first rendering digital work specifying a second usage fee to be reported for a use of said rendering digital work;
c) said server repository receiving a request to use said first digital work from said rendering repository;
d) said server repository determining if said request may be granted;
e) if said server repository determines that said request may not be granted, said server repository denying access to said first digital work;
f) if said server repository determines that said request may be granted, said server repository transmitting said digital work to said rendering repository;
g) said server repository transmitting a first fee transaction identifying said rendering repository as a payee for said first usage fee for use of said first digital work to a first credit server;
h) said rendering device rendering said first digital work using said rendering digital work; and
i) said rendering repository transmitting a second fee transaction identifying said rendering repository as a payee for said second usage fee for use of said rendering digital work to a second credit server.

9. The method as recited in Claim 8 further comprising the step of said rendering repository transmitting a third fee transaction identifying said rendering repository as a payee for said first usage fee for use of said first digital work to said second credit server.

10. The method as recited in Claim 9 wherein said rendering digital work is a set of coded rendering instructions for controlling said rendering device.

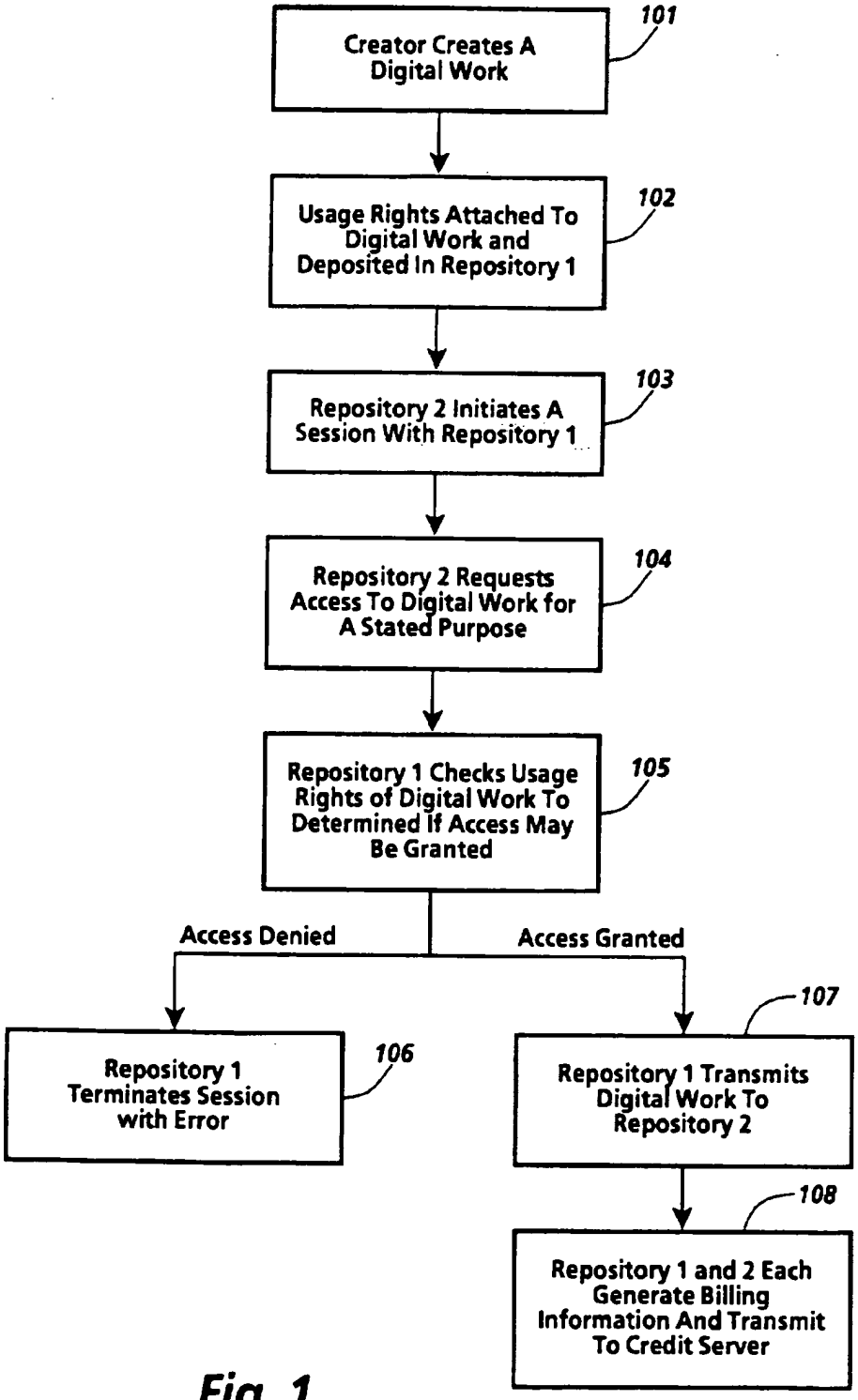


Fig. 1

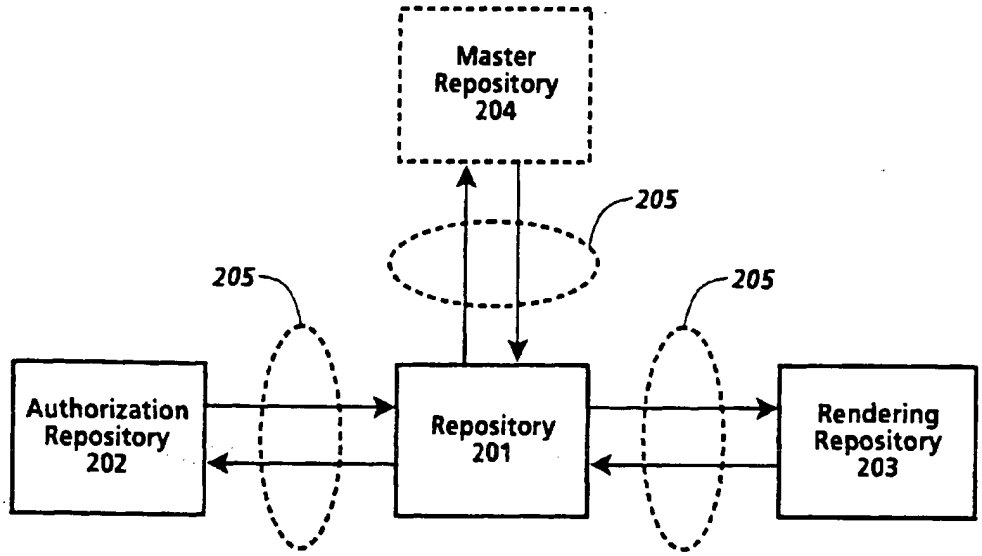


Fig. 2

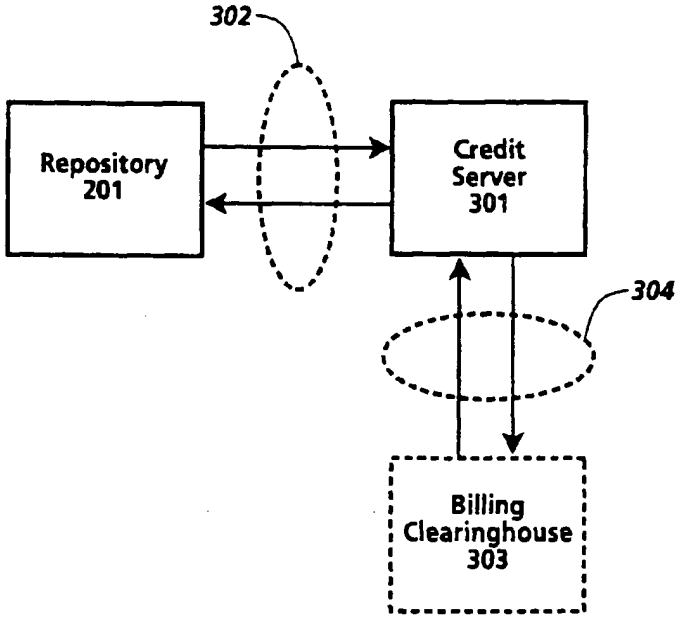


Fig. 3

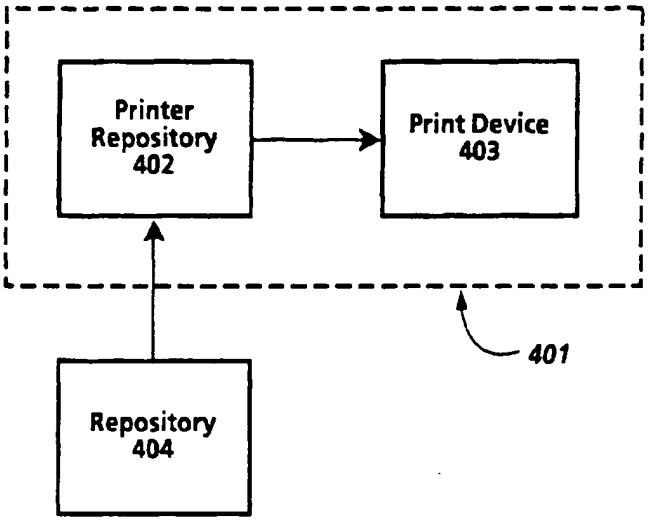


Fig. 4a

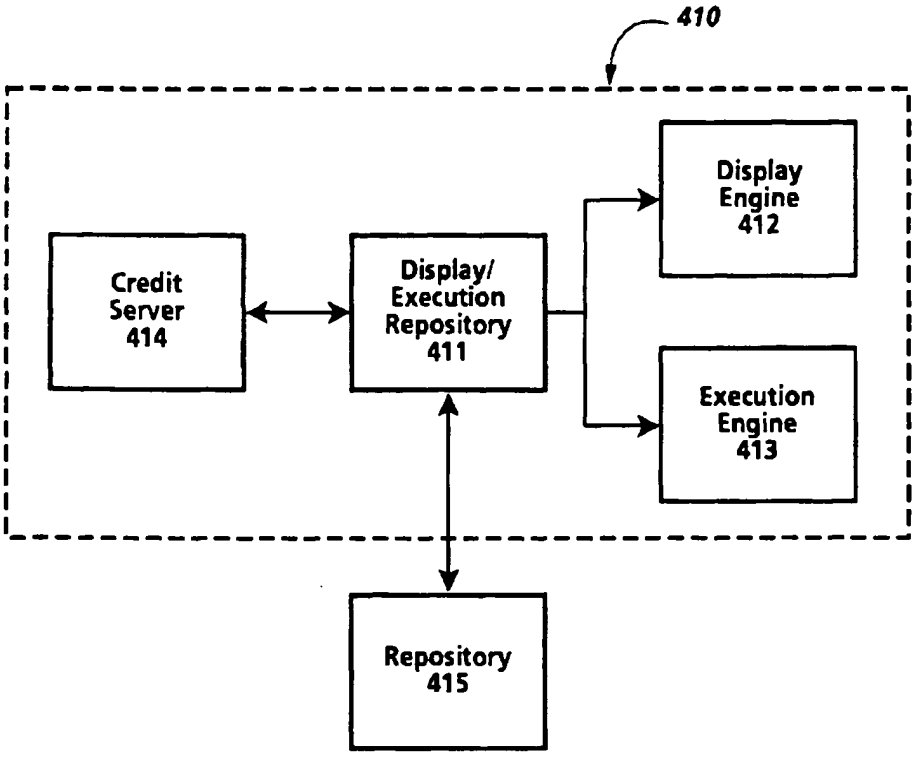


Fig. 4b

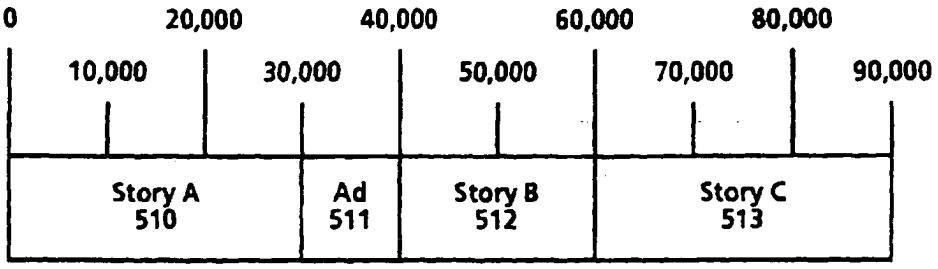


Fig. 5

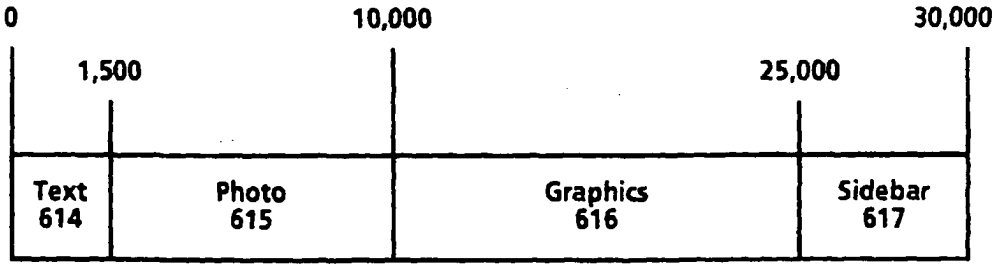


Fig. 6

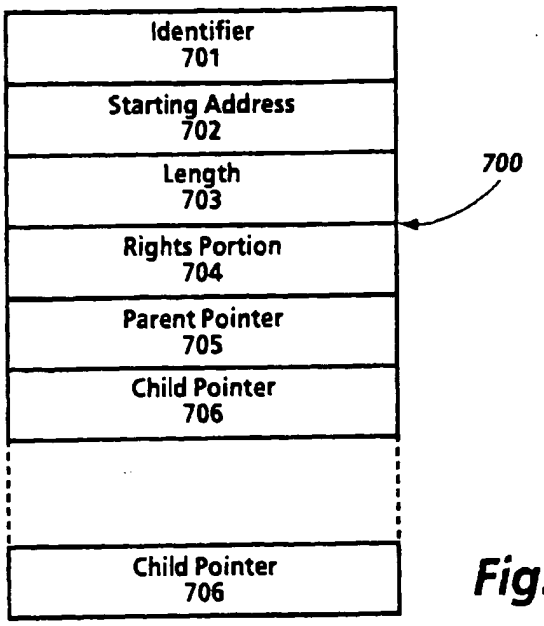


Fig. 7

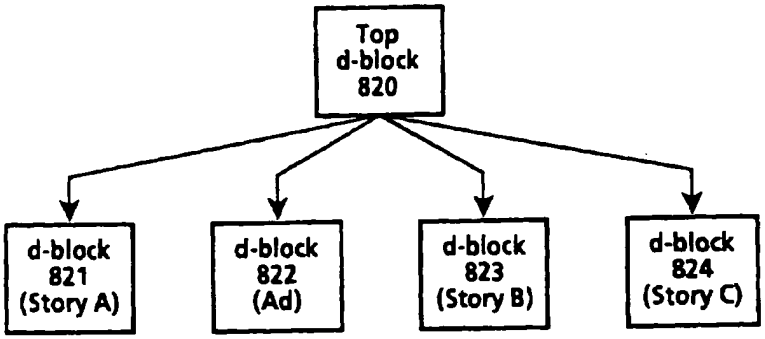


Fig. 8

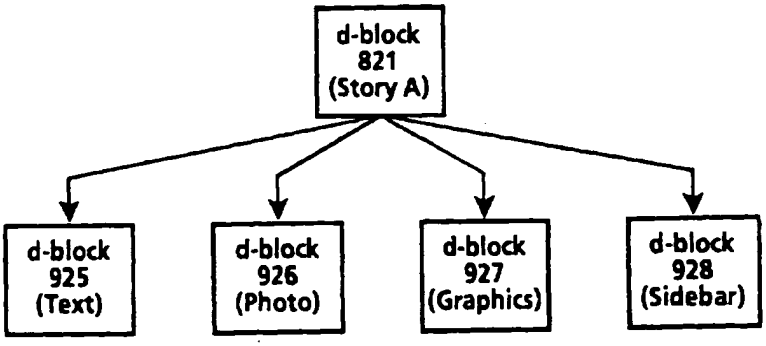


Fig. 9

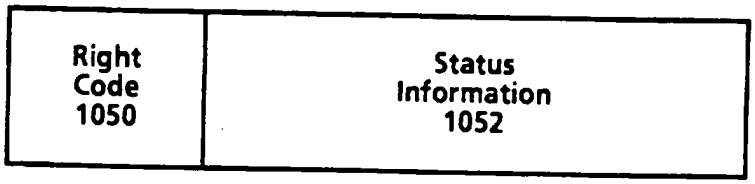


Fig.10

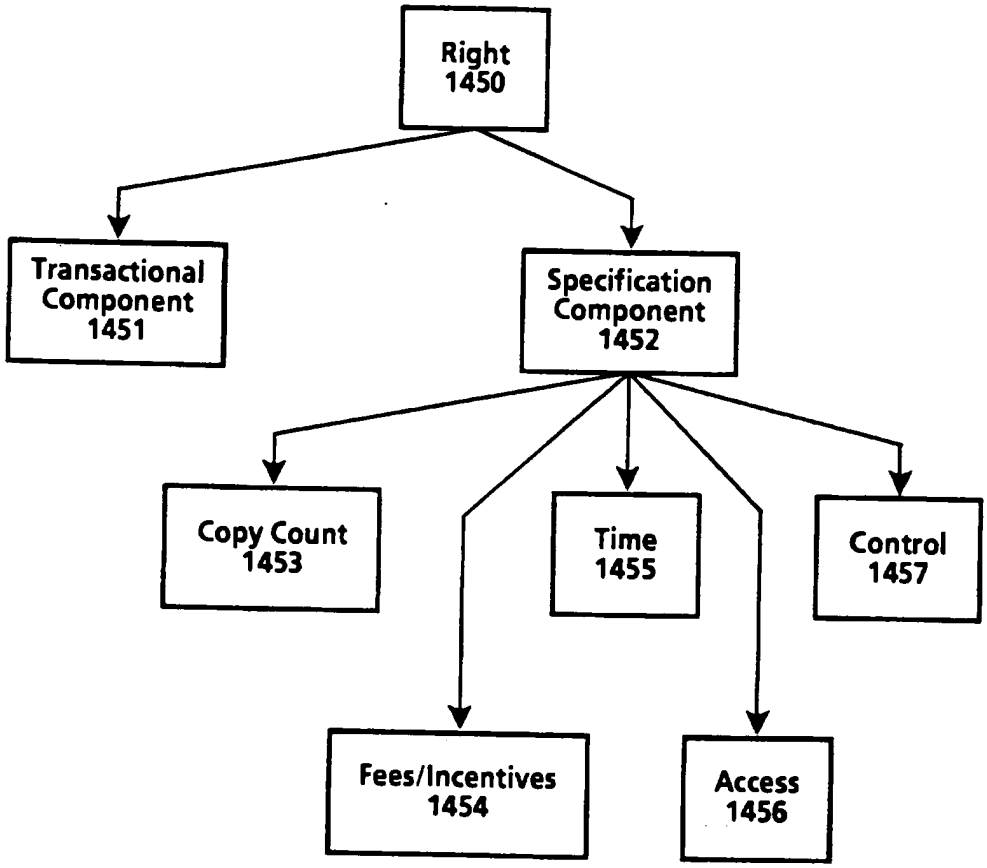


Fig.14

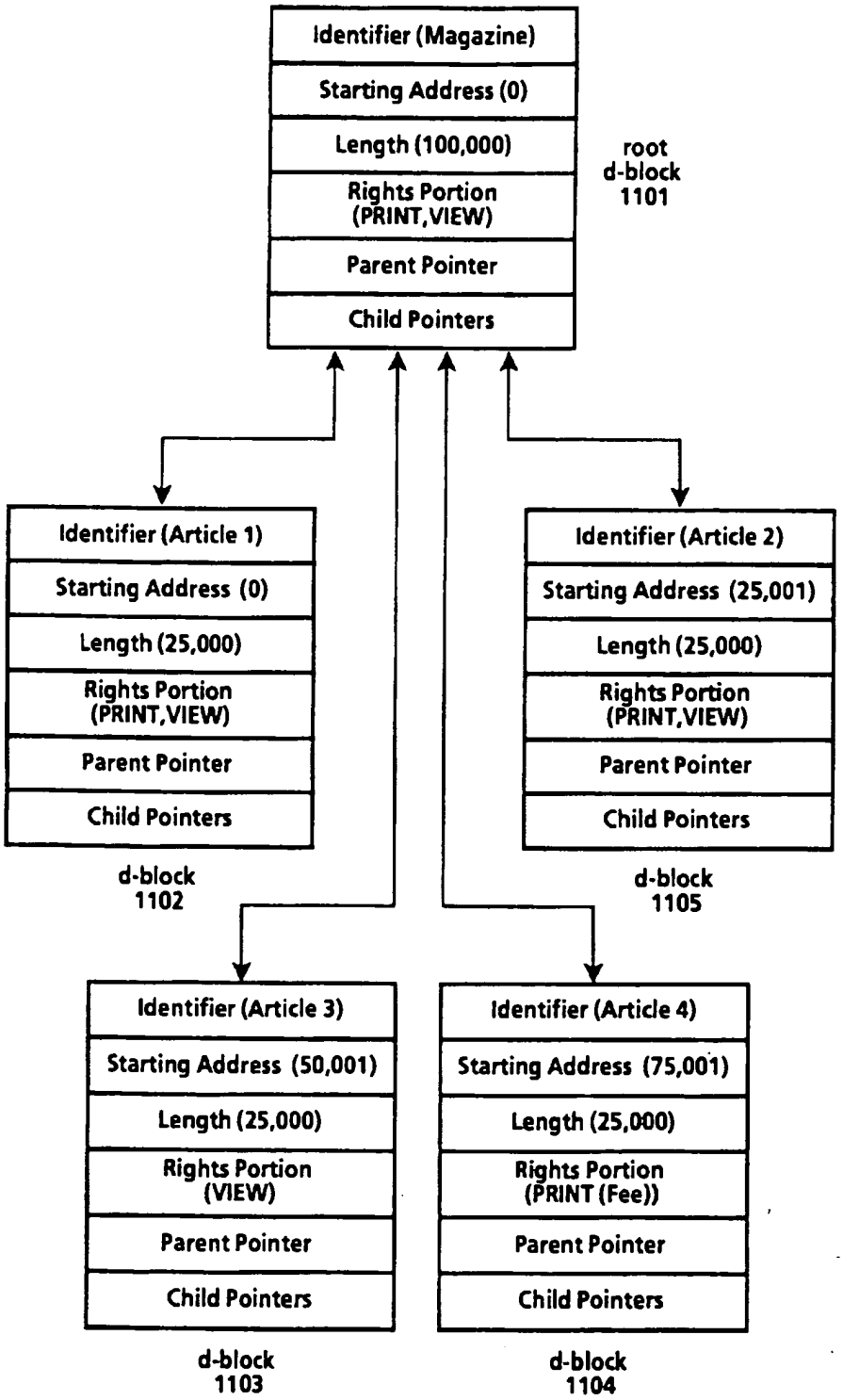


Fig.11

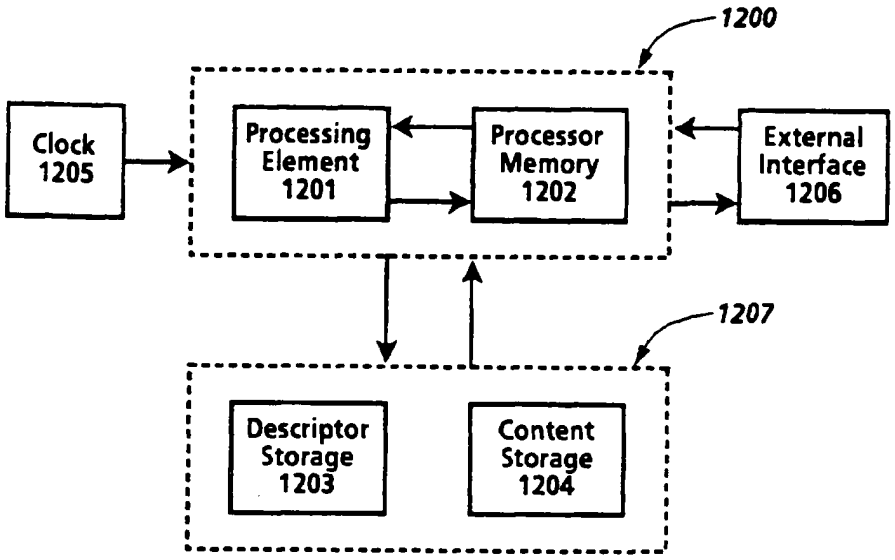


Fig.12

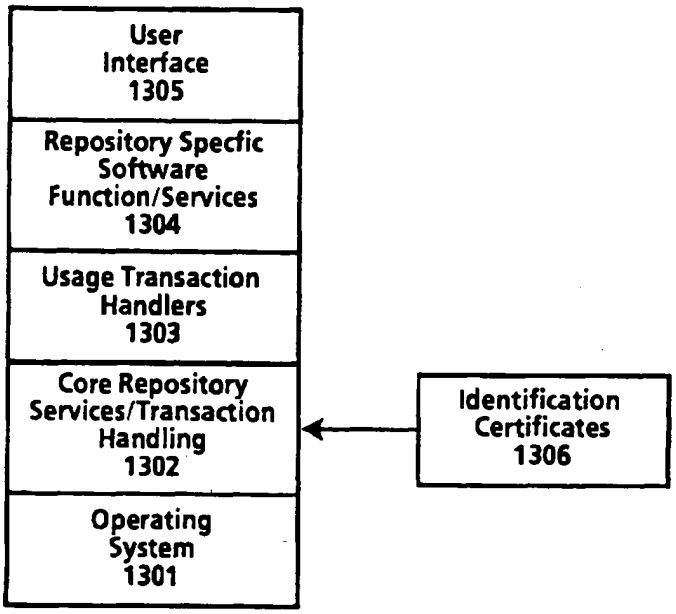


Fig.13

```
1501 ~Digital Work Rights: = (Rights*)
1502 ~Right : = (Right-Code {Copy-Count} {Control-Spec} {Time-Spec }
       {Access-Spec} {Fee-Spec})
  1503 ~Right-Code : = Render-Code | Transport-Code | File-Management-
         Code| Derivative-Works- Code | Configuration-Code
    1504 ~Render-Code : = [ Play : {Player: Player-ID} | Print: {Printer: Printer-ID}]
    1505 ~Transport-Code : = [Copy | Transfer | Loan {Remaining-Rights:
           Next-Set-of-Rights}]{(Next-Copy-Rights: Next-Set-of-Rights)}
    1506 ~File-Management-Code  : = Backup {Back-Up-Copy-Rights:
                                  Next-Set-of-Rights} | Restore | Delete | Folder
                                  | Directory {Name: Hide-Local | Hide-Remote}
                                  {Parts: Hide-Local | Hide-Remote}
    1507 ~Derivative-Works-Code : =  [Extract | Embed | Edit{Process:
                                  Process-ID}] {Next-Copy-Rights :
                                  Next-Set-of Rights}
    1508 ~Configuration-Code : = Install | Uninstall
    1509 ~Next-Set-of-Rights : = {(Add:  Set-Of-Rights)} {(Delete:
           Set-Of-Rights)} {(Replace: Set-Of-Rights )}{(Keep: Set-Of-Rights )}
1510 ~Copy-Count : = (Copies:positive-integer | 0 | Unlimited)
1511 ~Control-Spec : = (Control: {Restrictable | Unrestrictable}
                      {Unchargeable | Chargeable})
1512 ~Time-Spec : = ({Fixed-Interval | Sliding-Interval | Meter-Time}
                    Until: Expiration-Date)
  1513 ~Fixed-Interval : = From: Start-Time
  1514 ~Sliding-Interval : = Interval: Use-Duration
  1515 ~Meter-Time: = Time-Remaining: Remaining-Use
1516 ~Access-Spec : = ({SC: Security-Class} {Authorization: Authorization-ID*}
       {Other-Authorization: Authorization-ID*} {Ticket: Ticket-ID})
1517 ~Fee-Spec: = {Scheduled-Discount} Regular-Fee-Spec | Scheduled-Fee-Spec |
                  Markup-Spec
  1518 ~Scheduled-Discount: = Scheduled-Discount: (Scheduled-Discount:
                              (Time-Spec Percentage)*)
  1519 ~Regular-Fee-Spec : = ({Fee: | Incentive: } [Per-Use-Spec | Metered-Rate-
                             Spec | Best-Price-Spec | Call-For-Price-Spec]
                             {Min: Money-Unit Per: Time-Spec}{Max:
                             Money-Unit Per: Time-Spec} To: Account-ID)
    1520 ~Per-Use-Spec: = Per-Use: Money-unit
    1521 ~Metered-Rate-Spec : = Metered: Money-Unit Per: Time-Spec
    1522 ~Best-Price-Spec : = Best-Price: Money-unit Max: Money-unit
    1523 ~Call-For-Price-Spec : = Call-For -Price
  1524 ~Scheduled-Fee-Spec: = (Schedule: (Time-Spec Regular-Fee-Spec)* )
  1525 ~Markup-Spec: = Markup: percentage To: Account-ID
```

Fig.15

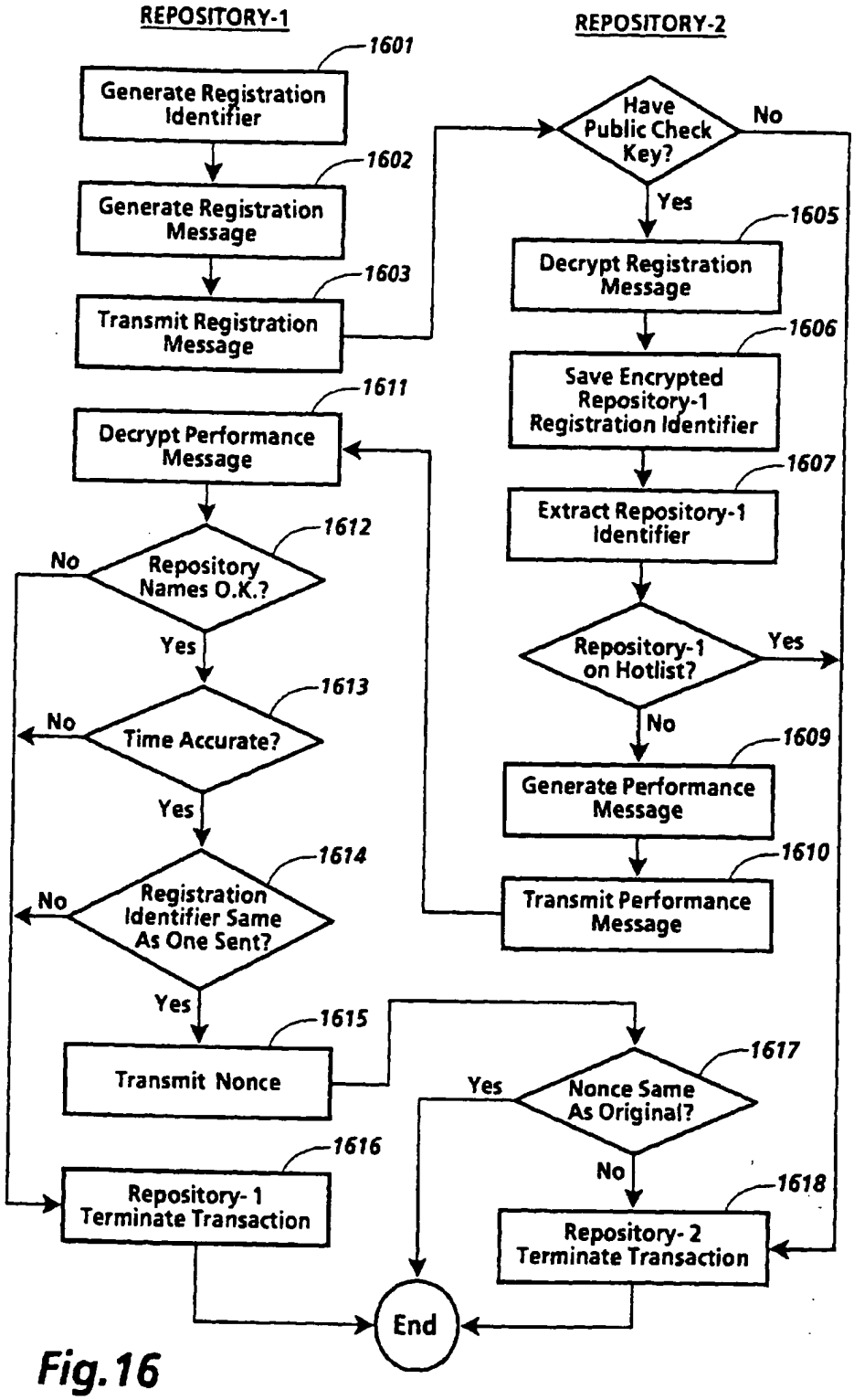


Fig.16

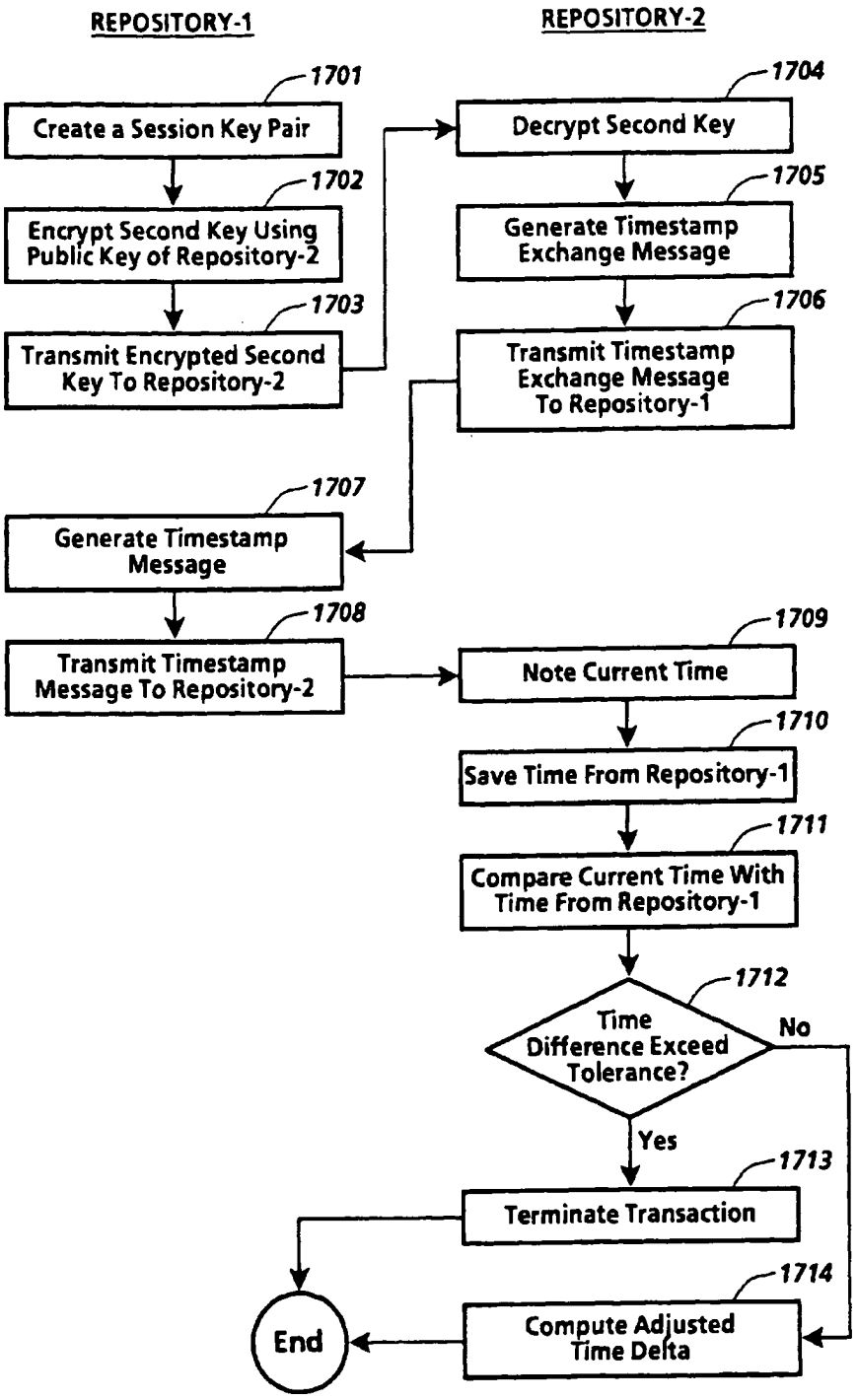


Fig.17

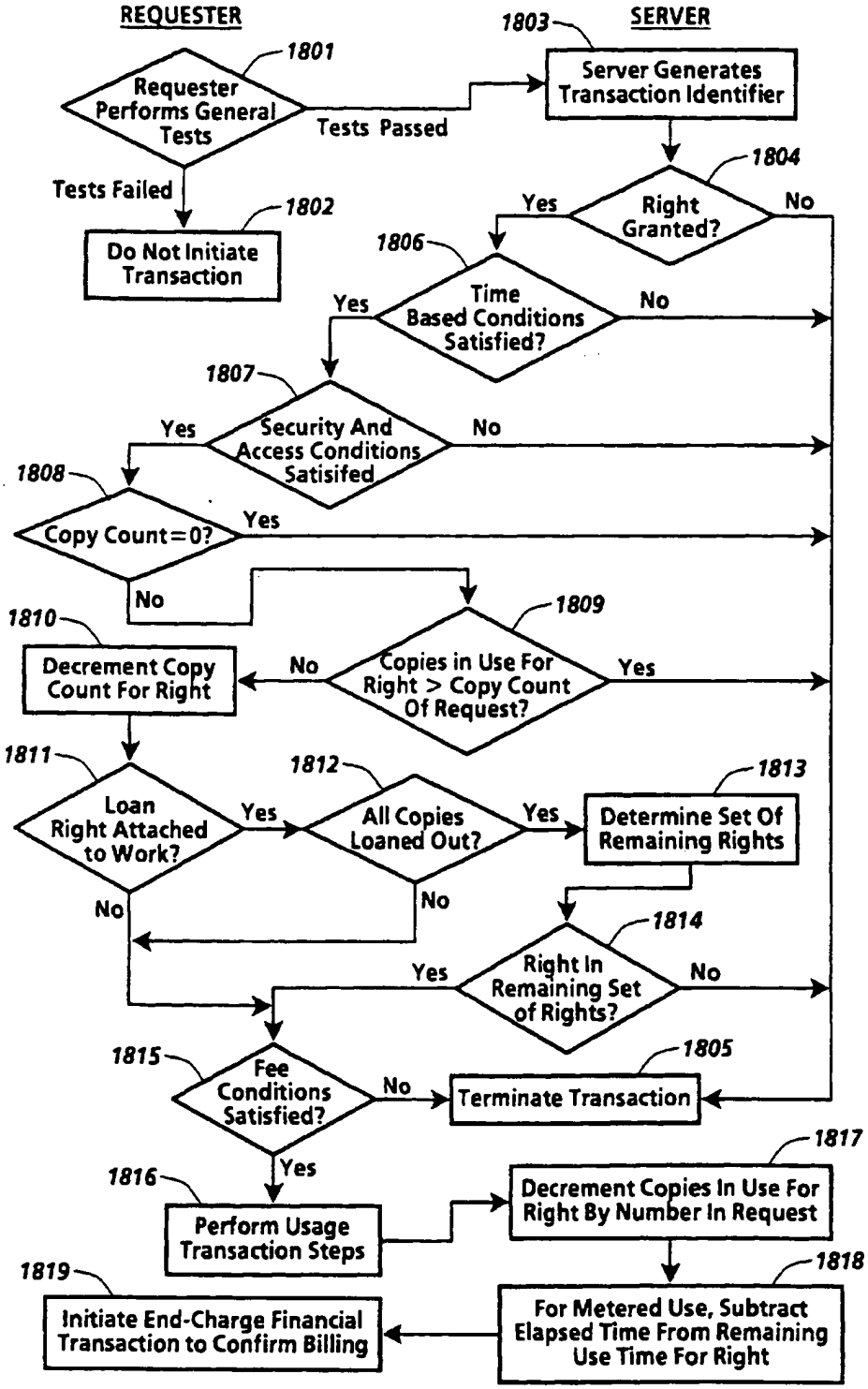


Fig.18

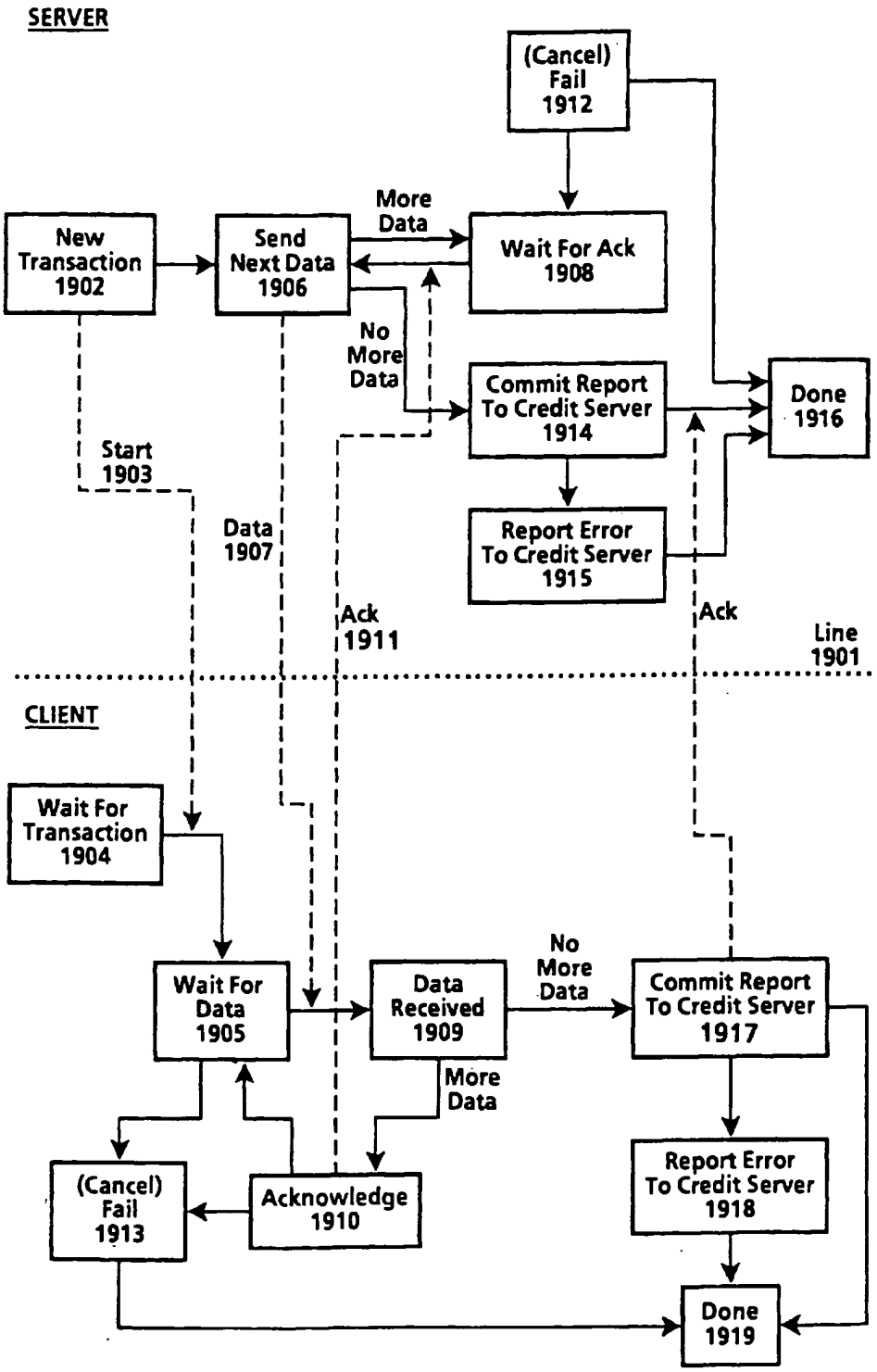


Fig.19

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 95 30 8414

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
A	WO-A-92 20022 (DIGITAL EQUIPMENT CORP.) * page 45, line 10 - page 64, line 17 * ---	1,3,5,6,8	G06F1/00 G06F17/60
A	TRANSACTIONS OF THE INSTITUTE OF ELECTRONICS, INFORMATION AND COMMUNICATION ENGINEERS OF JAPAN, vol. E73, no. 7, July 1990 TOKYO JP, pages 1133-1146, XP 000159229 MORI ET AL. 'SUPERDISTRIBUTION: THE CONCEPT AND THE ARCHITECTURE' * page 1135, left column, line 17 - page 1136, left column, line 40 * ---	1,3,5,6,8	
A	US-A-5 291 596 (MITA) * the whole document * ---	1,3,5,6,8	
A	GB-A-2 236 604 (SUN MICROSYSTEMS INC) * page 9, line 11 - page 20, line 15 * -----	1,3,5,6,8	

TECHNICAL FIELDS SEARCHED (Int.Cl.6)

G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	1 April 1996	Moens, R

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 818 748 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
14.01.1998 Bulletin 1998/03

(51) Int Cl.[6]: **G06F 17/60**

(21) Application number: **97304946.3**

(22) Date of filing: **07.07.1997**

(84) Designated Contracting States:
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE

(30) Priority: **08.07.1996 JP 178130/96**
21.05.1997 JP 130626/97

(71) Applicant: **Murakoshi, Hiromasa**
Koriyama-shi, Fukushima, 963 (JP)

(72) Inventor: **Kanno, Kazuhiro**
Koriyama-shi, Fukushima, 963-02 (JP)

(74) Representative:
Cross, Rupert Edward Blount et al
BOULT WADE TENNANT,
27 Furnival Street
London EC4A 1PQ (GB)

(54) Software management system and method

(57) An operation management system for managing the operation of a managed software product. When a management target function is executed, reference is made to a battery value and, if the value is zero or greater, the function is allowed to be executed. The battery value is decremented as the function is executed. A charge value is supplied on a charge disk, such as a floppy disk, to allow the user to increase the battery value and to extend the usage period of the managed software product. The charge value may be supplied over a communication line.

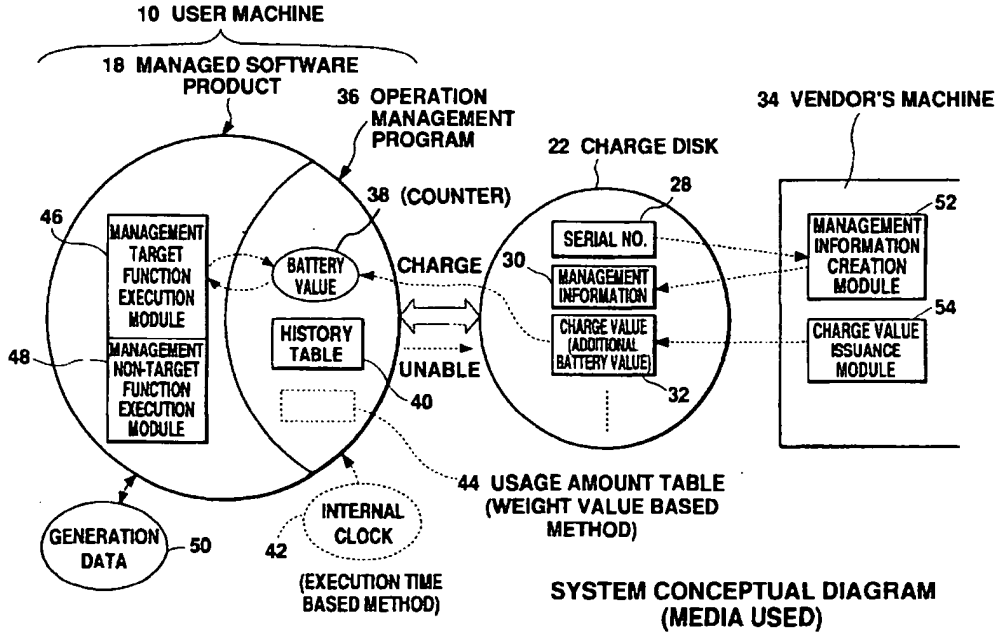


Fig. 3

EP 0 818 748 A2

Printed by Jouve, 75001 PARIS (FR)

Description

BACKGROUND OF THE INVENTION

Field of the Invention

This invention relates to an operation management system and an operation management method, and more particularly to software operation management or execution management.

Description of the Related Art

As computers and computer use become more common, more advanced technology is introduced and a variety of software products are developed for use in various fields. However, in many cases, the user finds it difficult to select a product from among a variety of software products that seem to meet the user's requirements; often, the user cannot find the best tool for his needs.

To reduce such a risk, a service has been available that supplies the user with a trial-use software product free of charge. However, most of these trial-use software products contain only function descriptions or provide the user with limited functions (e.g., save function and/or output function is/are not included). This makes it difficult for the user to evaluate the actual product (all the functions) correctly.

A sales system which charges the user according to how long the user actually uses a software product (including a trial use) would allow him to buy the product anytime he wants, to fully evaluate the product, and to precisely determine the requirements for continued use (including payment for it). Many users would find this type of sales system appealing and economical.

In Japanese Patent Laid-Open Publication No. Sho 59-41061 and Japanese Patent Laid-Open Publication No. Sho 63-153633, a system is disclosed that automatically prevents a program from being used when the usage count reaches a specified value. In Japanese Patent Laid-Open Publication No. Hei 1-147622 a system is disclosed which accumulates program execution time (total program execution time) and prevents the program from being used when the accumulation time reaches a specified amount. However, these systems do not disclose means for extending the program usage period. Japanese Patent Laid-Open Publication No. Hei 5-134949 discloses a system in which a program and expiry of the program are downloaded from a host computer to a user computer via a communication line. Also disclosed is a system in which a new expiry of the program is downloaded from the host computer to the user computer in order to update the expiry. However, the system only measures the execution time taken for executing the entire program, and does not include any means for changing the expiry on the user computer.

In Japanese Patent Laid-Open Publication No. Hei 7-234785, a system is disclosed that relates to a software rental system. This system connects a computer in a rental company to a user computer on which a rental software product is running over a communication line. When the time elapsed from the rental start time reaches the rental limit time, the system makes the program unavailable for use. (For example, the program is deleted.) To allow the user to update the rental period, the rental company sends a rental period extension program to the user's computer over a communication line. The user runs this program to extend the rental period of the program. A drawback of this system is that the user must pay for the software product regardless of whether the user has used it frequently or not. This means that the amount of money the user has to pay depends, not on how often he has used it, but on how long he has used it.

In Japanese Patent Laid-Open Publication No. Hei 7-244585, a system is disclosed that manages the program usage period. This system assigns a usage limit date to a program and, when the current date becomes greater than the limit date, the program product is made unavailable. To extend the usage limit date, the system reads update limit data from a recording medium containing that data and re-assigns a usage limit date based on the update limit data. This system is not reasonable because the amount of money the user has to pay does not depend on whether or not the user actually uses the program.

For example, during execution of a Computer Aided Design (CAD) software product, the user often spends much time thinking without entering data. In the system disclosed by the above mentioned Japanese Patent Laid-Open Publication No. Hei 7-234785 or Japanese Patent Laid-Open Publication No. Hei 7-244585, the user must pay for this thinking time. This places unwanted pressure on the user, especially when he must think carefully during program execution.

SUMMARY OF THE INVENTION

The present invention seeks to solve the problems associated with the art described above. In view of the foregoing, it is an object of the present invention to provide an operation management system and method which reasonably manage the operation of a managed software product.

It is another object of the present invention to provide an operation management system and method which levy a charge according to the actual usage amount of the managed software product (or the amount of the result generated by the managed software product).

It is still another object of the present invention to provide an operation management system and method which manage the operation according to the property of each function of the managed software product.

(1) To achieve the above objects, an operation manage-

ment system for managing the operation of a managed software product according to the present invention comprises: battery value management means for decrementing a battery value according to the operation amount of the managed software product; operation limit means for limiting the operation of the managed software product when the battery value has decreased to a specified limit value; and charge means for adding a charge value to the current battery value when the charge value is entered from external means.

The "battery value" mentioned above is a "virtual battery" which drives a managed software product. This battery value is preferably the value of a counter.

The battery value management means decrement the battery value according to the operation amount of the managed software product. When the battery value has reached a specified limit value (for example, 0), the operation limit means limit all of or a part of the operation of the managed software product. Upon receiving a charge value (additional battery value) from the external means, the charge means add the received value to the current battery value, thus extending the operation period. That is, the battery value is incremented, just as a battery is charged, to allow the continued use of the managed software product.

The managed software product described above is preferably a packaged application software program including a CAD program, game program, video program, language processor, music program, communication program, or a measurement program.

The battery value management means, operation management means, and charge means described above should be implemented preferably as software programs (management software programs) that run on a computer. The managed software product and the management software product may be separate, or the whole or a part of the management software product may be included in the managed software product.

A system according to the present invention is implemented on a general-purpose computer or special-purpose computer having such peripheral units as a disk drive, display, and input unit. The external means described above include recording media such as a magnetic disk or an optical disk and other host computers connected over a network.

(2) An operation management system according to the present invention may be applied to an application software product sales system. The following explains an example:

A vendor sells an application software product containing the operation management program according to the present invention. The operation management program has a battery value defined as the initial value. In addition to this product, the vendor sells recording media containing charge values (e.g., floppy disk (FD)). In this case, it is desirable that a variety of recording media, each containing a unique charge value, be supplied.

On the other hand, a user who bought the application software product may use the product until the battery value reaches zero. This allows the user to fully evaluate and examine the product. A user who wants to use the product after the battery value becomes zero must buy a recording medium containing a charge value to charge the battery. This enables him to add a charge value to the battery value and to use the product continuously.

If the specifications of the application software product do not satisfy the user's request, the user does not buy the recording medium. This prevents additional charges and reduces the cost to the user.

Considering an increase in the sales profit in recording media that will be produced in the future, a combination of a managed software product and the operation management program will lower prices significantly. The operation management system according to the present invention will increase the profits of both the user and the vendor, making it possible to build a very reasonable, economical system.

(3) In a preferred embodiment of the present invention, the battery value management means calculate the operation amount of each function of the managed software product, and subtracts a value corresponding to the operation amount from the battery value.

A continuous decrease in the battery value during execution of a managed software product, as in a conventional system, decrements the value even when the user is idle (input wait time), which places pressure on the user.

Calculating the operation amount of each function during execution of a managed software product, as in a system according to the present invention, decreases the battery value only when the managed software product is actually used, enabling the user to do operation without having to worry about time elapsed while thinking.

(4) In a preferred embodiment of the present invention, function category determination means are also available which determine if an execution instruction from the user activates a management target function or a management non-target function. And, the battery value management means decrement the battery value only when the management target function is executed.

For example, with the data generation function defined as a management target function and with other functions as management non-target functions, a cost can be levied only when new data are generated.

(5) In a preferred embodiment of the present invention, the battery value management means have a weight table containing an operation amount weight value for each of the management target functions. When any of the management target functions is executed, the battery value management means decrement the battery value by the weight value corresponding to the management target function.

In a preferred embodiment of the present invention,

the battery value management means measure the execution time of each of the management target functions and decrement the battery value by the value corresponding to the execution time.

This weight value system is able to calculate the operation amount regardless of the computer speed, which may differ among computers. In addition, by measuring time in this manner, the execution time is directly monitored and therefore the operation mount becomes proportional to the CPU load.

(6) In a preferred embodiment of the present invention, the operation limit means prevent only the management target functions from being executed when the battery value has decreased to a specified limit value; management non-target functions are executed.

For example, forcing a game program used at home to terminate when the battery value has reached a specified value does not cause a serious problem.

However, for a CAD program used in an office, forced termination when the battery value has reached a specified value may make already-produced data unavailable, possibly interrupting a job. Therefore, considering user's advantage and convenience, the embodiment keeps some functions operable even when the battery value has reached a specified value.

(7) A preferred embodiment of the present invention has remainder warning means for issuing a remainder warning message when the battery value has decremented to a specified warning value because a sudden inoperable condition in the managed software product without prior notice may cause the user unexpected damage. The remainder warning means alert the user to that condition before it occurs. In other words, the warning message prompts the user to determine whether to charge the battery value.

A preferred embodiment of the present invention has remainder display means for displaying the battery value on the screen during execution of the managed software product. This remainder display information keeps the user informed of the amount by which the managed software product will be able to continue operation without being charged.

It is also possible to program the system so that, upon detecting that the battery value has been charged to a specified value, the system can automatically disable operation management through the battery value to allow the user to use the product indefinitely.

(8) To achieve the above objects, a method for managing the operation of a managed software product according to the present invention comprises: a count value management step for changing a count value according to the operation amount of the managed software product; an operation limit step for limiting the operation of the managed software product when the count value has reached a specified limit value; and a charge step for charging the current count value or the limit value when a charge value is entered from external means.

The above count value is incremented or decremented according to the operation amount of the managed software product. When the count value is incremented, a charge value is added to the limit value; when the count value is decremented, a charge value is added to the current count value. In either case, the usage period is extended by charging the battery value.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a diagram showing a user machine used in the operation management system according to the present invention.

Fig. 2 is a diagram showing the data structure of a charge disk.

Fig. 3 is a diagram showing the concept of the operation management system according to the present invention.

Fig. 4 is a diagram showing an example of the history table.

Fig. 5 is a diagram showing an example of the usage amount table.

Fig. 6 is a flowchart showing the processing of the system when a management target function is executed in the execution time based method.

Fig. 7 is a flowchart showing the processing of the system when a management target function is executed in the weight value based method.

Fig. 8 is a flowchart showing the charge disk read processing.

Fig. 9 is a flowchart showing the charge processing.

Fig. 10 is a diagram showing a user machine used in another embodiment.

Fig. 11 is a diagram showing the structure of data sent from the host machine to a user machine.

Fig. 12 is a diagram showing the concept of the system in another embodiment.

Fig. 13 is a diagram showing an example of the user registration table.

Fig. 14 is a flowchart showing the operation of the user machine and a user machine in another embodiment.

Fig. 15 is a diagram showing another configuration of the system.

Fig. 16 is a diagram showing an example of an application according to the present invention.

Fig. 17 is a flowchart showing the function category determination processing.

DESCRIPTION OF PREFERRED EMBODIMENTS

Fig. 1 shows a user machine 10. This user machine 10 is a computer which executes various types of application programs under control of the operation system (OS). The user machine 10 is composed of a system unit 12, display 14, keyboard (not shown in the figure), output unit (not shown in the figure) such as a printer or plotter, and so forth. The system unit 12 contains a CD-ROM disk drive 16 which accesses a CD-ROM and

reads data from it and a floppy disk drive 20 which accesses a floppy disk (FD) and reads data from it.

The CD-ROM shown in Fig. 1 contains a managed software product 18. In this embodiment, the managed software product 18, such as a CAD software product, has an operation management program built in. The operation management program, designed for managing the operation of the managed software product 18, manages the operation using a "battery value" which will be described below. In the example shown in Fig. 1, the managed software product 18 is installed from the CD-ROM to the user machine 10; it may be installed from any other recording medium or via a communication line.

A charge disk 22, containing specified data (including a charge value) on a floppy disk, functions as a battery value charger. Inserting this charge disk 22 into the floppy disk drive 20 causes a charge value to be read and enables the user to extend the allowable operation period of the managed software product 18. In this embodiment, several charge disks 22, each containing a unique charge value, are supplied to allow the user to select or buy a desired charge disk 22 to add a desired charge value to the battery value.

The managed software product 18 and the charge disk 22 are usually supplied from the same vendor. In this embodiment, the managed software product 18 includes the operation management program. Of course, the managed software product 18 and the operation management program may be separately loaded into the user machine 10.

In Fig. 1, the display 14 has a remainder information area 24 where remainder information is displayed and a remainder warning area 26 where a warning message is displayed when the remainder drops below the specified amount. These areas will be described later.

Fig. 2 shows the data structure of the charge disk 22. As shown in Fig. 2, the charge disk 22 contains a serial number 28, management information 30, and charge value (additional battery value) 32. The serial number 28 is a unique identification number that is assigned when the floppy disk is formatted. Usually, this number is not copied when the disk is copied. The management information 30 is created when the serial number 28 is encrypted. This management information 30 is copied when the disk is copied. Therefore, when the disk is copied illegally, the serial number 28 and the management information 30 do not match, thereby making it easy to determine that the disk is copied illegally. Of course, any other conventional security system may also be used instead of this method.

The charge value 32 is an additional charge value to be added to the battery value that is decremented as the user uses the managed software product 18. Charging the battery value with this charge value enables the user to extend the usage period.

When the battery value is managed in the "execution time based method" in which the battery value is decremented by the execution time of each function, an additional time is recorded as the charge value 32. On the other hand, when the battery value is managed in the "weight value based method" in which the battery value is decremented by the weight value of each function, the additional value is recorded as the charge value 32. These methods will be described in more detail later.

Although a floppy disk is used as the charge disk 22 in the embodiment shown in Fig. 1, other types of recording media may also be used. Also, as shown in another embodiment that will be explained later, a charge value may be sent over a communication line.

Fig. 3 shows the concept of the operation management system which uses the charge disk 22. The system is composed primarily of the user machine 10, charge disk 22, and vendor's machine 34. In this embodiment, the managed software product 18 including the operation management program 36 is installed in the user machine 10.

The charge disk 22 is generated on the vendor's machine 34 owned by the vendor which sold the managed software product 18. More specifically, the vendor's machine 34 has two software modules: the management information creation module 52 and the charge value issuance module 54. The management information creation module 52 encrypts the serial number 28 recorded on the charge disk 22, and writes the resulting management information 30 back onto the charge disk 22. Note that the operation management program 36, which contains the encryption condition or the decryption condition, can check whether or not the serial number 28 agrees with the management information 30. The charge value issuance module 54 records the charge value 32, which has been set by the vendor, onto the charge disk 22. In the execution time based method, the charge value 32 is recorded, for example, as 100 hours, 200 hours, or 500 hours. Note that the operation management program 36 contains an initial battery value (for example, 100 hours).

The operation management program 36 has a counter 38 which decrements the battery value (battery value management function). In this embodiment, the operation management program 36 decrements the counter 38 each time a "management target function" provided by the managed software product 18 is executed. When the battery value, i.e., the counter value, has decremented to the limit value of 0, the operation management program 36 prevents management target functions from being executed. That is, in this embodiment, when the battery value has reached a specified limit value, the execution of the managed software product 18 is limited and, when the battery value is charged with the charge value 32 contained on the charge disk 22, the charge value is added to the battery value and the resulting value is used as a new battery value. The usage period of the managed software product 18 is thus extended.

A history table 40 managed by the operation man-

agement program 36 contains history information on charge values recorded on the charge disk 22. Fig. 4 shows an example. As shown in Fig. 4, the history table 40 is composed of three columns: FD serial number column 40A, charge data/time column 40B, and charge value column 40C. The table may have other columns as necessary.

Referring to Fig. 3 again, the following explains how the battery value is managed. When the battery value is managed in the "execution time based method" described above, the execution time of each management target function, measured based on the internal clock 42, is subtracted from the battery value. On the other hand, when the "weight value based method" described above is used, the battery value is managed based on the usage amount table 44. Fig. 5 shows an example of the usage amount table 44. In this embodiment, the table contains entries, each consisting of a function name 44A and the corresponding usage amount 44B. It should be noted that each usage amount is used as a weight value. For example, a weight value is pre-defined according to the processing time of each function. Therefore, when a management target function is executed, the corresponding usage amount (weight value) is subtracted from the battery value.

The managed software product 18 shown in Fig. 3 has many user interface programs as well as many internal functions and common functions used by the programs. These functions are classified roughly into two: management target functions and management non-target functions. Whenever the managed software product 18 attempts to execute a management target function, the operation management program 36 references the battery value and, when it is zero or greater, allows the managed software product 18 to execute that function. When the managed software product 18 attempts to execute a management non-target function, the operation management program 36 does not check the battery value. For example, when input/output function for processing generated data 50 from the managed software product 18 is defined as a management non-target function, the input/output processing is always executed on the generated data 50, even if the usage period of the managed software product 18 has expired. This ensures that the generated data 50 are always processed, thus protecting user assets. Examples of management non-target functions include the data display function, data print function, and data plotter output function.

Management target functions include the data generation function. For example, when the managed software product is a CAD software product, the data generation function includes the straight-line drawing function, curved-line drawing function, circle drawing function, area fill-in function, area hatching function, and character insertion function.

Fig. 3 conceptually shows management target function execution module 46 which executes management target functions and management non-target function execution module 48 which executes management non-target functions. In this embodiment, the battery value is decremented only when a management target function is activated. Note that the battery may be decremented when both a management target function and a management non-target function are activated.

In addition to the data described above, the charge disk 22 may contain other types of data. For example, it may contain the name of the managed software product 18 which accepts a charge value. In this case, the name of the managed software product 18 is used as follows. When the charge disk 22 is read, the operation management program 36 checks whether or not the name of the managed software recorded on the charge disk 22 matches that of the managed software product 18 installed in the user machine 10 and, only when they match, accepts the charge value 32.

The battery value described above is stored on the hard disk and then copied into the computer's RAM. The battery value in the RAM is decremented whenever a management target function is executed. Also, at an interval or as necessary, the battery value in the RAM replaces the battery value on the hard disk. This means that, even when the computer fails, the battery value is not erased. The battery value may also be maintained in some other way.

Fig. 17 is a flowchart showing how the operation management program operates when it accepts an instruction requesting the execution of a managed software product function. The following explains this processing in more detail.

Upon receiving from a user an instruction requesting the execution of a function of the managed software product while the managed software product is in execution (S601), the operation management program checks whether the requested function is a management target function or a management non-target function (S602). When the function is a management target function (S603), the operation management program performs the processing shown in Fig. 6 or Fig. 7 (S604). When the function is a management non-target function (S603), the program executes the function immediately. (S605). This processing is repeated whenever an execution instruction is received.

Next, referring to Fig. 3, the execution of a management target function in the execution time based method is explained with the use of Fig. 6.

When the user requests the execution of a management target function while the managed software product 18 shown in Fig. 3 is in execution, the routine shown in Fig. 6 is started. First, the management target function execution module 46 or the operation management program 36 reads the battery value to check if it is greater than zero. If the battery value is zero or less, the routine is terminated. That is, the requested management target function cannot be started. Note that a management non-target function is started even if the battery value is

zero.

In S102, the routine gets the start time from the internal clock 42 before starting the requested management target function and, in S103, starts the management target function. In S104, the routine gets the end time from the internal clock 42 and, in S105, subtracts the start time from the end time to calculate the processing time (execution time) of the processing executed in S103.

In S106, the routine subtracts the processing time calculated in S105 from the battery value. In S107, the routine checks if the resulting battery value is equal to or less than the warning value and, if so, displays a message in the remainder warning area 26 shown in Fig. 1. If the resulting battery value is greater than the warning value, the routine does not display the message. As shown in Fig. 1, the remainder information area 24 is displayed during execution of the managed software product 18 (see Fig. 1) to allow the user to check the remaining amount. This helps the user determine how long he can execute the managed software product 18.

Fig. 7 shows the processing of a management target function in the weight value based method.

When the execution of a management target function is requested as described above, the routine references the battery value in S201 to check if it is equal to or greater than 0. If it is, the routine executes the requested management target function in S202 and, in S203, references the usage amount table 44 shown in Fig. 5 to find the usage amount (weight value) of the executed management target function. Then, in S204, the routine subtracts the processing amount found in S203 from the battery value to find a new battery value. In S205, the routine checks if the battery value is less than the warning value and, if so, displays a message in the remainder warning area 26 in S206.

The "execution time based method" shown in Fig. 6 allows the user to manage operation using a physical amount that is easy to understand. In addition, the user can manage operation in a relatively simple configuration. On the other hand, the "weight value based method" shown in Fig. 7 gives the user the same result regardless of the CPU speed of the user's machine.

Next, referring to Fig. 3, the charge disk 22 read processing is explained with the use of Fig. 8.

This processing is started when the charge disk 22 is inserted into the floppy disk drive 20 as shown in Fig. 1. The routine reads the serial number in S301, and the management information in S302, both from the charge disk 22. In S303, the routine encrypts the serial number according to the encryption condition, or decrypts the management information according to the decryption condition, and compares the serial number with the management information. This comparison determines whether or not the charge disk 22 is legal. For example, when the disk is illegally copied, the management information 30 is copied, but the serial number 28 is not copied but replaced. This results in a mismatch between the serial number 28 and the management information 30, thereby making it possible to find an illegal copy.

In S304, the routine checks if the charge disk 22 is valid and, if it is not valid, terminates processing in S308. If it is valid, the routine references the history table 40, containing past charge history data, in S305 to check the validity of the charge value 32 recorded on the charge disk 22. To do so, the routine first checks to see if the serial number 28 of the charge disk 22 is in the history table 40. If the serial number is found, the routine takes the following steps to check if the charge value 32 recorded on the charge disk 22 is valid. The routine finds the charge value initially recorded on the charge disk 22 and, from that initial value, subtracts the actual charge value to find the remainder. The next time the battery value is charged, the routine compares the remainder with the charge value currently recorded on the charge disk. If the charge value on the charge disk 22 is greater than the remainder, the routine determines in S306 that the charge disk is not valid and terminates processing in S308. If the routine finds that the charge value 32 on the charge disk 22 is valid, it performs the charge processing, shown in Fig. 9, in S307.

Fig. 9 shows an example of charge processing. In S401, the routine references the counter 38 to read the current battery value and, in S402, reads the charge value from the charge disk 22. In S403, the routine asks the user to type an actual charge value that does not exceed the charge value 32 recorded on the charge disk 22. The user types the charge value, for example, from the keyboard. In S404, the routine checks that the specified charge value is less than the charge value on the charge disk 22. If the specified charge value is greater than the charge value on the charge disk 22, the routine asks the user to retype the charge value.

In S405, the routine adds the specified charge value to the battery value, thus charging the battery value. In S406, the routine subtracts the specified charge value from the initial charge value and writes the resulting value on the charge disk 22 as a new charge value 32 . If the initial charge value 32 is exhausted, the routine writes the value of 0 on the charge disk 22 to virtually erase the charge value. The value of 0 prevents the charge disk 22 from being re-used. In S407, a record relating to the charge processing is added to the history table 40.

In the above embodiment, the user specifies an actual charge value. Instead of having the user specify a value, a pre-defined charge value may be added to the battery value at that time.

Fig. 10 shows another embodiment according to the present invention. In the embodiment described above, the battery value is charged using a recording medium. In this embodiment, the battery value is charged via a communication line 60. For the same components as those used in the above embodiment, the same numbers are assigned and their descriptions are omitted.

The user machine 10 in Fig.10 is connected to the

host machine 62 via the communication line 60. From this host machine 62, send data 64 shown in Fig. 11 are sent to the user machine 10 to charge the battery value.

In Fig. 11, address information 68 specifies the address of the user machine 10. Management information 70 is created by encrypting the serial number on the recording medium containing the managed software product 18. A charge value 72, a value to be added to the battery value as with the above embodiment, is an additional period of time in the execution time based method, and is an additional amount in the weight value based method.

Fig. 12 illustrates the system concept of this embodiment.

As described above, the user machine 10 is connected to the host machine 62 via the communication line 60. That is, this host machine 62 is connected to each of a number of user machines 10 for integrated operation management. This host machine 62 has a management information creation module 76, charge value issuance module 78, user registration table 80, and billing module 82. The management information creation module 76 creates the management information 70 shown in Fig. 11, and the charge value issuance module 78 issues a charge value 72 in response to a request from the user machine 10. As shown in Fig. 13, the user registration table 80 is composed primarily of the user ID column 80A, user name column 80B, and request charge value column 80C. The billing module 82 references the user registration table 80 to automatically issue a bill for a requested amount whenever a charge value is issued, or at some specified interval.

Next, referring to Fig. 12, the operation of this embodiment is explained with the use of Fig. 14. The operation of the user machine 10 is shown in the left side of Fig. 14, while that of the host machine 62 is shown on the right.

First, in S501 and S502, the user machine 10 is connected to the host machine 62 via a communication line. In S503, the user machine 10 generates a request for a charge value that will be sent to the host machine 62. In this case, the request contains at least the serial number of the CD-ROM containing the managed software product 18 and information on the charge value. In S504, the user machine sends the request to the host machine and, in S505, the host machine receives the request.

In S506, the host machine checks the user registration table 80. If the host machine finds, in S507, that the requesting user is registered in the host machine 62, the management information creation module 76 creates management information based on the serial number in S508, and the charge value issuance module 78 generates a charge value in response to the request from the user. In S509, the host machine 62 sends the management information and the charge value to the user machine 10 as the send data 64 shown in Fig. 11. In S510, the user machine 10 receives the send data 64. In S511 and S512, the user machine 10 and the host machine 62 are disconnected.

In S513, the operation management program 36 compares the serial number 74 with the management information 70 to check to see if the data received by the user machine 10 are valid. This prevents the user from illegally charging the battery value. If it is found in S514 that the send data are valid, the charge processing is performed in S515. This charge processing is the same as that in Fig. 9.

As shown in Fig. 12, this embodiment may also use the execution time based method or the weight value based method in order to manage the battery value.

Although the battery value is charged over a communication line such as a telephone line in the above embodiment, it may also be charged over a communication satellite (satellite line).

In the above embodiments, the operation management program 36 is included in the managed software product 18. Of course, an external program can manage the operation of the managed software product 18. Fig. 15 shows the concept of such an embodiment.

As shown in Fig. 15, the operation system (OS) 83 is located between the hardware 81 and each of application programs 84, 86, and 88. The operation management program 36 according to the present invention may be located between the operation system 83 and the application program 84.

Operation management program 36 therefore functions as an interface program. Messages are exchanged between the operation management program 36 and the application program 84 according to some specific rule. Messages are also exchanged between the operation management program 36 and the operation system 83 according to a specific rule.

To execute a management target function in this configuration, the operation management program 36 references the battery value when it receives an execution request from the application program 84. If the battery value is not zero, the operation management program 36 sends an instruction to the operation system 83 while simultaneously decrementing the battery value by a value corresponding to the function. If the battery value is zero, the operation management program 36 sends a message back to the application program 84, indicating that the instruction cannot be executed.

To execute a management non-target function, the operation management program 36 does not reference the battery value when it receives an execution request from the application program 84 but instead sends the instruction directly to the operation system 83.

The battery value is decremented as management target functions are executed. Charging the battery value allows the user to extend the usage period of the application program 84, which may be supplied separately from the application program 84.

In the above embodiments, one operation management program manages one operation management program. It is also possible for one operation manage-

ment program to manage several application programs.

Fig. 16 shows an application of the present invention. The system shown in Fig. 16 is composed of one host machine 90 and several user machines 92. Within each user machine 92 are a managed software product 18 and the operation management program 36, which, in turn, contains the counter 38 where the battery value to be decremented is stored. In other words, the operation of the managed software product 18 is controlled by the value stored in the counter 38. To execute the managed software product 18 in this system, it is necessary to insert a battery disk 96 into the user machine 92 and to move the battery value from the battery disk 96 into the counter 38. The battery value is decremented as the operation of the managed software product 18 proceeds. When the user finishes the managed software product 18, a sequence of operations are executed to move the current counter value from the counter 38 to the battery disk 96. This initializes the counter 38 to zero just as it was before the battery disk 96 was inserted.

The host machine 90 has several disk drives into which a battery disk 96 is inserted to read the battery value that was returned to the battery disk 96. This host machine 90 is also used to charge the battery value on the battery disk 96.

Integrated management of the battery values on several battery disks 96 through the host machine 90 brings a benefit of integrally managing several managed software products 18.

This type of system may be used, for example, in a school or a business where many computers are installed. With an individual carrying his or her own portable battery disk 96, it is possible to check and control the software usage amount of each person. In this case, either the "execution time based method" or the "weight value based method" may be used.

Claims

1. An operation management system for managing the operation of a managed software product, comprising:

   battery value management means for decrementing a battery value according to the operation amount of said managed software product;
   operation limit means for limiting the operation of said managed software product when said battery value has decremented to a specified limit value; and
   charge means for adding a charge value to the current battery value when the charge value is entered from external means.

2. An operation management system according to claim 1, wherein said battery value management means find the operation amount for each execution of a function owned by said managed software product and subtract a value corresponding to said operation amount from said battery value.

3. An operation management system according to claim 2, further comprising:

   function category determination means for determining if a function to which an execution instruction is issued is a management target function or a management non-target function, wherein said battery value management means decrement said battery value only when said management target function is executed.

4. An operation management system according to claim 3, wherein

   said battery value management means has a weight table containing pairs of said management target function and a weight value representing said operation amount thereof, and
   said battery value management means subtract a weight value corresponding to said management target function from said battery value when said management target function is executed.

5. An operation management system according to claim 3, wherein, when said management target function is executed, said battery value management means measure the execution time and subtracts the execution time from said battery value.

6. An operation management system according to claim 3, wherein said operation limit means prevent said management target function from being executed but allows said management non-target function to be executed when said battery value has reached a limit value.

7. An operation management system according to claim 3, wherein said managed software product has a data generation function and a data output function and wherein said function category determination means determine said data generation function as said management target function and determine said data output function as said management non-target function.

8. An operation management system according to claim 1, further comprising remainder warning means for issuing a remainder warning when said battery value has decremented to a warning value.

9. An operation management system according to claim 1, further comprising remainder display

means for displaying said battery value during execution of said managed software product.

10. An operation management system for managing the operation of a managed software product, comprising:

battery value management means for decrementing a battery value according to the operation amount of said managed software product;
operation limit means for limiting the operation of said managed software product when said battery value has decremented to a specified limit value;
read means for reading a charge value from a recording medium containing the charge value thereon; and
charge means for adding said charge value to the current battery value.

11. An operation management system according to claim 10, further comprising erase means for erasing the charge value from said recording medium after said charge value is added.

12. An operation management system according to claim 10, further comprising:

specification means for allowing a user to specify an actual charge value by which the current battery value is to be actually charged, the actual charge value not exceeding the charge value recorded on said recording medium; and
rewrite means for rewriting the charge value on said recording medium with a remainder value after said actual charge value is added to the current battery value.

13. An operation management system according to claim 10, in which said recording medium contains not only said charge value, but also the identification number of the recording medium and management information generated through encryption of the identification number, said operation management system further comprising:

validity determination means for comparing said identification number with said management information considering the condition of said encryption to determine the validity of said recording medium.

14. An operation management system comprising:

a managed machine containing a managed software product; and
a managing machine connected to said managed machine with a communication line, wherein
said managed machine comprises:
battery value management means for decrementing a battery value according to the operation amount of said managed software product;
operation limit means for limiting the operation of said managed software product when said battery value has decremented to a specified limit value;
charge value receive means for receiving a charge value from said managing machine; and
charge means for adding said charge value to the current battery value, and wherein
said managing machine comprises:
charge value send means for sending said charge value to said managed machine.

15. An operation management system according to claim 14, wherein said managed machine further comprises:

notification means for notifying said managing machine of the identification number of a portable recording medium initially containing said managed software product; and
validity determination means for comparing management information sent from said managing machine with said identification number to determine the validity of the recording medium; and wherein said managing machine further comprises:
management information creation means for creating said management information generated by encrypting said notified identification number and for sending the management information to said managed machine.

16. An operation management system comprising:

at least one managed machine containing a managed software product; and
a managing machine for managing the operation of said managed machine, wherein said managed machine comprises:
a counter containing a battery value changing according to the operation amount of said managed software product;
first charge means for reading a battery value from a portable recording medium to store the battery value into said counter; and
first return means for writing the current battery value on said recording medium, and wherein,
said managing machine comprises:
second charge means for writing said battery value on said recording medium; and
second return means for reading said battery value from said recording medium.

17. An operation management method comprising:

a count value management step for changing a count value according to the operation amount of a managed software product;
an operation limit step for limiting the operation of said managed software product when said count value has reached a specified limit value; and
a charge step for charging the current count value or said limit value when a charge value is entered from external means.

18. A medium containing a management software product for managing the operation of a managed software product, wherein said managed software product and said management software product are executed on computers, said management software product comprising:

a module for changing a count value according to the operation amount of said managed software product;
a module for limiting the operation of said managed software product when said count value has reached a specified limit value; and
a module for charging the current count value or said limit value when a charge value is entered from external means.

19. A medium containing a charge value read by a management software product for use in managing the operation of a managed software product, wherein said managed software product and said management software product are executed on computers, said management software product comprising:

a module for changing a count value according to the operation amount of said managed software product;
a module for limiting the operation of said managed software product when said count value has reached a specified limit value; and
a module for charging the current count value or said limit value when said charge value is entered.

20. A computer system having an interface software product between an operation system and at least one application software product, wherein said interface software product comprises:

a module for changing a count value according to the operation amount of said application software product;
a module for limiting the operation of said application software product when said count value has reached a specified limit value; and
a module for charging the current count value or said limit value when a charge value is entered from external means.

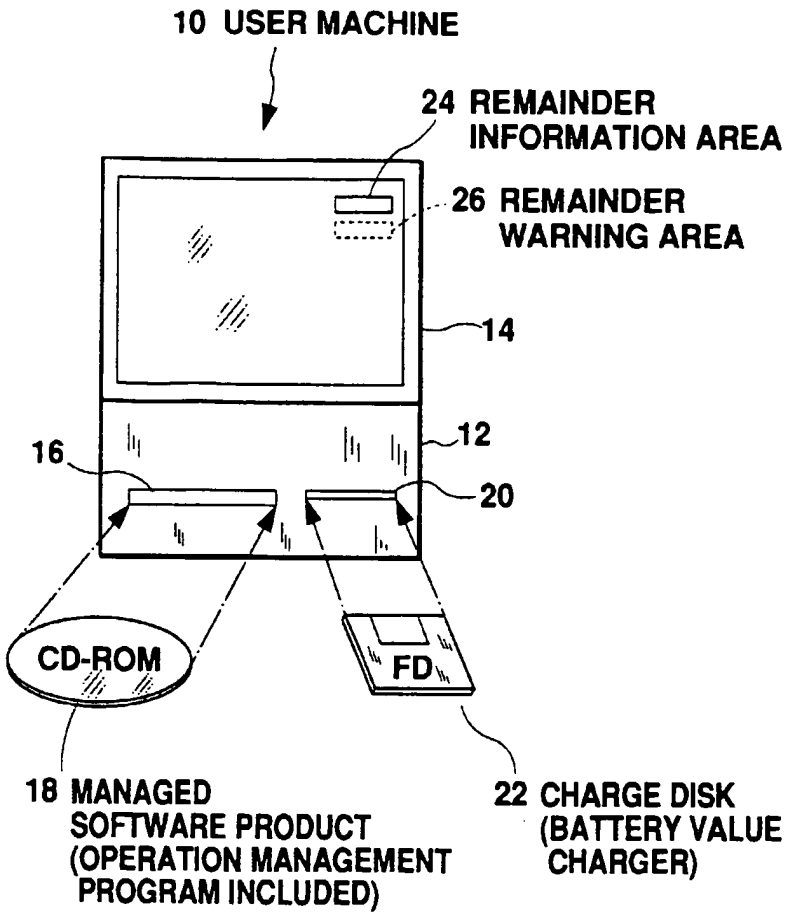


Fig. 1

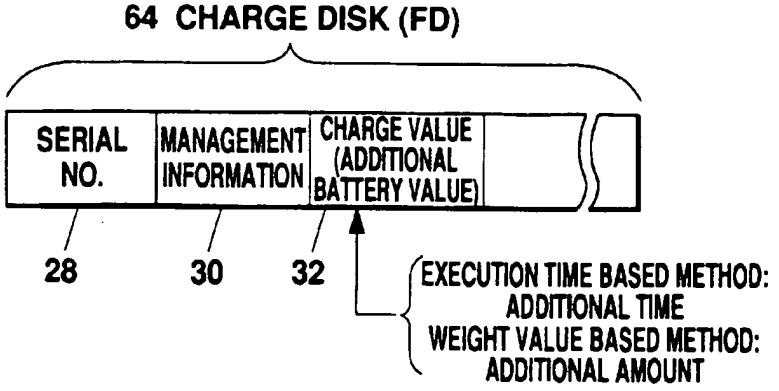


Fig. 2

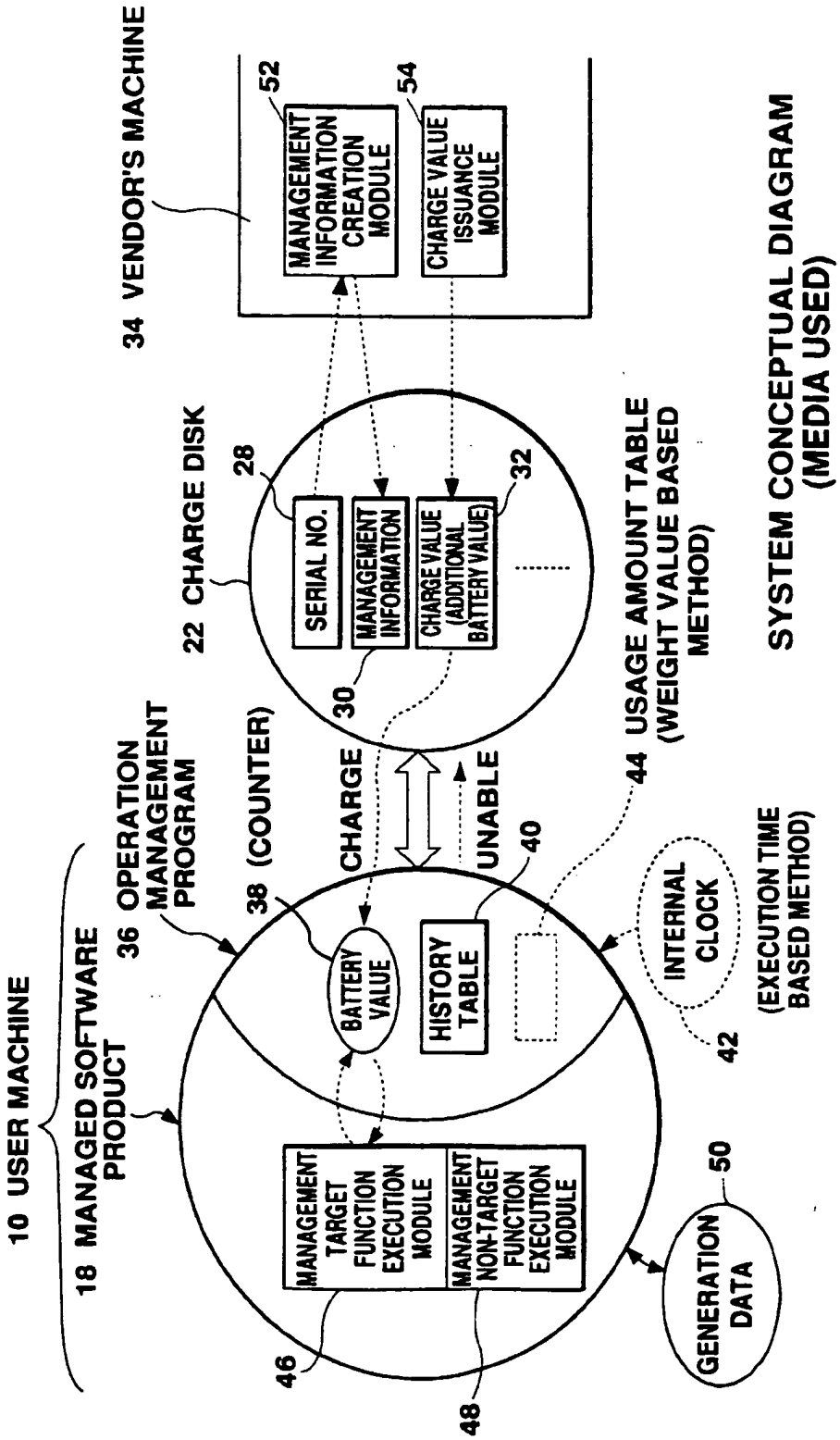


Fig. 3

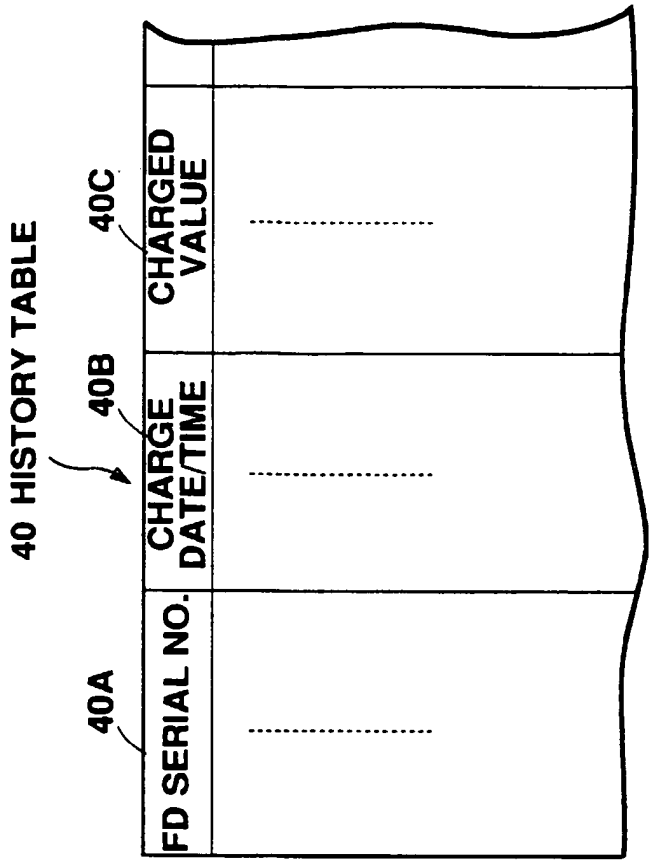


Fig. 5

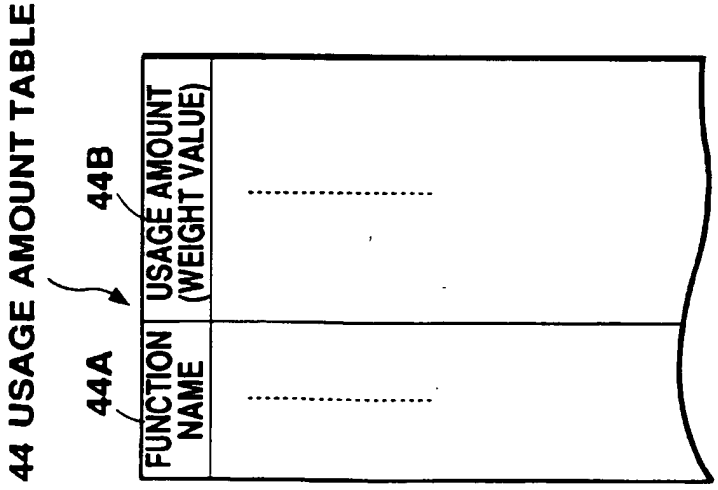


Fig. 4

MANAGEMENT TARGET FUNCTION EXECUTION
(EXECUTION TIME BASED METHOD)

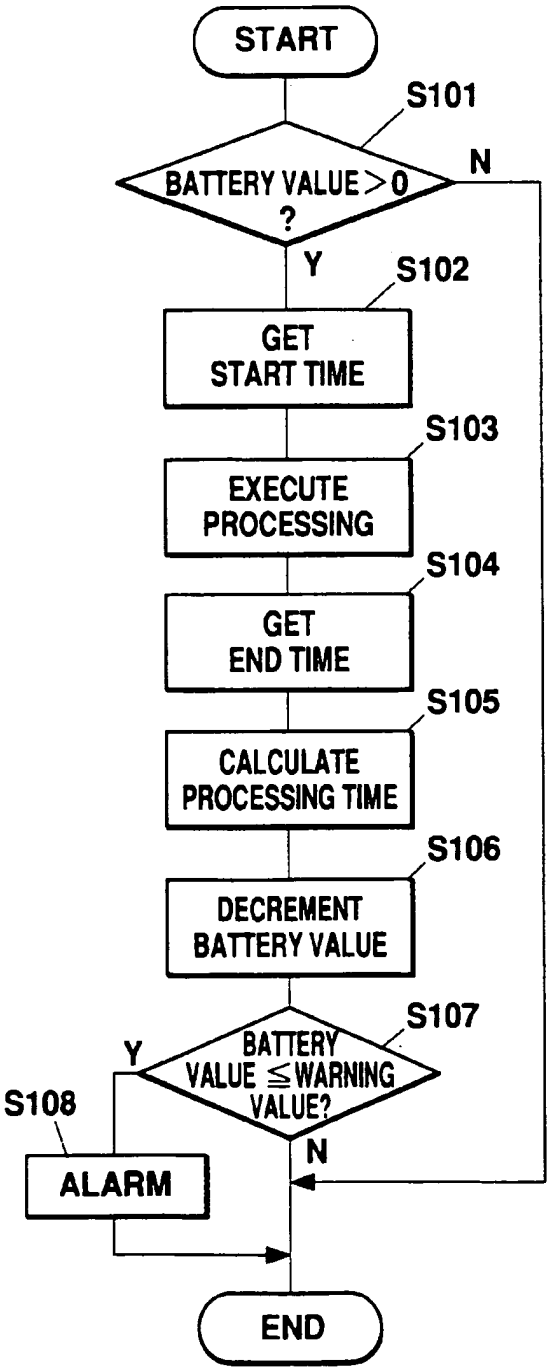


Fig. 6

MANAGEMENT TARGET FUNCTION EXECUTION
(WEIGHT VALUE BASED METHOD)

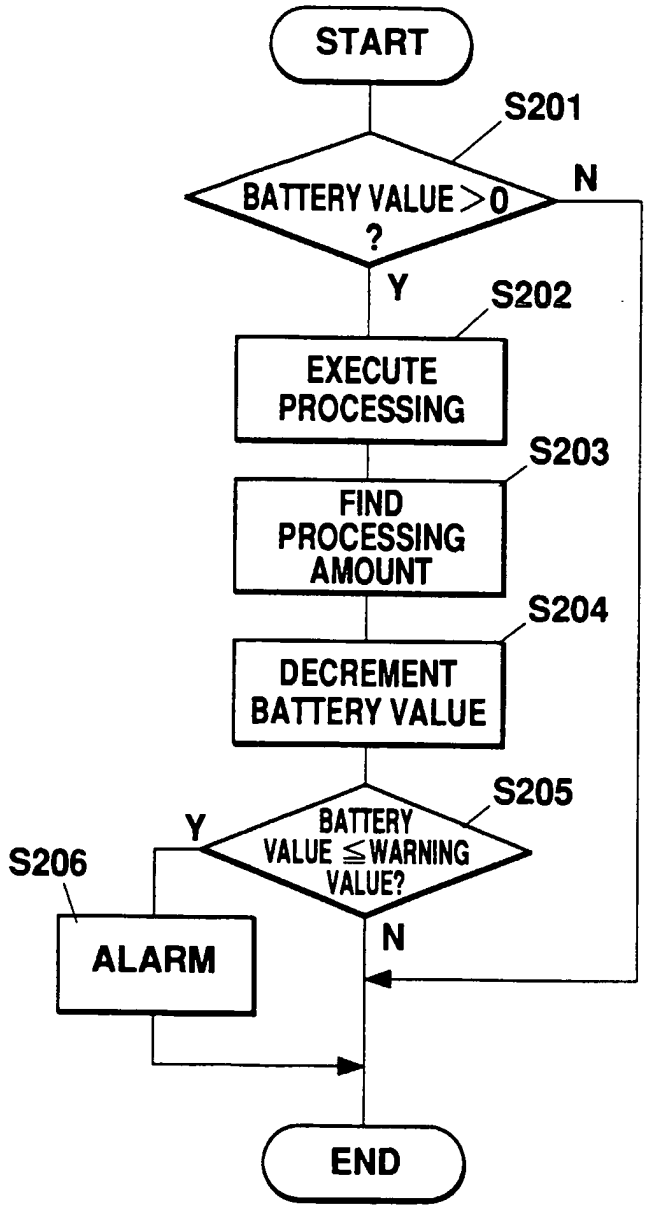


Fig. 7

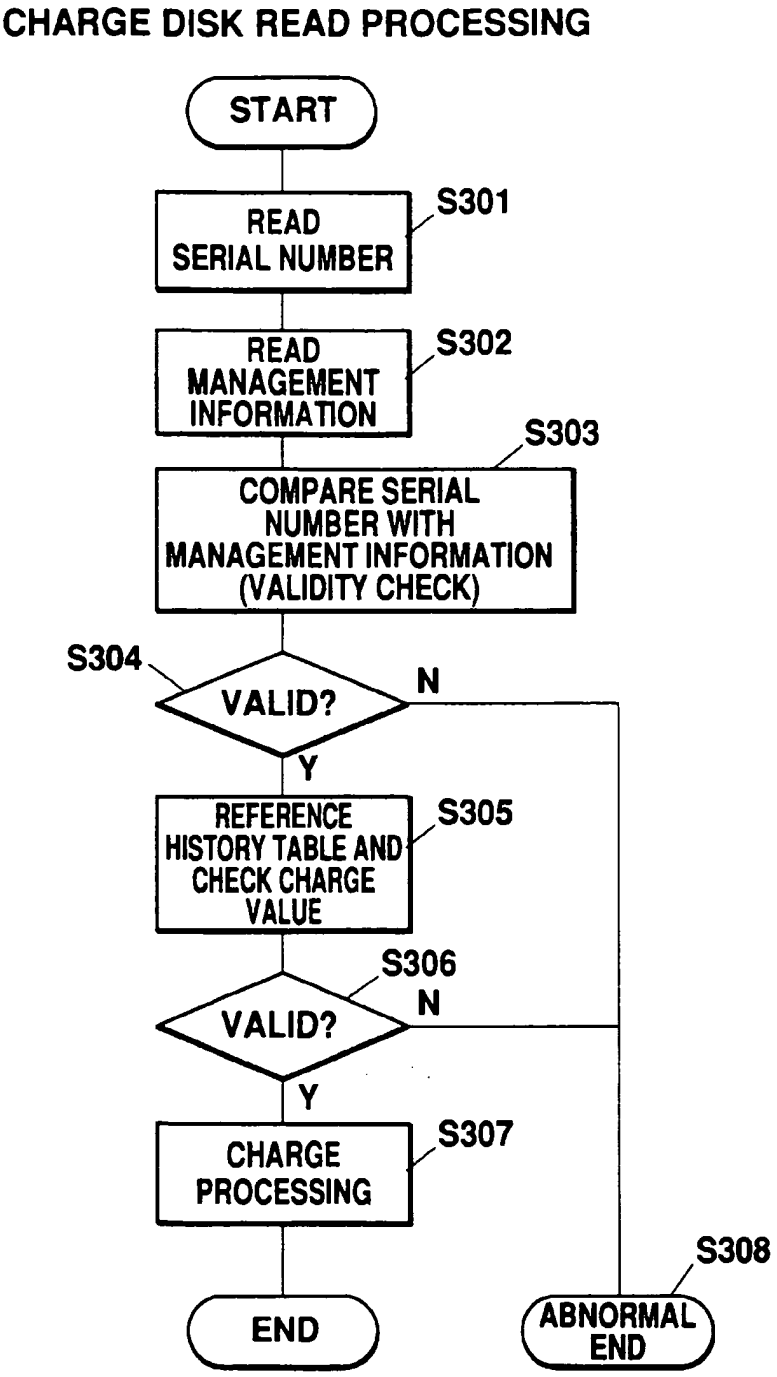


Fig. 8

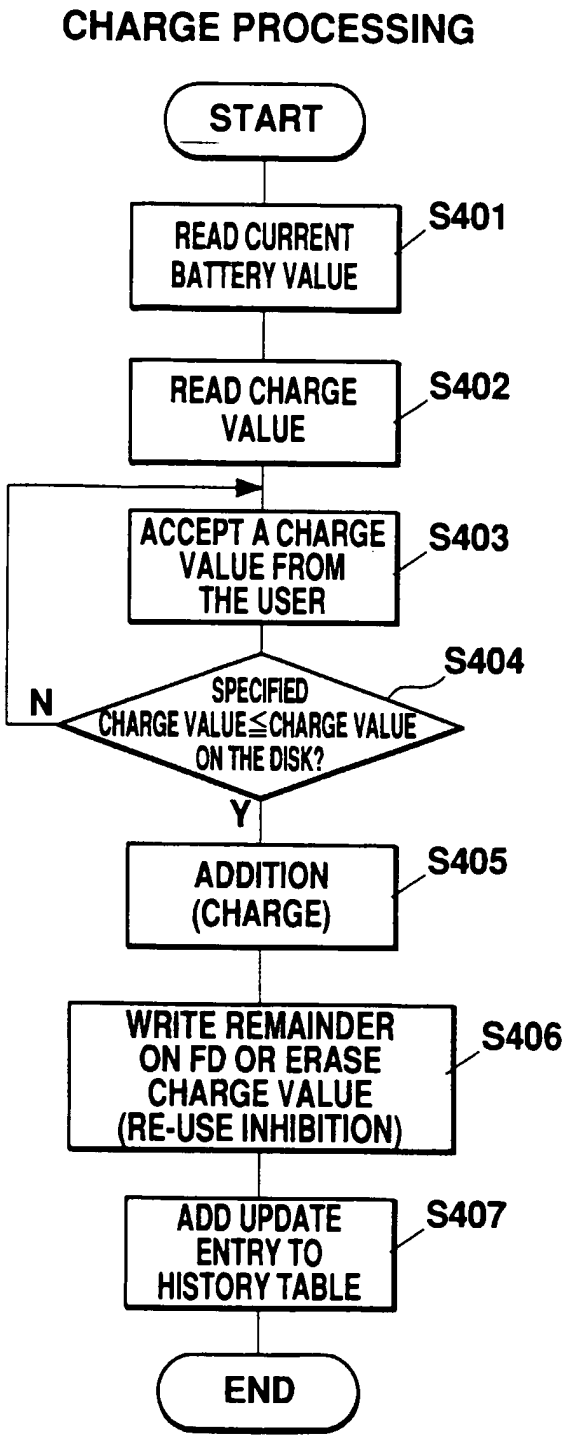


Fig. 9

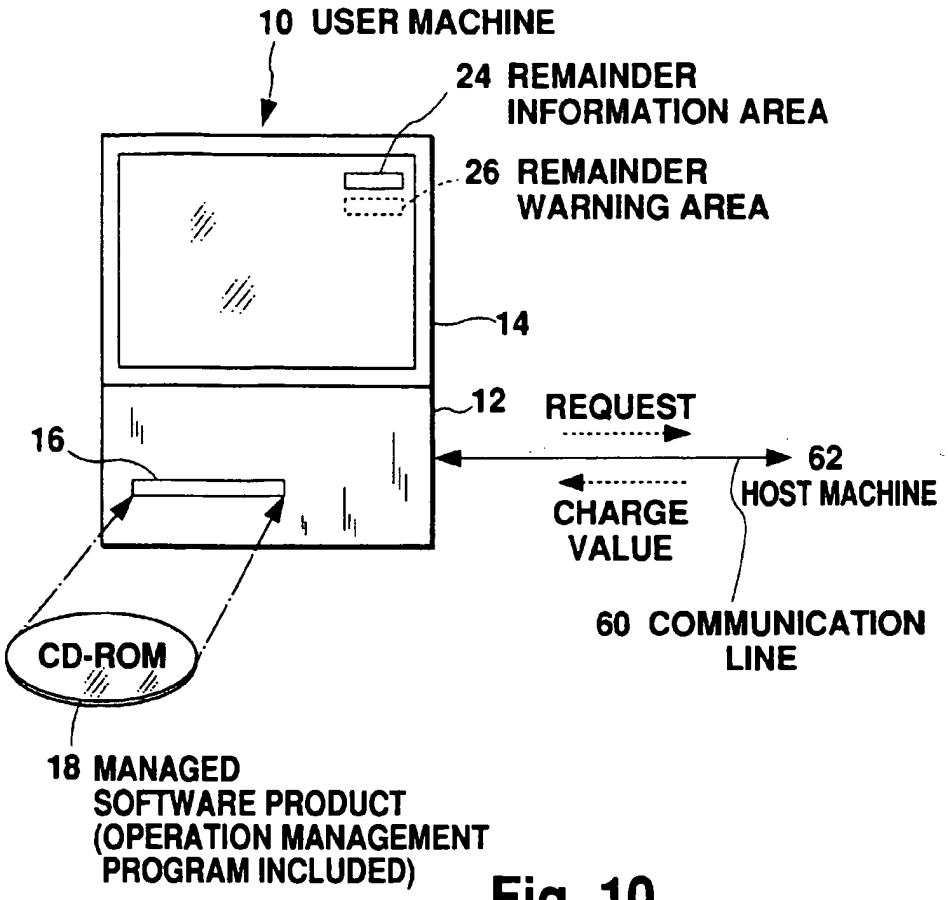


Fig. 10

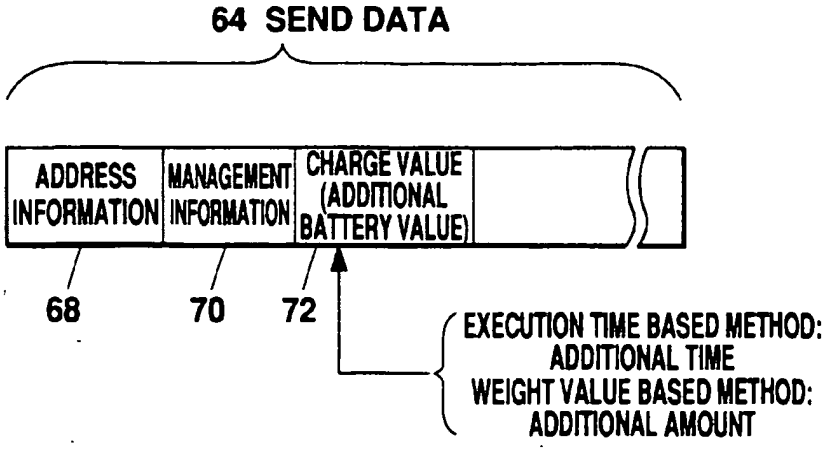


Fig. 11

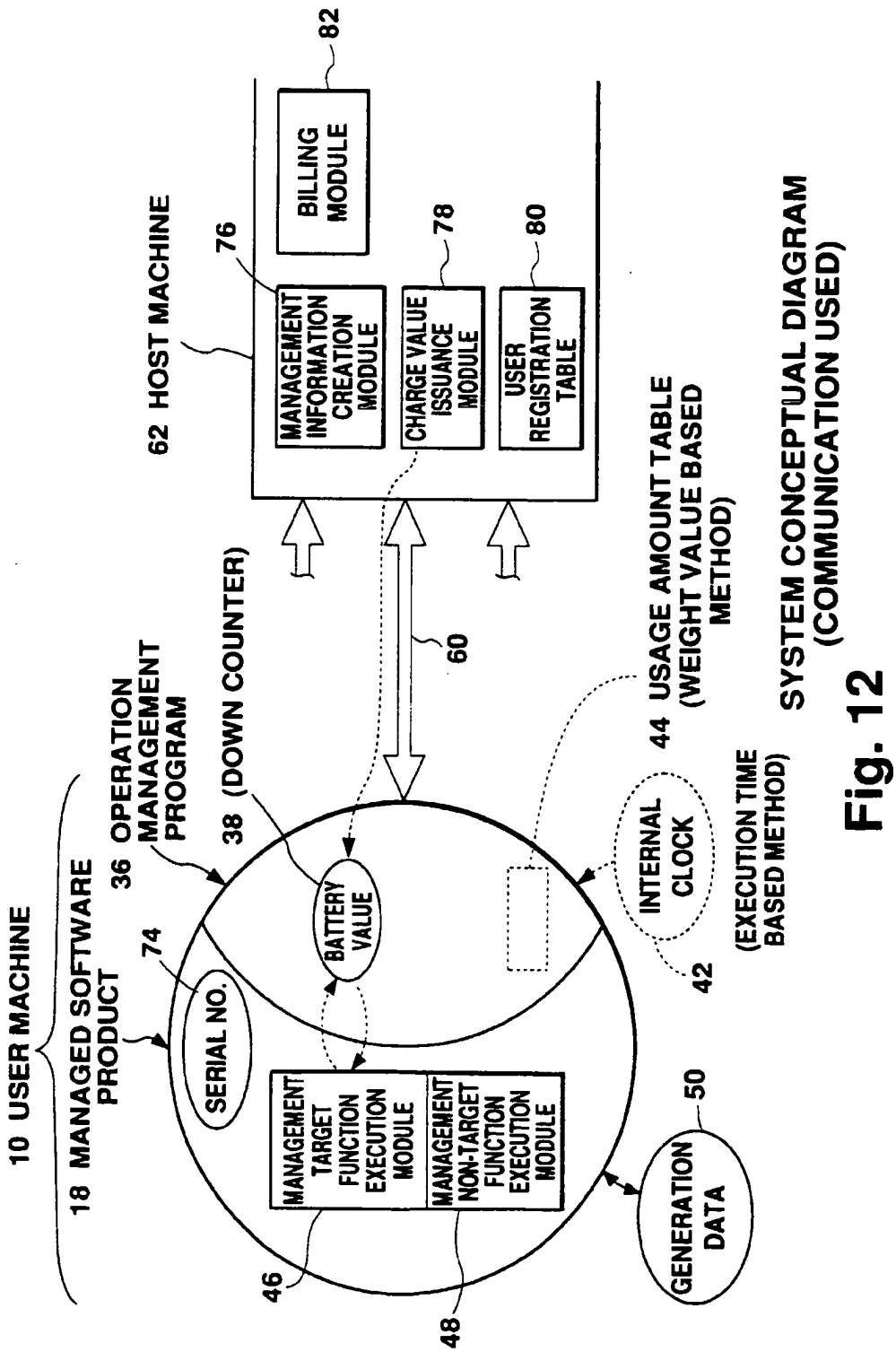


Fig. 12

SYSTEM CONCEPTUAL DIAGRAM (COMMUNICATION USED)

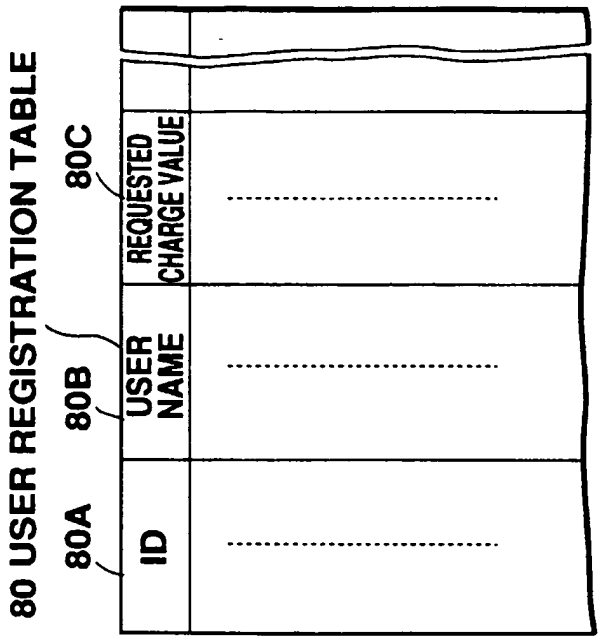


Fig. 13

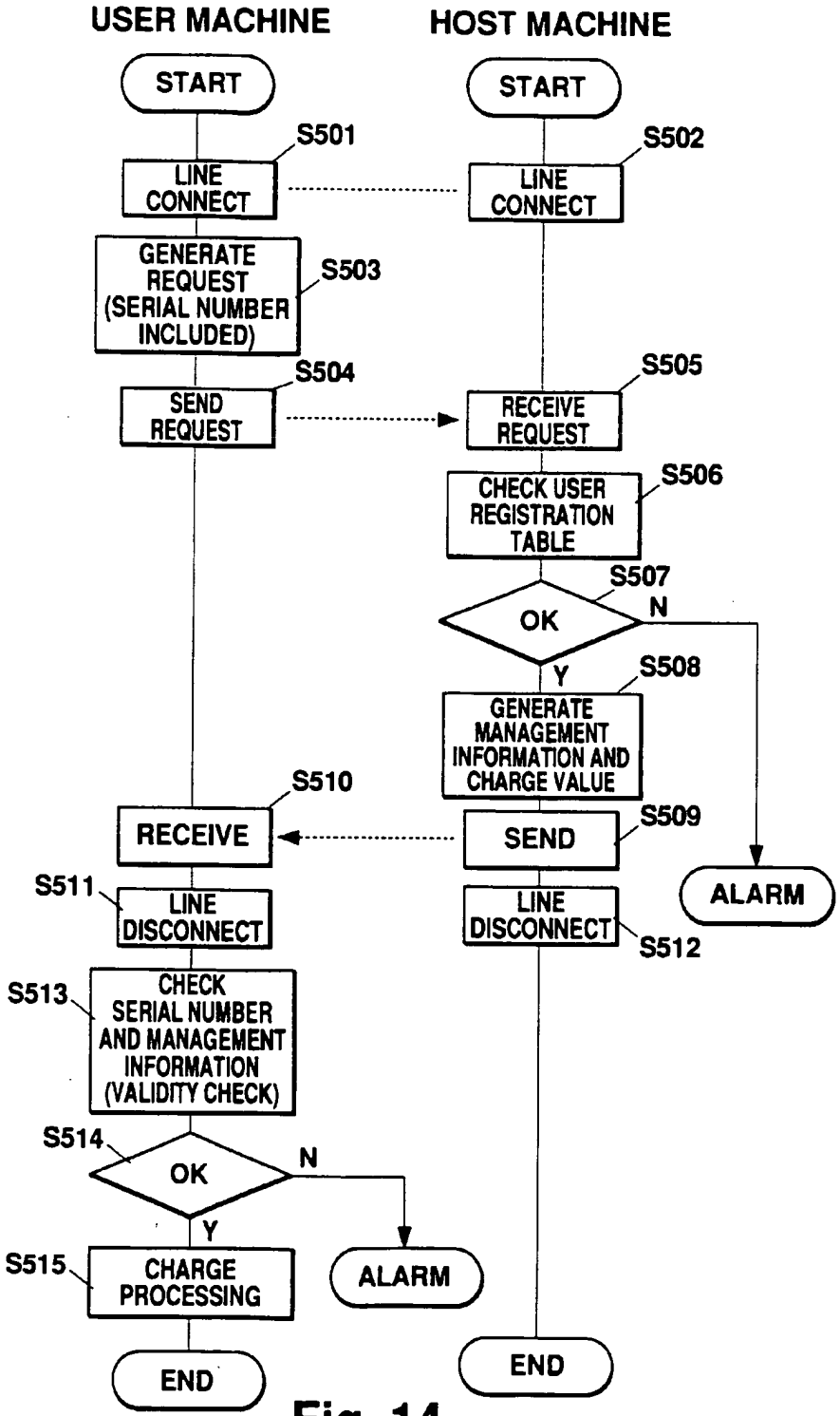


Fig. 14

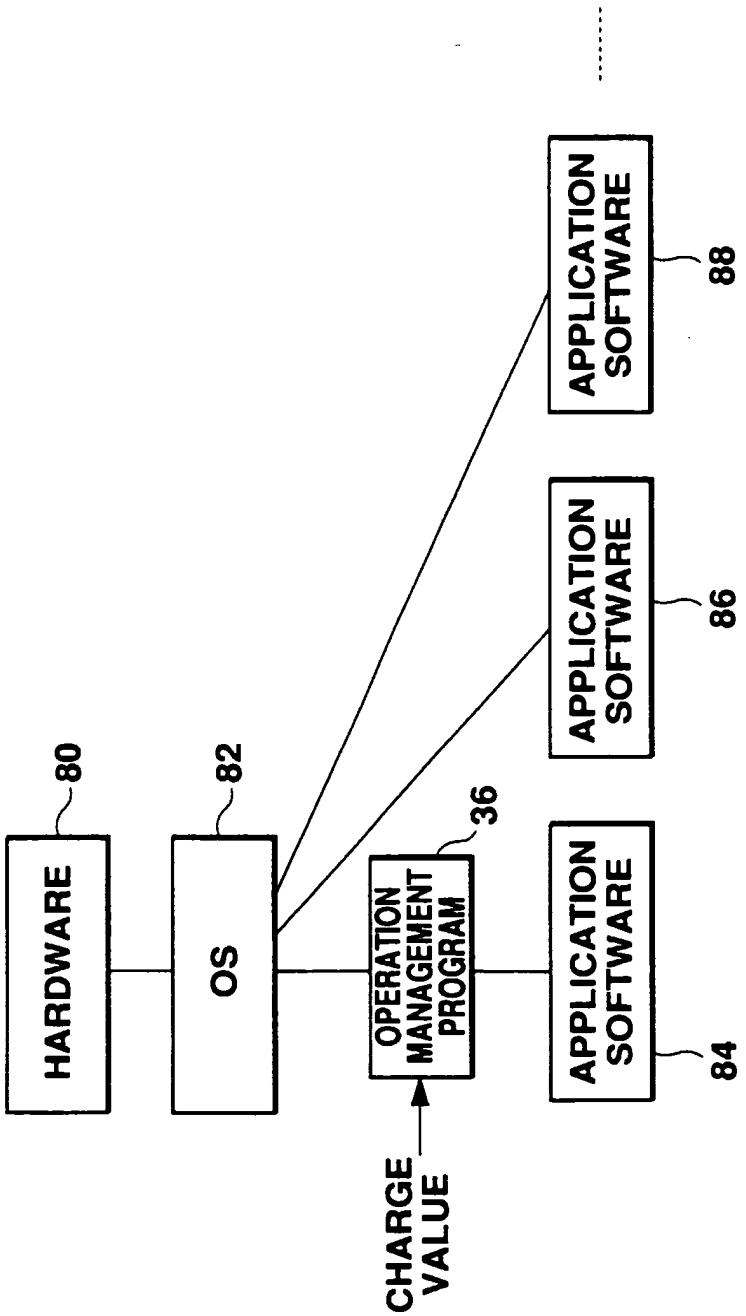


Fig. 15

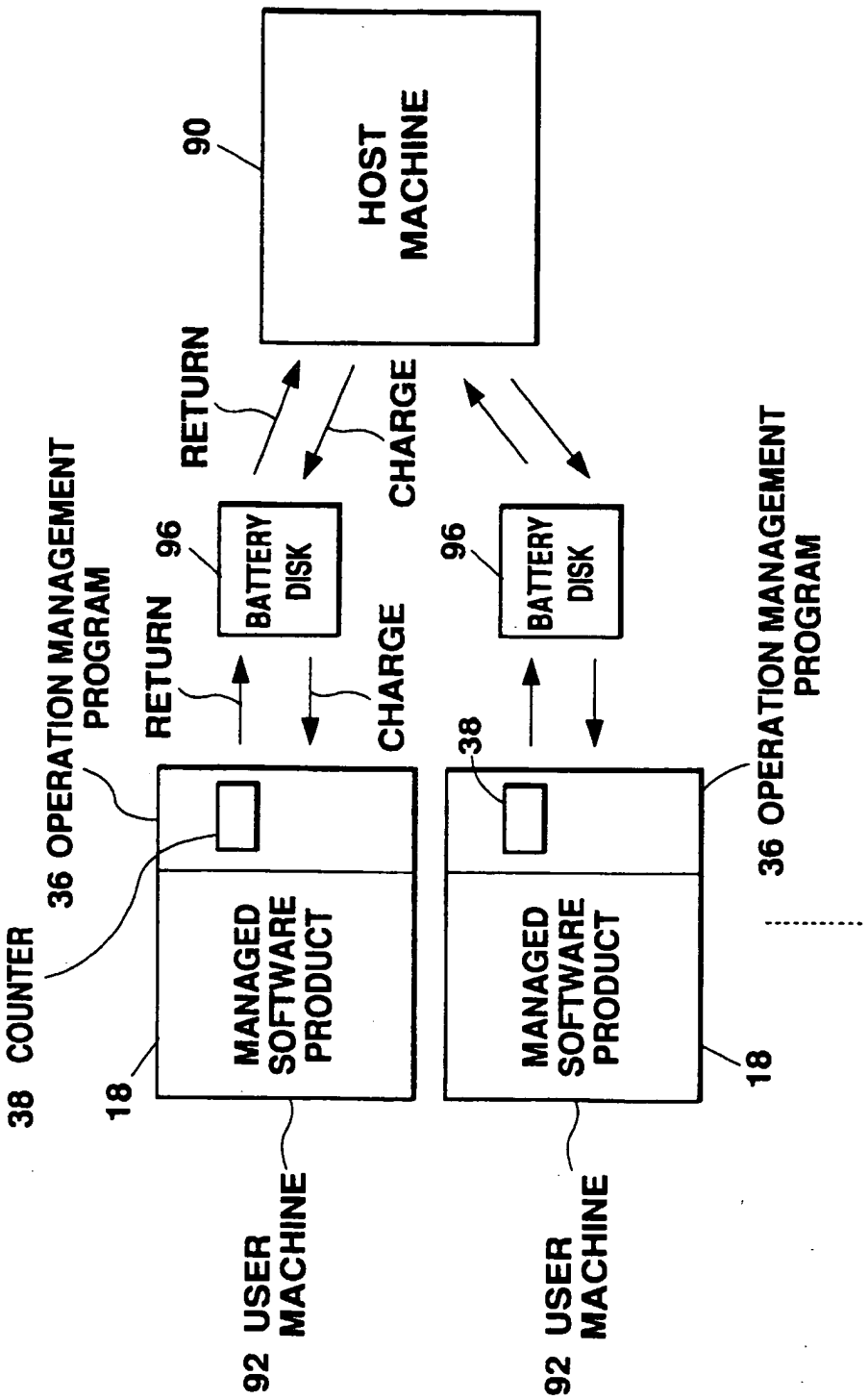


Fig. 16

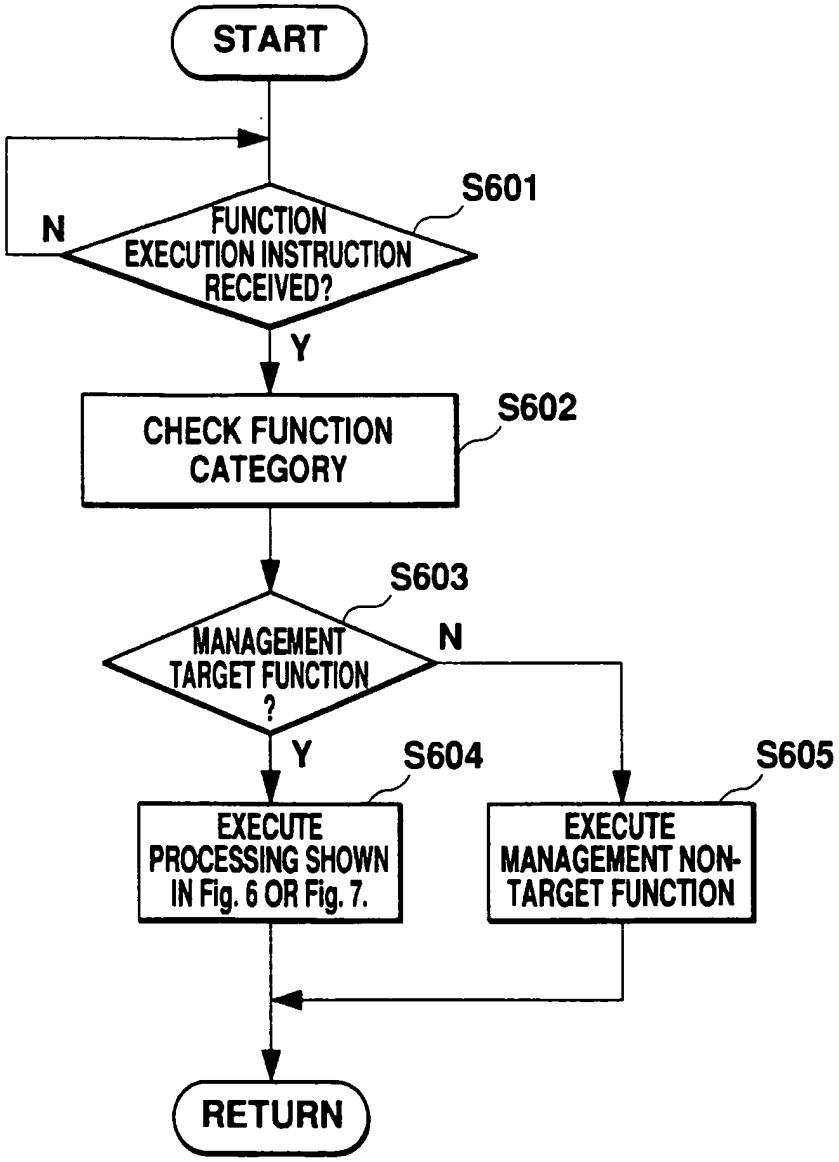


Fig. 17

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 715 245 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
05.06.1996 Bulletin 1996/23

(51) Int Cl.[6]: **G06F 1/00**

(21) Application number: **95308420.9**

(22) Date of filing: **23.11.1995**

(84) Designated Contracting States:
DE FR GB

(30) Priority: **23.11.1994 US 344042**

(71) Applicant: **XEROX CORPORATION**
Rochester New York 14644 (US)

(72) Inventors:
- **Stefik, Mark J.**
  Woodside, California 94062 (US)
- **Casey, Michalene M.**
  Morgan Hill, California 95037 (US)

(74) Representative: **Goode, Ian Roy**
Rank Xerox Ltd
Patent Department
Parkway
Marlow Buckinghamshire SL7 1YL (GB)

(54) System for controlling the distribution and use of digital works

(57) A system for controlling use and distribution of digital works, in which the owner of a digital work (101) attaches usage rights (102) to that work. Usage rights are granted by the "owner" of a digital work to "buyers" of the digital work. The usage rights define how a digital work may be used and further distributed by the buyer. Each right has associated with it certain optional specifications which outline the conditions and fees upon which the right may be exercised. Digital works are stored in a repository. A repository will process each request (103,104) to access a digital work by examining the corresponding usage rights (105). Digital work playback devices, coupled to the repository containing the work, are used to play, display or print the work. Access to digital works for the purposes of transporting between repositories (e.g. copying, borrowing or transfer) is carried out using a digital work transport protocol. Access to digital works for the purposes of replay by a digital work playback device(e.g. printing, displaying or executing) is carried out using a digital work playback protocol. Access is denied (106) or granted (107) depending whether the requesting repository has the required usage rights.

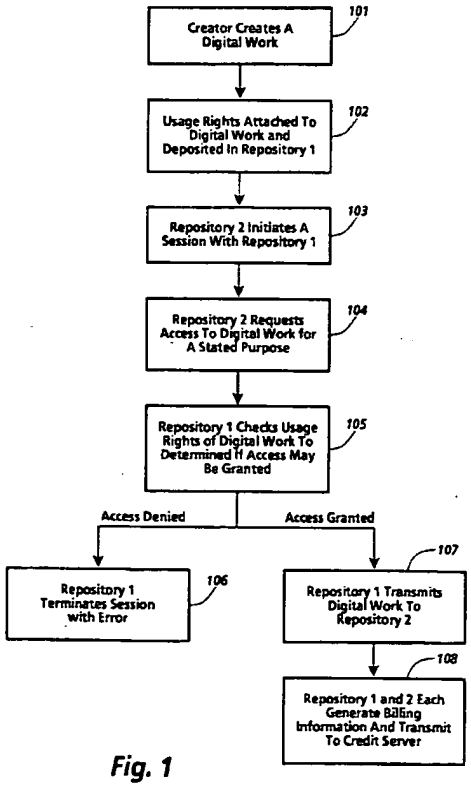


Fig. 1

EP 0 715 245 A1

Printed by Jouve, 75001 PARIS (FR)

Description

The present invention relates to the field of distribution and usage rights enforcement for digitally encoded works.

A fundamental issue facing the publishing and information industries as they consider electronic publishing is how to prevent the unauthorized and unaccounted distribution or usage of electronically published materials. Electronically published materials are typically distributed in a digital form and recreated on a computer based system having the capability to recreate the materials. Audio and video recordings, software, books and multimedia works are all being electronically published. Companies in these industries receive royalties for each accounted for delivery of the materials, e.g. the sale of an audio CD at a retail outlet. Any unaccounted distribution of a work results in an unpaid royalty (e.g. copying the audio recording CD to another digital medium.)

The ease in which electronically published works can be "perfectly" reproduced and distributed is a major concern. The transmission of digital works over networks is commonplace. One such widely used network is the Internet. The Internet is a widespread network facility by which computer users in many universities, corporations and government entities communicate and trade ideas and information. Computer bulletin boards found on the Internet and commercial networks such as CompuServ and Prodigy allow for the posting and retrieving of digital information. Information services such as Dialog and LEXIS/NEXIS provide databases of current information on a wide variety of topics. Another factor which will exacerbate the situation is the development and expansion of the National Information Infrastructure (the NII). It is anticipated that, as the NII grows, the transmission of digital works over networks will increase many times over. It would be desirable to utilize the NII for distribution of digital works without the fear of widespread unauthorized copying.

The most straightforward way to curb unaccounted distribution is to prevent unauthorized copying and transmission. For existing materials that are distributed in digital form, various safeguards are used. In the case of software, copy protection schemes which limit the number of copies that can be made or which corrupt the output when copying is detected have been employed. Another scheme causes software to become disabled after a predetermined period of time has lapsed. A technique used for workstation based software is to require that a special hardware device must be present on the workstation in order for the software to run, e.g., see US-A-4,932,054 entitled "Method and Apparatus for Protecting Computer Software Utilizing Coded Filter Network in Conjunction with an Active Coded Hardware Device." Such devices are provided with the software and are commonly referred to as dongles.

Yet another scheme is to distribute software, but which requires a "key" to enable its use. This is employed in distribution schemes where "demos" of the software are provided on a medium along with the entire product. The demos can be freely used, but in order to use the actual product, the key must be purchased. These schemes do not hinder copying of the software once the key is initially purchased.

It is an object of the present invention to provide an improved system and method for controlling the use and distribution of digital works.

The invention accordingly provides a system and method as claimed in the accompanying claims.

A system for controlling use and distribution of digital works is disclosed. A digital work is any written, aural, graphical or video based work including computer programs that has been translated to or created in a digital form, and which can be recreated using suitable rendering means such as software programs. The present invention allows the owner of a digital work to attach usage rights to the work. The usage rights for the work define how it may be used and distributed. Digital works and their usage rights are stored in a secure repository. Digital works may only be accessed by other secure repositories.

Usage rights for a digital work are embodied in a flexible and extensible usage rights grammar. Conceptually, a right in the usage rights grammar is a label attached to a predetermined behavior and conditions to exercising the right. For example, a COPY right denotes that a copy of the digital work may be made. A condition to exercising the right is the requester must pass certain security criteria. Conditions may also be attached to limit the right itself. For example, a LOAN right may be defined so as to limit the duration of which a work may be LOANed. Conditions may also include requirements that fees be paid.

A repository is comprised of a storage means for storing a digital work and its attached usage rights, an external interface for receiving and transmitting data, a processor and a clock. A repository has two primary operating modes, a server mode and a requester mode. When operating in a server mode, the repository is responding to requests to access digital works. When operating in requester mode, the repository is requesting access to a digital work.

Generally, a repository will process each request to access a digital work by examining the work's usage rights. For example, in a request to make a copy of a digital work, the digital work is examined to see if rights have been granted which would allow copies to be given out. If such a right has been granted, then conditions to exercise of the right are checked (e.g. a right to make 2 copies). If conditions associated with the right are satisfied, the copy can be made. Before transporting the digital work, any specified changes to the set of usage rights in the copy are attached to the copy of the digital work.

Repositories communicate utilizing a set of repository transactions. The repository transactions embody a set of

protocols for establishing secure sessions connections between repositories, and for processing access requests to the digital works.

Digital works are recreated on rendering systems. A rendering system is comprised of at least a rendering repository and a rendering device (e.g. a printer, display or audio system.) Rendering systems are internally secure. Access to digital works not contained within the rendering repository is accomplished via repository transactions with an external repository containing the desired digital work.

A system and method in accordance with the invention will now be described, by way of example, with reference to the accompanying drawings, in which:-

Figure 1 is a flowchart illustrating a simple instantiation of the operation of the currently preferred embodiment of the present invention.

Figure 2 is a block diagram illustrating the various repository types and the repository transaction flow between them in the currently preferred embodiment of the present invention.

Figure 3 is a block diagram of a repository coupled with a credit server in the currently preferred embodiment of the present invention.

Figures 4a and 4b are examples of rendering systems as may be utilized in the currently preferred embodiment of the present invention.

Figure 5 illustrates a contents file layout for a digital work as may be utilized in the currently preferred embodiment of the present invention.

Figure 6 illustrates a contents file layout for an individual digital work of the digital work of Figure 5 as may be utilized in the currently preferred embodiment of the present invention.

Figure 7 illustrates the components of a description block of the currently preferred embodiment of the present invention.

Figure 8 illustrates a description tree for the contents file layout of the digital work illustrated in Figure 5.

Figure 9 illustrates a portion of a description tree corresponding to the individual digital work illustrated in Figure 6.

Figure 10 illustrates a layout for the rights portion of a description block as may be utilized in the currently preferred embodiment of the present invention.

Figure 11 is a description tree wherein certain d-blocks have PRINT usage rights and is used to illustrate "strict" and "lenient" rules for resolving usage rights conflicts.

Figure 12 is a block diagram of the hardware components of a repository as are utilized in the currently preferred embodiment of the present invention.

Figure 13 is a block diagram of the functional (logical) components of a repository as are utilized in the currently preferred embodiment of the present invention.

Figure 14 is diagram illustrating the basic components of a usage right in the currently preferred embodiment of the present invention.

Figure 15 lists the usage rights grammar of the currently preferred embodiment of the present invention.

Figure 16 is a flowchart illustrating the steps of certificate delivery, hotlist checking and performance testing as performed in a registration transaction as may be performed in the currently preferred embodiment of the present invention.

Figure 17 is a flowchart illustrating the steps of session information exchange and clock synchronization as may be performed in the currently preferred embodiment of the present invention, after each repository in the registration transaction has successfully completed the steps described in Figure 16.

Figure 18 is a flowchart illustrating the basic flow for a usage transaction, including the common opening and closing step, as may be performed in the currently preferred embodiment of the present invention.

Figure 19 is a state diagram of server and client repositories in accordance with a transport protocol followed when moving a digital work from the server to the client repositories, as may be performed in the currently preferred embodiment of the present invention.

OVERVIEW

A system for controlling use and distribution of digital works is disclosed. The present invention is directed to supporting commercial transactions involving digital works.

Herein the terms "digital work", "work" and "content" refer to any work that has been reduced to a digital representation. This would include any audio, video, text, or multimedia work and any accompanying interpreter (e.g. software) that may be required for recreating the work. The term composite work refers to a digital work comprised of a collection of other digital works. The term "usage rights" or "rights" is a term which refers to rights granted to a recipient of a digital work. Generally, these rights define how a digital work can be used and if it can be further distributed. Each usage right may have one or more specified conditions which must be satisfied before the right may be exercised.

Figure 1 is a high level flowchart omitting various details but which demonstrates the basic operation of the present

invention. Referring to Figure 1, a creator creates a digital work, step 101. The creator will then determine appropriate usage rights and fees, attach them to the digital work, and store them in Repository 1, step 102. The determination of appropriate usage rights and fees will depend on various economic factors. The digital work remains securely in Repository 1 until a request for access is received. The request for access begins with a session initiation by another repository. Here a Repository 2 initiates a session with Repository 1, step 103. As will be described in greater detail below, this session initiation includes steps which helps to insure that the respective repositories are trustworthy. Assuming that a session can be established, Repository 2 may then request access to the Digital Work for a stated purpose, step 104. The purpose may be, for example, to print the digital work or to obtain a copy of the digital work. The purpose will correspond to a specific usage right. In any event, Repository 1 checks the usage rights associated with the digital work to determine if the access to the digital work may be granted, step 105. The check of the usage rights essentially involves a determination of whether a right associated with the access request has been attached to the digital work and if all conditions associated with the right are satisfied. If the access is denied, repository 1 terminates the session with an error message, step 106. If access is granted, repository 1 transmits the digital work to repository 2, step 107. Once the digital work has been transmitted to repository 2, repository 1 and 2 each generate billing information for the access which is transmitted to a credit server, step 108. Such double billing reporting is done to insure against attempts to circumvent the billing process.

Figure 2 illustrates the basic interactions between repository types in the present invention. As will become apparent from Figure 2, the various repository types will serve different functions. It is fundamental that repositories will share a core set of functionality which will enable secure and trusted communications. Referring to Figure 2, a repository 201 represents the general instance of a repository. The repository 201 has two modes of operation; a server mode and a requester mode. When in the server mode, the repository will be receiving and processing access requests to digital works. When in the requester mode, the repository will be initiating requests to access digital works. Repository 201 is general in the sense that its primary purpose is as an exchange medium for digital works. During the course of operation, the repository 201 may communicate with a plurality of other repositories, namely authorization repository 202, rendering repository 203 and master repository 204. Communication between repositories occurs utilizing a repository transaction protocol 205.

Communication with an authorization repository 202 may occur when a digital work being accessed has a condition requiring an authorization. Conceptually, an authorization is a digital certificate such that possession of the certificate is required to gain access to the digital work. An authorization is itself a digital work that can be moved between repositories and subjected to fees and usage rights conditions. An authorization may be required by both repositories involved in an access to a digital work.

Communication with a rendering repository 203 occurs in connection with the rendering of a digital work. As will be described in greater detail below, a rendering repository is coupled with a rendering device (e.g. a printer device) to comprise a rendering system.

Communication with a master repository 205 occurs in connection with obtaining an identification certificate. Identification certificates are the means by which a repository is identified as "trustworthy". The use of identification certificates is described below with respect to the registration transaction.

Figure 3 illustrates the repository 201 coupled to a credit server 301. The credit server 301 is a device which accumulates billing information for the repository 201. The credit server 301 communicates with repository 201 via billing transactions 302 to record billing transactions. Billing transactions are reported to a billing clearinghouse 303 by the credit server 301 on a periodic basis. The credit server 301 communicates to the billing clearinghouse 303 via clearinghouse transactions 304. The clearinghouse transactions 304 enable a secure and encrypted transmission of information to the billing clearinghouse 303.

RENDERING SYSTEMS

A rendering system is generally defined as a system comprising a repository and a rendering device which can render a digital work into its desired form. Examples of a rendering system may be a computer system, a digital audio system, or a printer. A rendering system has the same security features as a repository. The coupling of a rendering repository with the rendering device may occur in a manner suitable for the type of rendering device.

Figure 4a illustrates a printer as an example of a rendering system. Referring to Figure 4, printer system 401 has contained therein a printer repository 402 and a print device 403. It should be noted that the the dashed line defining printer system 401 defines a secure system boundary. Communications within the boundary are assumed to be secure. Depending on the security level, the boundary also represents a barrier intended to provide physical integrity. The printer repository 402 is an instantiation of the rendering repository 205 of Figure 2. The printer repository 402 will in some instances contain an ephemeral copy of a digital work which remains until it is printed out by the print engine 403. In other instances, the printer repository 402 may contain digital works such as fonts, which will remain and can be billed based on use. This design assures that all communication lines between printers and printing devices are

encrypted, unless they are within a physically secure boundary. This design feature eliminates a potential "fault" point through which the digital work could be improperly obtained. The printer device 403 represents the printer components used to create the printed output.

Also illustrated in Figure 4a is the repository 404. The repository 404 is coupled to the printer repository 402. The repository 404 represents an external repository which contains digital works.

Figure 4b is an example of a computer system as a rendering system. A computer system may constitute a "multi-function" device since it may execute digital works (e.g. software programs) and display digital works (e.g. a digitized photograph). Logically, each rendering device can be viewed as having its own repository, although only one physical repository is needed. Referring to Figure 4b, a computer system 410 has contained therein a display/execution repository 411. The display/execution repository 411 is coupled to display device, 412 and execution device 413. The dashed box surrounding the computer system 410 represents a security boundary within which communications are assumed to be secure. The display/execution repository 411 is further coupled to a credit server 414 to report any fees to be billed for access to a digital work and a repository 415 for accessing digital works stored therein.

STRUCTURE OF DIGITAL WORKS

Usage rights are attached directly to digital works. Thus, it is important to understand the structure of a digital work. The structure of a digital work, in particular composite digital works, may be naturally organized into an acyclic structure such as a hierarchy. For example, a magazine has various articles and photographs which may have been created and are owned by different persons. Each of the articles and photographs may represent a node in a hierarchical structure. Consequently, controls, i.e. usage rights, may be placed on each node by the creator. By enabling control and fee billing to be associated with each node, a creator of a work can be assured that the rights and fees are not circumvented.

In the currently preferred embodiment, the file information for a digital work is divided into two files: a "contents" file and a "description tree" file. From the perspective of a repository, the "contents" file is a stream of addressable bytes whose format depends completely on the interpreter used to play, display or print the digital work. The description tree file makes it possible to examine the rights and fees for a work without reference to the content of the digital work. It should be noted that the term description tree as used herein refers to any type of acyclic structure used to represent the relationship between the various components of a digital work.

Figure 5 illustrates the layout of a contents file. Referring to Figure 5, a digital work is comprised of story A 510, advertisement 511, story B 512 and story C 513. It is assumed that the digital work is stored starting at a relative address of 0. Each of the parts of the digital work are stored linearly so that story A 510 is stored at approximately addresses 0-30,000, advertisement 511 at addresses 30,001-40,000, story B 512 at addresses 40,001-60,000 and story C 513 at addresses 60,001-85K. The detail of story A 510 is illustrated in Figure 6. Referring to Figure 6, the story A 510 is further broken down to show text 614 stored at address 0-1500, soldier photo 615 at addresses 1501-10,000, graphics 616 stored at addresses 10,001-25,000 and sidebar 617 stored address 25,001-30,000. Note that the data in the contents file may be compressed (for saving storage) or encrypted (for security).

From Figures 5 and 6 it is readily observed that a digital work can be represented by its component parts as a hierarchy. The description tree for a digital work is comprised of a set of related descriptor blocks (d-blocks). The contents of each d-block is described with respect to Figure 7. Referring to Figure 7, a d-block 700 includes an identifier 701 which is a unique identifier for the work in the repository, a starting address 702 providing the start address of the first byte of the work, a length 703 giving the number of bytes in the work, a rights portion 704 wherein the granted usage rights and their status data are maintained, a parent pointer 705 for pointing to a parent d-block and child pointers 706 for pointing to the child d-blocks. In the currently preferred embodiment, the identifier 701 has two parts. The first part is a unique number assigned to the repository upon manufacture. The second part is a unique number assigned to the work upon creation. The rights portion 704 will contain a data structure, such as a look-up table, wherein the various information associated with a right is maintained. The information required by the respective usage rights is described in more detail below. D-blocks form a strict hierarchy. The top d-block of a work has no parent; all other d-blocks have one parent. The relationship of usage rights between parent and child d-blocks and how conflicts are resolved is described below.

A special type of d-block is a "shell" d-block. A shell d-block adds no new content beyond the content of its parts. A shell d-block is used to add rights and fee information, typically by distributors of digital works.

Figure 8 illustrates a description tree for the digital work of Figure 5. Referring to Figure 8, a top d-block 820 for the digital work points to the various stories and advertisements contained therein. Here, the top d-block 820 points to d-block 821 (representing story A 510), d-block 822 (representing the advertisement 511), d-block 823 (representing story B 512) and and d-block 824 (representing story C 513).

The portion of the description tree for Story A 510 is illustrated in Figure 9. D-block 925 represents text 614, d-block 926 represents photo 615, d-block 927 represents graphics 616 by and d-block 928 represents sidebar 617.

The rights portion 704 of a descriptor block is further illustrated in Figure 10. Figure 10 illustrates a structure which is repeated in the rights portion 704 for each right. Referring to Figure 10, each right will have a right code field 1050 and status information field 1052. The right code field 1050 will contain a unique code assigned to a right. The status information field 1052 will contain information relating to the state of a right and the digital work. Such information is indicated below in Table 1. The rights as stored in the rights portion 704 may typically be in numerical order based on the right code.

TABLE 1

DIGITAL WORK STATE INFORMATION		
Property	**Value**	**Use**
Copies-in-Use	Number	A counter of the number of copies of a work that are in use. Incremented when another copy is used; decremented when use is completed.
Loan-Period	Time-Units	Indicator of the maximum number of time-units that a document can be loaned out
Loaner-Copy	Boolean	Indicator that the current work is a loaned out copy of an authorized digital work.
Remaining-Time	Time-Units	Indicator of the remaining time of use on a metered document right.
Document-Descr	String	A string containing various identifying information about a document. The exact format of this is not specified, but it can include information such as a publisher name, author name, ISBN number, and so on.
Revenue-Owner	RO-Descr	A handle identifying a revenue owner for a digital work. This is used for reporting usage fees.
Publication-Date	Date-Descr	The date that the digital work was published.
History-list	History-Rec	A list of events recording the repostories and dates for operations that copy, transfer, backup, or restore a digital work.

The approach for representing digtal works by separating description data from content assumes that parts of a file are contiguous but takes no position on the actual representation of content. In particular, it is neutral to the question of whether content representation may take an object oriented approach. It would be natural to represent content as objects. In principle, it may be convenient to have content objects that include the billing structure and rights information that is represented in the d-blocks. Such variations in the design of the representation are possible and are viable altematives but may introduce processing overhead, e.g. the interpretation of the objects.

Digital works are stored in a repository as part of a hierarchical file system. Folders (also termed directories and sub-directories) contain the digital works as well as other folders. Digital works and folders in a folder are ordered in alphabetical order. The digital works are typed to reflect how the files are used. Usage rights can be attached to folders so that the folder itself is treated as a digital work. Access to the folder would then be handled in the same fashion as any other digital work As will be described in more detail below, the contents of the folder are subject to their own rights. Moreover, file management rights may be attached to the folder which define how folder contents can be managed.

ATTACHING USAGE RIGHTS TO A DIGITAL WORK

It is fundamental to the present invention that the usage rights are treated as part of the digital work. As the digital work is distributed, the scope of the granted usage rights will remain the same or may be narrowed. For example, when a digital work is transferred from a document server to a repository, the usage rights may include the right to loan a copy for a predetermined period of time (called the original rights). When the repository loans out a copy of the digital work, the usage rights in the loaner copy (called the next set of rights) could be set to prohibit any further rights to loan out the copy. The basic idea is that one cannot grant more rights than they have.

The attachment of usage rights into a digital work may occur in a variety of ways. If the usage rights will be the same for an entire digital work, they could be attached when the digital work is processed for deposit in the digital work server. In the case of a digital work having different usage rights for the various components, this can be done as the digital work is being created. An authoring tool or digital work assembling tool could be utilized which provides for an automated process of attaching the usage rights.

As will be described below, when a digital work is copied, transferred or loaned, a "next set of rights" can be specified. The"next set of rights" will be attached to the digital work as it is transported.

Resolving Conflicting Rights

Because each part of a digital work may have its own usage rights, there will be instances where the rights of a "contained part" are different from its parent or container part. As a result, conflict rules must be established to dictate when and how a right may be exercised. The hierarchical structure of a digital work facilitates the enforcement of such rules. A "strict" rule would be as follows: a right for a part in a digital work is sanctioned if and only if it is sanctioned for the part, for ancestor d-blocks containing the part and for all descendent d-blocks. By sanctioned, it is meant that (1) each of the respective parts must have the right, and (2) any conditions for exercising the right are satisfied.

It also possible to implement the present invention using a more lenient rule. In the more lenient rule, access to the part may be enabled to the descendent parts which have the right, but access is denied to the descendents which do not.

An example of applying both the strict rule and lenient is illustrated with reference to Figure 11. Referring to Figure 11, a root d-block 1101 has child d-blocks 1102-1105. In this case, root d-block represents a magazine, and each of the child d-blocks 1102-1105 represent articles in the magazine. Suppose that a request is made to PRINT the digital work represented by root d-block 1101 wherein the strict rule is followed. The rights for the root d-block 1101 and child d-blocks 1102-1105 are then examined. Root d-block 1101 and child d-blocks 1102 and 1105 have been granted PRINT rights. Child d-block 1103 has not been granted PRINT rights and child d-block 1104 has PRINT rights conditioned on payment of a usage fee.

Under the strict rule the PRINT right cannot be exercised because the child d-block does not have the PRINT right. Under the lenient rule, the result would be different. The digital works represented by child d-blocks 1102 and 1105 could be printed and the digital work represented by d-block 1104 could be printed so long as the usage fee is paid. Only the digital work represented by d-block 1103 could not be printed. This same result would be accomplished under the strict rule if the requests were directed to each of the individual digital works.

The present invention supports various combinations of allowing and disallowing access. Moreover, as will be described below, the usage rights grammar permits the owner of a digital work to specify if constraints may be imposed on the work by a container part. The manner in which digital works may be sanctioned because of usage rights conflicts would be implementation specific and would depend on the nature of the digital works.

REPOSITORIES

In the description of Figure 2, it was indicated that repositories come in various forms. All repositories provide a core set of services for the transmission of digital works. The manner in which digital works are exchanged is the basis for all transaction between repositories. The various repository types differ in the ultimate functions that they perform. Repositories may be devices themselves, or they may be incorporated into other systems. An example is the rendering repository 203 of Figure 2.

A repository will have associated with it a repository identifier. Typically, the repository identifier would be a unique number assigned to the repository at the time of manufacture. Each repository will also be classified as being in a particular security class. Certain communications and transactions may be conditioned on a repository being in a particular security class. The various security classes are described in greater detail below.

As a prerequisite to operation, a repository will require possession of an identification certificate. Identification certificates are encrypted to prevent forgery and are issued by a Master repository. A master repository plays the role of an authorization agent to enable repositories to receive digital works. Identification certificates must be updated on a periodic basis. Identification certificates are described in greater detail below with respect to the registration transaction.

A repository has both a hardware and functional embodiment. The functional embodiment is typically software executing on the hardware embodiment. Alternatively, the functional embodiment may be embedded in the hardware embodiment such as an Application Specific Integrated Circuit (ASIC) chip.

The hardware embodiment of a repository will be enclosed in a secure housing which if compromised, may cause the repository to be disabled. The basic components of the hardware embodiment of a repository are described with reference to Figure 12. Referring to Figure 12, a repository is comprised of a processing means 1200, storage system 1207, clock 1205 and external interface 1206. The processing means 1200 is comprised of a processor element 1201 and processor memory 1202. The processing means 1201 provides controller, repository transaction and usage rights transaction functions for the repository. Various functions in the operation of the repository such as decryption and/or decompression of digital works and transaction messages are also performed by the processing means 1200. The processor element 1201 may be a microprocessor or other suitable computing component. The processor memory 1202 would typically be further comprised of Read Only Memories (ROM) and Random Access Memories (RAM). Such memories would contain the software instructions utilized by the processor element 1201 in performing the functions of the repository.

The storage system 1207 is further comprised of descriptor storage 1203 and content storage 1204. The description tree storage 1203 will store the description tree for the digital work and the content storage will store the associated content. The description tree storage 1203 and content storage 1204 need not be of the same type of storage medium, nor are they necessarily on the same physical device. So for example, the descriptor storage 1203 may be stored on a solid state storage (for rapid retrieval of the description tree information), while the content storage 1204 may be on a high capacity storage such as an optical disk.

The clock 1205 is used to time-stamp various time based conditions for usage rights or for metering usage fees which may be associated with the digital works. The clock 1205 will have an uninterruptable power supply, e.g. a battery, in order to maintain the integrity of the time-stamps. The external interface means 1206 provides for the signal connection to other repositories and to a credit server. The external interface means 1206 provides for the exchange of signals via such standard interfaces such as RS-232 or Personal Computer Manufacturers Card Industry Association (PCMCIA) standards, or FDDI. The external interface means 1206 may also provide network connectivity.

The functional embodiment of a repository is described with reference to Figure 13. Referring to Figure 13, the functional embodiment is comprised of an operating system 1301, core repository services 1302, usage transaction handlers 1303, repository specific functions, 1304 and a user interface 1305. The operating system 1301 is specific to the repository and would typically depend on the type of processor being used. The operating system 1301 would also provide the basic services for controlling and interfacing between the basic components of the repository.

The core repository services 1302 comprise a set of functions required by each and every repository. The core repository services 1302 include the session initiation transactions which are defined in greater detail below. This set of services also includes a generic ticket agent which is used to "punch" a digital ticket and a generic authorization server for processing authorization specifications. Digital tickets and authorizations are specific mechanisms for controlling the distribution and use of digital works and are described in more detail below. Note that coupled to the core repository services are a plurality of identification certificates 1306. The identification certificates 1306 are required to enable the use of the repository.

The usage transactions handlers 1303 comprise functionality for processing access requests to digital works and for billing fees based on access. The usage transactions supported will be different for each repository type. For example, it may not be necessary for some repositories to handle access requests for digital works.

The repository specific functionality 1304 comprises functionality that is unique to a repository. For example, the master repository has special functionality for issuing digital certificates and maintaining encryption keys. The repository specific functionality 1304 would include the user interface implementation for the repository.

Repository Security Classes

For some digital works the losses caused by any individual instance of unauthorized copying is insignificant and the chief economic concern lies in assuring the convenience of access and low-overhead billing. In such cases, simple and inexpensive handheld repositories and network-based workstations may be suitable repositories, even though the measures and guarantees of security are modest.

At the other extreme, some digital works such as a digital copy of a first run movie or a bearer bond or stock certificate would be of very high value so that it is prudent to employ caution and fairly elaborate security measures to ensure that they are not copied or forged. A repository suitable for holding such a digital work could have elaborate measures for ensuring physical integrity and for verifying authorization before use.

By arranging a universal protocol, all kinds of repositories can communicate with each other in principle. However, creators of some works will want to specify that their works will only be transferred to repositories whose level of security is high enough. For this reason, document repositories have a ranking system for classes and levels of security. The security classes in the currently preferred embodiment are described in Table 2.

TABLE 2

REPOSITORY SECURITY LEVELS	
Level	Description of Security
0	Open system. Document transmission is unencrypted. No digital certificate is required for identification. The security of the system depends mostly on user honesty, since only modest knowledge may be needed to circumvent the security measures. The repository has no provisions for preventing unauthorized programs from running and accessing or copying files. The system does not prevent the use of removable storage and does not encrypt stored files.
1	Minimal security. Like the previous class except that stored files are minimally encrypted, including ones on removable storage.

TABLE 2 (continued)

REPOSITORY SECURITY LEVELS	
Level	Description of Security
2	Basic security. Like the previous class except that special tools and knowledge are required to compromise the programming, the contents of the repository, or the state of the clock. All digital communications are encrypted. A digital certificate is provided as identification. Medium level encryption is used. Repository identification number is unforgeable.
3	General security. Like the previous class plus the requirement of special tools are needed to compromise the physical integrity of the repository and that modest encryption is used on all transmissions. Password protection is required to use the local user interface. The digital clock system cannot be reset without authorization. No works would be stored on removable storage. When executing works as programs, it runs them in their own address space and does not give them direct access to any file storage or other memory containing system code or works. They can access works only through the transmission transaction protocol.
4	Like the previous class except that high level encryption is used on all communications. Sensors are used to record attempts at physical and electronic tampering. After such tampering, the repository will not perform other transactions until it has reported such tampering to a designated server.
5	Like the previous class except that if the physical or digital attempts at tampering exceed some preset thresholds that threaten the physical integrity of the repository or the integrity of digital and cryptographic barriers, then the repository will save only document description records of history but will erase or destroy any digital identifiers that could be misused if released to an unscrupulous party. It also modifies any certificates of authenticity to indicate that the physical system has been compromised. It also erases the contents of designated documents.
6	Like the previous class except that the repository will attempt wireless communication to report tampering and will employ noisy alarms.
10	This would correspond to a very high level of security. This server would maintain constant communications to remote security systems reporting transactions, sensor readings, and attempts to circumvent security.

The characterization of security levels described in Table 2 is not intended to be fixed. More important is the idea of having different security levels for different repositories. It is anticipated that new security classes and requirements will evolve according to social situations and changes in technology.

Repository User Interface

A user interface is broadly defined as the mechanism by which a user interacts with a repository in order to invoke transactions to gain access to a digital work, or exercise usage rights. As described above, a repository may be embodied in various forms. The user interface for a repository will differ depending on the particular embodiment. The user interface may be a graphical user interface having icons representing the digital works and the various transactions that may be performed. The user interface may be a generated dialog in which a user is prompted for information.

The user interface itself need not be part of the repository. As a repository may be embedded in some other device, the user interface may merely be a part of the device in which the repository is embedded. For example, the repository could be embedded in a "card" that is inserted into an available slot in a computer system. The user interface may be a combination of a display, keyboard, cursor control device and software executing on the computer system.

At a minimum, the user interface must permit a user to input information such as access requests and alpha numeric data and provide feedback as to transaction status. The user interface will then cause the repository to initiate the suitable transactions to service the request. Other facets of a particular user interface will depend on the functionality that a repository will provide.

CREDIT SERVERS

In the present invention, fees may be associated with the exercise of a right. The requirement for payment of fees is described with each version of a usage right in the usage rights language. The recording and reporting of such fees is performed by the credit server. One of the capabilities enabled by associating fees with rights is the possibility of

supporting a wide range of charging models. The simplest model, used by conventional software, is that there is a single fee at the time of purchase, after which the purchaser obtains unlimited rights to use the work as often and for as long as he or she wants. Alternative models, include metered use and variable fees. A single work can have different fees for different uses. For example, viewing a photograph on a display could have different fees than making a hardcopy or including it in a newly created work. A key to these alternative charging models is to have a low overhead means of establishing fees and accounting for credit on these transactions.

A credit server is a computational system that reliably authorizes and records these transactions so that fees are billed and paid. The credit server reports fees to a billing clearinghouse. The billing clearinghouse manages the financial transactions as they occur. As a result, bills may be generated and accounts reconciled. Preferably, the credit server would store the fee transactions and periodically communicate via a network with the billing clearinghouse for reconciliation. In such an embodiment, communications with the billing clearinghouse would be encrypted for integrity and security reasons. In another embodiment, the credit server acts as a "debit card" where transactions occur in "real-time" against a user account.

A credit server is comprised of memory, a processing means, a clock, and interface means for coupling to a repository and a financial institution (e.g. a modem). The credit server will also need to have security and authentication functionality. These elements are essentially the same elements as those of a repository. Thus, a single device can be both a repository and a credit server, provided that it has the appropriate processing elements for carrying out the corresponding functions and protocols. Typically, however, a credit server would be a card-sized system in the possession of the owner of the credit. The credit server is coupled to a repository and would interact via financial transactions as described below. Interactions with a financial institution may occur via protocols established by the financial institutions themselves.

In the currently preferred embodiment credit servers associated with both the server and the repository report the financial transaction to the billing clearinghouse. For example, when a digital work is copied by one repository to another for a fee, credit servers coupled to each of the repositories will report the transaction to the billing clearinghouse. This is desirable in that it insures that a transaction will be accounted for in the event of some break in the communication between a credit server and the billing clearinghouse. However, some implementations may embody only a single credit server reporting the transaction to minimize transaction processing at the risk of losing some transactions.

USAGE RIGHTS LANGUAGE

The present invention uses statements in a high level "usage rights language" to define rights associated with digital works and their parts. Usage rights statements are interpreted by repositories and are used to determine what transactions can be successfully carried out for a digital work and also to determine parameters for those transactions. For example, sentences in the language determine whether a given digital work can be copied, when and how it can be used, and what fees (if any) are to be charged for that use. Once the usage rights statements are generated, they are encoded in a suitable form for accessing during the processing of transactions.

Defining usage rights in terms of a language in combination with the hierarchical representation of a digital work enables the support of a wide variety of distribution and fee schemes. An example is the ability to attach multiple versions of a right to a work. So a creator may attach a PRINT right to make 5 copies for $10.00 and a PRINT right to make unlimited copies for $100.00. A purchaser may then choose which option best fits his needs. Another example is that rights and fees are additive. So in the case of a composite work, the rights and fees of each of the components works is used in determining the rights and fees for the work as a whole.

The basic contents of a right are illustrated in Figure 14. Referring to Figure 14, a right 1450 has a transactional component 1451 and a specifications component 1452. A right 1450 has a label (e.g. COPY or PRINT) which indicates the use or distribution privileges that are embodied by the right. The transactional component 1451 corresponds to a particular way in which a digital work may be used or distributed. The transactional component 1451 is typically embodied in software instructions in a repository which implement the use or distribution privileges for the right. The specifications components 1452 are used to specify conditions which must be satisfied prior to the right being exercised or to designate various transaction related parameters. In the currently preferred embodiment, these specifications include copy count 1453, Fees and Incentives 1454, Time 1455, Access and Security 1456 and Control 1457. Each of these specifications will be described in greater detail below with respect to the language grammar elements.

The usage rights language is based on the grammar described below. A grammar is a convenient means for defining valid sequence of symbols for a language. In describing the grammar the notation "[a | b | c] is used to indicate distinct choices among alternatives. In this example, a sentence can have either an "a", "b" or "c". It must include exactly one of them. The braces { } are used to indicate optional items. Note that brackets, bars and braces are used to describe the language of usage rights sentences but do not appear in actual sentences in the language.

In contrast, parentheses are part of the usage rights language. Parentheses are used to group items together in lists. The notation (x*) is used to indicate a variable length list, that is, a list containing one or more items of type x.

The notation (x)* is used to indicate a variable number of lists containing x.

Keywords in the grammar are words followed by colons. Keywords are a common and very special case in the language. They are often used to indicate a single value, typically an identifier. In many cases, the keyword and the parameter are entirely optional. When a keyword is given, it often takes a single identifier as its value. In some cases, the keyword takes a list of identifiers.

In the usage rights language, time is specified in an hours:minutes:seconds (or hh:mm:ss) representation. Time zone indicators, e.g. PDT for Pacific Daylight Time, may also be specified. Dates are represented as year/ month/day (or YYYY/MMM/DD). Note that these time and date representations may specify moments in time or units of time Money units are specified in terms of dollars.

Finally, in the usage rights language, various "things" will need to interact with each other. For example, an instance of a usage right may specify a bank account, a digital ticket, etc.. Such things need to be identified and are specified herein using the suffix "-ID."

The Usage Rights Grammar is listed in its entirety in Figure 15 and is described below.

Grammar element 1501 **"Digital Work Rights: = (Rights*)"** define the digital work rights as a set of rights. The set of rights attached to a digital work define how that digital work may be transferred, used, performed or played. A set of rights will attach to the entire digital work and in the case of compound digital works, each of the components of the digital work. The usage rights of components of a digital may be different.

Grammar element 1502 **"Right : = (Right-Code {Copy-Count} {Control-Spec} {Time-Spec} {Access-Spec} {Fee-Spec})"** enumerates the content of a right. Each usage right must specify a right code. Each right may also optionally specify conditions which must be satisfied before the right can be exercised. These conditions are copy count, control, time, access and fee conditions. In the currently preferred embodiment, for the optional elements, the following defaults apply: copy count equals 1, no time limit on the use of the right, no access tests or a security level required to use the right and no fee is required. These conditions will each be described in greater detail below.

It is important to note that a digital work may have multiple versions of a right, each having the same right code. The multiple version would provide alternative conditions and fees for accessing the digital work.

Grammar element 1503 **"Right-Code : = Render-Code I Transport-Code I File-Management-CodeI Derivative-Works- Code Configuration-Code"** distinguishes each of the specific rights into a particular right type (although each right is identified by distinct right codes). In this way, the grammar provides a catalog of possible rights that can be associated with parts of digital works. In the following, rights are divided into categories for convenience in describing them.

Grammar element 1504 **"Render-Code : = [Play: {Player: Player-ID} I Print: {Printer: Printer-ID}]"** lists a category of rights all involving the making of ephemeral, transitory, or non-digital copies of the digital work. After use the copies are erased.

- Play — A process of rendering or performing a digital work on some processor. This includes such things as playing digital movies, playing digital music, playing a video game, running a computer program, or displaying a document on a display.
- Print — To render the work in a medium that is not further protected by usage rights, such as printing on paper.

Grammar element 1505 **"Transport-Code : = [Copy I Transfer I Loan {Remaining-Rights: Next-Set-of-Rights}] {(Next-Copy-Rights: Next-Set of Rights)}"** lists a category of rights involving the making of persistent, usable copies of the digital work on other repositories. The optional Next-Copy-Rights determine the rights on the work after it is transported. If this is not specified, then the rights on the transported copy are the same as on the original. The optional Remaining-Rights specify the rights that remain with a digital work when it is loaned out. If this is not specified, then the default is that no rights can be exercised when it is loaned out.

- Copy — Make a new copy of a work
- Transfer — Moving a work from one repository to another.
- Loan — Temporarily loaning a copy to another repository for a specified period of time.

Grammar element 1506 **"File-Management-Code : = Backup {Back-Up-Copy-Rights: Next-Set -of Rights}I Restore I Delete I Folder I Directory {Name:Hide-Local I Hide - Remote}{Parts:Hide-Local I Hide-Remote}"** lists a category of rights involving operations for file management, such as the making of backup copies to protect the copy owner against catastrophic equipment failure.

Many software licenses and also copyright law give a copy owner the right to make backup copies to protect against catastrophic failure of equipment. However, the making of uncontrolled backup copies is inherently at odds with the ability to control usage, since an uncontrolled backup copy can be kept and then restored even after the authorized copy was sold.

The File management rights enable the making and restoring of backup copies in a way that respects usage rights, honoring the requirements of both the copy owner and the rights grantor and revenue owner. Backup copies of work descriptions (including usage rights and fee data) can be sent under appropriate protocol and usage rights control to other document repositories of sufficiently high security. Further rights permit organization of digital works into folders which themselves are treated as digital works and whose contents may be "hidden" from a party seeking to determine the contents of a repository.

- Backup To make a backup copy of a digital work as protection against media failure.
- Restore To restore a backup copy of a digital work.
- Delete To delete or erase a copy of a digital work.
- Folder To create and name folders, and to move files and folders between folders.
- Directory To hide a folder or its contents.

Grammar element 1507 **"Derivative-Works-Code: [Extract | Embed | Edit {Process: Process-ID}] {Next-Copy-Rights : Next-Set-of Rights}"** lists a category of rights involving the use of a digital work to create new works.

- Extract To remove a portion of a work, for the purposes of creating a new work.
- Embed To include a work in an existing work.
- Edit To alter a digital work by copying, selecting and modifying portions of an existing digital work.

Grammar element 1508 **"Configuration-Code : = Install | Uninstall"** lists a category of rights for installing and uninstalling software on a repository (typically a rendering repository.) This would typically occur for the installation of a new type of player within the rendering repository.

- Install: To install new software on a repository.
- Uninstall: To remove existing software from a repository.

Grammar element 1509 **"Next-Set-of-Rights : = {(Add : Set-Of-Rights)} {(Delete: Set-Of-Rights)} {(Replace: Set-Of-Rights)} {(Keep: Set-Of-Rights)}"** defines how rights are carried forward for a copy of a digital work. If the Next-Copy-Rights is not specified, the rights for the next copy are the same as those of the current copy. Otherwise, the set of rights for the next copy can be specified. Versions of rights after Add: are added to the current set of rights. Rights after Delete: are deleted from the current set of rights. If only right codes are listed after Delete:, then all versions of rights with those codes are deleted. Versions of rights after Replace: subsume all versions of rights of the same type in the current set of rights.

If Remaining-Rights is not specified, then there are no rights for the original after all Loan copies are loaned out. If Remaining-Rights is specified, then the Keep: token can be used to simplify the expression of what rights to keep behind. A list of right codes following keep means that all of the versions of those listed rights are kept in the remaining copy. This specification can be overridden by subsequent Delete: or Replace: specifications.

Copy Count Specification

For various transactions, it may be desirable to provide some limit as to the number of "copies" of the work which may be exercised simultaneously for the right. For example, it may be desirable to limit the number of copies of a digital work that may be loaned out at a time or viewed at a time.

Grammar element 1510 **"Copy-Count : = (Copies: positive-integer | 0 | unlimited)"** provides a condition which defines the number of "copies" of a work subject to the right . A copy count can be 0, a fixed number, or unlimited. The copy-count is associated with each right, as opposed to there being just a single copy-count for the digital work. The Copy-Count for a right is decremented each time that a right is exercised. When the Copy-Count equals zero, the right can no longer be exercised. If the Copy-Count is not specified, the default is one.

Control Specification

Rights and fees depend in general on rights granted by the creator as well as further restrictions imposed by later distributors. Control specifications deal with interactions between the creators and their distributors governing the imposition of further restrictions and fees. For example, a distributor of a digital work may not want an end consumer of a digital work to add fees or otherwise profit by commercially exploiting the purchased digital work.

Grammar element 1511 **"Control-Spec : = (Control: {Restrictable | Unrestrictable} {Unchargeable | Chargeable})"** provides a condition to specify the effect of usage rights and fees of parents on the exercise of the right. A

digital work is restrictable if higher level d-blocks can impose further restrictions (time specifications and access specifications) on the right. It is unrestrictable if no further restrictions can be imposed. The default setting is restrictable. A right is unchargeable if no more fees can be imposed on the use of the right. It is chargeable if more fees can be imposed. The default is chargeable.

Time Specification

It is often desirable to assign a start date or specify some duration as to when a right may be exercised. Grammar element 1512 **"Time-Spec : = ({Fixed-Interval | Sliding-Interval | Meter-Time} Until: Expiration-Date)"** provides for specification of time conditions on the exercise of a right. Rights may be granted for a specified time. Different kinds of time specifications are appropriate for different kinds of rights. Some rights may be exercised during a fixed and predetermined duration. Some rights may be exercised for an interval that starts the first time that the right is invoked by some transaction. Some rights may be exercised or are charged according to some kind of metered time, which may be split into separate intervals. For example, a right to view a picture for an hour might be split into six ten minute viewings or four fifteen minute viewings or twenty three minute viewings.

The terms "time" and "date" are used synonymously to refer to a moment in time. There are several kinds of time specifications. Each specification represents some limitation on the times over which the usage right applies. The Expiration-Date specifies the moment at which the usage right ends. For example, if the Expiration-Date is "Jan 1, 1995," then the right ends at the first moment of 1995. If the Expiration-Date is specified as "forever", then the rights are interpreted as continuing without end. If only an expiration date is given, then the right can be exercised as often as desired until the expiration date.

Grammar element 1513 **"Fixed-Interval := From: Start-Time"** is used to define a predetermined interval that runs from the start time to the expiration date.

Grammar element 1514 **"Sliding-Interval : = Interval: Use-Duration"** is used to define an indeterminate (or "open") start time. It sets limits on a continuous period of time over which the contents are accessible. The period starts on the first access and ends after the duration has passed or the expiration date is reached, whichever comes first. For example, if the right gives 10 hours of continuous access, the use-duration would begin when the first access was made and end 10 hours later.

Grammar element 1515 **"Meter-Time: = Time-Remaining: Remaining-Use"** is used to define a "meter time," that is, a measure of the time that the right is actually exercised. It differs from the Sliding-Interval specification in that the time that the digital work is in use need not be continuous. For example, if the rights guarantee three days of access, those days could be spread out over a month. With this specification, the rights can be exercised until the meter time is exhausted or the expiration date is reached, whichever comes first.

Remaining-Use: = Time-Unit
Start-Time: = Time-Unit
Use-Duration: = Time-Unit
All of the time specifications include time-unit specifications in their ultimate instantiation.

Security Class and Authorization Specification

The present invention provides for various security mechanisms to be introduced into a distribution or use scheme. Grammar element 1516 **"Access-Spec : = ({SC: Security-Class} {Authorization: Authorization-ID*} {Other-Authorization: Authorization-ID*} {Ticket: Ticket-ID})"** provides a means for restricting access and transmission. Access specifications can specify a required security class for a repository to exercise a right or a required authorization test that must be satisfied.

The keyword **"SC:"** is used to specify a minimum security level for the repositories involved in the access. If **"SC:"** is not specified, the lowest security level is acceptable.

The optional **"Authorization:"** keyword is used to specify required authorizations on the same repository as the work. The optional **"Other-Authorization:"** keyword is used to specify required authorizations on the other repository in the transaction.

The optional **"Ticket:"** keyword specifies the identity of a ticket required for the transaction. A transaction involving digital tickets must locate an appropriate digital ticket agent who can "punch" or otherwise validate the ticket before the transaction can proceed. Tickets are described in greater detail below.

In a transaction involving a repository and a document server, some usage rights may require that the repository have a particular authorization, that the server have some authorization, or that both repositories have (possibly different) authorizations. Authorizations themselves are digital works (hereinafter referred to as an authorization object) that can be moved between repositories in the same manner as other digital works. Their copying and transferring is

subject to the same rights and fees as other digital works. A repository is said to have an authorization if that authorization object is contained within the repository.

In some cases, an authorization may be required from a source other than the document server and repository. An authorization object referenced by an Authorization-ID can contain digital address information to be used to set up a communications link between a repository and the authorization source. These are analogous to phone numbers. For such access tests, the communication would need to be established and authorization obtained before the right could be exercised.

For one-time usage rights, a variant on this scheme is to have a digital ticket. A ticket is presented to a digital ticket agent, whose type is specified on the ticket. In the simplest case, a certified generic ticket agent, available on all repositories, is available to "punch" the ticket. In other cases, the ticket may contain addressing information for locating a "special" ticket agent. Once a ticket has been punched, it cannot be used again for the same kind of transaction (unless it is unpunched or refreshed in the manner described below.) Punching includes marking the ticket with a timestamp of the date and time it was used. Tickets are digital works and can be copied or transferred between repositories according to their usage rights.

In the currently preferred embodiment, a "punched" ticket becomes "unpunched" or "refreshed" when it is copied or extracted. The Copy and Extract operations save the date and time as a property of the digital ticket. When a ticket agent is given a ticket, it can simply check whether the digital copy was made after the last time that it was punched. Of course, the digital ticket must have the copy or extract usage rights attached thereto.

The capability to unpunch a ticket is inportant in the following cases:

- A digital work is circulated at low cost with a limitation that it can be used only once.
- A digital work is circulated with a ticket that can be used once to give discounts on purchases of other works.
- A digital work is circulated with a ticket (included in the purchase price and possibly embedded in the work) that can be used for a future upgrade.

In each of these cases, if a paid copy is made of the digital work (including the ticket) the new owner would expect to get a fresh (unpunched) ticket, whether the copy seller has used the work or not. In contrast, loaning a work or simply transferring it to another repository should not revitalize the ticket.

Usage Fees and Incentives Specification

The billing for use of a digital work is fundamental to a commercial distribution system. Grammar Element 1517 **"Fee-Spec: = {Scheduled-Discount} Regular-Fee-Spec | Scheduled-Fee-Spec | Markup-Spec"** provides a range of options for billing for the use of digital works.

A key feature of this approach is the development of low-overhead billing for transactions in potentially small amounts. Thus, it becomes feasible to collect fees of only a few cents each for thousands of transactions.

The grammar differentiates between uses where the charge is per use from those where it is metered by the time unit. Transactions can support fees that the user pays for using a digital work as well as incentives paid by the right grantor to users to induce them to use or distribute the digital work.

The optional scheduled discount refers to the rest of the fee specification--discounting it by a percentage over time. If it is not specified, then there is no scheduled discount. Regular fee specifications are constant over time. Scheduled fee specifications give a schedule of dates over which the fee specifications change. Markup specifications are used in d-blocks for adding a percentage to the fees already being charged.

Grammar Element 1518 **"Scheduled-Discount: = (Scheduled-Discount: (Time-Spec Percentage)*)"** A Scheduled-Discount is a essentially a scheduled modifier of any other fee specification for this version of the right of the digital work. (It does not refer to children or parent digital works or to other versions of rights.). It is a list of pairs of times and percentages. The most recent time in the list that has not yet passed at the time of the transaction is the one in effect. The percentage gives the discount percentage. For example, the number 10 refers to a 10% discount.

Grammar Element 1519 **"Regular-Fee-Spec : = ({Fee: | Incentive: } [Per-Use-Spec | Metered-Rate-Spec | Best-Price-Spec | Call-For-Price-Spec] {Min: Money-Unit Per: Time-Spec)(Max: Money-Unit Per: Time-Spec} To: Account-ID)"** provides for several kinds of fee specifications.

Fees are paid by the copy-owner/user to the revenue-owner if Fee: is specified. Incentives are paid by the revenue-owner to the user if Incentive: is specified. If the Min: specification is given, then there is a minimum fee to be charged per time-spec unit for its use. If the Max: specification is given, then there is a maximum fee to be charged per time-spec for its use. When Fee: is specified, Account-ID identifies the account to which the fee is to be paid. When Incentive: is specified, Account-ID identifies the account from which the fee is to be paid.

Grammar element 1520 **"Per-Use-Spec: = Per-Use: Money-unit"** defines a simple fee to be paid every time the right is exercised, regardless of how much time the transaction takes.

Grammar element 1521 **"Metered-Rate-Spec : = Metered: Money-Unit Per: Time-Spec"** defines a metered-rate fee paid according to how long the right is exercised. Thus, the time it takes to complete the transaction determines the fee.

Grammar element 1522 **"Best-Price-Spec := Best-Price: Money-unit Max: Money-unit"** is used to specify a best-price that is determined when the account is settled. This specification is to accommodate special deals, rebates, and pricing that depends on information that is not available to the repository. All fee specifications can be combined with tickets or authorizations that could indicate that the consumer is a wholesaler or that he is a preferred customer, or that the seller be authorized in some way. The amount of money in the Max: field is the maximum amount that the use will cost. This is the amount that is tentatively debited from the credit server. However, when the transaction is ultimately reconciled, any excess amount will be returned to the consumer in a separate transaction.

Grammar element 1523 **"Call-For-Price-Spec:= Call-For-Price"** is similar to a **"Best-Price-Spec"** in that it is intended to accommodate cases where prices are dynamic. **A Call-For-Price Spec** requires a communication with a dealer to determine the price. This option cannot be exercised if the repository cannot communicate with a dealer at the time that the right is exercised. It is based on a secure transaction whereby the dealer names a price to exercise the right and passes along a deal certificate which is referenced or included in the billing process.

Grammar element 1524 **"Scheduled-Fee-Spec: = (Schedule: (Time-Spec Regular-Fee-Spec)*)"** is used to provide a schedule of dates over which the fee specifications change. The fee specification with the most recent date not in the future is the one that is in effect. This is similar to but more general than the scheduled discount. It is more general, because it provides a means to vary the fee agreement for each time period.

Grammar element 1525 **"Markup-Spec: = Markup: percentage To: Account-ID"** is provided for adding a percentage to the fees already being charged. For example, a 5% markup means that a fee of 5% of cumulative fee so far will be allocated to the distributor. A markup specification can be applied to all of the other kinds of fee specifications. It is typically used in a shell provided by a distributor. It refers to fees associated with d-blocks that are parts of the current d-block. This might be a convenient specification for use in taxes, or in distributor overhead.

REPOSITORY TRANSACTIONS

When a user requests access to a digital work, the repository will initiate various transactions. The combination of transactions invoked will depend on the specifications assigned for a usage right. There are three basic types of transactions, Session Initiation Transactions, Financial Transactions and Usage Transactions. Generally, session initiation transactions are initiated first to establish a valid session. When a valid session is established, transactions corresponding to the various usage rights are invoked. Finally, request specific transactions are performed.

Transactions occur between two repositories (one acting as a server), between a repository and a document playback platform (e.g. for executing or viewing), between a repository and a credit server or between a repository and an authorization server. When transactions occur between more than one repository, it is assumed that there is a reliable communication channel between the repositories. For example, this could be a TCP/IP channel or any other commercially available channel that has built-in capabilities for detecting and correcting transmission errors. However, it is not assumed that the communication channel is secure. Provisions for security and privacy are part of the requirements for specifying and implementing repositories and thus form the need for various transactions.

Message Transmission

Transactions require that there be some communication between repositories. Communication between repositories occurs in units termed as messages. Because the communication line is assumed to be unsecure, all communications with repositories that are above the lowest security class are encrypted utilizing a public key encryption technique. Public key encryption is a well known technique in the encryption arts. The term key refers to a numeric code that is used with encryption and decryption algorithms. Keys come in pairs, where "writing keys" are used to encrypt data and "checking keys" are used to decrypt data. Both writing and checking keys may be public or private. Public keys are those that are distributed to others. Private keys are maintained in confidence.

Key management and security is instrumental in the success of a public key encryption system. In the currently preferred embodiment, one or more master repositories maintain the keys and create the identification certificates used by the repositories.

When a sending repository transmits a message to a receiving repository, the sending repository encrypts all of its data using the public writing key of the receiving repository. The sending repository includes its name, the name of the receiving repository, a session identifier such as a nonce (described below), and a message counter in each message.

In this way, the communication can only be read (to a high probability) by the receiving repository, which holds the private checking key for decryption. The auxiliary data is used to guard against various replay attacks to security. If

messages ever arrive with the wrong counter or an old nonce, the repositories can assume that someone is interfering with communication and the transaction terminated.

The respective public keys for the repositories to be used for encryption are obtained in the registration transaction described below.

Session Initiation Transactions

A usage transaction is carried out in a session between repositories. For usage transactions involving more than one repository, or for financial transactions between a repository and a credit server, a registration transaction is performed. A second transaction termed a login transaction, may also be needed to initiate the session. The goal of the registration transaction is to establish a secure channel between two repositories who know each others identities. As it is assumed that the communication channel between the repositories is reliable but not secure, there is a risk that a non-repository may mimic the protocol in order to gain illegitimate access to a repository.

The registration transaction between two repositories is described with respect to Figures 16 and 17. The steps described are from the perspective of a "repository-1" registering its identity with a "repository-2". The registration must be symmetrical so the same set of steps will be repeated for repository-2 registering its identity with repository-1. Referring to Figure 16, repository-1 first generates an encrypted registration identifier, step 1601 and then generates a registration message, step 1602. A registration message is comprised of an identifier of a master repository, the identification certificate for the repository-1 and an encrypted random registration identifier. The identification certificate is encrypted by the master repository in its private key and attests to the fact that the repository (here repository-1) is a bona fide repository. The identification certificate also contains a public key for the repository, the repository security level and a timestamp (indicating a time after which the certificate is no longer valid.) The registration identifier is a number generated by the repository for this registration. The registration identifier is unique to the session and is encrypted in repository-1's private key. The registration identifier is used to improve security of authentication by detecting certain kinds of communications based attacks. Repository-1 then transmits the registration message to repository-2, step 1603.

Upon receiving the registration message, repository-2 determines if it has the needed public key for the master repository, step 1604. If repository-2 does not have the needed public key to decrypt the identification certificate, the registration transaction terminates in an error, step 1618.

Assuming that repository-2 has the proper public key the identification certificate is decrypted, step 1605. Repository-2 saves the encrypted registration identifier, step 1606, and extracts the repository identifier, step 1607. The extracted repository identifier is checked against a "hotlist" of compromised document repositories, step 1608. In the currently preferred embodiment, each repository will contain "hotlists" of compromised repositories. If the repository is on the "hotlist', the registration transaction terminates in an error per step 1618. Repositories can be removed from the hotlist when their certificates expire, so that the list does not need to grow without bound. Also, by keeping a short list of hotlist certificates that it has previously received, a repository can avoid the work of actually going through the list. These lists would be encrypted by a master repository. A minor variation on the approach to improve efficiency would have the repositories first exchange lists of names of hotlist certificates, ultimately exchanging only those lists that they had not previously received. The "hotlists" are maintained and distributed by Master repositories.

Note that rather than terminating in error, the transaction could request that another registration message be sent based on an identification certificate created by another master repository. This may be repeated until a satisfactory identification certificate is found, or it is determined that trust cannot be established.

Assuming that the repository is not on the hotlist, the repository identification needs to be verified. In other words, repository-2 needs to validate that the repository on the other end is really repository-1. This is termed performance testing and is performed in order to avoid invalid access to the repository via a counterfeit repository replaying a recording of a prior session initiation between repository-1 and repository-2. Performance testing is initiated by repository-2 generating a performance message, step 1609. The performance message consists of a nonce, the names of the respective repositories, the time and the registration identifier received from repository-1. A nonce is a generated message based on some random and variable information (e.g. the time or the temperature.) The nonce is used to check whether repository-1 can actually exhibit correct encrypting of a message using the private keys it claims to have, on a message that it has never seen before. The performance message is encrypted using the public key specified in the registration message of repository-1. The performance message is transmitted to repository-1, step 1610, where it is decrypted by repository-1 using its private key, step 1611. Repository-1 then checks to make sure that the names of the two repositories are correct, step 1612, that the time is accurate, step 1613 and that the registration identifier corresponds to the one it sent, step 1614. If any of these tests fails, the transaction is terminated per step 1616. Assuming that the tests are passed, repository-1 transmits the nonce to repository-2 in the clear, step 1615. Repository-2 then compares the received nonce to the original nonce, step 1617. If they are not identical, the registration transaction terminates in an error per step 1618. If they are the same, the registration transaction has successfully completed.

At this point, assuming that the transaction has not terminated, the repositories exchange messages containing session keys to be used in all communications during the session and synchronize their clocks. Figure 17 illustrates the session information exchange and clock synchronization steps (again from the perspective of repository-1.) Referring to Figure 17, repository-1 creates a session key pair, step 1701. A first key is kept private and is used by repository-1 to encrypt messages. The second key is a public key used by repository-2 to decrypt messages. The second key is encrypted using the public key of repository-2, step 1702 and is sent to repository-2, step 1703. Upon receipt, repository-2 decrypts the second key, step 1704. The second key is used to decrypt messages in subsequent communications. When each repository has completed this step, they are both convinced that the other repository is bona fide and that they are communicating with the original. Each repository has given the other a key to be used in decrypting further communications during the session. Since that key is itself transmitted in the public key of the receiving repository only it will be able to decrypt the key which is used to decrypt subsequent messages.

After the session information is exchanged, the repositories must synchronize their clocks. Clock synchronization is used by the repositories to establish an agreed upon time base for the financial records of their mutual transactions. Referring back to Figure 17, repository-2 initiates clock synchronization by generating a time stamp exchange message, step 1705, and transmits it to repository-1, step 1706. Upon receipt, repository-1 generates its own time stamp message, step 1707 and transmits it back to repository-2, step 1708. Repository-2 notes the current time, step 1709 and stores the time received from repository-1, step 1710. The current time is compared to the time received from repository-1, step 1711. The difference is then checked to see if it exceeds a predetermined tolerance (e.g. one minute), step 1712. If it does, repository-2 terminates the transaction as this may indicate tampering with the repository, step 1713. If not repository-2 computes an adjusted time delta, step 1714. The adjusted time delta is the difference between the clock time of repository-2 and the average of the times from repository-1 and repository-2.

To achieve greater accuracy, repository-2 can request the time again up to a fixed number of times (e.g. five times), repeat the clock synchronization steps, and average the results.

A second session initiation transaction is a Login transaction. The Login transaction is used to check the authenticity of a user requesting a transaction. A Login transaction is particularly prudent for the authorization of financial transactions that will be charged to a credit server. The Login transaction involves an interaction between the user at a user interface and the credit server associated with a repository. The information exchanged here is a login string supplied by the repository/credit server to identify itself to the user, and a Personal Identification Number (PIN) provided by the user to identify himself to the credit server. In the event that the user is accessing a credit server on a repository different from the one on which the user interface resides, exchange of the information would be encrypted using the public and private keys of the respective repositories.

Billing Transactions

Billing Transactions are concerned with monetary transactions with a credit server. Billing Transactions are carried out when all other conditions are satisfied and a usage fee is required for granting the request. For the most part, billing transactions are well understood in the state of the art. These transactions are between a repository and a credit server, or between a credit server and a billing clearinghouse. Briefly, the required transactions include the following:

- Registration and LOGIN transactions by which the repository and user establish their bona fides to a credit server. These transactions would be entirely internal in cases where the repository and credit server are implemented as a single system.
- Registration and LOGIN transactions, by which a credit server establishes its bona fides to a billing clearinghouse.
- An Assign-fee transaction to assign a charge. The information in this transaction would include a transaction identifier, the identities of the repositories in the transaction, and a list of charges from the parts of the digital work. If there has been any unusual event in the transaction such as an interruption of communications, that information is included as well.
- A Begin-charges transaction to assign a charge. This transaction is much the same as an assign-fee transaction except that it is used for metered use. It includes the same information as the assign-fee transaction as well as the usage fee information. The credit-server is then responsible for running a clock.
- An End-charges transaction to end a charge for metered use. (In a variation on this approach, the repositories would exchange periodic charge information for each block of time.)
- A report-charges transaction between a personal credit server and a billing clearinghouse. This transaction is invoked at least once per billing period. It is used to pass along information about charges. On debit and credit cards, this transaction would also be used to update balance information and credit limits as needed.

All billing transactions are given a transaction ID and are reported to the credit severs by both the server and the client. This reduces possible loss of billing information if one of the parties to a transaction loses a banking card and

provides a check against tampering with the system.

Usage Transactions

After the session initiation transactions have been completed, the usage request may then be processed. To simplify the description of the steps carried out in processing a usage request, the term requester is used to refer to a repository in the requester mode which is initiating a request, and the term server is used to refer to a repository in the server mode and which contains the desired digital work. In many cases such as requests to print or view a work, the requester and server may be the same device and the transactions described in the following would be entirely internal. In such instances, certain transaction steps, such as the registration transaction, need not be performed.

There are some common steps that are part of the semantics of all of the usage rights transactions. These steps are referred to as the common transaction steps. There are two sets -- the "opening" steps and the "closing" steps. For simplicity, these are listed here rather than repeating them in the descriptions of all of the usage rights transactions.

Transactions can refer to a part of a digital work, a complete digital work, or a Digital work containing other digital works. Although not described in detail herein, a transaction may even refer to a folder comprised of a plurality of digital works. The term "work" is used to refer to what ever portion or set of digital works is being accessed.

Many of the steps here involve determining if certain conditions are satisfied. Recall that each usage right may have one or more conditions which must be satisfied before the right can be exercised. Digital works have parts and parts have parts. Different parts can have different rights and fees. Thus, it is necessary to verify that the requirements are met for ALL of the parts that are involved in a transaction For brevity, when reference is made to checking whether the rights exist and conditions for exercising are satisfied, it is meant that all such checking takes place for each of the relevant parts of the work.

Figure 18 illustrates the initial common opening and closing steps for a transaction. At this point it is assumed that registration has occurred and that a "trusted" session is in place. General tests are tests on usage rights associated with the folder containing the work or some containing folder higher in the file system hierarchy. These tests correspond to requirements imposed on the work as a consequence of its being on the particular repository, as opposed to being attached to the work itself. Referring to Figure 18, prior to initiating a usage transaction, the requester performs any general tests that are required before the right associated with the transaction can be exercised, step, 1801. For example, install, uninstall and delete rights may be implemented to require that a requester have an authorization certificate before the right can be exercised. Another example is the requirement that a digital ticket be present and punched before a digital work may be copied to a requester. If any of the general tests fail, the transaction is not initiated, step, 1802. Assuming that such required tests are passed, upon receiving the usage request, the server generates a transaction identifier that is used in records or reports of the transaction, step 1803. The server then checks whether the digital work has been granted the right corresponding to the requested transaction, step 1804. If the digital work has not been granted the right corresponding to the request, the transaction terminates, step 1805. If the digital work has been granted the requested right, the server then determines if the various conditions for exercising the right are satisfied. Time based conditions are examined, step 1806. These conditions are checked by examining the time specification for the the version of the right. If any of the conditions are not satisfied, the transaction terminates per step 1805.

Assuming that the time based conditions are satisfied, the server checks security and access conditions, step 1807. Such security and access conditions are satisfied if: 1) the requester is at the specified security class, or a higher security class, 2) the server satisfies any specified authorization test and 3) the requester satisfies any specified authorization tests and has any required digital tickets. If any of the conditions are not satisfied, the transaction terminates per step 1805.

Assuming that the security and access conditions are all satisfied, the server checks the copy count condition, step 1808. If the copy count equals zero, then the transaction cannot be completed and the transaction terminates per step 1805.

Assuming that the copy count does not equal zero, the server checks if the copies in use for the requested right is greater than or equal to any copy count for the requested right (or relevant parts), step 1809. If the copies in use is greater than or equal to the copy count, this indicates that usage rights for the version of the transaction have been exhausted. Accordingly, the server terminates the transaction, step 1805. If the copy count is less than the copies in use for the transaction the transaction can continue, and the copies in use would be incremented by the number of digital works requested in the transaction, step 1810.

The server then checks if the digital work has a "Loan" access right, step 1811. The "Loan" access right is a special case since remaining rights may be present even though all copies are loaned out. If the digital work has the "Loan" access right, a check is made to see if all copies have been loaned out, step 1812. The number of copies that could be loaned is the sum of the Copy-Counts for all of the versions of the loan right of the digital work. For a composite work, the relevant figure is the minimal such sum of each of the components of the composite work. If all copies have been loaned out, the remaining rights are determined, step 1813. The remaining-rights is determined from the remaining

rights specifications from the versions of the Loan right. If there is only one version of the Loan right, then the determination is simple. The remaining rights are the ones specified in that version of the Loan right, or none if Remaining-Rights: is not specified. If there are multiple versions of the Loan right and all copies of all of the versions are loaned out, then the remaining rights is taken as the minimum set (intersection) of remaining rights across all of the versions of the loan right. The server then determines if the requested right is in the set of remaining rights, step 1814. If the requested right is not in the set of remaining rights, the server terminates the transaction, step 1805.

If Loan is not a usage right for the digital work or if all copies have not been loaned out or the requested right is in the set of remaining rights, fee conditions for the right are then checked, step 1815. This will initiate various financial transactions between the repository and associated credit server. Further, any metering of usage of a digital work will commence. If any financial transaction fails, the transaction terminates per step 1805.

It should be noted that the order in which the conditions are checked need not follow the order of steps 1806-1815.

At this point, right specific steps are now performed and are represented here as step 1816. The right specific steps are described in greater detail below.

The common closing transaction steps are now performed. Each of the closing transaction steps are performed by the server after a successful completion of a transaction. Referring back to Figure 18, the copies in use value for the requested right is decremented by the number of copies involved in the transaction, step 1817. Next, if the right had a metered usage fee specification, the server subtracts the elapsed time from the Remaining-Use-Time associated with the right for every part involved in the transaction, step 1818. Finally, if there are fee specifications associated with the right, the server initiates End-Charge financial transaction to confirm billing, step 1819.

Transmission Protocol

An important area to consider is the transmission of the digital work from the server to the requester. The transmission protocol described herein refers to events occurring after a valid session has been created. The transmission protocol must handle the case of disruption in the communications between the repositories. It is assumed that interference such as injecting noise on the communication channel can be detected by the integrity checks (e.g., parity, checksum, etc.) that are built into the transport protocol and are not discussed in detail herein.

The underlying goal in the transmission protocol is to preclude certain failure modes, such as malicious or accidental interference on the communications channel. Suppose, for example, that a user pulls a card with the credit server at a specific time near the end of a transaction. There should not be a vulnerable time at which "pulling the card" causes the repositories to fail to correctly account for the number of copies of the work that have been created. Restated, there should be no time at which a party can break a connection as a means to avoid payment after using a digital work.

If a transaction is interrupted (and fails), both repositories restore the digital works and accounts to their state prior to the failure, modulo records of the failure itself.

Figure 19 is a state diagram showing steps in the process of transmitting information during a transaction. Each box represents a state of a repository in either the server mode (above the central dotted line 1901) or in the requester mode (below the dotted line 1901). Solid arrows stand for transitions between states. Dashed arrows stand for message communications between the repositories. A dashed message arrow pointing to a solid transition arrow is interpreted as meaning that the transition takes place when the message is received. Unlabeled transition arrows take place unconditionally. Other labels on state transition arrows describe conditions that trigger the transition.

Referring now to Figure 19, the server is initially in a state 1902 where a new transaction is initiated via start message 1903. This message includes transaction information including a transaction identifier and a count of the blocks of data to be transferred. The requester, initially in a wait state 1904 then enters a data wait state 1905.

The server enters a data transmit state 1906 and transmits a block of data 1907 and then enters a wait for acknowledgement state 1908. As the data is received, the requester enters a data receive state 1909 and when the data blocks are completely received it enters an acknowledgement state 1910 and transmits an Acknowledgement message 1911 to the server.

If there are more blocks to send, the server waits until receiving an Acknowledgement message from the requester. When an Acknowledgement message is received it sends the next block to the requester and again waits for acknowledgement. The requester also repeats the same cycle of states.

If the server detects a communications failure before sending the last block, it enters a cancellation state 1912 wherein the transaction is cancelled. Similarly, if the requester detects a communications failure before receiving the last block it enters a cancellation state 1913.

If there are no more blocks to send, the server commits to the transaction and waits for the final Acknowledgement in state 1914. If there is a communications failure before the server receives the final Acknowledgement message, it still commits to the transaction but includes a report about the event to its credit server in state 1915. This report serves two purposes. It will help legitimize any claims by a user of having been billed for receiving digital works that were not completely received. Also it helps to identify repositories and communications lines that have suspicious patterns of

use and interruption. The server then enters its completion state 1916.

On the requester side, when there are no more blocks to receive, the requester commits to the transaction in state 1917. If the requester detects a communications failure at this state, it reports the failure to its credit server in state 1918, but still commits to the transaction. When it has committed, it sends an acknowledgement message to the server. The server then enters its completion state 1919.

The key property is that both the server and the requester cancel a transaction if it is interrupted before all of the data blocks are delivered, and commits to it if all of the data blocks have been delivered.

There is a possibility that the server will have sent all of the data blocks (and committed) but the requester will not have received all of them and will cancel the transaction. In this case, both repositories will presumably detect a communications failure and report it to their credit server. This case will probably be rare since it depends on very precise timing of the communications failure. The only consequence will be that the user at the requester repository may want to request a refund from the credit services -- and the case for that refund will be documented by reports by both repositories.

To prevent loss of data, the server should not delete any transferred digital work until receiving the final acknowledgement from the requester. But it also should not use the file. A well known way to deal with this situation is called "two-phase commit" or 2PC.

Two-phase commit works as follows. The first phase works the same as the method described above. The server sends all of the data to the requester. Both repositories mark the transaction (and appropriate files) as uncommitted. The server sends a ready-to-commit message to the requester. The requester sends back an acknowledgement. The server then commits and sends the requester a commit message. When the requester receives the commit message, it commits the file.

If there is a communication failure or other crash, the requester must check back with the server to determine the status of the transaction. The server has the last word on this. The requester may have received all of the data, but if it did not get the final message, it has not committed. The server can go ahead and delete files (except for transaction records) once it commits, since the files are known to have been fully transmitted before starting the 2PC cycle.

There are variations known in the art which can be used to achieve the same effect. For example, the server could use an additional level of encryption when transmitting a work to a client. Only after the client sends a message acknowledging receipt does it send the key. The client then agrees to pay for the digital work. The point of this variation is that it provides a clear audit trail that the client received the work. For trusted systems, however, this variation adds a level of encryption for no real gain in accountability.

The transaction for specific usage rights are now discussed.

The Copy Transaction

A Copy transaction is a request to make one or more independent copies of the work with the same or lesser usage rights. Copy differs from the extraction right discussed later in that it refers to entire digital works or entire folders containing digital works. A copy operation cannot be used to remove a portion of a digital work.

- The requester sends the server a message to initiate the Copy Transaction. This message indicates the work to be copied, the version of the copy right to be used for the transaction, the destination address information (location in a folder) for placing the work, the file data for the work (including its size), and the number of copies requested.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the client according to the transmission protocol. If a Next-Set-Of-Rights has been provided in the version of the right, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted. In any event, the Copy-Count field for the copy of the digital work being sent right is set to the number-of-copies requested.
- The requester records the work contents, data, and usage rights and stores the work. It records the date and time that the copy was made in the properties of the digital work.
- The repositories perform the common closing transaction steps.

The Transfer Transaction

A Transfer transaction is a request to move copies of the work with the same or lesser usage rights to another repository. In contrast with a copy transaction, this results in removing the work copies from the server.

- The requester sends the server a message to initiate the Transfer Transaction. This message indicates the work to be transferred, the version of the transfer right to be used in the transaction, the destination address information for placing the work, the file data for the work, and the number of copies involved.

- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted.

In either case, the Copy-Count field for the transmitted rights are set to the number-of-copies requested.

- The requester records the work contents, data, and usage rights and stores the work.
- The server decrements its copy count by the number of copies involved in the transaction.
- The repositories perform the common closing transaction steps.
- If the number of copies remaining in the server is now zero, it erases the digital work from its memory.

The Loan Transaction

A loan transaction is a mechanism for loaning copies of a digital work. The maximum duration of the loan is determined by an internal parameter of the digital work. Works are automatically returned after a predetermined time period.

- The requester sends the server a message to initiate the Transfer Transaction. This message indicates the work to be loaned, the version of the loan right to be used in the transaction, the destination address information for placing the work, the number of copies involved, the file data for the work, and the period of the loan.
- The server checks the validity of the requested loan period, and ends with an error if the period is not valid. Loans for a loaned copy cannot extend beyond the period of the original loan to the server.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted, as modified to reflect the loan period.
- The requester records the digital work contents, data, usage rights, and loan period and stores the work.
- The server updates the usage rights information in the digital work to reflect the number of copies loaned out.
- The repositories perform the common closing transaction steps.
- The server updates the usage rights data for the digital work. This may preclude use of the work until it is returned from the loan. The user on the requester platform can now use the transferred copies of the digital work. A user accessing the original repository cannot use the digital work , unless there are copies remaining. What happens next depends on the order of events in time.

Case 1. If the time of the loan period is not yet exhausted and the requester sends the repository a Return message.

- The return message includes the requester identification, and the transaction ID.
- The server decrements the copies-in-use field by the number of copies that were returned. (If the number of digital works returned is greater than the number actually borrowed, this is treated as an error.) This step may now make the work available at the server for other users.
- The requester deactivates its copies and removes the contents from its memory.

Case 2. If the time of the loan period is exhausted and the requester has not yet sent a Return message.

- The server decrements the copies-in-use field by the number digital works that were borrowed.
- The requester automatically deactivates its copies of the digital work. It terminates all current uses and erases the digital work copies from memory. One question is why a requester would ever return a work earlier than the period of the loan, since it would be returned automatically anyway. One reason for early return is that there may be a metered fee which determines the cost of the loan. Returning early may reduce that fee.

The Play Transaction

A play transaction is a request to use the contents of a work. Typically, to "play" a work is to send the digital work through some kind of transducer, such as a speaker or a display device. The request implies the intention that the contents will not be communicated digitally to any other system. For example, they will not be sent to a printer, recorded on any digital medium, retained after the transaction or sent to another repository.

This term "play" is natural for examples like playing music, playing a movie, or playing a video game. The general

form of play means that a "player" is used to use the digital work. However, the term play covers all media and kinds of recordings. Thus one would "play" a digital work, meaning, to render it for reading, or play a computer program, meaning to execute it. For a digital ticket the player would be a digital ticket agent.

- The requester sends the server a message to initiate the play transaction. This message indicates the work to be played, the version of the play right to be used in the transaction, the identity of the player being used, and the file data for the work.
- The server checks the validity of the player identification and the compatibility of the player identification with the player specification in the right. It ends with an error if these are not satisfactory.
- The repositories perform the common opening transaction steps.
- The server and requester read and write the blocks of data as requested by the player according to the transmission protocol. The requester plays the work contents, using the player.
- When the player is finished, the player and the requester remove the contents from their memory.
- The repositories perform the common closing transaction steps.

The Print Transaction

A Print transaction is a request to obtain the contents of a work for the purpose of rendering them on a "printer." We use the term "printer" to include the common case of writing with ink on paper. However, the key aspect of "printing" in our use of the term is that it makes a copy of the digital work in a place outside of the protection of usage rights. As with all rights, this may require particular authorization certificates.

Once a digital work is printed, the publisher and user are bound by whatever copyright laws are in effect. However, printing moves the contents outside the control of repositories. For example, absent any other enforcement mechanisms, once a digital work is printed on paper, it can be copied on ordinary photocopying machines without intervention by a repository to collect usage fees. If the printer to a digital disk is permitted, then that digital copy is outside of the control of usage rights. Both the creator and the user know this, although the creator does not necessarily give tacit consent to such copying, which may violate copyright laws.

- The requester sends the server a message to initiate a Print transaction. This message indicates the work to be played, the identity of the printer being used, the file data for the work, and the number of copies in the request.
- The server checks the validity of the printer identification and the compatibility of the printer identification with the printer specification in the right. It ends with an error if these are not satisfactory.
- The repositories perform the common opening transaction steps.
- The server transmits blocks of data according to the transmission protocol.
- The requester prints the work contents, using the printer.
- When the printer is finished, the printer and the requester remove the contents from their memory.
- The repositories perform the common closing transaction steps.

The Backup Transaction

A Backup transaction is a request to make a backup copy of a digital work, as a protection against media failure. In the context of repositories, secure backup copies differ from other copies in three ways: (1) they are made under the control of a Backup transaction rather than a Copy transaction, (2) they do not count as regular copies, and (3) they are not usable as regular copies. Generally, backup copies are encrypted.

Although backup copies may be transferred or copied, depending on their assigned rights, the only way to make them useful for playing, printing or embedding is to restore them.

The output of a Backup operation is both an encrypted data file that contains the contents and description of a work, and a restoration file with an encryption key for restoring the encrypted contents. In many cases, the encrypted data file would have rights for "printing" it to a disk outside of the protection system, relying just on its encryption for security. Such files could be stored anywhere that was physically safe and convenient. The restoration file would be held in the repository. This file is necessary for the restoration of a backup copy. It may have rights for transfer between repositories.

- The requester sends the server a message to initiate a backup transaction. This message indicates the work to be backed up, the version of the backup right to be used in the transaction, the destination address information for placing the backup copy, the file data for the work.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester. If a Next-Set-Of-Rights has been provided,

those rights are transmitted as the rights for the work. Otherwise, a set of default rights for backup files of the original are transmitted by the server.

- The requester records the work contents, data, and usage rights. It then creates a one-time key and encrypts the contents file. It saves the key information in a restoration file.
- The repositories perform the common closing transaction steps.

In some cases, it is convenient to be able to archive the large, encrypted contents file to secure offline storage, such as a magneto-optical storage system or magnetic tape. This creation of a non-repository archive file is as secure as the encryption process. Such non-repository archive storage is considered a form of "printing" and is controlled by a print right with a specified "archive-printer." An archive-printer device is programmed to save the encrypted contents file (but not the description file) offline in such a way that it can be retrieved.

The Restore Transaction

A Restore transaction is a request to convert an encrypted backup copy of a digital work into a usable copy. A restore operation is intended to be used to compensate for catastrophic media failure. Like all usage rights, restoration rights can include fees and access tests including authorization checks.

- The requester sends the server a message to initiate a Restore transaction. This message indicates the work to be restored, the version of the restore right for the transaction, the destination address information for placing the work, and the file data for the work.
- The server verifies that the contents file is available (i.e. a digital work corresponding to the request has been backed-up.) If it is not, it ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The server retrieves the key from the restoration file. It decrypts the work contents, data, and usage rights.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, a set of default rights for backup files of the original are transmitted by the server.
- The requester stores the digital work.
- The repositories perform the common closing transaction steps.

The Delete Transaction

A Delete transaction deletes a digital work or a number of copies of a digital work from a repository. Practically all digital works would have delete rights.

- The requester sends the server a message to initiate a delete transaction. This message indicates the work to be deleted, the version of the delete right for the transaction.
- The repositories perform the common opening transaction steps.
- The server deletes the file, erasing it from the file system.
- The repositories perform the common closing transaction steps.

The Directory Transaction

A Directory transaction is a request for information about folders, digital works, and their parts. This amounts to roughly the same idea as protection codes in a conventional file system like TENEX, except that it is generalized to the full power of the access specifications of the usage rights language.

The Directory transaction has the important role of passing along descriptions of the rights and fees associated with a digital work. When a user wants to exercise a right, the user interface of his repository implicitly makes a directory request to determine the versions of the right that are available. Typically these are presented to the user -- such as with different choices of billing for exercising a right. Thus, many directory transactions are invisible to the user and are exercised as part of the normal process of exercising all rights.

- The requester sends the server a message to initiate a Directory transaction. This message indicates the file or folder that is the root of the directory request and the version of the directory right used for the transaction.
- The server verifies that the information is accessible to the requester. In particular, it does not return the names of any files that have a HIDE-NAME status in their directory specifications, and it does not return the parts of any folders or files that have HIDE-PARTS in their specification. If the information is not accessible, the server ends

the transaction with an error.

- The repositories perform the common opening transaction steps.
- The server sends the requested data to the requester according to the transmission protocol.
- The requester records the data.
- The repositories perform the common closing transaction steps.

The Folder Transaction

A Folder transaction is a request to create or rename a folder, or to move a work between folders. Together with Directory rights, Folder rights control the degree to which organization of a repository can be accessed or modified from another repository.

- The requester sends the server a message to initiate a Folder transaction. This message indicates the folder that is the root of the folder request, the version of the folder right for the transaction, an operation, and data. The operation can be one of create, rename, and move file. The data are the specifications required for the operation, such as a specification of a folder or digital work and a name.
- The repositories perform the common opening transaction steps.
- The server performs the requested operation -- creating a folder, renaming a folder, or moving a work between folders.
- The repositories perform the common closing transaction steps.

The Extract Transaction

A extract transaction is a request to copy a part of a digital work and to create a new work containing it. The extraction operation differs from copying in that it can be used to separate a part of a digital work from d-blocks or shells that place additional restrictions or fees on it. The extraction operation differs from the edit operation in that it does not change the contents of a work, only its embedding in d-blocks. Extraction creates a new digital work.

- The requester sends the server a message to initiate an Extract transaction. This message indicates the part of the work to be extracted, the version of the extract right to be used in the transaction, the destination address information for placing the part as a new work, the file data for the work, and the number of copies involved.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the new work. Otherwise, the rights of the original are transmitted. The Copy-Count field for this right is set to the number-of-copies requested.
- The requester records the contents, data, and usage rights and stores the work. It records the date and time that new work was made in the properties of the work.
- The repositories perform the common closing transaction steps.

The Embed Transaction

An embed transaction is a request to make a digital work become a part of another digital work or to add a shell d-block to enable the adding of fees by a distributor of the work.

- The requester sends the server a message to initiate an Embed transaction. This message indicates the work to be embedded, the version of the embed right to be used in the transaction, the destination address information for placing the part as a a work, the file data for the work, and the number of copies involved.
- The server checks the control specifications for all of the rights in the part and the destination. If they are incompatible, the server ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the new work. Otherwise, the rights of the original are transmitted. The Copy-Count field for this right is set to the number-of-copies requested.
- The requester records the contents, data, and usage rights and embeds the work in the destination file.
- The repositories perform the common closing transaction steps.

The Edit Transaction

An Edit transaction is a request to make a new digital work by copying, selecting and modifying portions of an existing digital work. This operation can actually change the contents of a digital work. The kinds of changes that are permitted depend on the process being used. Like the extraction operation, edit operates on portions of a digital work. In contrast with the extract operation, edit does not affect the rights or location of the work. It only changes the contents. The kinds of changes permitted are determined by the type specification of the processor specified in the rights. In the currently preferred embodiment, an edit transaction changes the work itself and does not make a new work. However, it would be a reasonable variation to cause a new copy of the work to be made.

- The requester sends the server a message to initiate an Edit transaction. This message indicates the work to be edited, the version of the edit right to be used in the transaction, the file data for the work (including its size), the process-ID for the process, and the number of copies involved.
- The server checks the compatibility of the process-ID to be used by the requester against any process-ID specification in the right. If they are incompatible, it ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The requester uses the process to change the contents of the digital work as desired. (For example, it can select and duplicate parts of it; combine it with other information; or compute functions based on the information. This can amount to editing text, music, or pictures or taking whatever other steps are useful in creating a derivative work.)
- The repositories perform the common closing transaction steps.

The edit transaction is used to cover a wide range of kinds of works. The category describes a process that takes as its input any portion of a digital work and then modifies the input in some way. For example, for text, a process for editing the text would require edit rights. A process for "summarizing" or counting words in the text would also be considered editing. For a music file, processing could involve changing the pitch or tempo, or adding reverberations, or any other audio effect. For digital video works, anything which alters the image would require edit rights. Examples would be colorizing, scaling, extracting still photos, selecting and combining frames into story boards, sharpening with signal processing, and so on.

Some creators may want to protect the authenticity of their works by limiting the kinds of processes that can be performed on them. If there are no edit rights, then no processing is allowed at all. A processor identifier can be included to specify what kind of process is allowed. If no process identifier is specified, then arbitrary processors can be used. For an example of a specific process, a photographer may want to allow use of his photograph but may not want it to be colorized. A musician may want to allow extraction of portions of his work but not changing of the tonality.

Authorization Transactions

There are many ways that authorization transactions can be defined. In the following, our preferred way is to simply define them in terms of other transactions that we already need for repositories. Thus, it is convenient sometimes to speak of "authorization transactions," but they are actually made up of other transactions that repositories already have.

A usage right can specify an authorization-ID, which identifies an authorization object (a digital work in a file of a standard format) that the repository must have and which it must process. The authorization is given to the generic authorization (or ticket) server of the repository which begins to interpret the authorization.

As described earlier, the authorization contains a server identifier, which may just be the generic authorization server or it may be another server. When a remote authorization server is required, it must contain a digital address. It may also contain a digital certificate.

If a remote authorization server is required, then the authorization process first performs the following steps:

- The generic authorization server attempts to set up the communications channel. (If the channel cannot be set up, then authorization fails with an error.)
- When the channel is set up, it performs a registration process with the remote repository. (If registration fails, then the authorization fails with an error.)
- When registration is complete, the generic authorization server invokes a "Play" transaction with the remote repository, supplying the authorization document as the digital work to be played, and the remote authorization server (a program) as the "player." (If the player cannot be found or has some other error, then the authorization fails with an error.)
- The authorization server then "plays" the authorization. This involves decrypting it using either the public key of the master repository that issued the certificate or the session key from the repository that transmitted it. The authorization server then performs various tests. These tests vary according to the authorization server. They

include such steps as checking issue and validity dates of the authorization and checking any hot-lists of known invalid authorizations. The authorization server may require carrying out any other transactions on the repository as well, such as checking directories, getting some person to supply a password, or playing some other digital work. It may also invoke some special process for checking information about locations or recent events. The "script" for such steps is contained within the authorization server.

- If all of the required steps are completed satisfactorily, the authorization server completes the transaction normally, signaling that authorization is granted.

The Install Transaction

An Install transaction is a request to install a digital work as runnable software on a repository. In a typical case, the requester repository is a rendering repository and the software would be a new kind or new version of a player. Also in a typical case, the software would be copied to file system of the requester repository before it is installed.

- The requester sends the server an Install message. This message indicates the work to be installed, the version of the Install right being invoked, and the file data for the work (including its size).
- The repositories perform the common opening transaction steps.
- The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error.
- The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step certifies the software.)
- The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.)
- The requester retrieves the instructions in the compatibility-checking script and follows them. If the software is not compatible with the repository, the installation transaction ends with an error. (This step checks platform compatibility.)
- The requester retrieves the instructions in the installation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error. Note that the installation process puts the runnable software in a place in the repository where it is no longer accessible as a work for exercising any usage rights other than the execution of the software as part of repository operations in carrying out other transactions.
- The repositories perform the common closing transaction steps.

The Uninstall Transaction

An Uninstall transaction is a request to remove software from a repository. Since uncontrolled or incorrect removal of software from a repository could compromise its behavioral integrity, this step is controlled.

- The requester sends the server an Uninstall message. This message indicates the work to be uninstalled, the version of the Uninstall right being invoked, and the file data for the work (including its size).
- The repositories perform the common opening transaction steps.
- The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error.
- The requester checks whether the software is installed. If the software is not installed, the transaction ends with an error.
- The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step authenticates the certification of the software, including the script for uninstalling it.)
- The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.)
- The requester retrieves the instructions in the uninstallation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error.
- The repositories perform the common closing transaction steps.

Claims

1. A system for secure distribution and control of digital works between repositories comprising:

means for creating usage rights, each instance of a usage right representing a specific instance of how a digital work may be used or distributed;
means for attaching a created set of usage rights to a digital work;
a communications medium for coupling repositories to enable exchange of repository transaction messages;
a plurality of general repositories for storing and securely exchanging digital works with attached usage rights, each of said general repositories comprising:
a storage means for storing digital works and their attached usage rights;
an identification certificate for indicating that the associated general repository is secure;
an external interface for removably coupling to said communications medium;
a session initiation transaction processing means for establishing a secure and trusted session with another repository, said session initiation transaction processing means using said identification certificate;
a usage transaction processing means having a requester mode of operation for generating usage repository transaction messages to request access to digital works stored in another general repository, said usage repository transaction message specifying a usage right, said usage transaction processing means further having a server mode of operation for determining if a request for access to a digital work stored in said storage means may be granted, said request being granted only if the usage right specified in said request is attached to said digital work; and
an input means coupled to said usage transaction processing means for enabling user created signals to cause generation of a usage repository transaction message to request access to digital works.

2. The system as recited in Claim 1 further comprising a rendering system, said rendering system comprising:

a rendering repository for securely accessing digital works from a general repository, said rendering repository comprising;
a storage means for storing digital works and their attached usage rights;
an identification certificate, said identification certificate for indicating that the rendering repository is secure;
an external interface for removably coupling to said communications medium;
a session initiation transaction processing means for establishing a secure and trusted session with a general repository, said session initiation transaction processing means using said identification certificate;
a usage transaction processing means for generating usage repository transaction messages to request access to digital works stored in a general repository, said usage repository transaction message specifying a usage right;
an input means coupled to said usage transaction processing means for enabling user created signals to cause generation of usage repository transaction messages to request access to digital works;
a rendering device for rendering digital works.

3. The system as recited in Claim 1 wherein said means for creating usage rights is further for the specification of different sets of usage rights to be attached to digital works when a corresponding usage right is exercised.

4. The system as recited in Claim 1 wherein said usage rights grammar further defines means for specifying conditions which must be satisfied before a usage right may be exercised and said usage transaction processing means in said server mode is further comprised of means for determining if specified conditions for a usage right are satisfied before access is granted.

5. The system as recited in Claim 1 wherein a first usage right enables copying of a digital work and specification of a revenue owner who is paid a fee whenever a copy of said digital work is made.

6. A method for controlling distribution and use of digital works comprising the steps of:

a) attaching a set of usage rights to a digital work, each of said usage rights defining a specific instance of how a digital work may be used or distributed, said usage right specifying one or more conditions which must be satisfied in order for said usage right to be exercised and a next set of usage rights to be attached to a distributed digital work;
b) storing said digital work and its attached usage rights in a first repository;

c) a second repository initiating a request to access said digital work in said first repository, said request identifying a usage right representing how said second repository desires to use said digital work;
d) said first repository receiving said request from said second repository;
e) said first repository determining if the identified usage right is attached to said digital work;
f) said first repository denying access to said digital work if said identified usage right is not attached to said digital work;
g) if said identified usage right is attached to said digital work, said first repository determining if conditions specified by said usage right are satisfied;
h) if said conditions are not satisfied, said first repository denying access to said digital work;
i) if said conditions are satisfied, said first repository attaching a next set of usage rights to said digital work, said next set of usage rights specifying how said second repository may use and distribute said digital work; and
j) said first repository transmitting said digital work and said attached next set of usage rights to said second repository.

7. The method as recited in Claim 6 wherein said step of a second repository initiating a request to access said digital work in said first repository is further comprised of the steps of:

c1) said second repository initiating establishment of a trusted session with said first repository;
c2) said first repository performing a set of registration transaction steps with said second repository, successful completion of said set of registration transaction steps indicating that said first repository is a trusted repository;
c3) said second repository performing said set of registration transaction steps with said first repository, successful completion of said set of registration transaction steps indicating that said second repository is a trusted repository;
c4) if said first repository and said second repository each successfully complete said set of registration steps, said first and second repository exchanging session encryption and decryption keys for secure transmission of subsequent communications between said first and second repository; and
c5) if said first repository or said second repository cannot successfully complete said set of registration transaction steps, terminating said session.

8. A system for controlling distribution and use of digital works comprising:

means for attaching usage rights to said digital work, said usage rights indicating how a recipient may use and and subsequently distribute said digital work;
a communications medium for coupling repositories to enable distribution of digital works;
a plurality of repositories for managing exchange of digital works based on usage rights attached to said digital works, each of said plurality of repositories comprising:
a storage means for storing digital works and their attached usage rights;
a processor operating responsive to coded instructions;
a memory means coupled to said processor for storing coded instruction to enable said processor to operate in a first server mode for processing access requests to digital works and for attaching usage rights to digital works when transmitted to another of said plurality of repositories, a second requester mode for initiating requests to access digital works, and a session initiation mode for establishing a trusted session with another of said plurality of repositories over said communications medium;
a clock;
a repository interface for coupling to said communications medium.

9. The system as recited in Claim 8 further comprising a plurality of rendering systems for rendering of digital works, each of said rendering systems comprising:

a repository for secure receipt of a digital work; and
a rendering device having means for converting digital works to signals suitable for rendering of said digital works.

10. A method for secure access of digital works stored on a server repository, said digital works having associated therewith one or more usage rights for specifying how said digital work may be used or distributed, said method comprising the steps of:

a) a requesting repository performing a first registration transaction with a server repository, said first regis-

tration transaction for establishing to said server repository that said requesting repository is trustworthy;
b) concurrently with step a), said server repository responding with a second registration transaction, said second registration transaction for establishing to said requesting repository that said server repository is trustworthy;
c) if either said first registration transaction or said second registration transaction fails, said server repository denying access to said digital work;
d) if said first registration transaction and said second registration transaction are successful, said requesting repository initiating a usage transaction with respect to a digital work stored in said server repository, said usage transaction indicating a request to access a digital work and specifying a particular usage right;
e) determining if said usage transaction may be completed by comparing said particular usage right specified in said usage transaction and usage rights associated with said digital work;
f) if said particular usage right is not one of said usage rights associated with said digital work, denying access to said digital work; and
g) if said particular usage right is one of said usage rights associated with said digital work, granting access to said digital work and performing usage transaction steps associated with said particular usage right.

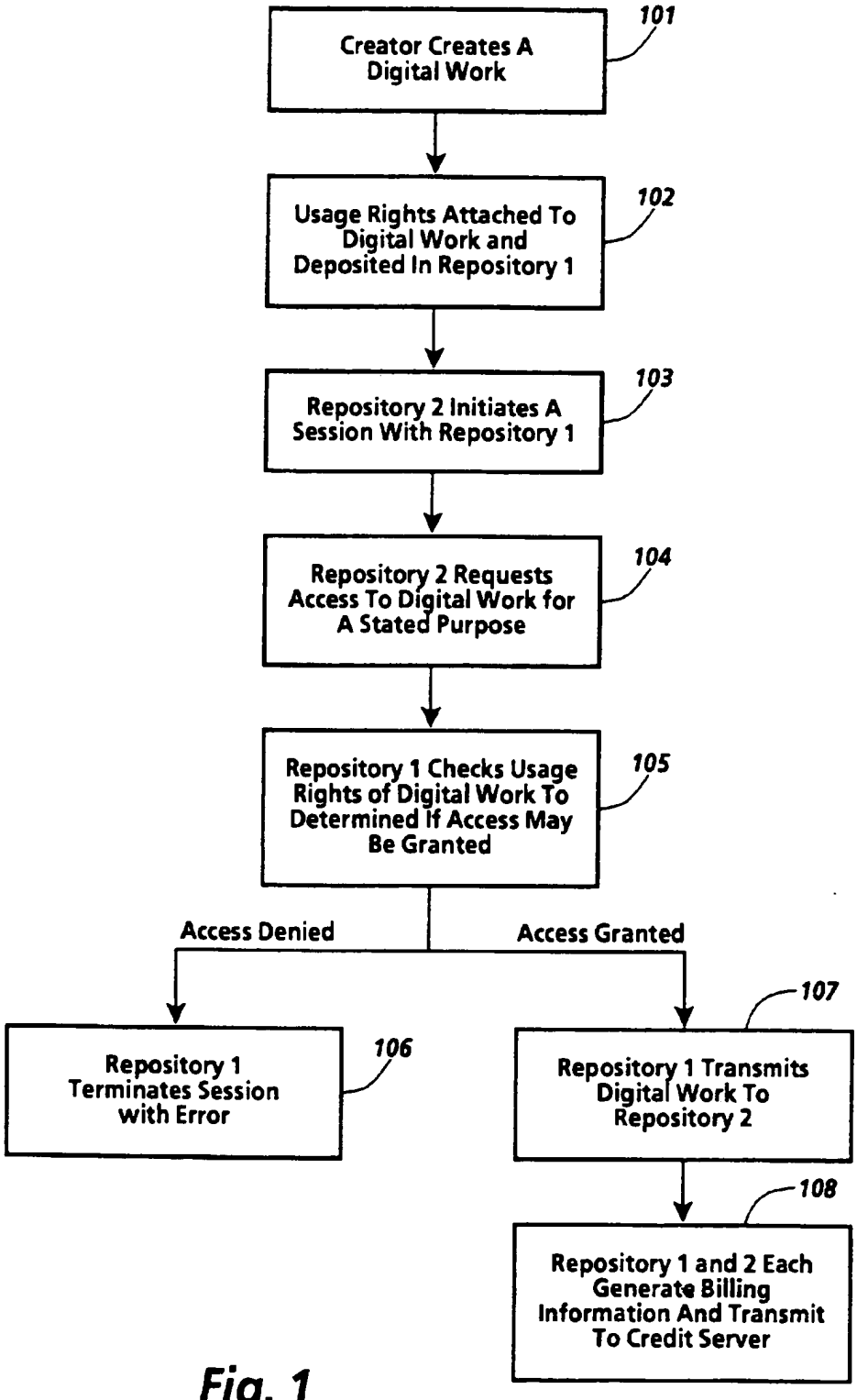


Fig. 1

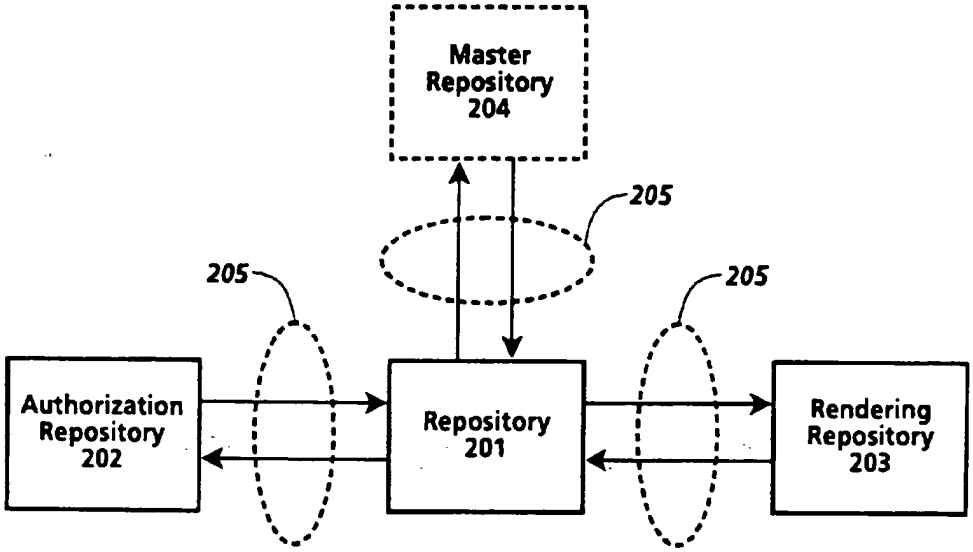


Fig. 2

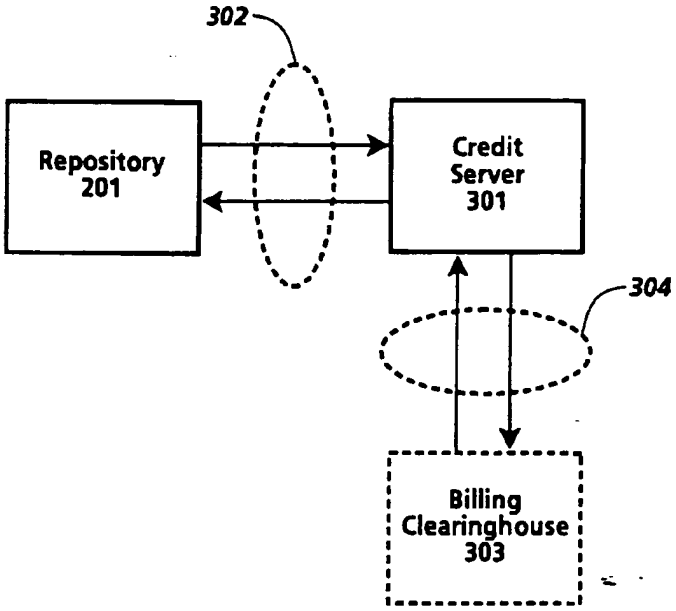


Fig. 3

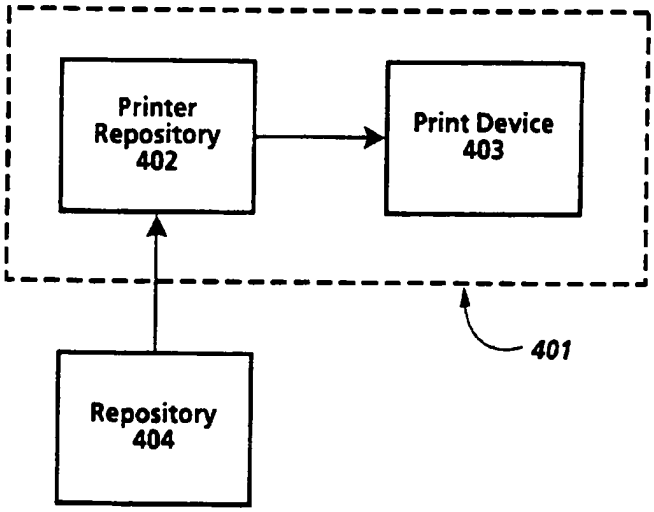


Fig. 4a

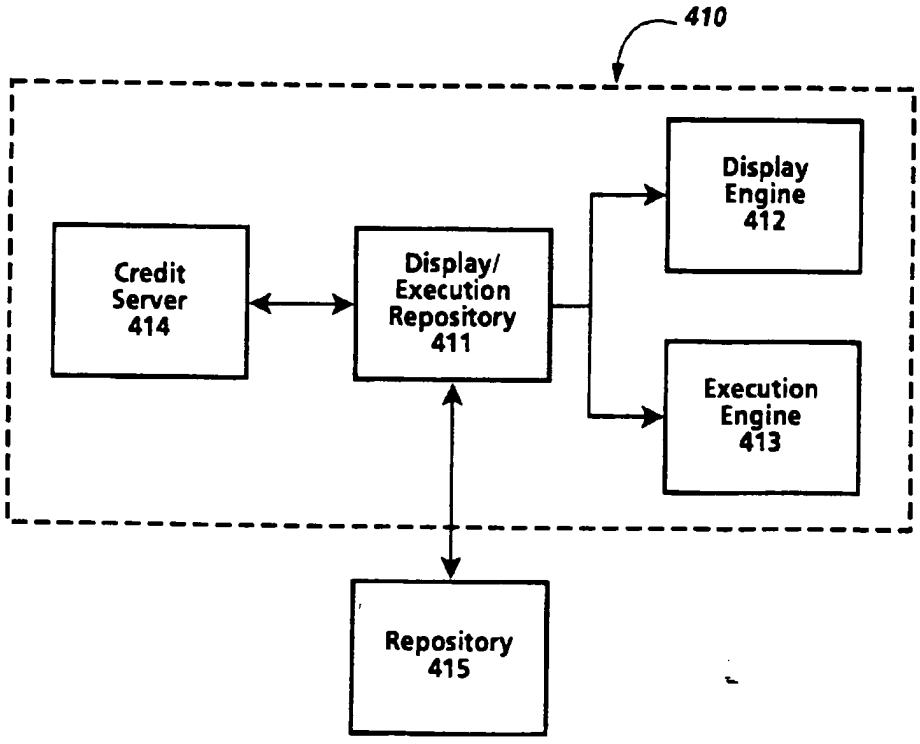


Fig. 4b

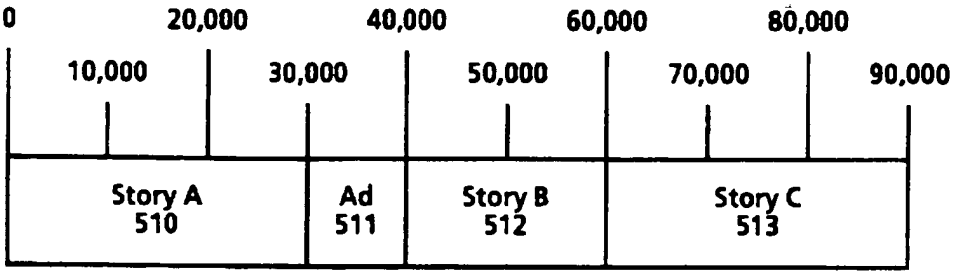


Fig. 5

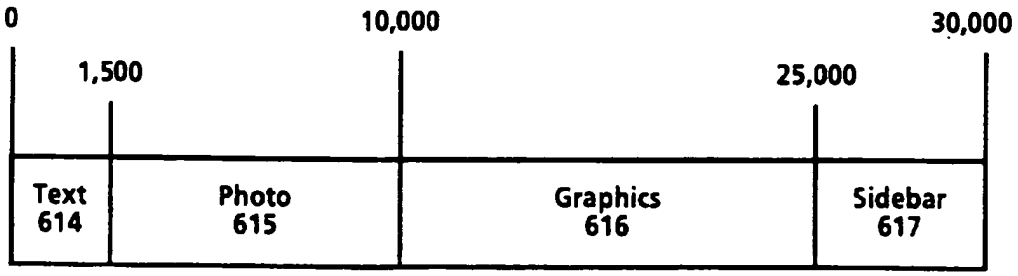


Fig. 6

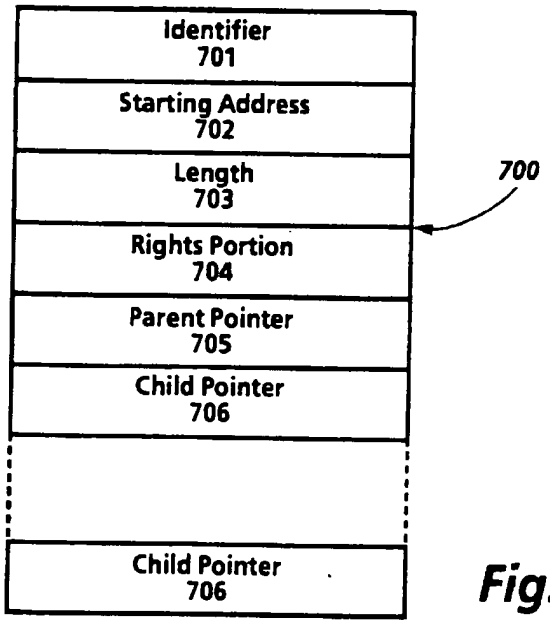


Fig. 7

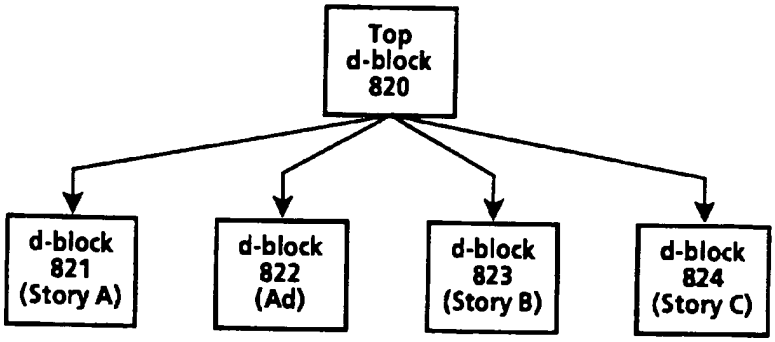


Fig. 8

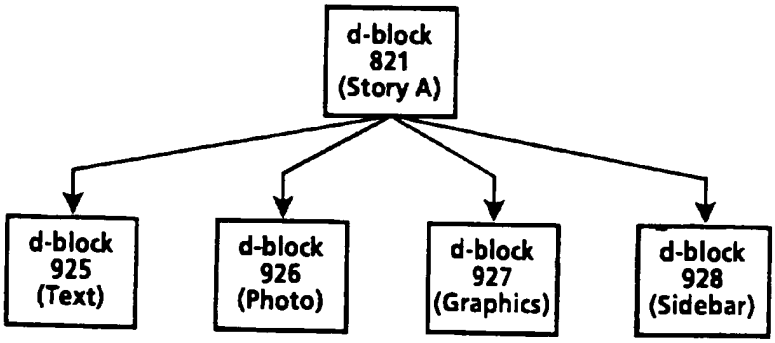


Fig. 9

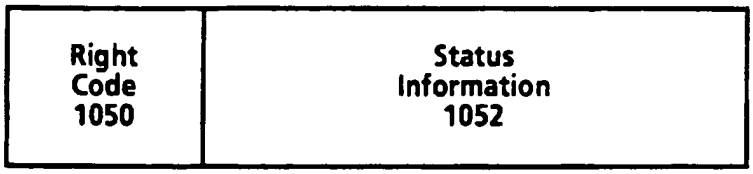


Fig.10

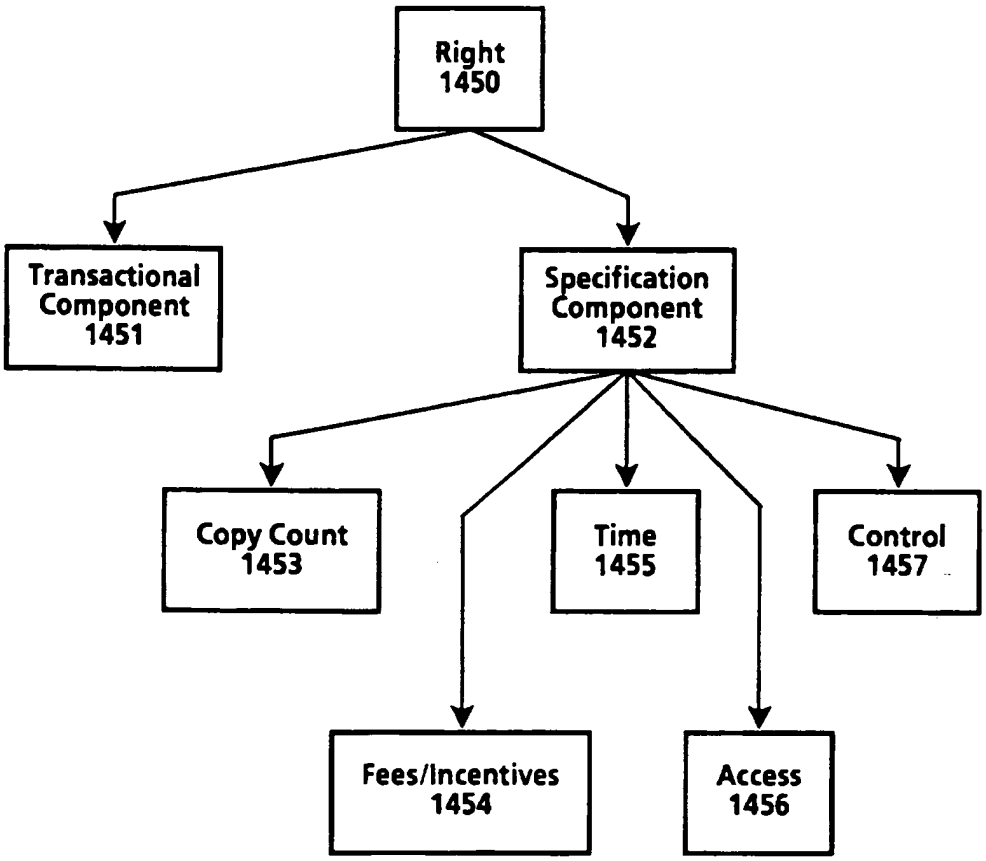


Fig.14

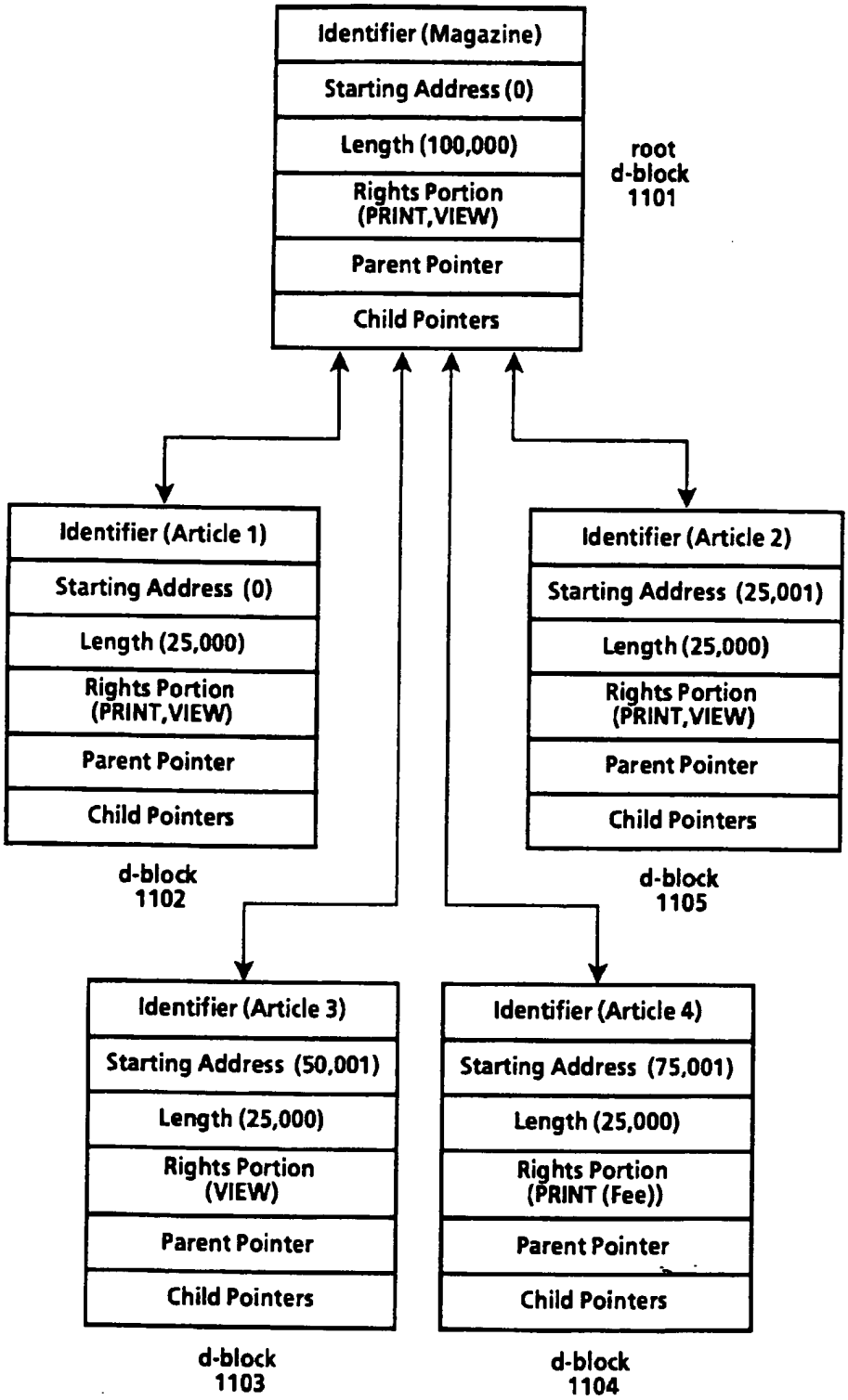


Fig.11

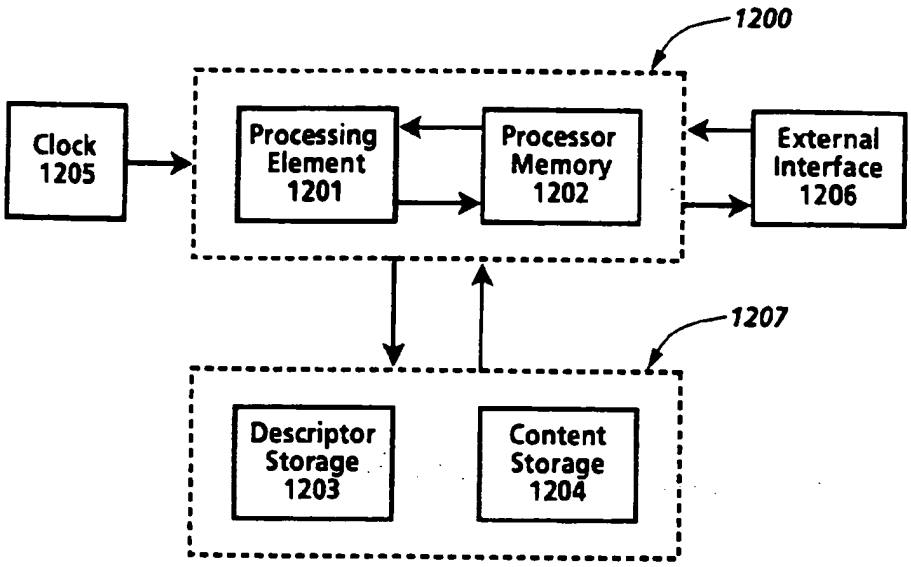


Fig.12

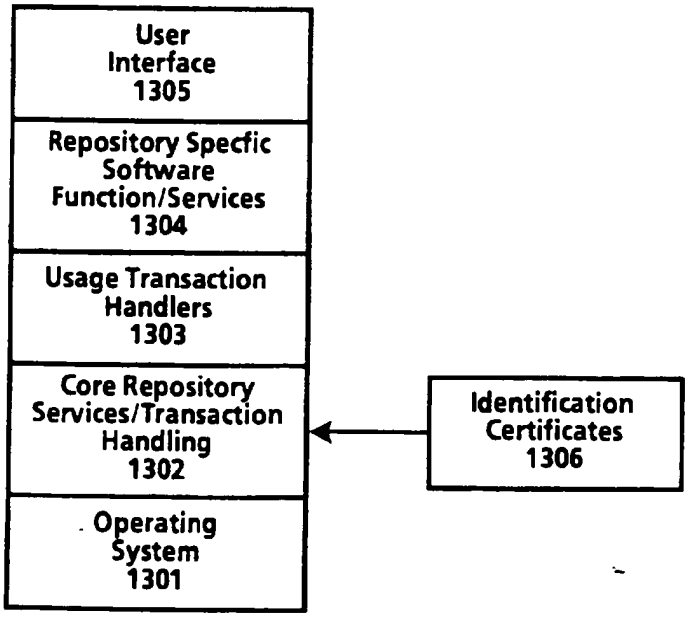


Fig.13

1501 ~ **Digital Work Rights**: = (Rights*)
1502 ~ **Right** : = (Right-Code {Copy-Count} {Control-Spec} {Time-Spec } {Access-Spec} {Fee-Spec})
1503 ~ **Right-Code** : = Render-Code | Transport-Code | File-Management-Code| Derivative-Works- Code | Configuration-Code
1504 ~ **Render-Code** : = [Play : {Player: Player-ID} | Print: {Printer: Printer-ID}]
1505 ~ **Transport-Code** : = [Copy | Transfer | Loan {Remaining-Rights: Next-Set-of-Rights}]{(Next-Copy-Rights: Next-Set-of-Rights)}
1506 ~ **File-Management-Code** : = Backup {Back-Up-Copy-Rights: Next-Set-of-Rights} | Restore | Delete | Folder | Directory {Name: Hide-Local | Hide-Remote} {Parts: Hide-Local | Hide-Remote}
1507 ~ **Derivative-Works-Code** : = [Extract | Embed | Edit{Process: Process-ID}] {Next-Copy-Rights : Next-Set-of Rights}
1508 ~ **Configuration-Code** : = Install | Uninstall
1509 ~ **Next-Set-of-Rights** : = {(Add: Set-Of-Rights)} {(Delete: Set-Of-Rights)} {(Replace: Set-Of-Rights)}{(Keep: Set-Of-Rights)}
1510 ~ **Copy-Count** : = (Copies:positive-integer | 0 | Unlimited)
1511 ~ **Control-Spec** : = (Control: {Restrictable | Unrestrictable} {Unchargeable | Chargeable})
1512 ~ **Time-Spec** : = ({Fixed-Interval | Sliding-Interval | Meter-Time} Until: Expiration-Date)
1513 ~ **Fixed-Interval** : = From: Start-Time
1514 ~ **Sliding-Interval** : = Interval: Use-Duration
1515 ~ **Meter-Time**: = Time-Remaining: Remaining-Use
1516 ~ **Access-Spec** : = ({SC: Security-Class} {Authorization: Authorization-ID*} {Other-Authorization: Authorization-ID*} {Ticket: Ticket-ID})
1517 ~ **Fee-Spec**: = {Scheduled-Discount} Regular-Fee-Spec | Scheduled-Fee-Spec | Markup-Spec
1518 ~ **Scheduled-Discount**: = Scheduled-Discount: (Scheduled-Discount: (Time-Spec Percentage)*)
1519 ~ **Regular-Fee-Spec** : = ({Fee: | Incentive: } [Per-Use-Spec | Metered-Rate-Spec | Best-Price-Spec | Call-For-Price-Spec] {Min: Money-Unit Per: Time-Spec}{Max: Money-Unit Per: Time-Spec} To: Account-ID)
1520 ~ **Per-Use-Spec**: = Per-Use: Money-unit
1521 ~ **Metered-Rate-Spec** : = Metered: Money-Unit Per: Time-Spec
1522 ~ **Best-Price-Spec** : = Best-Price: Money-unit Max: Money-unit
1523 ~ **Call-For-Price-Spec** : = Call-For -Price
1524 ~ **Scheduled-Fee-Spec**: = (Schedule: (Time-Spec Regular-Fee-Spec)*)
1525 ~ **Markup-Spec**: = Markup: percentage To: Account-ID

Fig.15

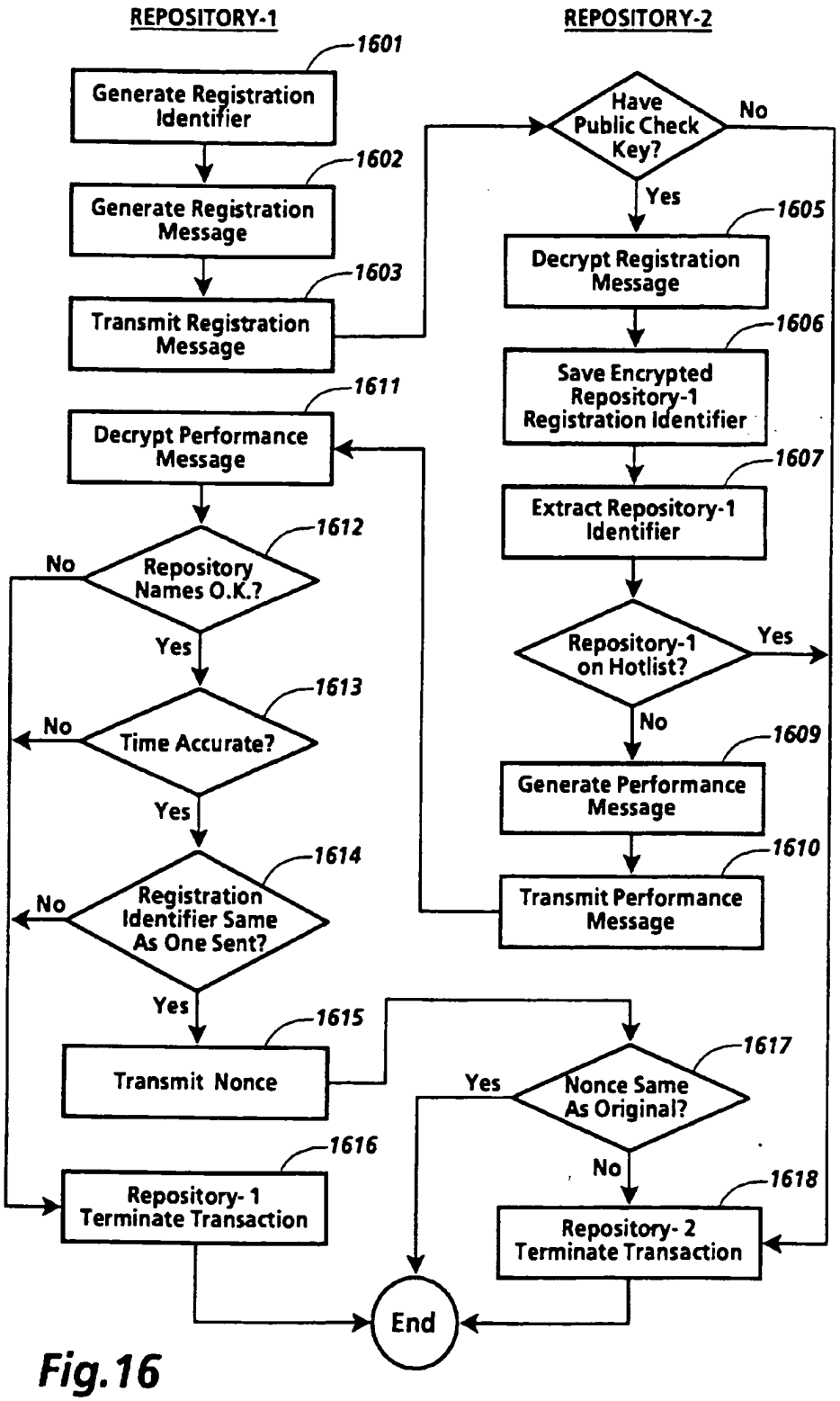


Fig.16

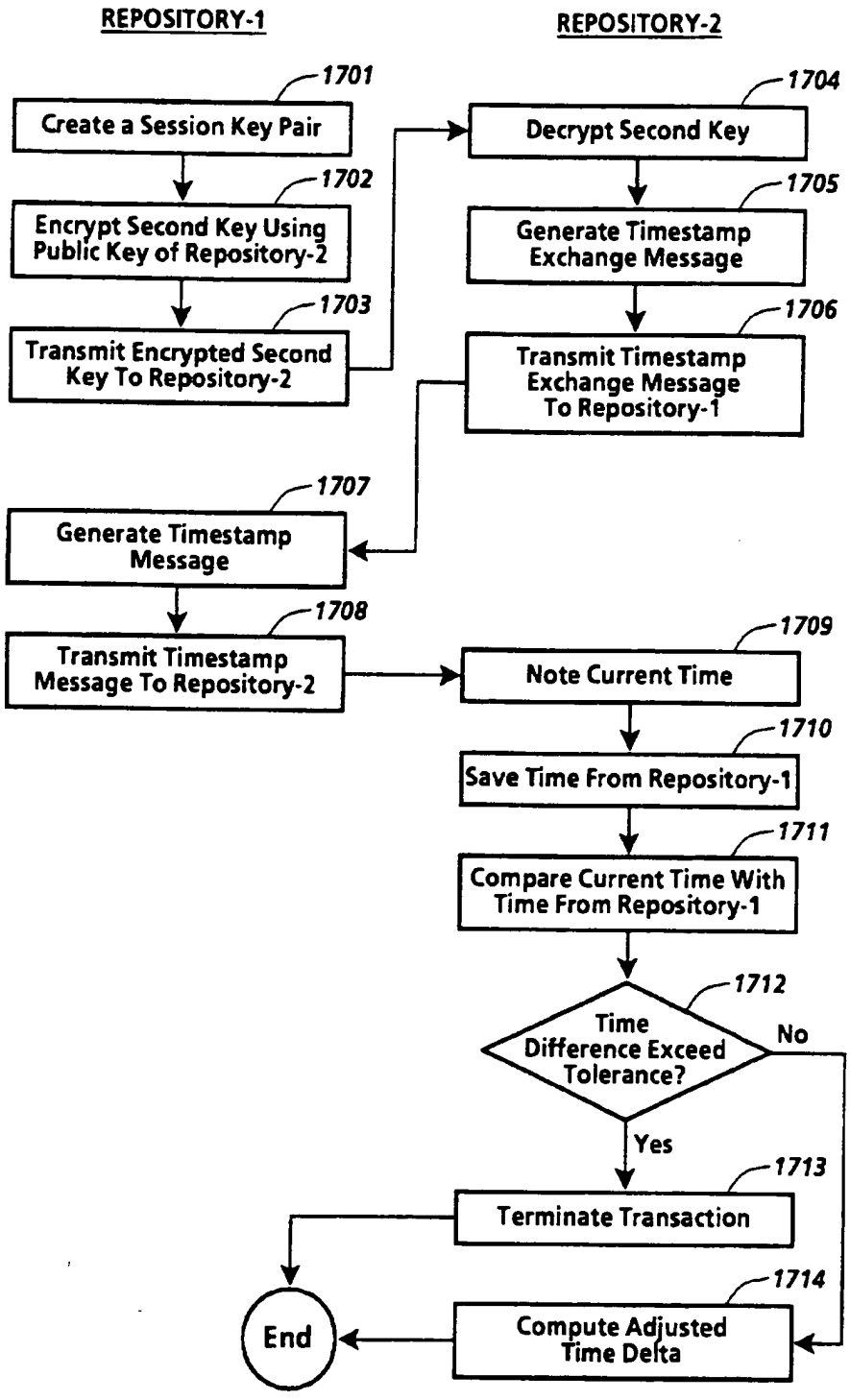


Fig.17

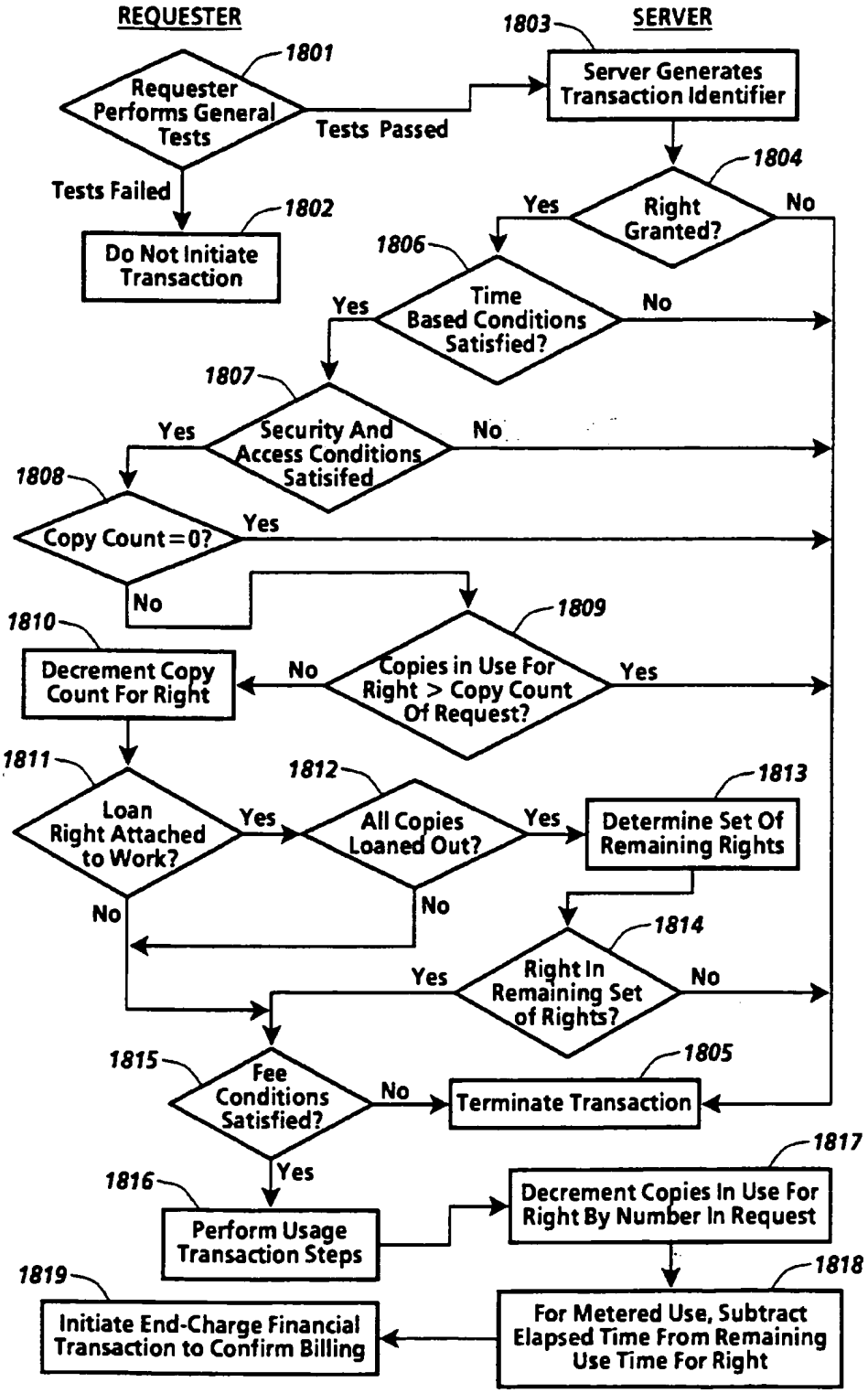


Fig.18

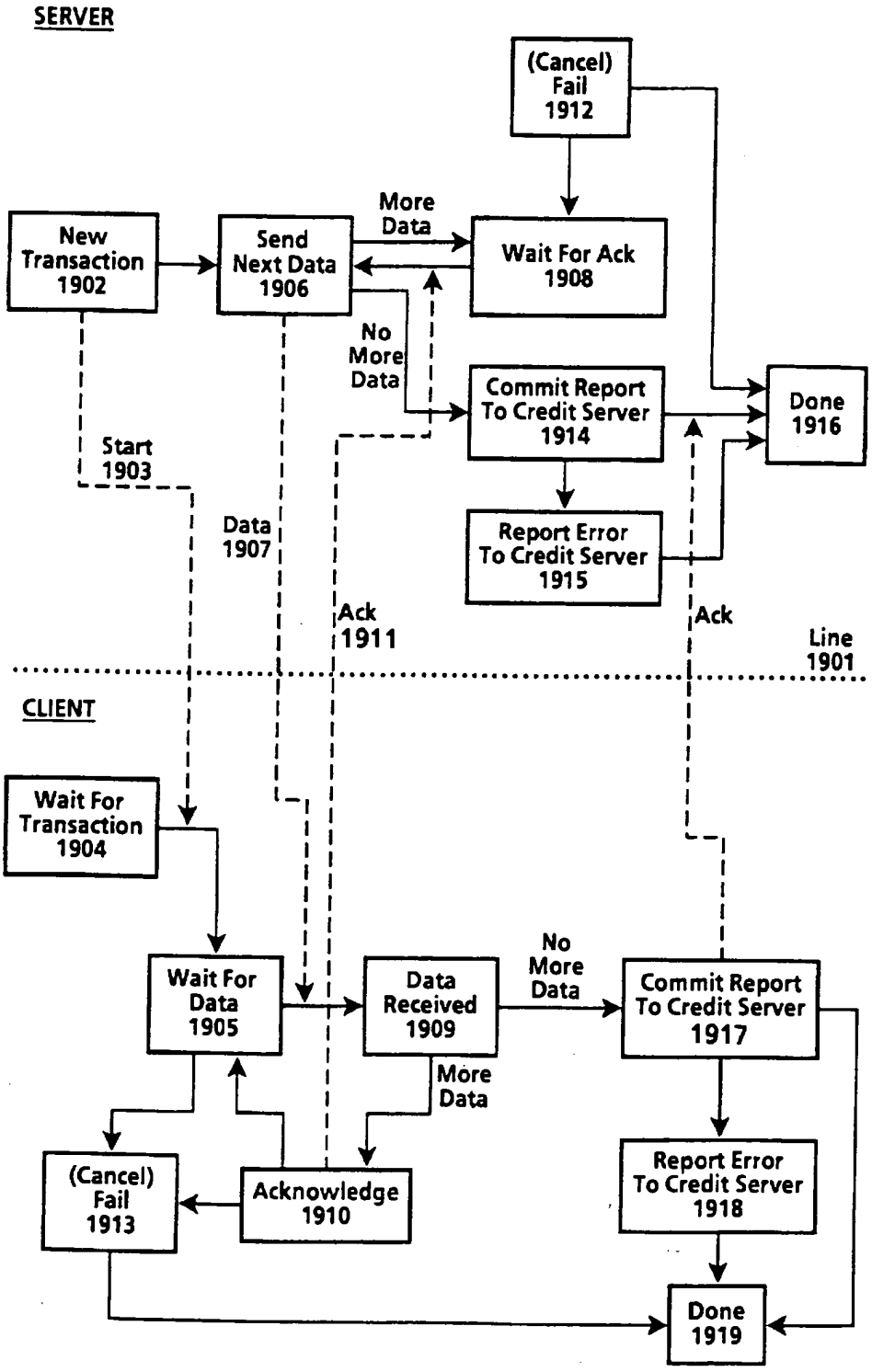


Fig.19

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 95 30 8420

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
A	WO-A-92 20022 (DIGITAL EQUIPMENT CORP.) * page 45, line 10 - page 64, line 17 * ---	1,6,8,10	G06F1/00
A	GB-A-2 236 604 (SUN MICROSYSTEMS INC) * page 9, line 11 - page 20, line 15 * ---	1,6,8,10	
A	US-A-5 291 596 (MITA) * the whole document * -----	1,6,8,10	

TECHNICAL FIELDS SEARCHED (Int.Cl.6)

G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	1 April 1996	Moens, R

EPO FORM 1503 03.82 (P04C01)

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 731 404 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
11.09.1996 Bulletin 1996/37

(51) Int. Cl.[6]: **G06F 1/00**, G06F 19/00

(21) Application number: **96100832.3**

(22) Date of filing: **22.01.1996**

(84) Designated Contracting States:
DE FR GB

(30) Priority: **07.03.1995 US 401484**

(71) Applicant: **International Business Machines Corporation**
Armonk, N.Y. 10504 (US)

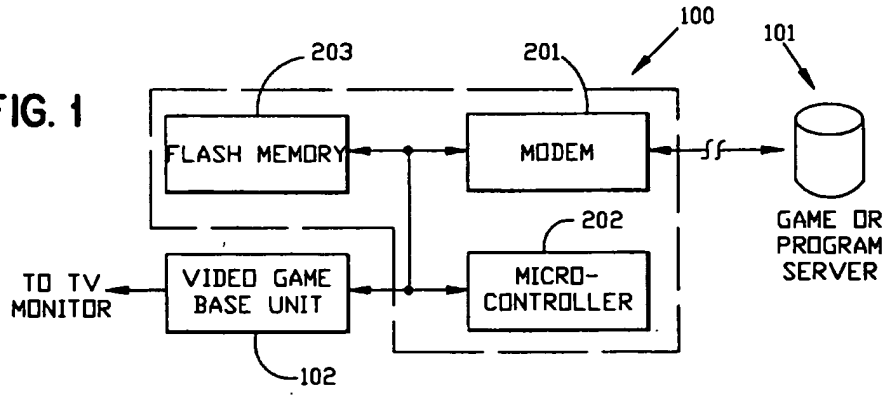
(72) Inventors:
- **Bakoglu, Halil Burhan**
  Ossining, New York 10562 (US)
- **Chen, Inching**
  Wappingers Falls, New York 12590 (US)
- **Lean, Andy Geng-Chyun**
  Merrick, New York 11566 (US)
- **Maruyama, Kiyoshi**
  Chappaqua, New York 10514 (US)
- **Yue, Chung-wai**
  Yorktown Heights, New York 10598 (US)

(74) Representative: **Rach, Werner, Dr.**
IBM Deutschland
Informationssysteme GmbH,
Patentwesen und Urheberrecht
70548 Stuttgart (DE)

(54) A universal electronic video game renting/distributing system

(57) A video game cartridge that can be plugged into a video game machine to enable a user to request and play a video game for a predetermined number of video frames. The cartridge has a receiver for receiving the video game program and the predetermined frame count in response to a request from the user. The program and frame count is then stored in a memory of the cartridge. Finally, the cartridge has a counter which changes its value when the user is actively playing the video game program. The counter ceases to change its value when the user is not playing the video game program. When the counter reaches a predetermined limit, the user is no longer authorized to play the video game program.

FIG. 1



EP 0 731 404 A1

Printed by Rank Xerox (UK) Business Services
2.13.4/3.4

Description

Technical Field

This invention relates to a video game cartridge for receiving video game programs from a remote server.

Description of the Prior Art

Today, there are many video games available for purchase or for rental at stores. Generally, there is no trial or test playing of the games in the stores, and there is no return on purchased games once the game package has been opened. Therefore, a person who is interested in any game has to buy it before playing it and thus may face the risk of not liking the game later. There is no return or refund of the game since the package has been opened. A person who rents a game from a store has to go through the usual VCR tape rental trouble of driving to the store, picking up the game and then later returning the game to the store.

To make video game rental easier for the consumer, Sega has created the Sega Channel. In this service, via cable and using a cable adapter unit which is plugged into the Sega Genesis game machine, people can play games that are downloaded to the cable adapter. It requires the on-line Sega Channel connection as well as the special adapter while the game is being played.

Down loading a software program to a personal computer over the modem connection exists today. Such software can come with a limited life where the life can be specified by expiration date, or time, or the number of times of the software usage. These schemes in limiting the software usage is not applicable to down loading video games to cartridges which are plugged into existing video game base units because these game base units do not have timer device built in. Thus a new scheme for controlling the usage of the game is needed.

The US Patent 4,905,280 to J.D. Wiedemer, et al describes a method for real time down loading of broadcast programs for pay-per-view or for subscription. Descrambling of broadcast programs is done by codes on a replaceable memory module, which is delivered to a subscriber by the service provider. This patent is applicable to the "purchase" of software content or real-time service, but it is not applicable to limiting the life of rented software.

US patent 5,251,909 to Reed et al describes software renting or distributing schemes in which access is granted to a subscriber prior to the actual programs being transmitted. This patent describes an off-line process and is not applicable to delivering software for rental purposes.

Summary of the Invention

It is an object of this invention to provide a portable video game cartridge which can be plugged into a video game machine base unit, such as Nintendos, Sega Genesis&tm. video game machine or Atari's Jaguar&tm. video game machine. The cartridge will allow a video game program to be used by receiving the video program over a telephone network or cable system.

The current invention describes a way of distributing and controlling the usage of a video game program (or any software program) by using a "watchdog mechanism" and by limiting the "life" of a game by limiting the total number of graphic frames that a video machine can generate. It offers a simple and effective way of software renting and distribution where game machines have no timer.

It is also an object of this invention to prevent piracy of video programs and programs in general by storing the frame count in a random location of the memory that is unknown to a potential pirate, especially if the count itself is encrypted. Since the count is part of the video game program or program execution path, the video game or program cannot be used without knowledge of the count.

This invention is generally an apparatus and method for enabling a user to request and use a program where the user receives the program and a frame count indicating the number of frames of the program that the user is authorized to execute or use. This program and the frame count is then stored in a memory. When the user is actively providing input to the program, the frame count changes. The frame count will cease to change when the user is not providing input to the program. When the count reaches a predetermined limit, the user is prevented from continuing use of the program.

This invention is a video game cartridge which can be plugged into a video game machine for enabling a user to receive and play a video game for a predetermined number of frames. The cartridge has a receiver for receiving the video program and for receiving a frame count indicating the number of video frames of the video game program that the user is authorized to play. The video program and frame count is then stored in a memory of the cartridge. The cartridge also has a counter which changes the frame count when the user is actively playing the video game program. When the user is not playing the video game program, the counter ceases to change its count. Finally when the counter reaches a predetermined limit, the user is prevented from further playing the video game program.

Brief Description of the Drawings

FIG. 1 schematically illustrates the major components of the video game cartridge along with a video game machine and a remote server.

FIG. 2 is a functional diagram showing the functions of each of the major components of the video game cartridge.

FIG. 3 schematically illustrates the flow chart for the watch "dog mechanism".

Description of the Preferred Embodiment

FIG. 1 illustrates a sample diagram of a electronic game or program renting system setup. The dotted line encloses the portable and programmable game cartridge unit 100 that can be plugged into a video game machine base unit 102, such as Sega Genesis&tm. video game machine, and remotely be connected to a video game server 101 via a modem connection. The connection to the remote video server can be through cable TV, or other telecommunication facilities.

When a video game base unit 102 is powered on, a user could either play a game (or games) stored in the programmable game cartridge 100 or place an order of a new game (either for rental or for purchase) to the game or program server 101. The cartridge 100 contains screen assistance (and voice assistance) to help place an order for a video game program to the server 101.

FIG. 2 illustrates the components of the video game cartridge unit 100. It consists of modem 201, microcontroller 202, flash memory 203 and an interface 204 to the video game base unit 102. The modem 201 performs the interface to the telephone or cable network. It can optionally perform decompression of received game or software if necessary. The received game is stored in flash memory 203. The game comes with its "life" which is indicated by the total number of graphic frames the video game machine 102 is authorized to generate when the game is actively played. For example, the game machine could render game graphics frame by frame at the rate of thirty framers per second.

After the number of graphic frames is exhausted, further playing of the game is prevented by the following mechanism. The flash memory 203 also stores a "watchdog mechanism" which keeps track of the remaining life of the game. An hourglass routine is embedded in the watchdog mechanism which is executed by microcontroller 202. This watchdog mechanism updates and tracks down a specified register in the flash memory 203 with its location randomly determined by the game server 101 in FIG. 1 during the down loading of the game.

The use of expiration date or time for voiding the game is an obvious approach if the video game base unit 102 comes with a timer. Since this patent application assumes a game base unit 102 which has no timer (which is the case of many existing game machines), the "life" of the rented game is determined by the total number of graphic frames that the base game unit can generate. This "life", or frame count, is what a renter gets when a game is down loaded. It is stored into a location in the flash memory 203. The location into which the frame count is stored in the flash memory is determined randomly by the video server at the time of the game down loading. The video game can resume at any time when it is being turned on, provided there is available frame count stored in the designated random location. The microcontroller 202 can pick up the frame count and allow the renting period, and thus the game or software, to be continued. As the rented game is being played, the frame count is decremented. When the user turns off the power, the hourglass routine in memory 203 will first store the remaining frame count to a random location in the non- volatile memory 203 and then shut down the game. The rental expires when there is no frame count remaining. The microcontroller 202 will not allow any portion of the game to be played by the game base unit 102 when the frame count reaches zero.

FIG. 3 illustrates the watchdog mechanism embedded with the video game program execution path that contains the hourglass routine which serves as part of the watchdog mechanism which can expire the game. When the user starts the game, the frame count is first fetched (305) and checked (306). If the frame count reaches zero, the game is over even though the game unit still has its power on (306N). If the frame count is still greater than zero (306Y), the scanner continues to monitor the game player's input in playing the video game (307). No active input (307) means the player is not playing the video game, and the scanner continues to monitor the player inputs from the key pad connected to the video game. When there is no active input, the video game will not render any game graphic frames. Therefore, the game program execution path will fall through decisions 308 and 309 and immediately return to continue scanning (307). When the game is not actively played and the player leaves the game machine's power on, the game will be sitting idle without rendering any new graphic frames. The frame count will not be consumed until the player becomes active again in playing the game as detected by the scanner (307 and 308).

If the player's input has been detected as active (307), a check is made to see if graphic rendering is required (309). Graphics rendering is required when the game program determines that the input signals from the key pad connected to the video game are valid signals. If rendering is required (309Y), the frame counter will be decremented (301). The hourglass routine (301 and 302) decrements the frame count and checks for any frame count left.

If the count is valid (302Y), then the program flows back to (310) which is the game program main collections, and then at the same time, 302 Y:sup.':esup. branches to check for power-off condition (303).

If the user decides to power-off the game, the watchdog mechanism will go through decision (303) and the shutdown routine (304) to store any remaining frame count in the flash memory. The shutdown routine stores the remaining frame count in the flash memory and exits the game. In summary flowchart components (301-306) and their associated flash memory form the "watchdog mechanism" that contains the hourglass rou-

tine (301 and 302) to keep track of the games "life" (remaining frame count). The watchdog mechanism also insures that the game can be resumed if there is still a valid frame count in the flash memory. Microcontroller (202) can also give advance warning when the rental is about to expire. Rental extension, if desired, can be downloaded again by the server (101) through a telephone or cable connection. Thus, server (101) in FIG. 1 has complete control over the game playing time, which should reflect the user's request for renting the game.

Although this embodiment was described in terms of a video game program in a cartridge, this invention can be extended to software programs in general. As long as the programs monitor user inputs, a scanner and watchdog mechanism can be implemented in similar fashion using a non-volatile memory.

The watchdog mechanism can even be made more secure by encrypting the frame count, which is stored at a random location in the memory. Even if the would-be pirate stumbles across the count in the memory, he/she wouldn't know what he/she found.

Claims

1. An apparatus for enabling a user to request and use a program, said apparatus comprising:

   a. a receiver for receiving the program and a frame count indicating a number of frames of the program that is authorized to be executed by the user;

   b. a memory for storing the program and the frame count received by the receiver; and

   c. a counter for changing the frame count when the user is actively providing input to the program, wherein the counter ceases to change its count when the user is not providing input to the program, and wherein the user is prevented from continuing use of the program when the counter reaches a predetermined limit.

2. An apparatus as recited in claim 1, further comprising:

   means for randomly determining an address in the memory in which the frame count is to be stored, and wherein the address is unknown to the user.

3. A method of enabling a user to request and use a program, said method comprising:

   a. receiving the game program and a frame count indicating a number of frames of the program that is authorized to be used by the user in response to a request;

   b. a memory for storing the program and the frame count; and

   c. changing the frame count when the user is actively using the program, wherein the frame count ceases to change when the user is not using the program and wherein the user is prevented from continuing use of the program when the counter reaches a predetermined limit.

4. A method as recited in claim 3, wherein the frame count is stored in a randomly determined location in the memory.

5. A video game cartridge which can be plugged into, for operation with, a video game machine to enable a user to request and play a video game program which is received from a remotely located server, said video game cartridge comprising:

   a. a receiver for receiving from the server the video game program and a frame count indicating a number of frames of the video game program that is authorized to be played by the user in response to a request;

   b. a memory for storing the video game program and the frame count received by the receiver; and

   c. a counter for changing the frame count when the user is actively playing the video game program, wherein the counter ceases to change its count when the user is not playing the video game program, and wherein the user is prevented from further playing the video game program when the counter reaches a predetermined limit, indicating that the user has played said video game for the number of frames.

6. A video game cartridge as recited in claim 5, further comprising:

   means for randomly determining an address in the memory in which the frame count is to be stored.

7. A video game cartridge as recited in claim 5, further comprising:

   a modem for transmitting to the server the request from the user to play a video game program.

8. A video game cartridge, as recited in claim 5, wherein said memory is a non-volatile memory.

9. A video game cartridge, as recited in claim 8, wherein the frame count indicated in the counter is stored in the memory when power for the video game machine is turned off.

10. A video game cartridge, as recited in claim 9, further comprising:

a means for fetching the frame count stored in the memory when power for said game machine is turned on.

11. A video game cartridge which can be plugged into, for operation with, a video game machine to enable a user to request and play a video game program which is received from a remotely located server, said video game cartridge comprising:

a. a modem for transmitting from the user over a telephone or cable network a request to receive the video game from the server, and for receiving the video game program and frame count from the server over the telephone or cable network, the frame count indicating a predetermined number of frames of the video game program that is authorized to be played by the user in response to the request;

b. a non-volatile memory for storing the video game program and the frame count;

c. a counter for changing the frame count when the player is actively playing the video game;

d. a means for storing the changed frame count of the counter in the memory when the power to the video game machine is turned off; and

e. a means for fetching the changed frame count stored in the memory in step (d) when the player resumes playing the video game, wherein the user is prevented from further playing of the video game program when the frame count of the counter reaches a predetermined limit, indicating that the user has played said video game for the predetermined number of frames.

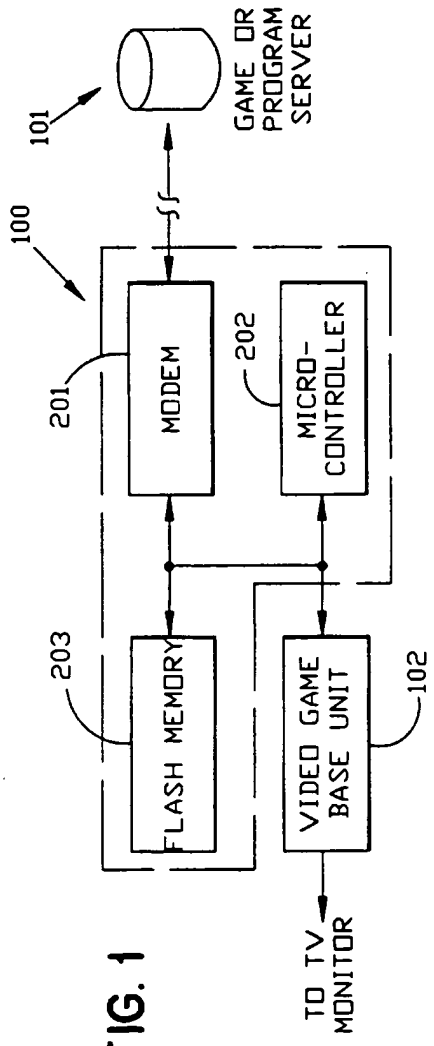


FIG. 1

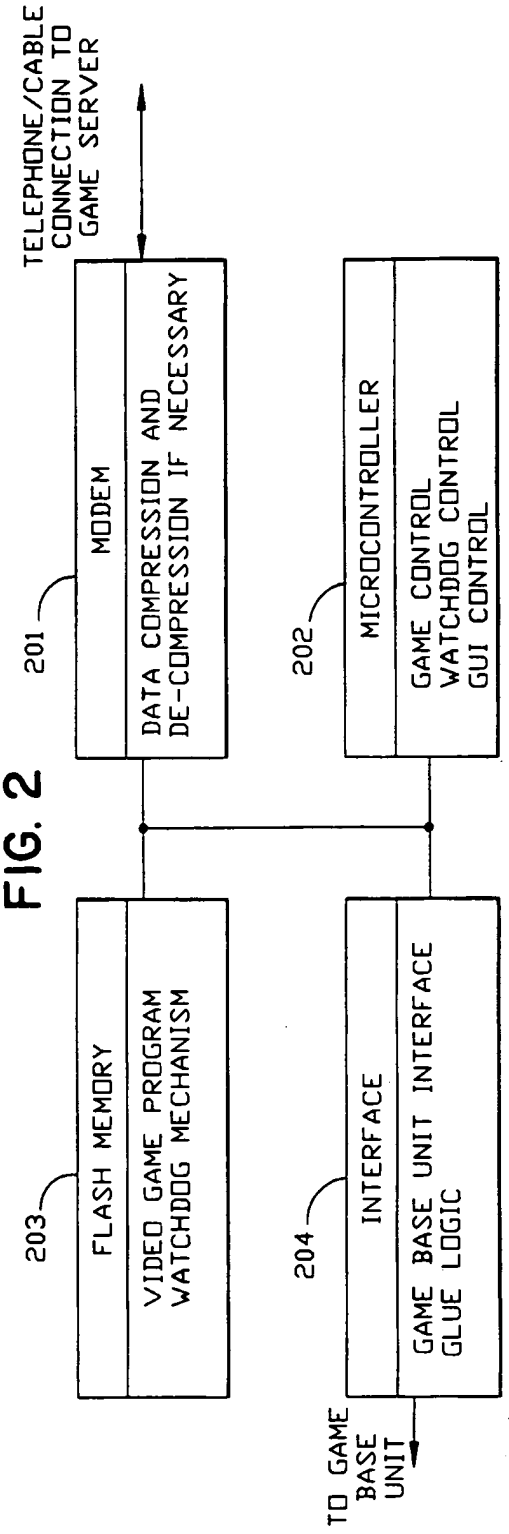
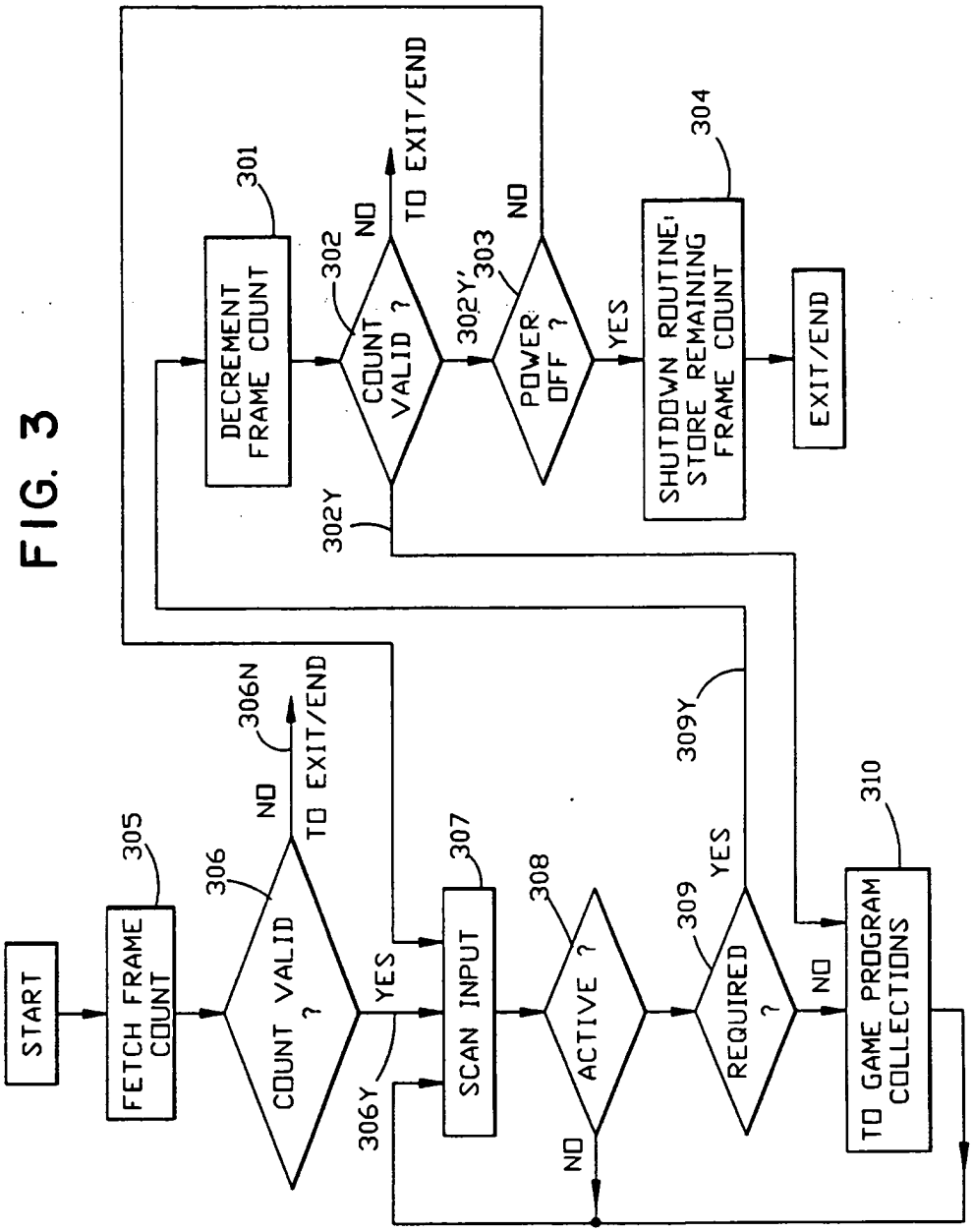


FIG. 2

FIG. 3



European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 96 10 0832

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
P,A	EP-A-0 671 711 (SEGA ENTERPRISES KK) 13 September 1995 * the whole document * ---	1-11	G06F1/00 G06F19/00
A	IBM TECHNICAL DISCLOSURE BULLETIN, vol. 37, no. 3, 1 March 1994, pages 413-417, XP000441522 "MULTIMEDIA MIXED OBJECT ENVELOPES SUPORTING A GRADUATED FEE SCHEME VIA ENCRYPTION" * page 413, line 1 - page 414, line 14 * ---	1-11	
A	WO-A-93 01550 (INFOLOGIC SOFTWARE INC) 21 January 1993 * page 1, line 1 - page 8, line 32 * * claims 1-3 * -----	1-11	
			TECHNICAL FIELDS SEARCHED (Int.Cl.6) G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	14 June 1996	Powell, D

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 763 936 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
19.03.1997 Bulletin 1997/12

(51) Int Cl.6: H04N 5/913

011

(21) Application number: 96306507.3

(22) Date of filing: 06.09.1996

(84) Designated Contracting States:
DE FR GB

(30) Priority: 18.09.1995 KR 9530444

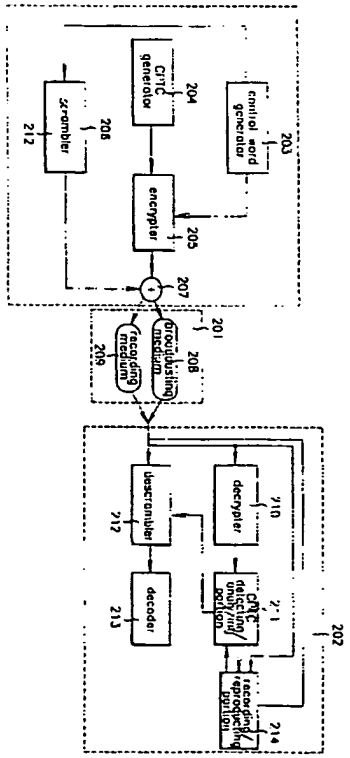
(71) Applicant: LG ELECTRONICS INC.
Seoul (KR)

(72) Inventors:
- Kim, Yung Gil, c/o LG Elec. Video-Media R&D
Seoul (KR)
- Park, Tae Joon
Seoul (KR)

(74) Representative:
Cross, Rupert Edward Blount et al
BOULT WADE TENNANT
27 Furnival Street
London EC4A 1PQ (GB)

(54) Illegal view/copy protection method and apparatus for digital broadcasting system

(57) An illegal view/copy protection method for a digital broadcasting system is disclosed including an audio/video signal transmission step (200,201) for multiplexing and transmitting audio/video bit stream scrambled in control words (206) and information where the control words and CPTC information for illegal view/copy protection are encrypted (208); and an audio/video reception step (202) for decrypting (210) the transmitted bit stream to analyze the CPTC information and control words (211), deciding whether recording is allowed or not to be recorded on cassette tape, and using the control words, performing descrambling (212) and decoding (213) to output audio/video signals to a monitor, thereby protecting copyright.



F I G.16

EP 0 763 936 A2

Printed by Jouve, 75001 PARIS (FR)

Description

Background of the Invention

The present invention relates to an illegal view/copy protection method and apparatus for a digital broadcasting system, in which digital broadcasting performed through broadcasting media such as cable, satellite and terrestrial broadcasting, or through prerecorded media such as video cassette tapes, is prevented from being illegally viewed or copied to thereby protect its copyright.

For conventional systems for copyright protection on digital media, there are Macrovision's intellectual property protection system (IPPS), which is disclosed in US Patent No. 5,315,448, and the integrated receiver/decoder (IRD), a conditional receiving system for digital broadcasting media, for receiving DirecTV's satellite broadcasting currently transmitted in the US.

The Macrovision's IPPS disclosed in US Patent No. 5,315,448 is a copy protection system for a hybrid digital VCR having digital recording functions for both a digital input signal and an analog input signal.

As shown in Figs. 1 and 2, in operating its copy protection function, Macrovision's IPPS detects, when a digital signal is input, copy protection control bits from an input signal, and when an analog signal is input, detects the analog copy protection waveform from the input signal.

More specifically, as shown in Fig. 2, a signal in which the analog copy protection waveform generated from an analog copy protection generator is added to the analog video output of the output signals of the digital VCR is output and displayed to be normal on an analog TV but distorted on an analog VCR, as shown in Fig. 1. In digital recording of the input signal, the copy protection control bits are changed to prevent digital copy or to permit one-time digital copy.

Referring to Fig. 3, the IPPS comprises an analog copy protection detector (ACP) 2 for detecting the analog copy protection waveform from an input analog NTSC video signal 1, an A/D converter 3 for A/D-converting analog NTSC video signal 1 input according to the signal output from the ACP detector, an AC bit detector 5 for detecting the AC bit from input digital video signal 4, an SCPS bit detector 6 for detecting the SCPS from input digital video signal 4, an AC bit adder 7 for adding the AC bit to input digital video signal 4 according to the SCPS bit output from SCPS bit detector 6, a switch 8 for outputting a signal output from AC bit adder 7 according to the AC bit output from AC bit detector 5, a switch 9 for selecting and outputting the signal output from A/D converter 3 and switch 8, a digital tape deck mechanism/circuit 10 for digitally recording the signal output from switch 9 and outputting a digital video signal, an AC bit detector 11 for detecting the AC bit from the signal output from digital tape deck mechanism/circuit 10, an ACP signal generator 12 for generating the ACP signal from the signal output from AC bit detector 11, and a D/A converter 13 for adding the ACP signal output from ACP signal generator 12 to the signal output from digital tape deck mechanism/circuit 10 and D/A converting the added result which is output as an analog NTSC video signal.

The operation of the IPPS will be explained below.

The copy protection control bits are made up of the AC and SCPS bits. The AC bit is added to recorded digital video data so that if the AC bit is set, digital copy is prohibited and if the SCPS bit is set, one-time digital copy is allowed.

In playback, when the AC bit is detected by AC bit detector 11, the analog copy protection waveform generated from ACP signal generator 12 is added to the analog video signal, which is output to D/A converter 13. Here, as the position of the copy protection control bits of the digital video data, an area of an MPEC-2 digital copy protection header where one-bit copyright flag and one-bit original-or-copy flag of a PES header are placed is used, or a transport-private-data field area of the transport header of the MPEC-2 is used.

The analog copy protection waveform is a signal which is severely distorted when inserted into the analog NTSC waveform and directly coupled to the analog TV. A method of generating such a signal is presented in US Patents Nos. 4,613,603 and 4,914,694. Using this method, the IPPS generates the analog copy protection waveform.

Referring to Fig. 4, the IRD, as a conditional receiving system for digital broadcasting media, for receiving the DirecTV's satellite broadcasting currently transmitted in US comprises an outdoor unit (ODU) 21 made up of a satellite antenna for receiving 12GHz-satellite broadcasting signals and a low noise block converter (LNB) for converting down the received satellite broadcasting signal into a 1GHz-signal, an IRD 20 for receiving satellite broadcasting from ODU 21 and offering audio and video services to a subscriber's TV or monitor, and an access card 22 required for conditional access (CA) for conditional reception.

Here, IRD 20 performs forward error correction (FEC), decoding, transport demultiplexing, MPEG decoding, NTSC encoding, and audio processing which is a D/A conversion.

Access card 22, whose size is similar to that of a general credit card, has a built-in IC. With this, the card receives CA-related information through a broadcast bit stream and telephone line, that is, a telco MODEM, in order to decide whether a user, subscriber, -selected channel can be viewed or not and to collect its subscription fee.

As shown in Fig. 4, IRD 20 comprises an IR receiver 25 for receiving and processing the subscriber's remote controller input, a telco MODEM 26 which is a general MODEM coupled to the telephone line, a microcomputer 27 made up of an NDC verifier code including software for the CA function and IRD software for IRD driving, a tuner/demodulator/FEC 28 for selecting one channel of

the signal received through ODU 21 and converting the selected channel into a digital bit stream for the purpose of error correction, a transport IC 29 for selecting one program of bit streams output from tuner/demodulator/FEC 28 and multiplexed with various programs, and converting the selected program into a bit stream decodable in the MPEG video decoder and MPEG audio decoder, a card reader interface 23 for data communication between transport IC 29 and access card 22, a system memory 24 coupled to transport IC 29 and for intermediate buffering of data, an MPEG video decoder 30 for expanding a video bit stream compressed in the MPEG format, a frame memory 31 for storing video data expanded in MPEG video decoder 30 in units of frame, an encode/sync/anti-tape/D/A 33 for converting the digital video data expanded in MPEG video decoder 30 into the analog NTSC format and inserting horizontal and vertical sync signals H-Sync and V-Sync and a Macrovision-mode analog copy protection signal in the conversion process, an RF modulator 34 for modulating an NTSC signal of the baseband output from encode/sync/anti-tape/D/A 33 into the RF band, an MPEG audio decoder 32 for expanding the audio bit stream compressed in the MPEG format, and a D/A 35 for converting the expanded digital audio data output from MPEG audio decoder 32 into analog.

Here, in the procedure of conversion into decodable bit stream in the MPEG video and audio decoders from transport IC 29, it is decided whether a program selected through communication with access card 22 can be viewed or not. If the bit stream is scrambled, its descrambling is performed with the access card's permission.

During the process of encode/sync/anti-tape/D/A 33 prior to NTSC video output, the analog copy protection waveform is added to prohibit copying to the analog VCR.

IRD 20 employs a CA system for conditional reception so that a subscriber views programs provided through a broadcasting medium such as satellite broadcasting.

In IRD 20, the NDC verifier code, which is software, and access card 22, which is a smart card for CA, are used to support CA function. A descrambler 36 is contained in transport IC 29.

The detailed block diagram of CA unit 37 and transport IC 29 for operating the CA function in a manner generally used in digital broadcasting is shown in Fig. 5.

More specifically, CA unit 37, included in smart card 22, is made up of smart card 38 for CA and microcomputer 39 operated with CA software.

The CA function is performed when the following two kinds of data are transmitted from a broadcasting station to the IRD. In other words, there are two types of data such as entitlement control message (ECM) or control word packet (CWP), and entitlement management message (EMM) or conditional access packet (CAP).

The EMM is accessed, through the telephone line or satellite broadcasting, to the smart card of the respective IRD at the data rate of 200kbps. The broadcasting station can access all of subscribers' smart cards in a manner that the EMM is transmitted along with ID or address. The EMM has information required to make a control word (CW) for descrambling from the ECM information. The ECM, information in which the control word is encrypted, is transmitted at a speed over 10 per second.

For satellite broadcasting, there are Europe's DVB, Korea's DBS, US' echoster, and the like, aside from DirecTV. Their CA function commonly uses the ECM and EMM information, though different means is provided for the respective broadcastings.

The conventional Macrovision's IPPS is a system having a good performance with respect to the copy protection of analog NTSC video signal. This is an appropriate copyright protection means when a program supplied through a digital medium is converted into analog audio/video signal and recorded or copied through an analog VCR.

However, the IPPS cannot guarantee a satisfactory protection if digital data is recorded or copied using a digital recording medium such as digital VCR. This is because the IPPS uses a method of operating the header's flag bits, without employing, to digital data, encoding methods such as scrambling and encryption. By doing so, hacking is easy to perform only by modulating the flag bits, resulting in very low security.

Summary of the Invention

It would therefore be desirable to provide an illegal view/copy protection method and apparatus for a digital broadcasting system in which intellectual properties supplied via digital media and protected by copyright are prohibited from being illegally recorded or copied using a digital recording medium such as digital VCR by a user.

It would also be desirable to provide an illegal view/copy protection method and apparatus for a digital broadcasting system in which data recorded on a cassette tape is always scrambled to make its hacking difficult and protect its copyright.

It would also be desirable to provide an illegal view/copy protection method and apparatus for a digital broadcasting system in which copyright is protected appropriately for respective media which are divided into broadcasting media and pre-recorded media.

It would also be desirable to provide an illegal view/copy protection method and apparatus for a digital broadcasting system in which intellectual properties supplied from a program provider are reproduced to be viewed on screen, copying of the intellectual properties copied and the number of copy are controlled arbitrarily, and fee for recording and copying is collected for the purpose of copyright protection.

According to a first aspect of the present invention, there is provided an illegal view/copy protection method for a digital broadcasting system comprising: an audio/video signal transmission step for multiplexing and transmitting audio/video bit stream scrambled in control words and information where the control words and CPTC information for illegal view/copy protection are encrypted; and an audio/video reception step for decrypting the transmitted bit stream to analyze the CPTC information and control words, deciding whether recording is allowed or not to be recorded on cassette tape, and using the control words, performing descrambling and decoding to output audio/video signals to a monitor.

According to a second aspect of the present invention, there is provided an illegal view/copy protection apparatus for a digital broadcasting system comprising: a program producing portion for multiplexing information encrypted both with the control word for scrambling and the CPTC information for prohibiting illegal view/copy, and the audio/video bit stream scrambled in control words, to thereby make a program; a distribution medium portion for distributing programs made in the program producing portion through a transmission medium; and a program receiving portion for detecting and analyzing the CPTC information from the bit stream transmitted from the distribution medium portion and the bit stream reproduced from cassette tape, and descrambling and decoding the bit stream transmitted from the distribution medium portion.

Brief Description of the Attached Drawings

Figs. 1 and 2 illustrate the operation state of a conventional IPPS;

Fig. 3 is a block diagram of a conventional IPPS;

Fig. 4 is a block diagram of an IRD system;

Fig. 5 shows a configuration of general hardware performing CA function;

Figs. 6A and 6B show formats of CPTC information of an embodiment of the present invention;

Fig. 7 shows a state of generation copy indicating the number of tape recopiable;

Figs. 8A-8D show the recording positions of the CPTC information of an embodiment of the present invention;

Fig. 9 is a flowchart of showing the transmission step of an illegal view/copy protection method embodying the present invention;

Fig. 10 is a flowchart of showing the reception step of an illegal view/copy protection method embodying the present invention;

Fig. 11 is a flowchart of the CPTC information analyzing step of Fig. 10;

Fig. 12 is a flowchart of showing the reproduction/rerecording step of an illegal view/copy protection method embodying the present invention;

Fig. 13 shows the format of an EMM lookup table;

Fig. 14 shows the format of a tape state signal;

Fig. 15 is a flowchart of showing the EMM processing step;

Fig. 16 is a block diagram of the whole configuration of an illegal view/copy protection apparatus embodying the present invention;

Fig. 17 is a block diagram of one embodiment of the program receiving portion of Fig. 16;

Fig. 18 is a block diagram of another embodiment of the program receiving portion of Fig. 16;

Fig. 19 is a block diagram of still another embodiment of the program receiving portion of Fig. 16;

Fig. 20 is a block diagram of yet another embodiment of the program receiving portion of Fig. 16;

Fig. 21 is a block diagram of the IRD shown in Figs. 17, 19 and 20;

Fig. 22 is a block diagram of the IRD and DVCR of Fig. 18;

Fig. 23 illustrates the flow of signals of Fig. 21;

Fig. 24 is a block diagram of one embodiment of the smart card of Fig. 17;

Fig. 25 is a block diagram of another embodiment of the smart card of Fig. 17; and

Fig. 26 is a block diagram of the DVCR of Fig. 17.

Detailed Description of the Invention

An illegal view/copy protection method for a digital broadcasting system embodying the present invention is performed by audio/video signal transmission and audio/video reception steps.

In the audio/video signal transmission step, audio/video bit stream scrambled in control words and information where the control words and CPTC information for illegal view/copy protection are encrypted are multiplexed and transmitted.

In the audio/video reception step, the bit stream transmitted in the audio/video signal transmission step is decrypted to analyze the CPTC information and control words. By doing so, it is decided whether recording is allowed or not. This result is recorded on cassette tape. Using the control words, descrambling and decoding are performed, and then audio/video signals are output to a monitor. Here, the CPTC information separately manages the ECM, EMM and control words, and contains CA information, to thereby control illegal view/copy protection. The CPTC information will be described with reference to Figs. 6A and 6B.

The CPTC information is formatted in a generational copy control field for limiting the number of copy available in order to control the depth of generational copy, and a reproducibility control field for limiting the reproduction of a copied program in order to control the number of copyiable tapes. As shown in Fig. 6A, formatting is performed containing a descrambling information field where part of the control words for descrambling are recorded, or containing a CA field where CA information for conditional access is recorded, as shown in Fig. 6B.

The CPTC information may be encrypted separately to be multiplexed with scrambled digital data, or contained in the ECM information for CA for encryption and multiplexing. Here, the generational copy control field is made up of a permissible generational field for limiting the number of copy permissible and a present generational field for indicating the present generation of a program copied. If the present generation stored in the present generational field is greater than or equal to the permissible generation stored in the permissible generational field, recording or copying is impossible.

A reproduction control field is made up of a reproducible number field for limiting the number of reproducing a copied program, and a maximum reproducible time field for limiting time to reproduce the copied program.

Here, the reproducible number stored in the reproducible number field implements a conditional-number reproducibility function according to the current reproduction number of cassette tape. The maximum reproducible time stored in the maximum reproducible time field implements the conditional-time reproducibility function of copied cassette tape according to the current time information of digital hardware.

The CPTC information may allow the copied cassette tape to be always reproducible, make it never reproducible, allow it to be reproducible as many as a limited number, or make the copied cassette tape reproducible for a limited time after recording or copying.

Using the permissible generational field and present generational field of the generational copy control field, the reproducible number field of the reproduction control field, and data of the maximum reproducible time field, the depth of generation copy, recopying of copied cassette tape, and reproduction time and number are controlled. This process controls the number of copiable cassette tape copied, and reproduction time and number.

In other words, as shown in Fig. 7, information stored in the permissible generational field and present generational field is used to allow first and second generation copy to be perform. Information stored in the reproducible number field and maximum reproducible time field is used to allow reproduction as many as a limited number or for a limited time.

In order to prohibit illegal recording or copy of a program protected by copyright law, collect fee for recording or copy, or arbitrarily control the number of reproducible copied tape to be made from a program supplied by a provider, the depth of generation copy and reproduction of copy tape are controlled to decide how long the first generation recording and copy and second generation copy are made possible.

For this purpose, the copy tape made to be always reproducible, it is made never to be reproducible, it is made to be reproducible as many as a limited number, or it is made to be reproducible for a limited time after recording or copy.

The data recorded on cassette tape contains scrambled audio/video bit stream and CPTC information. The CPTC information is recorded on a recording medium, that is, a rental tape, to prohibit illegal view/copy.

In other words, as shown in Fig. 8A, the CPTC information is overwritten on the scrambled audio/video bit stream for the error effect and recorded on cassette tape. Otherwise, as shown in Fig. 8B, the CPTC information is recorded on a portion of the audio track of cassette tape, on the control track of cassette tape as shown in Fig. 8C, or on the video track of cassette tape as shown in Fig. 8D.

In other words, as shown in Fig. 8A, the CPTC information is overwritten in a predetermined position in the form of error after parities for error correction, that is, inner and outer parities, are added to the scrambled digital data. This method reduces error correction capability but requires no additional tape area for recording the CPTC information. Further, during interleaving and decoding of ECC, the CPTC information is recognized as an error and removed, obtaining the scrambled digital data. Here, the CPTC information is detected separately.

In case that the CPTC information is recorded in part of audio track or control track, as shown in Figs. 8B and 8C, the audio head or control head must be additionally used as the means for detecting the CPTC so that audio track and control track are additionally accessed to detect the CPTC information.

The audio/video signal transmission step using the CPTC information will be explained with reference to Fig. 9.

One embodiment of the audio/video signal transmission step is to transmit an audio/video signal not containing the CA information for conditional access. This, having only the copy protection function, is used in case that a program which can be provided to all viewers is transmitted.

As shown in Fig. 9, the first embodiment of the audio/video signal transmission step comprises the steps of: encoding (100) the audio/video bit stream; generating (105) a control word for scrambling; scrambling (104) for the encoded audio/video bit stream using the generated control word; generating (102) CPTC information for illegal view/copy protection; encrypting (103) for encrypting the control word and CPTC information; and multiplexing and transmitting (106) the scrambled audio/video bit stream and encrypted CPTC information.

In other words, in step 100, the audio/video bit stream is encoded. In step 105, the control word for scrambling is generated. In step 104, the encoded audio/video bit stream is scrambled using the generated control word. In step 102, the CPTC information for illegal view/copy protection is generated. In step 103, the CPTC information and CA information are encrypted using the generated control word. The scrambled audio/video bit stream, encrypted CPTC information and CA

information are multiplexed and transmitted through a transmission medium in step 106. The audio/video signal transmitted through the first embodiment of the audio/video signal transmission step is received through one embodiment of an audio/video reception step.

Referring to Fig. 10, the first embodiment of the audio/video reception step comprises the steps of filtering (110) the transmitted bit stream and decrypting (111) the CPTC information; analyzing (113 and 114) the CPTC information to generate a control word and a signal for controlling the protection of copyright and to update the CPTC information; deciding (115) whether to allow recording according to the signal for controlling the protection of copyright to record the scrambled and transmitted bit stream on cassette tape; and descrambling and decoding (116 and 117) the transmitted bit stream in the control word and outputting an audio/video signal.

In other words, the bit stream transmitted in the first embodiment of the audio/video signal transmission step is filtered and the CPTC information is decrypted in steps 110 and 111. The CPTC information is analyzed to generate the control word and the signal for controlling the protection of copyright, and the CPTC information is updated in steps 113 and 114. Whether to allow recording is determined by the generated signal for controlling the protection of copyright so that the scrambled and transmitted bit stream is recorded on cassette tape in step 115. Then, the transmitted bit stream is descrambled and decoded in control words and output as an audio/video signal in steps 116 and 117. Here, all of the control word is contained in the CPTC information.

Referring to Fig. 11, the CPTC information analyzing step comprises the steps of detecting (130, 131, 132 and 133) the permissible generation of the permissible generational field for limiting the available number of copy of a program of the CPTC information and the present generation of the present generational field indicating the present generation of the program copied, to thereby perform copy-impossible and update the CPTC information; and detecting (134, 135, 136 and 137) the reproducible number of the reproducible number field for limiting the number of reproduction of copied programs of the CPTC information, the maximum reproducible time of the maximum reproducible time field for limiting time to reproduce the copied program, and the number and time of reproduction of tape, to thereby process reproduction-impossible.

The copying number limiting step comprises the steps of: comparing (130) the permissible generation of the permissible generational field and the present generation of the present generational field and deciding whether the permissible generation is below the present generation; if the permissible generation is below the present generation, generating (131) an output disable signal to make copying impossible and destroying the control word; and if the permissible generation is not below the present generation, increasing (132) the present invention by '1' and recording the result on cassette tape. If the permissible generation is not below the present generation, the CPTC information is updated in step 133, instead of increasing the present generation by '1.'

In order to control generation copy, the permissible generation of the permissible generational field and the present generation of the present generational field are compared in step 130. If the permissible generation is below the present generation, the output disable signal is generated to make copying impossible and the control word is destroyed in step 131. If the permissible generation is not below the present generation, the present generation is increased by '1' and thus recorded on cassette tape in step 132. This enables generation copy. Here, it can be possible that generation copy is limited by updating the CPTC information, instead of increasing the present generation by '1.'

The reproduction limiting step comprises the steps of: comparing the reproducible number of the reproducible number field and the reproduction number of tape and deciding (134) whether the reproducible number is below the reproduction number of tape; if the reproducible number is not below the reproduction number of tape, comparing the maximum reproducible time and reproduction time of tape, and deciding (135) whether the maximum reproducible time is below the reproduction time of tape; if the maximum reproducible time is not below reproduction time of tape, turning off (136) an enable erase signal to thereby enable the copied program to be reproduced; if the reproducible number is below the reproduction number of tape or the maximum reproducible time is below the reproduction time of tape, turning on (137) the enable erase signal to make the reproduction of the copied program impossible so that part of or the whole program recorded on cassette tape is erased.

In order to control reproduction, the reproducible number of the reproducible number field and the reproduction number of tape are compared in step 134. If the reproducible number is not below the reproduction number of tape, the maximum reproducible time of the maximum reproducible time field and the reproduction time of tape are compared and it is decided whether the maximum reproducible time is below the reproduction time of tape in step 135. In other words, though reproducible, whether it is limited by the reproducible time must be checked. If the maximum reproducible time is not below the reproduction time of tape, the enable erase signal is turned off in step 136 to thereby make the copied program reproducible. If the reproducible number is below the reproduction number of tape or the maximum reproducible time is below the reproduction time of tape, the enable erase signal is turned on to prohibit the reproduction of the copied program. By doing so, part of or the whole program recorded on cassette tape is erased to make copy and reproduction impossible in step 137.

Here, the current time is transmitted to the user by

a provider along with a program. In this case, the copyright protection system implements limited time reproduction using transmitted time information. In this method, the program provider manages the whole users' time so that time modulation by a user cannot occur. Therefore, this is very secure.

The bit stream transmitted in the first embodiment of the audio/video signal transmission step contains ECM and EMM. Part of the control word may be contained in the CPTC information. Its remainder may be contained in the ECM or EMM. The whole control word is contained in the ECM or EMM.

The audio/video signal containing the control word and transmitted according to the audio/video signal transmission step is received according to another embodiment of the audio/video reception step.

Referring to Fig. 10, the second embodiment of the audio/video reception step comprises the steps of filtering (110) the transmitted bit stream and decrypting (111) the CPTC information and control word; filtering (118) the control word; analyzing (113 and 114) the CPTC information to generate a control word and a signal for controlling the protection of copyright and to update the CPTC information; deciding (115) whether to allow recording according to the signal for controlling the protection of copyright to record the scrambled and transmitted bit stream on cassette tape; and descrambling and decoding (116 and 117) the transmitted bit stream in control words and outputting an audio/video signal.

In other words, the bit stream transmitted in the audio/video signal transmission step is filtered and the CPTC information and control word are decrypted in steps 110 and 111. The control word is filtered in step 118. The decrypted CPTC information is analyzed to generate the control word and the signal for controlling the protection of copyright, and the CPTC information is updated in steps 113 and 114. Whether to allow recording is determined by the generated signal for controlling the protection of copyright so that the scrambled and transmitted bit stream is recorded on cassette tape in step 115. Then, the transmitted bit stream is descrambled and decoded in control words and output as an audio/video signal in steps 116 and 117.

Referring to Fig. 11, in the same manner as the first embodiment of the audio/video reception step, the CPTC information analyzing step comprises the steps of: generating the control words; detecting (130, 131, 132 and 133) the permissible generation of the permissible generational field for limiting the available number of copy of a program of the CPTC information and the present generation of the present generational field indicating the present generation of the program copied, to thereby perform copy-impossible and update the CPTC information; and detecting (134, 135, 136 and 137) the reproducible number of the reproducible number field for limiting the number of reproduction of copied programs of the CPTC information, the maximum reproducible time of the maximum reproducible time field for limiting time to reproduce the copied program, and the number and time of reproduction of tape, to thereby process reproduction-impossible.

The copying number limiting step comprises the steps of: comparing (130) the permissible generation of the permissible generational field and the present generation of the present generational field and deciding whether the permissible generation is below the present generation; if the permissible generation is below the present generation, generating (131) an output disable signal to make copying impossible and destroying the control word; and if the permissible generation is not below the present generation, increasing (132) the present invention by '1' and recording the result on cassette tape. If the permissible generation is not below the present generation, the CPTC information is updated in step 133, instead of increasing the present generation by '1.'

The reproduction limiting step comprises the steps of: comparing the reproducible number of the reproducible number field and the reproduction number of tape and deciding (134) whether the reproducible number is below the reproduction number of tape; if the reproducible number is not below the reproduction number of tape, comparing the maximum reproducible time and reproduction time of tape, and deciding (135) whether the maximum reproducible time is below the reproduction time of tape; if the maximum reproducible time is not below reproduction time of tape, turning off (136) an enable erase signal to thereby enable the copied program to be reproduced; if the reproducible number is below the reproduction number of tape or the maximum reproducible time is below the reproduction time of tape, turning on (137) the enable erase signal to make the reproduction of the copied program impossible so that part of or the whole program recorded on cassette tape is erased.

Another embodiment of the audio/video signal transmission step is to transmit an audio/video signal containing the CA information for conditional access. This, having the illegal reception and copy protection functions, is used in case that a program which can be provided to limited viewers is transmitted.

As shown in Fig. 9, the second embodiment of the audio/video signal transmission step comprises the steps of: encoding (100) the audio/video bit stream; generating (105) a control word for scrambling; scrambling (104) for the encoded audio/video bit stream using the generated control word; generating (102) CPTC information for illegal view/copy protection; generating (101) CA information for conditional reception; encrypting (103) for encrypting the CPTC information and CA information; and multiplexing and transmitting (106) the scrambled audio/video bit stream and encrypted CPTC information and CA information.

In other words, in step 100, the audio/video bit stream is encoded. In step 105, the control word for scrambling is generated. In step 104, the encoded au-

dio/video bit stream is scrambled using the generated control word. In step 102, the CPTC information for illegal view/copy protection is generated. In step 101, CA information for conditional reception is generated. In step 103, the CPTC information and CA information are encrypted using the generated control word. The scrambled audio/video bit stream, encrypted CPTC information and CA information are multiplexed and transmitted through a transmission medium in step 106. The audio/video signal transmitted through the second embodiment of the audio/video signal transmission step is received through the second embodiment of the audio/video reception step.

Referring to Fig. 10, the second embodiment of the audio/video reception step comprises the steps of: filtering (110) the transmitted bit stream and decrypting (111) the CPTC information; analyzing (112, 113 and 114) the CPTC information and CA information to generate a control word and a signal for controlling the protection of copyright and to update the CPTC information; deciding (115) whether to allow recording according to the signal for controlling the protection of copyright to record the scrambled and transmitted bit stream on cassette tape; and descrambling and decoding (116 and 117) the transmitted bit stream and outputting an audio/video signal.

Referring to Fig. 11, in the same manner as the first embodiment of the audio/video reception step, the CPTC information analyzing step comprises the steps of: generating a control word; detecting (130, 131, 132 and 133) the permissible generation of the permissible generational field for limiting the available number of copy of a program of the CPTC information and the present generation of the present generational field indicating the present generation of the program copied, to thereby perform copy-impossible and update the CPTC information; and detecting (134, 135, 136 and 137) the reproducible number of the reproducible number field for limiting the number of reproduction of copied programs of the CPTC information, the maximum reproducible time of the maximum reproducible time field for limiting time to reproduce the copied program, and the number and time of reproduction of tape, to thereby process reproduction-impossible.

In the same manner as the first embodiment of the audio/video reception step, the copying number limiting step comprises the steps of: comparing (130) the permissible generation of the permissible generational field and the present generation of the present generational field and deciding whether the permissible generation is below the present generation; if the permissible generation is below the present generation, generating (131) an output disable signal to make copying impossible and destroying the control word; and if the permissible generation is not below the present generation, increasing (132) the present invention by '1' and recording the result on cassette tape. If the permissible generation is not below the present generation, the CPTC information is updated in step 133.

The reproduction limiting step comprises the steps of: comparing the reproducible number of the reproducible number field and the reproduction number of tape and deciding (134) whether the reproducible number is below the reproduction number of tape; if the reproducible number is not below the reproduction number of tape, comparing the maximum reproducible time and reproduction time of tape, and deciding (135) whether the maximum reproducible time is below the reproduction time of tape; if the maximum reproducible time is not below reproduction time of tape, turning off (136) an enable erase signal to thereby enable the copied program to be reproduced; if the reproducible number is below the reproduction number of tape or the maximum reproducible time is below the reproduction time of tape, turning on (137) the enable erase signal to make the reproduction of the copied program impossible so that part of or the whole program recorded on cassette tape is erased.

The bit stream transmitted in the second embodiment of the audio/video signal transmission step contains ECM and EMM. Part of the control word may be contained in the CPTC information. Its remainder may be contained in the ECM or EMM. The whole control word is contained in the ECM or EMM.

The audio/video signal containing the control word and transmitted according to the audio/video signal transmission step is received according to another embodiment of the audio/video reception step. The audio/video signal transmitted in the audio/video signal transmission step containing the control word is received according to still another embodiment of the audio/video reception step.

Referring to Fig. 10, the third embodiment of the audio/video reception step comprises the steps of: filtering (110) the transmitted bit stream and decrypting (111) the CPTC information and CA information; analyzing (112, 113, 114 and 118) the CPTC information and CA information and filtering the control word to generate a control word and a signal for controlling the protection of copyright and to update the CPTC information; deciding (115) whether to allow recording according to the signal for controlling the protection of copyright to record the scrambled and transmitted bit stream on cassette tape; and descrambling and decoding (116 and 117) the transmitted bit stream and outputting an audio/video signal.

Referring to Fig. 11, in the same manner as the first embodiment of the audio/video reception step, the CPTC information analyzing step comprises the steps of: generating the control words; detecting (130, 131, 132 and 133) the permissible generation of the permissible generational field for limiting the available number of copy of a program of the CPTC information and the present generation of the present generational field indicating the present generation of the program copied, to thereby perform copy-impossible and update the CPTC information; and detecting (134, 135, 136 and

137) the reproducible number of the reproducible number field for limiting the number of reproduction of copied programs of the CPTC information, the maximum reproducible time of the maximum reproducible time field for limiting time to reproduce the copied program, and the number and time of reproduction of tape, to thereby process reproduction-impossible.

The copying number limiting step comprises the steps of: comparing (130) the permissible generation of the permissible generational field and the present generation of the present generational field and deciding whether the permissible generation is below the present generation; if the permissible generation is below the present generation, generating (131) an output disable signal to make copying impossible and destroying the control word; and if the permissible generation is not below the present generation, increasing (132) the present invention by '1' and recording the result on cassette tape, and if the permissible generation is not below the present generation, updating the CPTC information in step 133.

The reproduction limiting step comprises the steps of: comparing the reproducible number of the reproducible number field and the reproduction number of tape and deciding (134) whether the reproducible number is below the reproduction number of tape; if the reproducible number is not below the reproduction number of tape, comparing the maximum reproducible time and reproduction time of tape, and deciding (135) whether the maximum reproducible time is below the reproduction time of tape; if the maximum reproducible time is not below reproduction time of tape, turning off (136) an enable erase signal to thereby enable the copied program to be reproduced; if the reproducible number is below the reproduction number of tape or the maximum reproducible time is below the reproduction time of tape, turning on (137) the enable erase signal to make the reproduction of the copied program impossible so that part of or the whole program recorded on cassette tape is erased.

The illegal view/copy protection method for digital broadcasting system embodying the present invention, after the audio/video signal transmission step and audio/video reception step, further comprises a reproduction and rerecording step of: decrypting the bit stream recorded and reproduced on cassette tape, analyzing the CPTC information, deciding whether to allow rerecording, recording the result on cassette tape, filtering the control word, and performing descrambling and decoding to output an audio/video signal.

Referring to Fig. 12, the audio/video reproduction and rerecording step comprises the steps of: filtering (120) the bit stream recorded and reproduced on video tape, and decrypting (121) the CPTC information; analyzing (122 and 123) the CPTC information to generate control words and a signal for controlling the protection of copyright and update the CPTC information; deciding (124) whether to allow recording according to the signal of controlling the protection of copyright, and recording the scrambled and transmitted bit stream on cassette tape; descrambling and decoding (125 and 126) the transmitted bit stream in control words to output an audio/video signal; and deciding whether to allow post-reproduction according to the signal for controlling the protection of copyright to thereby erase part of or the whole data recorded on cassette tape.

Here, EMM may contain information required for decoding information in order to perform the illegal view/copy protection method of a broadcasting system. In this case, a step of storing and processing the EMM is added in the audio/video reproduction and rerecording step.

In the EMM storing and processing step, in case that the EMM is updated by a broadcasting station for the purpose of copyright protection, the EMM having information required to decode the CPTC information is stored in order to continuously reproduce programs of copied cassette tape.

Here, an ID number indicative of updating the EMM is recorded on cassette tape. The EMM is stored to which the updating state and the ID number of cassette tape are mapped.

The EMM storing and processing step comprises the steps of: storing all EMM to be updated and corresponding ID information; selecting the latest EMM in recording cassette tape; recording a corresponding ID number; and selecting an EMM corresponding to the ID number recorded on cassette tape in reproducing the cassette tape.

As shown in Fig. 13, all EMMs (EMM1, EMM2, EMM3,...) to be updated on the EMM lookup table and corresponding ID information (ID1, ID2, ID3,...) are mapped and stored.

Referring to Figs. 14 and 15, in recording a program on cassette tape, that is, when recording is indicated in the recording/reproduction mode, an ID number corresponding to the latest, the final, EMM, is recorded. Thereafter, in reproducing the cassette tape, that is, when reproduction is indicated in the recording/reproduction mode, an EMM corresponding to the ID number recorded on cassette tape is selected from the EMM lookup table so that the recorded program is reproduced according to the reproducible number of the reproducible number field and the reproduction number recorded on the video tape.

Referring to Fig. 16, an illegal view/copy protection apparatus of digital broadcasting system embodying the present invention comprises a program producing portion 200, distribution medium portion 201, and program receiving portion 202.

Program producing portion 200 offers programs, in which information encrypted both with the control word for scrambling and the CPTC information for prohibiting illegal view/copy, and the audio/video bit stream scrambled in control words are multiplexed to make a program.

Distribution medium portion 201 distributes pro-

grams made in program producing portion 200 through a transmission medium.

Program receiving portion 202 detects and analyzes the CPTC information from the bit stream transmitted from distribution medium portion 201 and the bit stream reproduced from cassette tape, and descrambles and decodes the bit stream transmitted from distribution medium portion 201. The descrambled and decoded bit stream is displayed or recorded on cassette tape.

Program producing portion 200 comprises a control word generator 203 for generating a control word for scrambling, a CPTC generator 204 for generating the CPTC information for prohibiting illegal view/copy, a scrambling portion 206 for scrambling the audio/video bit stream using the control word output from control word generator 203, an encrypting portion 205 for encrypting the control word output from control word generator 203 and the CPTC information output from CPTC generator 204, and an adder 207 for multiplexing the signals output from scrambling portion 206 and encrypting portion 205 and transmitting them to distribution medium portion 201.

Distribution medium portion 201 comprises a broadcasting medium 208 for distributing the program made by program producing portion 200 through cable, satellite or terrestrial broadcasting, and a recording medium 209 for distributing the program made by program producing portion 200 through cassette tape.

Program receiving portion 202 comprises a decrypting portion 210 for decrypting the bit stream transmitted from broadcasting medium 208, a CPTC detecting/analyzing portion 211 for detecting and analyzing the CPTC information from the bit stream output from decrypting portion 210 and recording medium 209, and outputting signals for controlling the control word and illegal view/copy, a descrambling portion 212 for descrambling the bit stream transmitted from broadcasting medium 208 and recording medium 209 and the bit stream reproduced from cassette tape, a decoding portion 213 for decoding and displaying the signal output from descrambling portion 212, and a recording/reproducing portion 214 for recording the bit stream transmitted from broadcasting medium 208 and recording medium 209 according to the signal output from CPTC detecting/analyzing portion 211, and reproducing cassette tape, to thereby output the result to descrambling portion 212 and CPTC detecting/analyzing portion 211.

The operation of an illegal view/copy protection apparatus for a digital broadcasting system embodying the present invention will be described below.

Control word generator 203 generates a control word for scrambling, and CPTC generator 204 generates the CPTC information for prohibiting illegal view/copy. Scrambling portion 206 scrambles the audio/video bit stream using the generated control word. Encrypting portion 205 encrypts the CPTC information output from CPTC generator 204 using the generated control word. The audio/video bit stream scrambled in scrambling portion 206 is multiplexed with the encrypted CPTC information in adder 207. The multiplexed result is transmitted to a reception port through distribution medium portion 201.

The signal output from adder 207 is transmitted to program receiving portion 202 through broadcasting medium 208 such as cable, satellite, and terrestrial broadcastings, or through recording medium 209 made of cassette tape such as rental tape.

The bit stream transmitted through broadcasting medium 208 is decrypted in decrypting portion 210. The CPTC information is detected and analyzed in CPTC detecting/analyzing portion 211 so that signals for controlling the control word and illegal view/copy are output. Here, the bit stream transmitted to cassette tape through recording medium 209 is reproduced in recording/reproducing portion 214 and input to descrambling portion 212 and CPTC detecting/analyzing portion 211. The bit stream transmitted from broadcasting medium 208 and the bit stream reproduced from recording medium 209 through recording/reproducing portion 214 are descrambled in descrambling portion 212 according to the control word output from CPTC detecting/analyzing portion 211. The signal output from descrambling portion 212 is decoded in decoding portion 213 and displayed. The bit stream transmitted from broadcasting medium 208 and recording medium 209 is recorded on cassette tape in a recording/reproducing portion 214 according to the signal output from CPTC detecting/analyzing portion 211.

Data received from program receiving portion 202 and recorded on cassette tape is made up of the scrambled audio/video bit stream and CPTC information. The configuration of the program receiving portion having decrypting portion 210, CPTC detecting/analyzing portion 211, descrambling portion 212, decoding portion 213 and recording/reproducing portion 214 will be explained with reference to Figs. 17, 18, 19, and 20.

One embodiment of the program receiving portion of Fig. 17 receives and processes data transmitted via a broadcasting medium. Specifically, this embodiment performs conditional access and copy protection.

Referring to Fig. 17, the first embodiment of the program receiving portion comprises an IRD 222 for receiving, decoding and descrambling the bit stream transmitted from broadcasting medium 208, outputting analog audio/video data to be displayed and outputting scrambled digital audio/video data to be recorded on cassette tape, a smart card 221 for decrypting the bit stream output from IRD 222, detecting/analyzing the CPTC information, and outputting the control word and signals for controlling illegal view/copy to IRD 222 in order to perform conditional access and copy protection, a DVCR 223 for recording the digital audio/video data and CPTC information scrambled and output from IRD 222 on cassette tape, and reproducing the scrambled digital audio/video data and CPTC information recorded on cassette tape to be output to IRD 222, and a lookup table 224 for,

in case that the EMM is updated by a broadcasting station for the purpose of copyright protection, storing EMM having information required to decode the CPTC information, and outputting CPTC information corresponding in reproduction to smart card 221 in order to continuously reproduce the program of copied cassette tape. Here, lookup table 221 is mapped and processed as shown in Figs. 13, 14 and 15.

The operation of the first embodiment of the program receiving portion will be described below.

In case that a bit stream, that is, a program, is received through a broadcasting medium, the received audio/video data is scrambled digital audio/video data.

The received bit stream is decoded in IRD 222 and decrypted in smart card 221. Its CPTC information is detected and analyzed so that a signal for controlling the control word and illegal view/copy is output to IRD 222.

IRD 222 descrambles the decoded bit stream using the bit stream output from smart card 221 and signals for controlling illegal view/copy. The descrambled bit stream is output to display analog audio/video data. IRD 222 outputs the scrambled digital audio/video data and CPTC information to DVCT 223 in order to record them on cassette tape.

The scrambled digital audio/video data and CPTC information output from IRD 222 is recorded on cassette tape in DVCR 223. They are in turn reproduced in DVCR 223 and processed in the same manner that the bit stream transmitted via the broadcasting medium is descrambled and processed in IRD 222 and smart card 221. The processed result is output to be displayed on a monitor, or output to the DVCR and recopied.

Here, reproduction and recopy are made possible by the data stored in the permissible generational field, present generational field, reproducible number field, and maximum reproducible time field contained in the CPTC information.

Updated EMM is mapped and stored in lookup table 224 so that, when the EMM is updated through a broadcasting signal in a broadcasting station in order to protect copyright, the program of cassette tape copied can be continuously reproduced.

Lookup table 224 reads out the EMM containing information required to decode the CPTC information in reproducing the cassette tape. Corresponding CPTC information is output to smart card 221 to enable reproduction.

Another embodiment of the program receiving portion shown in Fig. 18 is to receive and process data transmitted through a recording medium, for instance, rental tape.

The second embodiment of the program receiving portion, as shown in Fig. 18, comprises a DVCR 232 for detecting/analyzing the CPTC information from the bit stream transmitted from the recording medium, outputting a control word and signals for controlling illegal view/copy, and reproducing scrambled digital audio/video data, and an IRD 231 for receiving the control word and signals for controlling illegal view/copy output from DVCR 232, descrambling the scrambled digital audio/video data, and outputting analog audio/video data to be displayed or recorded.

The second embodiment of the program receiving portion is to perform CPTC detection and processing carried out in the smart card of the first embodiment of the program receiving portion shown in Fig. 17. The operation of the second embodiment of the program receiving portion will be described below.

In case that the bit stream is received through the recording medium, the audio/video data reproduced through the DVCR is scrambled digital audio/video data.

The bit stream recorded in DVCR 232 is reproduced. Its CPTC information is detected and analyzed so that the control word and signal for controlling illegal view/copy is output to IRD 231. The bit stream reproduced from DVCR 232 is decoded in IRD 231. The decoded bit stream is descrambled according to the control word and signal for controlling illegal view/copy output from DVCR 232 so that analog audio/video data is output to be displayed.

IRD 231 outputs the scrambled digital audio/video data and CPTC information to DVCR 232 to record them on cassette tape. The scrambled digital audio/video data and CPTC information output from IRD 231 is recorded on cassette tape and recopied in DVCR 223.

Here, reproduction and recopy are made possible by the data stored in the permissible generational field, present generational field, reproducible number field, and maximum reproducible time field contained in the CPTC information.

Referring to Fig. 19, still another embodiment of the program receiving portion is to receive and process data transmitted through a recording medium, performing copy protection (CP).

As shown in Fig. 19, the third embodiment of the program receiving portion comprises a DVCR 243 for reproducing the scrambled digital audio/video data and CPTC information recorded on cassette tape through a recording medium, and outputting them to IRD 242, an IRD 242 for decoding/descrambling the bit stream transmitted from DVCR 243, and outputting analog audio/video data to be displayed, and a smart card 241 for decrypting the bit stream output from IRD 242, detecting/analyzing the CPTC, and outputting the control word and signals for controlling copying to IRD 222 to thereby perform CP. The operation of the third embodiment of the program receiving portion will be explained below.

In case that the bit stream is received via a recording medium, that is, through rental tape, the reproduced audio/video data is scrambled digital audio/video data.

The scrambled digital audio/video data and CPTC information reproduced from DVCR 243 are decoded in IRD 242 and decrypted in smart card 241. The CPTC information is detected and analyzed so that the control word and signal for controlling copying are output to IRD 242.

IRD 242 descrambles the decoded bit stream using the CPTC information output from smart card 241 and signals for controlling copying so that analog audio/video data is output to be displayed.

IRD 242 outputs the scrambled digital audio/video data and CPTC information to DVCR 243 in order to record them on cassette tape. The scrambled digital audio/video data and CPTC information output from IRD 242 are recorded on cassette tape in DVCR 243.

Here, reproduction and recopy are made possible by the data stored in the permissible generational field, present generational field, reproducible number field, and maximum reproducible time field contained in the CPTC information.

Referring to Fig. 20, yet another embodiment of the program receiving portion is to receive and process data transmitted through a recording medium, performing conditional access and CP. This embodiment is made in such a manner that in case of using the same CPTC information as the broadcasting medium, the smart card is commonly used.

As shown in Fig. 20, the fourth embodiment of the program receiving portion comprises a DVCR 253 for reproducing the scrambled digital audio/video data and CPTC information recorded on cassette tape through a recording medium, and outputting them to IRD 252, an IRD 252 for decoding/descrambling the bit stream transmitted from DVCR 253, and outputting analog audio/video data to be displayed, and a smart card 251 for decrypting the bit stream output from IRD 252, detecting/analyzing the CPTC, and outputting the control word and signals for controlling copying to IRD 252 to thereby perform CA and CP. The operation of the third embodiment of the program receiving portion will be explained below.

In case that the bit stream is received via a recording medium, that is, through rental tape and the DVCR, the reproduced audio/video data is scrambled digital audio/video data.

The scrambled digital audio/video data and CPTC information reproduced from DVCR 253 are decoded in IRD 252 and decrypted in smart card 251. The CPTC information is detected and analyzed so that the control word and signal for controlling copying are output back to IRD 252.

IRD 252 descrambles the decoded bit stream using the CPTC information output from smart card 251 and signals for controlling illegal view/copy so that analog audio/video data is output to be displayed.

IRD 252 outputs the scrambled digital audio/video data and CPTC information to DVCR 253 in order to record them on cassette tape. The scrambled digital audio/video data and CPTC information output from IRD 222 are recorded on cassette tape in DVCR 253.

Here, reproduction and recopy are made possible by the data stored in the permissible generational field, present generational field, reproducible number field, and maximum reproducible time field contained in the CPTC information.

IRD 222, 242, or 252 shown in Fig. 17, 19 or 20 is made in the following configuration as shown in Fig. 21.

Referring to Fig. 21, IRD 222, 242 or 252 comprises a recording/digital output controller 262 for decoding the bit stream transmitted from the broadcasting medium and DVCR, outputting to smart card 221, receiving the control word and signals for controlling illegal view/copy output from smart card 221, and controlling the output of the scrambled digital audio/video data for the purpose of recording and displaying; a descrambler 263 for descrambling the scrambled digital audio/video data output from recording/digital output controller 262 according to the control word output from recording/digital output controller 262, and a display processing portion 264 for processing and outputting the digital audio/video data output from descrambler 263 to be displayed. Here, DVCR 265 performs reproduction mainly. DVCR 223 of the program receiving portion of Fig. 18 combines recording therewith. The operation of IRD 266 will be described below.

The signal output to smart card 261 from recording/digital output controller 262 of IRD 266 is ECM, EMM and CPTC information. The signals output from smart card 261 to IRD 266 are the control word used to descramble and display the bit stream, and a signal for controlling copy protection.

Recording/digital output controller 262 communicates with the smart card, performs recording according to the signals of copy protection, outputs them to the digital output port in order to record them in another set, and outputs the control word and bit stream to descrambler 263.

When output to the recording/digital output port, updated ECM, EMM and CPTC information are output in addition to the scrambled data from recording/digital output controller 262 so that a copy different from the original script, that is, the broadcast or rental tape.

The ECM, EMM and CPTC are transmitted in various combinations. For the first combination, the ECM, EMM and CPTC are independently combined. The second combination is that the CPTC is included in the ECM and the EMM is independently combined. The third is that the CPTC is included in the EMM and the ECM is independently combined.

IRD 231 and DVCR 232 of Fig. 18 use the smart card, and additionally requires a CPTC detection and processing portion in the DVCR, which will be shown in Fig. 22.

DVCR 232 comprises a CPTC detecting/processing portion 276 for detecting/analyzing the CPTC information from the bit stream transmitted from recording medium 209, and outputting the control word and signals for illegal view/copy, and a reproducing portion 277 for reproducing the bit stream transmitted from recording medium 209 and outputting it to the IRD.

IRD 231 comprises a digital output controller 272 for receiving the control word and signals for controlling

illegal view/copy output from CPTC detecting/processing portion 276, and controlling the output of the scrambled digital audio/video data output from reproducing portion 277 in order to display them, a descrambler 273 for descrambling the scrambled digital audio/video data output from digital output controller 262 according to the control word output from digital output controller 262, and a display processing portion 274 for processing and outputting the digital audio/video data output from descrambler 273 in order to display them. The operation of IRD 276 and DVCR 275 will be described below.

CPTC detecting/processing portion 276 operates separately when reproducing portion 277 reproduces the scrambled data so that the CPTC information is detected from the cassette tape.

IRD 276 receives the scrambled data, CPTC information and control word from CPTC detecting/processing portion 276 and reproducing portion 277 from DVCR 275. Therefore, for normal descrambling, the scrambled data and control word are supplied to scrambler 273 from digital output controller 272. To the digital output port, only the scrambled data is output. For this reason, in case that the reproduced data is scrambled, copying is made impossible, and vise versa.

Commonly, in order to control tape copying, the depth of generation copy and the reproduction of tape to be copied are used together. As shown in Fig. 7, this yields the effect of controlling the number of copiable tape.

However, in order to allow copying tape to be reproducible as many as a predetermined number or for a predetermined time, it is necessary to perform communication between the smart card and DVCR.

Referring to Fig. 23, tape state information such as the reproduction number of the current tape is transmitted to smart card 261 from DVCR 265. In order to erase the tape, an enable erase signal is transmitted to DVCR 265 from smart card 261, and the erase head of the DVCR operates.

For tape erasing methods, the whole area of tape is erased by the full-width erase head, or only the control track is erased using the control head. In case that the CPTC is contained in the EMM, signals are input and output between the DVCR and smart card.

As the signals input to IRD 266, there are a broadcasting signal transmitted from a broadcasting medium and a signal reproduced from DVCR 265. The broadcasting signal input to IRD 266 is the scrambled digital data and a control signal having the EMM, ECM and CPTC information. The EMM and ECM are required for CA, the CPTC for copyright protection.

The scrambled digital data is input to descrambler 263. The control signal is input to smart card 261 for performing CA and CP. Using the control signal, smart card 261 restores control word CW and outputs it to descrambler 263. Descrambler 263 descrambles it using the control word.

The ECM output from smart card 261 is output to DVCR 265 or to an external port. This ECM is updated from the ECM input for copyright protection. The output disable signal output from smart card 261 is a signal to instruct IRD 266 to prohibit recording or copying. This signal is input to recording/digital output controller 262. The tape state signal is output to smart card 261 from DVCR 265 in order to inform the state of tape.

The signal output to DVCR 265 from smart card 261 for the purpose of a predetermined-number reproduction or predetermined-time reproduction is an erase enable signal. The signal for allowing recorded and copied tape to be reproducible even though the EMM information of the smart card is changed is an ID signal.

The ID signal is mapped and stored with corresponding EMM in the lookup table of smart card 261. If necessary, the EMM corresponding to the ID signal is output.

As shown in Fig. 24, the smart card comprises an ECM filter 301 for filtering the ECM from the bit stream output from the IRD, a CPTC/tape state signal filter 302 for filtering the CPTC information and the tape state signal indicative of the state of tape from the bit stream output from the IRD, an EMM filter 303 for filtering the EMM from the bit stream output from the IRD, a lookup table 304 for, in case that the EMM is updated for copyright protection by a broadcasting station, storing the previous EMM containing information required to decode the CPTC information, and outputting CPTC information corresponding in reproduction in order to continuously reproduce the program of cassette tape copied, an EMM processing portion 307 for processing the EMM using the EMM output from EMM filter 303 and lookup table 304 and the tape state signal output from CPTC/tape state signal filter 302, a CPTC processing portion 306 for processing the CPTC information using the signals output from CPTC/tape state signal filter 302 and EMM processing portion 307, and a CA processing portion 305 for outputting control word CW using the signals output from ECM filter 301 and EMM processing portion 307.

In case that the CPTC information is contained in the EMM, as shown in Fig. 25, smart card 221 comprises an ECM filter 311 for filtering the ECM from the bit stream output from the IRD, an EMM filter 312 for filtering the EMM containing the EMM from the bit stream output from the IRD, a tape state signal filter 313 for filtering the tape state signal output from the IRD, a lookup table 314 for, in case that the EMM is updated for copyright protection by a broadcasting station, storing the previous EMM containing information required to decode the CPTC information, and outputting CPTC information corresponding in reproduction in order to continuously reproduce the program of cassette tape copied, an EMM processing portion 317 for processing the EMM using the EMM output from EMM filter 312 and lookup table 314 and the tape state signal output from tape state signal filter 313, a CPTC processing portion 316 for processing the CPTC information using the signals

output from EMM filter 312 and tape state signal filter 313, to thereby output ECM, enable erase signal and ID signal, and a CA processing portion 315 for outputting control word CW using the signals output from ECM filter 311 and EMM processing portion 317.

ECM filter 301 or 311, CPTC/tape state signal filter 302, EMM filter 303 or 312, and tape state signal filter 313 extract ECM, CPTC, tape state signal and EMM, respectively. CA processing portion 305 or 315 generates a control word and performs CA. EMM processing portion 307 or 317 outputs the EMM information to CA processing portion 305 or 315 and CPTC processing portion 306 or 316, and additionally stores the received EMM to the lookup table.

In case that the scrambled digital data and encoded CPTC information are recorded on tape and that the EMM information required to decode the CPTC information is changed, the reproduction of tape is made impossible. According to this fact, the previous EMM is stored in a memory such as the EEPROM of the smart card as shown in Figs. 13 and 14, which is the same as described before.

Specifically, the lookup table is divided into two fields and stores ID information and EMM information, as shown in Fig. 13. In recording and copying, the ID information is recorded on tape, as shown in Fig. 14 in order to select corresponding EMM from the ID information recorded in the reproduction of tape.

In other words, referring to Fig. 14, EMM processing portion 307 receives a recording/playback signal indicating that the current DVCR mode is recording or playback, ID, and tape state signal having information of reproduction number of tape, selects a proper EMM from the lookup table, outputs it to CPTC processing portion 306 or 316 and CA processing portion 305 or 315, and transmits the ID information for the purpose of recording and copying to record it on tape.

Referring to Fig. 11, CPTC processing portion 306 or 316 performs copyright protection for recording or copying. The CPTC information or ECM containing the CPTC information is input to output the output disable signal, enable erase signal, and the CPTC or ECM containing the CPTC.

In order to control generation copy, CPTC processing portion 306 or 316, in case that the permissible generation of the permissible generational field is greater than the present generation recorded on tape, the present generational field is increased by 1 and encrypted again. If not, the output disable signal is generated to prohibit recording and copying.

In order to control reproduction, in case that the reproducible number of tape is greater than the reproducible number of the reproducible number field or the maximum reproducible time of the maximum reproducible time field is greater than the current time, CPTC processing portion 306 or 316 generates enable erase signal to operate the erase head of the DVCR.

In case that time delay produced when the CPTC or the ECM containing the CPTC is encrypted again becomes a problem to solve, CPTC processing portion 306 or 316 transmits the current generation signal to the DVCR and records it on tape, not modifying the CPTC or the ECM containing the CPTC.

The illegal view/copy protection apparatus for a digital broadcasting system embodying the present invention has means for recording and reproducing the reproduction number information of tape in the DVCR in order to implement the predetermined-number reproducibility of recorded or copied tape. Here, the reproduction number information of tape is updated and recorded again during tape reproduction.

As shown in Fig. 26, the DVCR comprises a deck mechanism 406, a recording/reproducing portion 405 for recording digital data on cassette tape according to the deck mechanism and reproducing the digital data recorded on cassette tape, a reproduction number detecting/updating portion 401 for detecting/updating the reproduction number from the digital data reproduced from recording/reproducing portion 405, and outputting it to the IRD in order to rerecord it in recording/reproducing portion 405, a digital data processing portion 402 for processing the digital data reproduced from recording/reproducing portion 405, outputting it to the IRD, and outputting switching position information for recording and reproducing, a recording/playback switching portion 404 for outputting a switching signal for controlling the reproduction number, the reproduction of digital data and the recording of the updated reproduction number using the switching position information output from digital data processing portion 402, and an error correction encoder/decoder 403 for correcting the error of data output from digital data processing portion 402, and encoding and decoding the data to be output to digital data processing portion 402.

In order to update and rerecord the reproduction number information of tape during playback, the reproduction number information of tape is recorded using an encoding algorithm. Otherwise, the information is recorded as clear data not encoded.

The recording position of the reproduction number information of tape uses part of audio, control and video tracks. For error correction to the reproduction number information of tape, a repetition coding is employed. The operation of the DVCR will be described below.

When reproduced by recording/reproducing portion 405 with the cassette tape loaded on deck mechanism 406, the reproduced digital data is input to reproduction number detecting/updating portion 401 and digital data processing portion 402 so that its reproduction number is detected and the digital data is processed and output.

The reproduction number detected in reproduction number detecting/updating portion 401 is updated, that is, increased by 1, and applied to recording/reproducing portion 405.

Digital data processing portion 402 applies the reproduced digital data output from recording/reproducing

portion 405 to error correction encoder/decoder 403 to perform error correction, encoding and decoding. The result is output to the IRD to be displayed or recorded. At the same time, the switching position information is output to recording/reproducing switching portion 404 in order to output a switching signal.

The switching signal output from recording/reproducing switching portion 404 controls recording/reproducing portion, to thereby record the updated reproduction number output from reproduction number detecting/updating portion 401, that is, the reproduction number added by 1, on tape.

Recording/reproducing switching portion 404 controls the reproduction number, the reproduction of digital data recorded on tape, and the recording of the updated reproduction number.

In another method of implementing the predetermined-number reproducibility of recorded or copied tape, an identifier is given to all tape used for a user to record broadcast programs, and the identifier given to tape and the reproducibility number information of tape corresponding to the identifier are handled together in the smart card.

Here, the smart card has a memory device which can be updated, such as EEPROM. The identifier and corresponding reproducible number information are stored in the memory device. For every reproduction of tape, the reproducible number information is updated and whether to playback is determined.

In conclusion, the described embodiments have the following advantages.

First, by adding CPTC information to data supplied, and by allowing a digital program to be normally viewed only when a CPTC detecting/analyzing means and descrambling/decrypting means are present at the receiving stage, illegal viewing is prohibited.

Second, to enhance copyright protection, data recorded on cassette tape is always scrambled digital data, and its CPTC information is encrypted to be recorded on cassette tape. A code for prohibiting viewable data from being restored from the cassette tape only with the scrambled data and CPTC information, and allowing the data to be viewable is provided in a device excluding the cassette tape. Otherwise, restoring of viewable data is made possible only with the scrambled data and CPTC information, making illegal copy impossible.

Third, using a method of restoring the viewable data only with the scrambled digital data and CPTC, rental tape is made to supply tape. Otherwise, using a method of prohibiting the viewable data from being restored only with the scrambled digital data and CPTC, rental tape is made to supply tape and smart card peculiar to a program provider as one set. Using the smart card for broadcasting medium, the rental tape is made to prohibit the viewable data from being restored only with the scrambled digital data and CPTC. Among the three methods of supplying tape only, one method is selected. Digital hardware for reproducing the data outputs only the scrambled digital data to an external port, making impossible the restoring of viewable data from the output data, without the smart card.

Fourth, the described embodiment prohibits illegal recording and copying of a program protected by copyright law, collects fee for recording or copying, and freely controls the reproducible number of copied tape which can be made from a program supplied by a program supplier, protecting copyright.

Fifth, the described embodiment can be used as a copyright protection system having a high security and multifunction with respect to a program through a broadcasting medium such as satellite and terrestrial broadcastings, or, at the same time, as a copy protection system having a high security to a program through a recording medium such as rental tape.

Sixth, the descibed embodiment is employed to digital hardware such as broadcasting receiver and digital VCR, to thereby perfectly protect a program supplier's copyright and activates digital media because of various software supplied through the digital media.

Claims

1. An illegal view/copy protection method for a digital broadcasting system comprising:

   an audio/video signal transmission step for multiplexing and transmitting audio/video bit stream scrambled in control words and information where the control words and CPTC information for illegal view/copy protection are encrypted; and

   an audio/video reception step for decrypting the transmitted bit stream to analyze the CPTC information and control words, deciding whether recording is allowed or not to be recorded on cassette tape, and using the control words, performing descrambling and decoding to output audio/video signals to a monitor.

2. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 1, wherein said CPTC information is formatted in a generational copy control field for limiting the number of copy available, and a reproducibility control field for limiting the reproduction of a copied program.

3. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 2, wherein said CPTC information is formatted further containing a descrambling information field where part of the control words for descrambling are recorded.

4. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 2, wherein said CPTC information is formatted further contain-

ing a CA field where CA information for conditional access is recorded.

5. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 2, wherein said generational copy control field is made up of a permissible generational field for limiting the number of copy permissible and a present generational field for indicating the present generation of a program copied.

6. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 2, wherein said reproduction control field is made up of a reproducible number field for limiting the number of reproducing a copied program, and a maximum reproducible time field for limiting time to reproduce the copied program.

7. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 1, wherein the data recorded on cassette tape contains scrambled audio/video bit stream and CPTC information.

8. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 7, wherein said CPTC information is overwritten on the scrambled audio/video bit stream for the error effect and recorded on cassette tape.

9. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 7, wherein said CPTC information is recorded on a portion of any of the audio track of cassette tape, the control track of cassette tape, or the video track of cassette tape.

10. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 1, wherein said audio/video signal transmission step comprises the steps of: encoding the audio/video bit stream;

generating a control word for scrambling;
scrambling for the encoded audio/video bit stream using the generated control word;
generating CPTC information for illegal view/copy protection;
encrypting for encrypting the control word and CPTC information; and
multiplexing and transmitting the scrambled audio/video bit stream and encrypted CPTC information.

11. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 1, wherein said audio/video signal transmission step comprises the steps of:

encoding the audio/video bit stream;
generating a control word for scrambling;
scrambling for the encoded audio/video bit stream using the generated control word;
generating CPTC information for illegal view/copy protection;
generating conditional access information for conditional reception;
encrypting for encrypting the CPTC information and CA information; and
multiplexing and transmitting the scrambled audio/video bit stream and encrypted CPTC information and conditional access information.

12. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 1 or claim 11, wherein said audio/video reception step comprises the steps of:

filtering the transmitted bit stream and decrypting the CPTC information;
analyzing the CPTC information to generate a control word and a signal for controlling the protection of copyright and to update the CPTC information;
deciding whether to allow recording according to the signal for controlling the protection of copyright to record the scrambled and transmitted bit stream on cassette tape; and
descrambling and decoding the transmitted bit stream in the control word and outputting an audio/video signal.

13. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 12, wherein said all of the control word is contained in the CPTC information.

14. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 1, wherein said bit stream transmitted contains ECM and EMM.

15. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 14, wherein said audio/video reception step comprises the steps of:

filtering the transmitted bit stream and decrypting the CPTC information and control word;
filtering the control word;
analyzing the CPTC information to generate a control word and a signal for controlling the protection of copyright and to update the CPTC information;
deciding whether to allow recording according to the signal for controlling the protection of copyright to record the scrambled and transmit-

ted bit stream on cassette tape; and
descrambling and decoding the transmitted bit stream in control words and outputting an audio/video signal.

16. An illegal view/copy protection method for a digital broadcasting system as claimed in any of claims 12, 14 or 15, wherein said CPTC information analyzing step comprises the steps of:

generating a control word;
detecting a permissible generation of a permissible generational field for limiting the available number of copy of a program of the CPTC information and the present generation of the present generational field indicating the present generation of the program copied, to thereby perform copy-impossible and update the CPTC information; and
detecting the reproducible number of the reproducible number field for limiting the number of reproduction of copied programs of the CPTC information, the maximum reproducible time of the maximum reproducible time field for limiting time to reproduce the copied program, and the number and time of reproduction of tape, to thereby process reproduction-impossible.

17. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 12 or claim 16, wherein said copying number limiting step comprises the steps of:

comparing the permissible generation of the permissible generational field and the present generation of the present generational field and deciding whether the permissible generation is below the present generation;
if the permissible generation is below the present generation, generating an output disable signal to make copying impossible and destroying the control word; and
if the permissible generation is not below the present generation, increasing the present invention by '1' and recording the result on cassette tape.

18. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 17, wherein said copying number limiting step further comprises the step of, if the permissible generation is not below the present generation, updating the CPTC information.

19. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 16 or claim 17, wherein said reproduction limiting step comprises the steps of:

comparing the reproducible number of the reproducible number field and the reproduction number of tape and deciding whether the reproducible number is below the reproduction number of tape;
if the reproducible number is not below the reproduction number of tape, comparing the maximum reproducible time and reproduction time of tape, and deciding whether the maximum reproducible time is below the reproduction time of tape;
if the maximum reproducible time is not below reproduction time of tape, turning off an enable erase signal to thereby enable the copied program to be reproduced; and
if the reproducible number is below the reproduction number of tape or the maximum reproducible time is below the reproduction time of tape, turning on the enable erase signal to make the reproduction of the copied program impossible so that part of or the whole program recorded on cassette tape is erased.

20. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 14 or claim 15, wherein part of the control word is contained in the CPTC information.

21. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 20, wherein the remainder of the control word is contained in the ECM.

22. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 20, wherein the remainder of the control word is contained in the EMM.

23. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 14 or claim 15, wherein the whole control word is contained in the ECM.

24. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 14 or claim 15, wherein the whole control word is contained in the EMM.

25. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 14, further comprising a reproduction and rerecording step of: decrypting the bit stream recorded and reproduced on cassette tape, analyzing the CPTC information, deciding whether to allow rerecording, recording the result on cassette tape, filtering the control word, and performing descrambling and decoding to output an audio/video signal.

26. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 25, wherein said audio/video reproduction and rerecording step comprises the steps of:

filtering the bit stream recorded and reproduced on video tape, and decrypting the CPTC information;
analyzing the CPTC information to generate control words and a signal for controlling the protection of copyright and update the CPTC information;
deciding whether to allow recording according to the signal of controlling the protection of copyright, and recording the scrambled and transmitted bit stream on cassette tape; and
descrambling and decoding the transmitted bit stream in control words to output an audio/video signal.

27. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 26, wherein said audio/video reproduction and rerecording step comprises the step of deciding whether to allow post-reproduction according to the signal for controlling the protection of copyright to thereby erase part of or the whole data recorded on cassette tape.

28. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 25, wherein said EMR contains information required for decoding information

29. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 28, further comprising the step of storing and processing EMM in which, in case that the EMM is updated by a broadcasting station for the purpose of copyright protection, the EMM having information required to decode the CPTC information is stored in order to continuously reproduce programs of copied cassette tape.

30. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 29, wherein an ID number indicative of updating the EMM is recorded on said cassette tape.

31. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 30, wherein the EMM is stored to which the updating state and the ID number of cassette tape are mapped.

32. An illegal view/copy protection method for a digital broadcasting system as claimed in claim 31, wherein said EMM storing and processing step comprises the steps of:

storing all EMM to be updated and corresponding ID information;
selecting the latest EMM in recording cassette tape;
recording a corresponding ID number; and
selecting an EMM corresponding to the ID number recorded on cassette tape in reproducing the cassette tape.

33. An illegal view/copy protection apparatus for a digital broadcasting system comprising:

a program producing portion for multiplexing information encrypted both with the control word for scrambling and the CPTC information for prohibiting illegal view/copy, and the audio/video bit stream scrambled in control words, to thereby make a program;
a distribution medium portion for distributing programs made in said program producing portion through a transmission medium; and
a program receiving portion for detecting and analyzing the CPTC information from the bit stream transmitted from said distribution medium portion and the bit stream reproduced from cassette tape, and descrambling and decoding the bit stream transmitted from said distribution medium portion.

34. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 33, wherein said program producing portion comprising:

a control word generator for generating a control word for scrambling;
a CPTC generator for generating the CPTC information for prohibiting illegal view/copy;
a scrambling portion for scrambling the audio/video bit stream using the control word output from said control word generator;
an encrypting portion for encrypting the control word output from said control word generator and the CPTC information output from said CPTC generator; and
an adder for multiplexing the signals output from said scrambling portion and encrypting portion and transmitting them to said distribution medium portion.

35. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 33, wherein said distribution medium portion comprises:

a broadcasting medium for distributing the program made by said program producing portion through cable, satellite or terrestrial broadcast-

ing; and
a recording medium for distributing the program made by said program producing portion through cassette tape.

36. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 35, wherein said program receiving portion comprises:

a decrypting portion for decrypting the bit stream transmitted from said broadcasting medium;
a CPTC detecting/analyzing portion for detecting and analyzing the CPTC information from the bit stream output from said decrypting portion and recording medium, and outputting signals for controlling the control word and illegal view/copy;
a descrambling portion for descrambling the bit stream transmitted from said broadcasting medium and recording medium and the bit stream reproduced from cassette tape;
a decoding portion for decoding and displaying the signal output from said descrambling portio; and
a recording/reproducing portion for recording the bit stream transmitted from said broadcasting medium and recording medium according to the signal output from said CPTC detecting/analyzing portion, and reproducing cassette tape, to thereby output the result to said descrambling portion and CPTC detecting/analyzing portion.

37. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 33, wherein said CPTC information is formatted in a generational copy control field for limiting the number of copy available, and a reproducibility control field for limiting the reproduction of a copied program.

38. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 37, wherein said CPTC information is formatted further containing a descrambling information field where the whole or part of the control words for descrambling are recorded.

39. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 37, wherein said CPTC information is formatted further containing a CA field where CA information for conditional access is recorded.

40. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 37,wherein said generational copy control field is made up of a permissible generational field for limiting the number of copy permissible and a present generational field for indicating the present generation of a program copied.

41. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 37, wherein said reproduction control field is made up of a reproducible number field for limiting the number of reproducing a copied program, and a maximum reproducible time field for limiting time to reproduce the copied program.

42. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 33, wherein the data recorded on cassette tape contains scrambled audio/video bit stream and CPTC information.

43. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 42, wherein said CPTC information is overwritten on the scrambled audio/video bit stream for the error effect and recorded on cassette tape.

44. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 42, wherein said CPTC information is recorded on a portion of any of the audio track of cassette tape, the control track of cassette tape, or the video track of cassette tape.

45. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 33, wherein said all of the control word is contained in the CPTC information.

46. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 33, wherein said bit stream transmitted contains ECM and EMM.

47. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 46, wherein part of the control word is contained in the CPTC information.

48. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 47, wherein the remainder of the control word is contained in the ECM.

49. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 47, wherein the remainder of the control word is contained in the EMM.

50. An illegal view/copy protection apparatus for a dig-

ital broadcasting system as claimed in claim 46, wherein the whole control word is contained in the ECM.

51. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 46, wherein the whole control word is contained in the EMM.

52. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 46, wherein said program receiving portion comprises:

an IRD for receiving, decoding and descrambling the bit stream transmitted from said broadcasting medium, outputting analog audio/video data to be displayed and outputting scrambled digital audio/video data to be recorded on cassette tape; and
a smart card for decrypting the bit stream output from said IRD, detecting/analyzing the CPTC information, and outputting the control word and signals for controlling illegal view/copy to said IRD in order to perform conditional access and copy protection.

53. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 52,wherein said program receiving portion further comprises a lookup table for, in case that the EMM is updated by a broadcasting station for the purpose of copyright protection, storing EMM having information required to decode the CPTC information, and outputting CPTC information corresponding in reproduction to said smart card in order to continuously reproduce the program of copied cassette tape.

54. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 52, wherein said program receiving portion further comprises a DVCR for recording the digital audio/video data and CPTC information scrambled and output from said IRD on cassette tape, and reproducing the scrambled digital audio/video data and CPTC information recorded on cassette tape to be output to said IRD.

55. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 54, wherein said smart card comprises:

an ECM filter for filtering the ECM from the bit stream output from said IRD;
a CPTC/tape state signal filter for filtering the CPTC information and the tape state signal indicative of the state of tape from the bit stream output from said IRD;
an EMM filter for filtering the EMM from the bit stream output from said IRD;
a lookup table for, in case that the EMM is updated for copyright protection by a broadcasting station, storing the previous EMM containing information required to decode the CPTC information, and outputting CPTC information corresponding in reproduction in order to continuously reproduce the program of cassette tape copied;
an EMM processing portion for processing the EMM using the EMM output from said EMM filter and lookup table and the tape state signal output from said CPTC/tape state signal filter;
a CPTC processing portion for processing the CPTC information using the signals output from said CPTC/tape state signal filter and EMM processing portion; and
a CA processing portion for outputting control word CW using the signals output from said ECM filter and EMM processing portion.

56. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 54, wherein said smart card comprises:

an ECM filter for filtering the ECM from the bit stream output from said IRD;
an EMM filter for filtering the EMM containing the EMM from the bit stream output from said IRD;
a tape state signal filter for filtering the tape state signal output from said IRD;
a lookup table for, in case that the EMM is updated for copyright protection by a broadcasting station, storing the previous EMM containing information required to decode the CPTC information, and outputting CPTC information corresponding in reproduction in order to continuously reproduce the program of cassette tape copied;
an EMM processing portion for processing the EMM using the EMM output from said EMM filter and lookup table and the tape state signal output from said tape state signal filter;
a CPTC processing portion for processing the CPTC information using the signals output from said EMM filter and tape state signal filter, to thereby output ECM, enable erase signal and ID signal; and
a CA processing portion for outputting control word CW using the signals output from said ECM filter and EMM processing portion.

57. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 54, wherein said DVCR comprises:

a deck mechanism;
a recording/reproducing portion for recording digital data on cassette tape according to said deck mechanism and reproducing the digital data recorded on cassette tape;
a reproduction number detecting/updating portion for detecting/updating the reproduction number from the digital data reproduced from said recording/reproducing portion, and outputting it to said IRD in order to rerecord it in said recording/reproducing portion;
a digital data processing portion for processing the digital data reproduced from said recording/reproducing portion, outputting it to said IRD, and outputting switching position information for recording and reproducing;
a recording/playback switching portion for outputting a switching signal for controlling the reproduction number, the reproduction of digital data and the recording of the updated reproduction number using the switching position information output from said digital data processing portion; and
an error correction encoder/decoder for correcting the error of data output from said digital data processing portion, and encoding and decoding the data to be output to said digital data processing portion.

58. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 35, wherein said program receiving portion comprises:

a DVCR for detecting/analyzing the CPTC information from the bit stream transmitted from said recording medium, outputting a control word and signals for controlling illegal view/copy, and reproducing scrambled digital audio/video data; and
an IRD for receiving the control word and signals for controlling illegal view/copy output from said DVCR 232, descrambling the scrambled digital audio/video data, and outputting analog audio/video data to be displayed or recorded.

59. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 58, wherein said DVCR comprises:

a CPTC detecting/processing portion for detecting/analyzing the CPTC information from the bit stream transmitted from said recording medium, and outputting the control word and signals for illegal view/copy; and
a reproducing portion for reproducing the bit stream transmitted from said recording medium and outputting it to said IRD.

60. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 59, wherein said IRD comprises:

a digital output controller for receiving the control word and signals for controlling illegal view/copy output from said CPTC detecting/processing portion, and controlling the output of the scrambled digital audio/video data output from said reproducing portion in order to display them;
a descrambler for descrambling the scrambled digital audio/video data output from said digital output controller according to the control word output from said digital output controller; and
a display processing portion for processing and outputting the digital audio/video data output from said descrambler in order to display them.

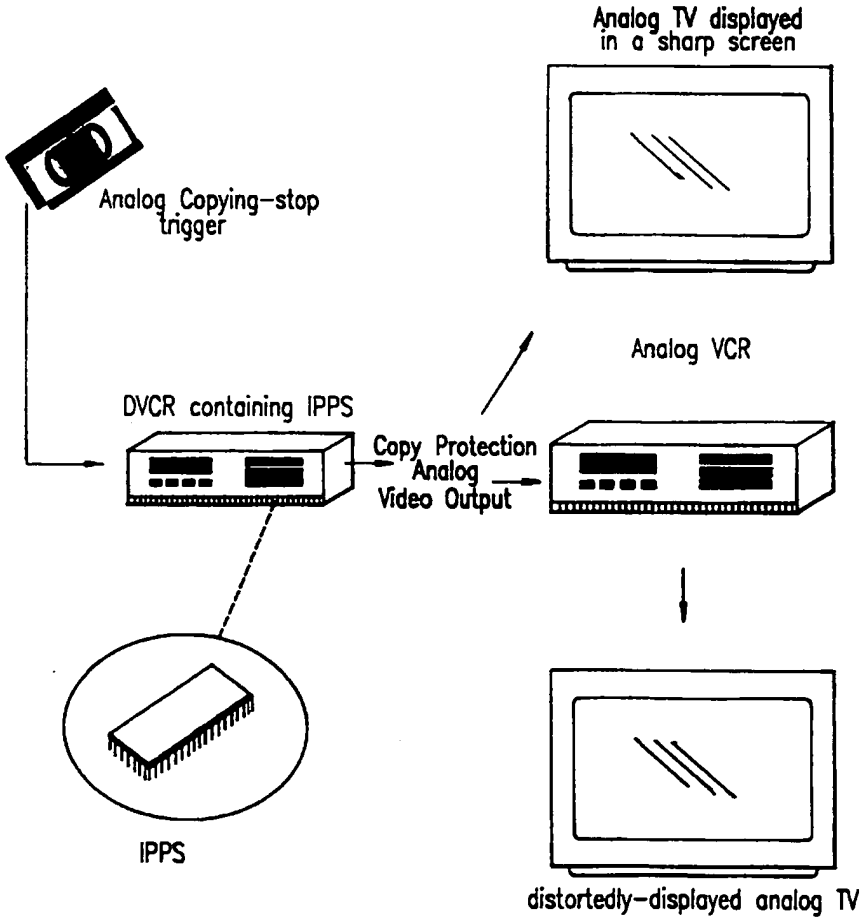
61. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 35, wherein said program receiving portion comprises:

a DVCR for reproducing the scrambled digital audio/video data and CPTC information recorded on cassette tape through a recording medium, and outputting them to said IRD;
an IRD for decoding/descrambling the bit stream transmitted from said DVCR, and outputting analog audio/video data to be displayed; and
a smart card for decrypting the bit stream output from said IRD, detecting/analyzing the CPTC, and outputting the control word and signals for controlling copying to said IRD to thereby perform copy protection and/or conditional access.

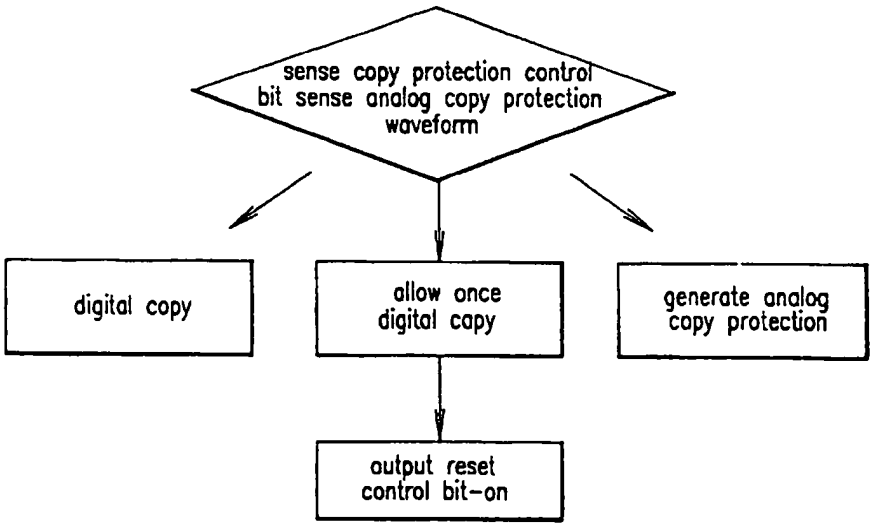
62. An illegal view/copy protection apparatus for a digital broadcasting system as claimed in claim 54 or claim 61, wherein said IRD comprises:

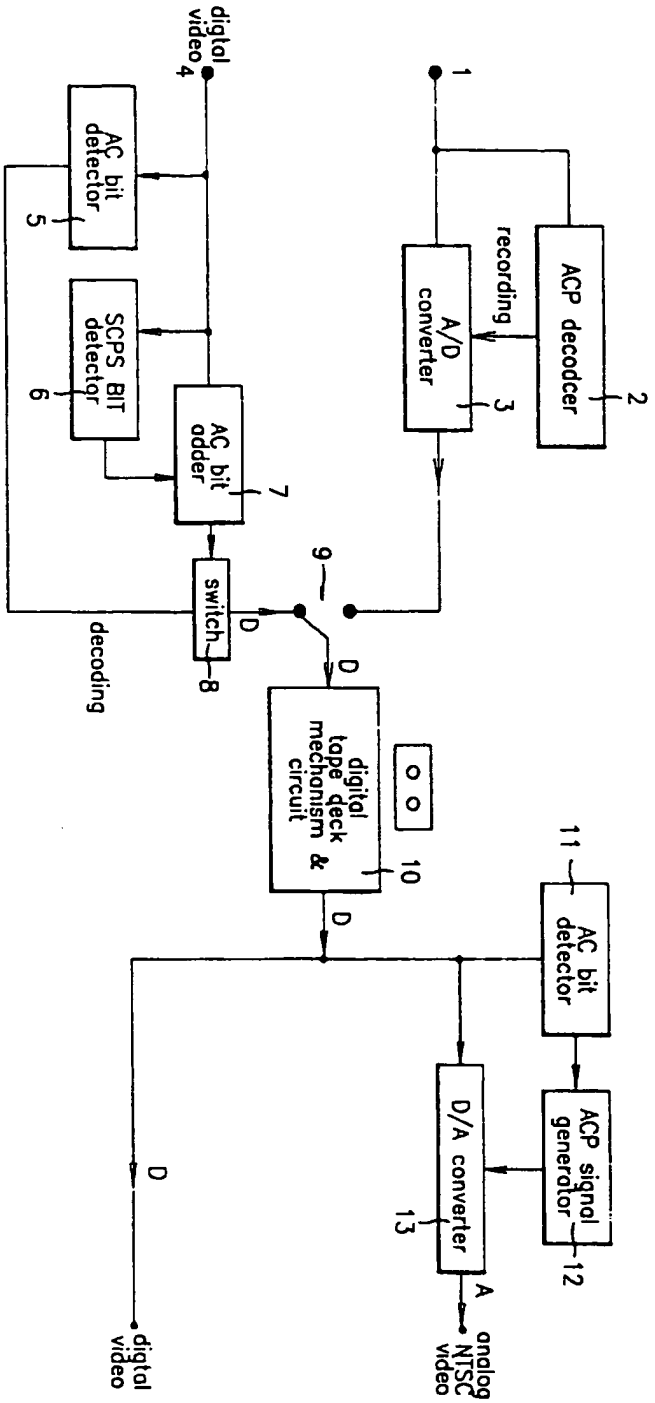
a recording/digital output controller for decoding the bit stream transmitted from the broadcasting medium and DVCR, outputting to said smart card, receiving the control word and signals for controlling illegal view/copy output from said smart card, and controlling the output of the scrambled digital audio/video data for the purpose of recording and displaying;
a descrambler for descrambling the scrambled digital audio/video data output from said recording/digital output controller according to the control word output from said recording/digital output controller; and
a display processing portion for processing and outputting the digital audio/video data output from said descrambler to be displayed.

F I G.1
(conventional art)



F I G.2
(conventional art)





F I G.3

FIG.4

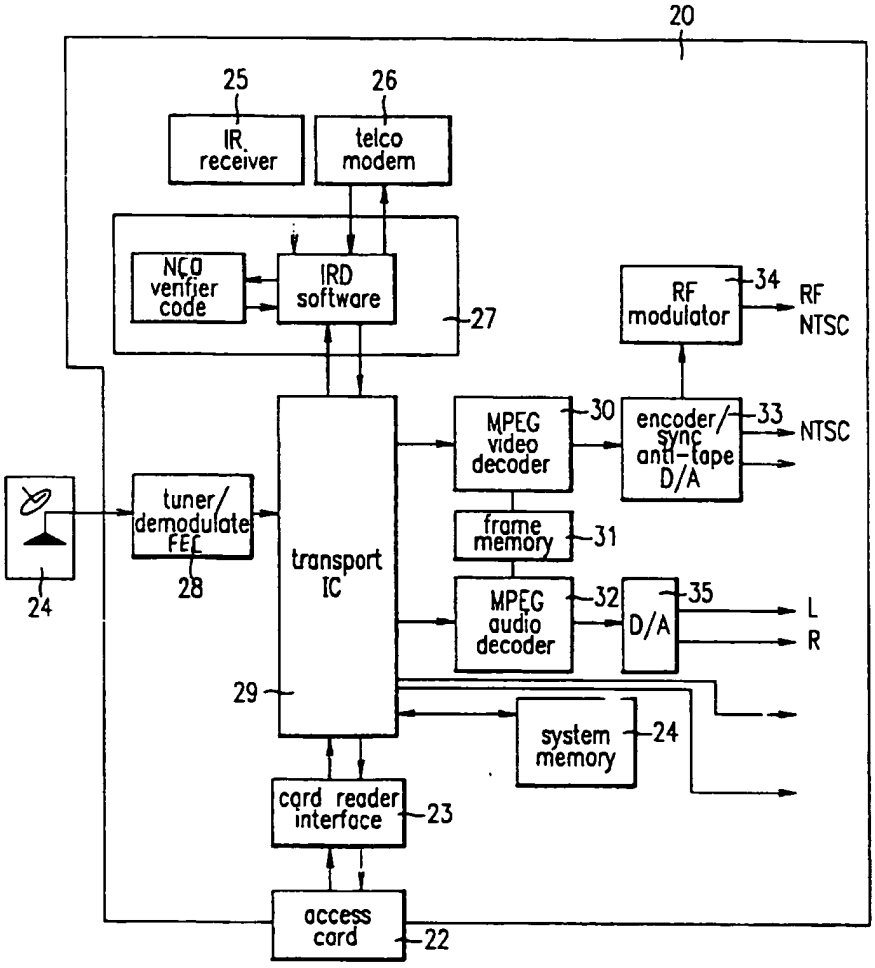


FIG.5

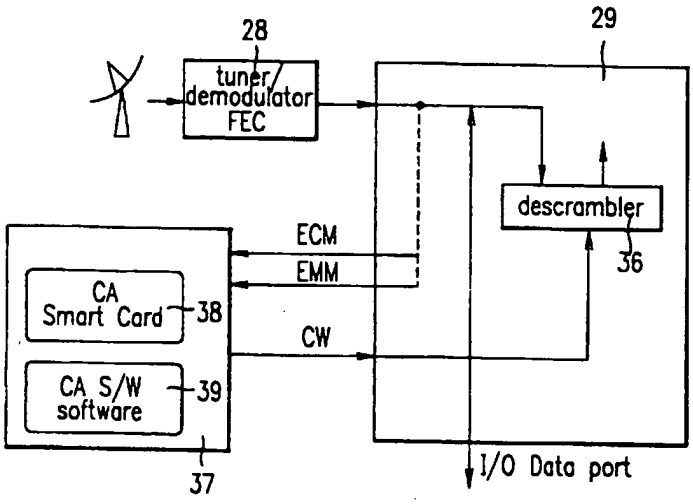


FIG.6a

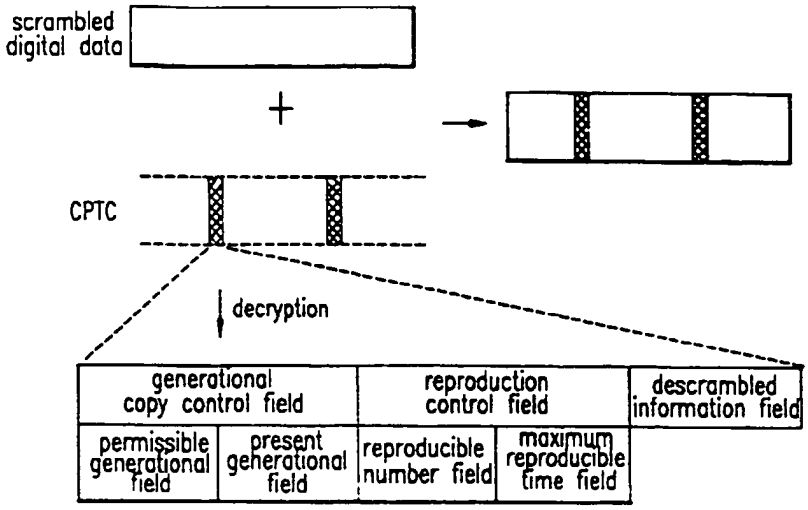


FIG.6b

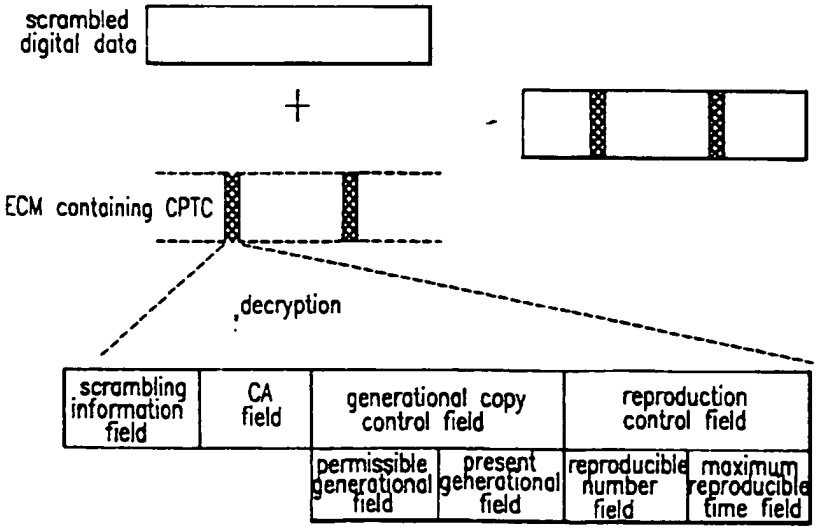


FIG.7

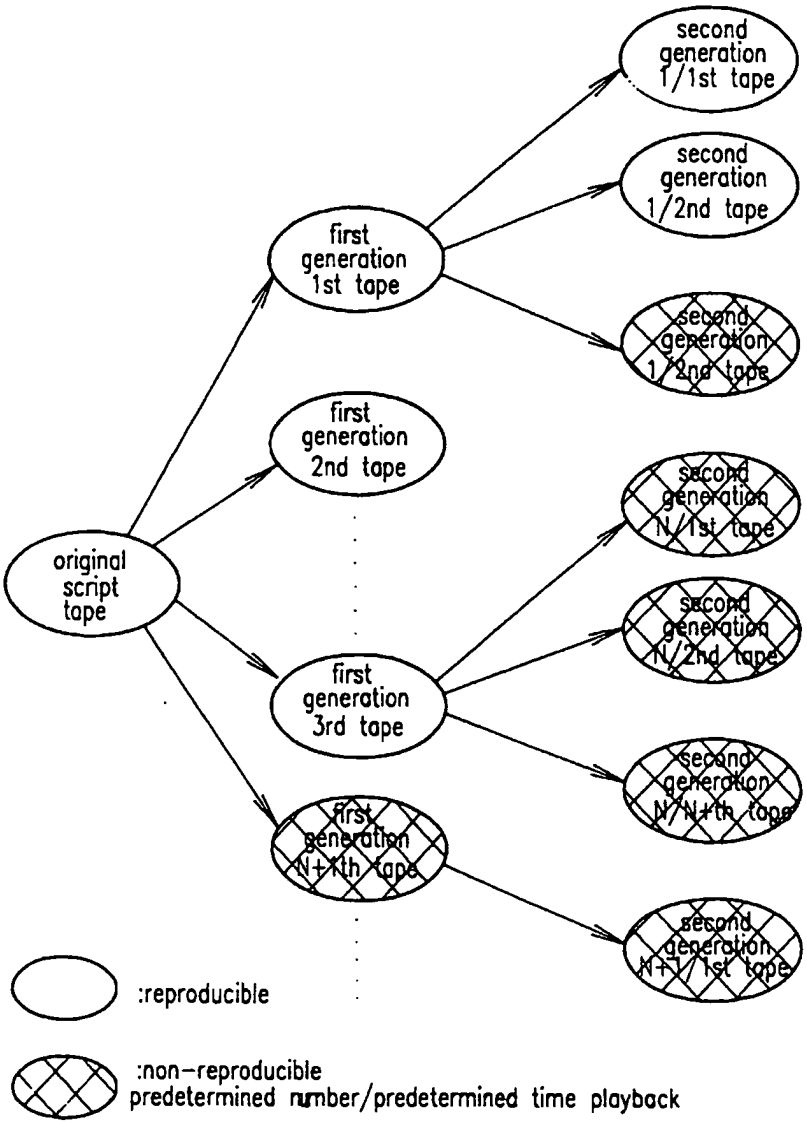


FIG.8a

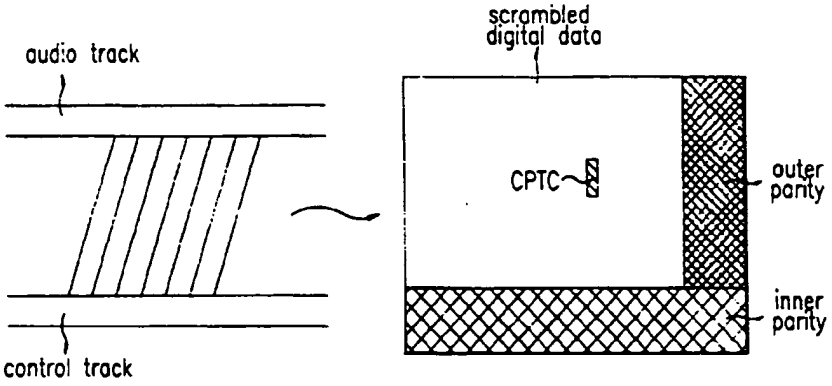


FIG.8b

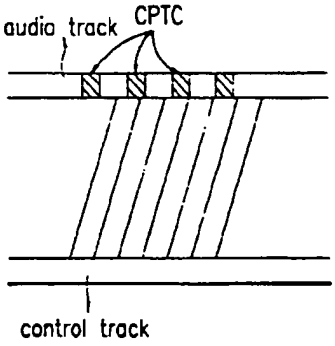


FIG.8c

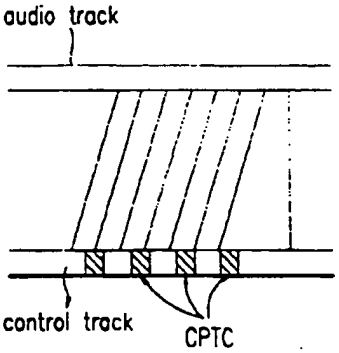


FIG.8d

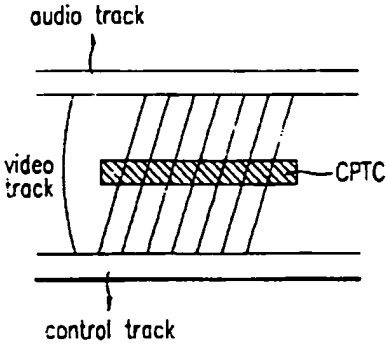


FIG.9

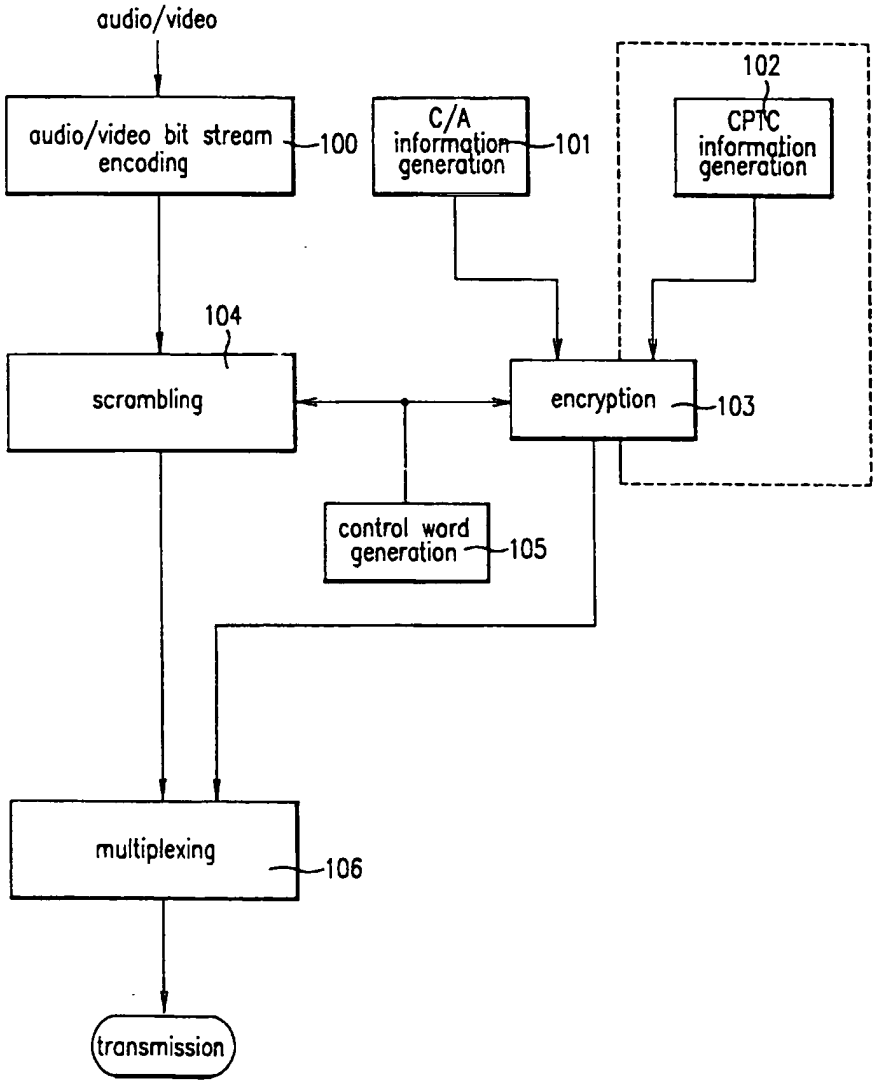
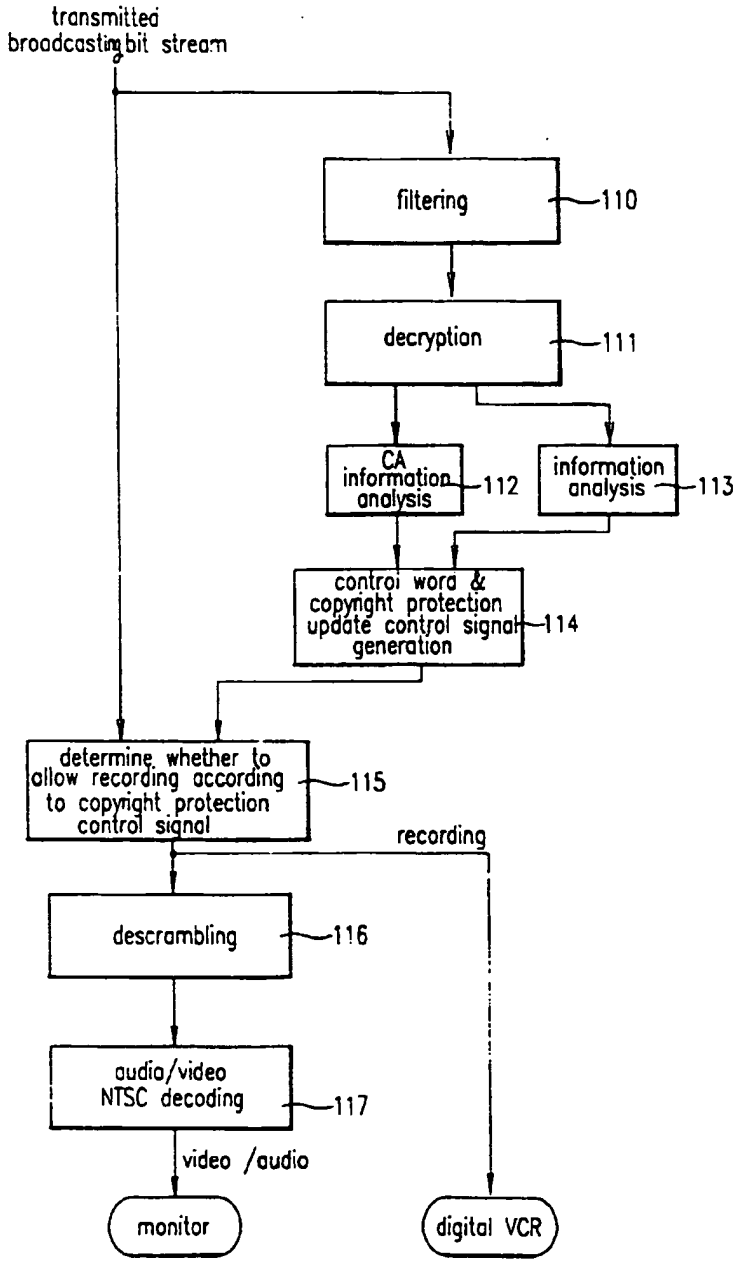
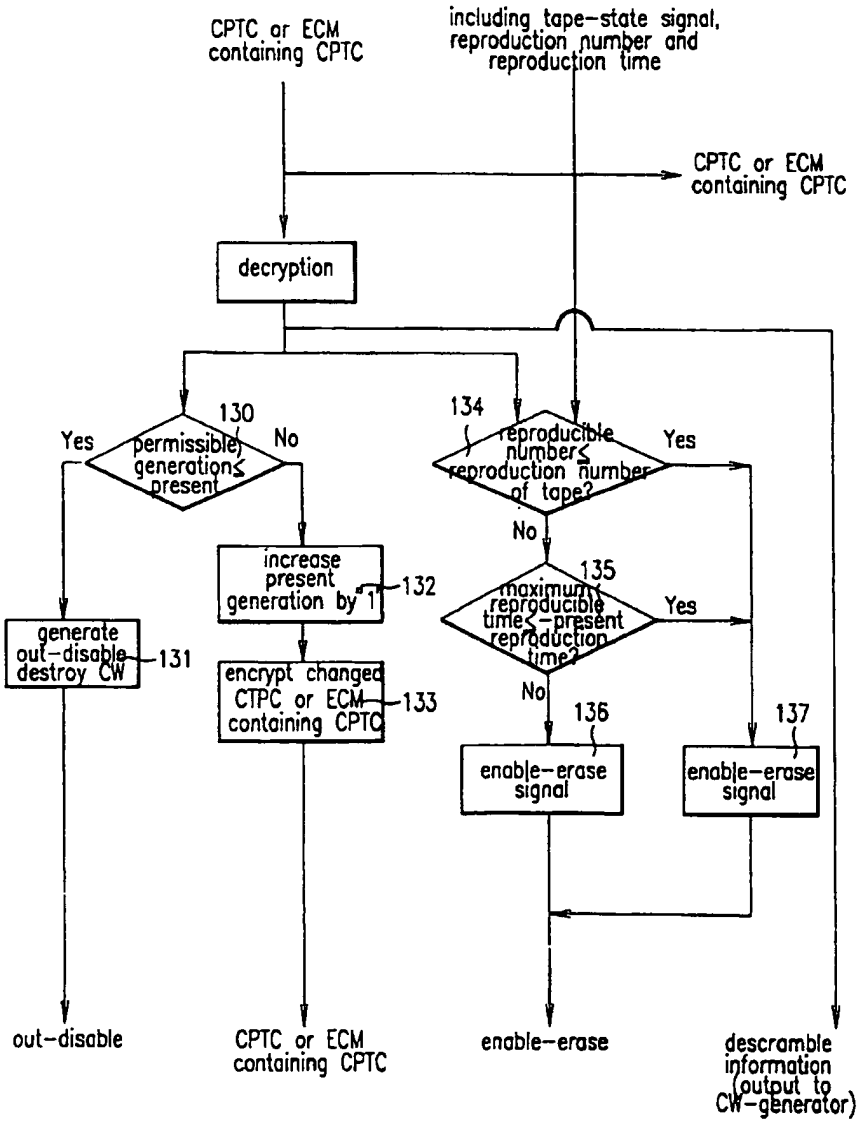


FIG.10



F I G.11



F I G.12

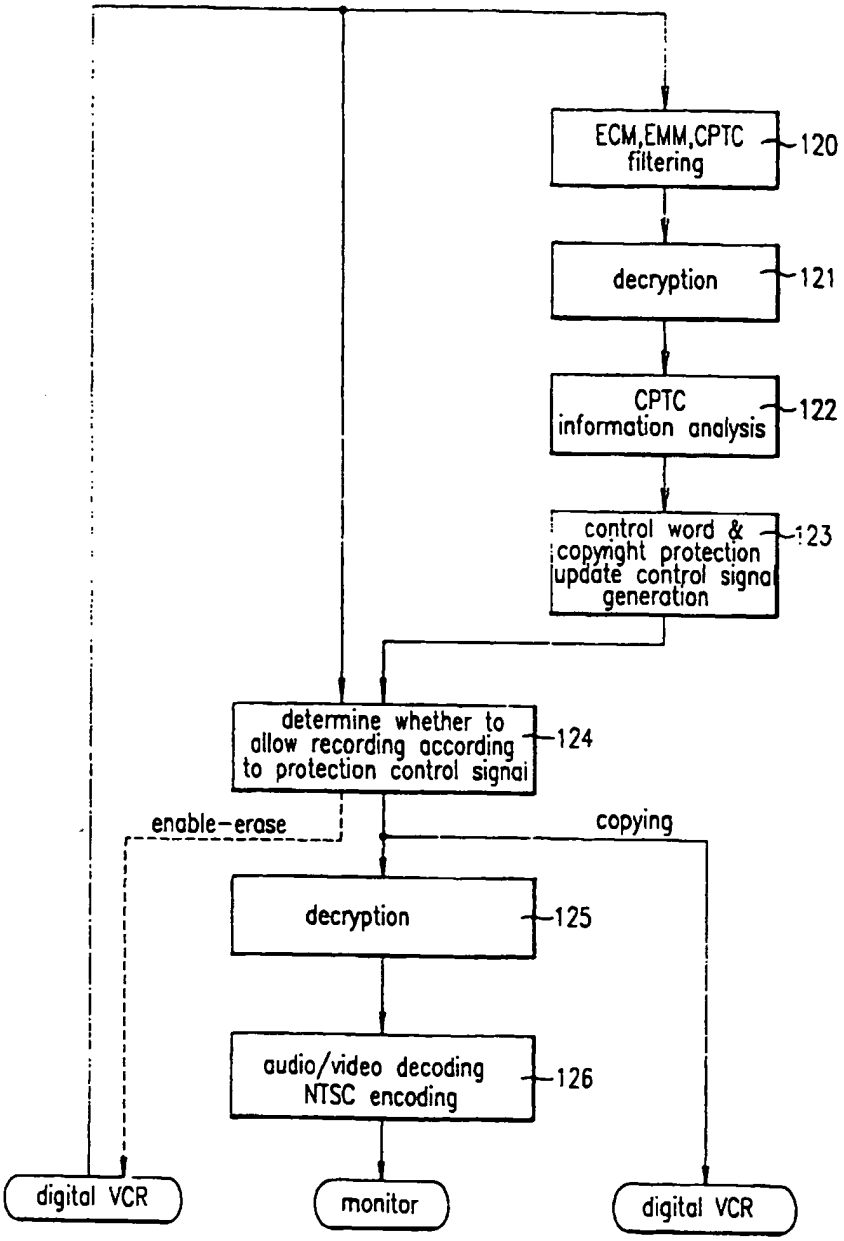


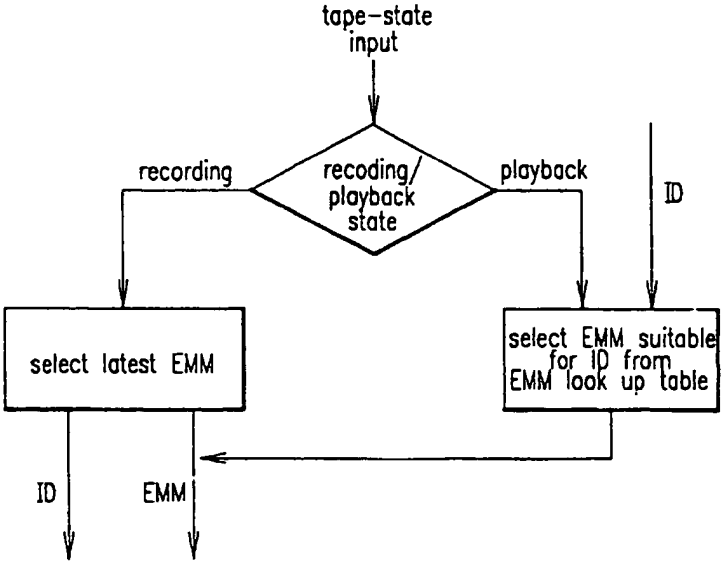
FIG.13

ID_1	EMM_1
ID_2	EMM_2
ID_3	EMM_3
⋮	⋮
ID_n	EMM_n

FIG.14

recording/reproduction state	ID	reproduction number

FIG.15



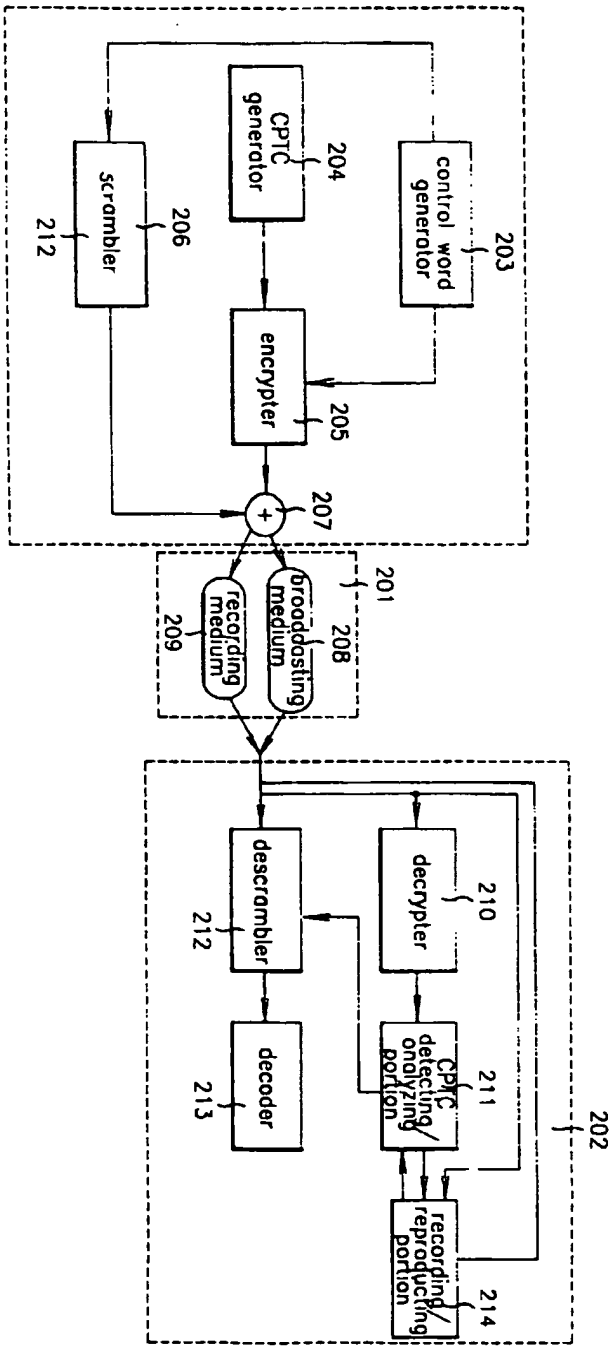


FIG.16

FIG.17a

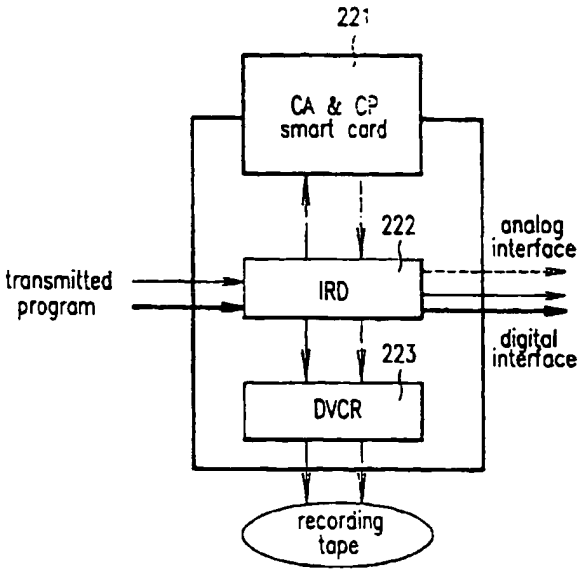


FIG.17b

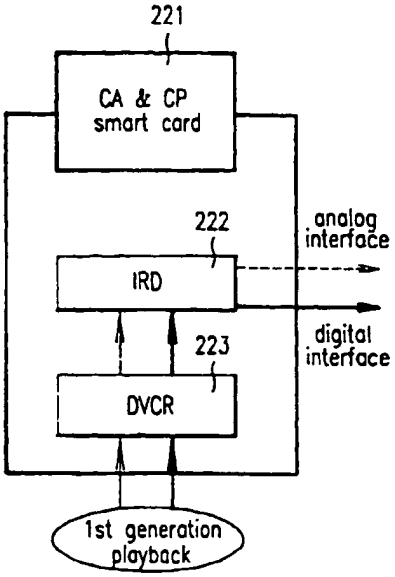


FIG.18

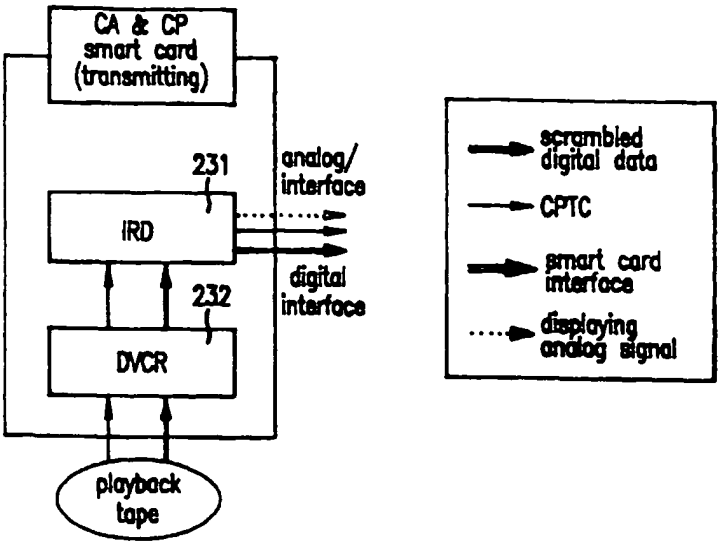


FIG.19

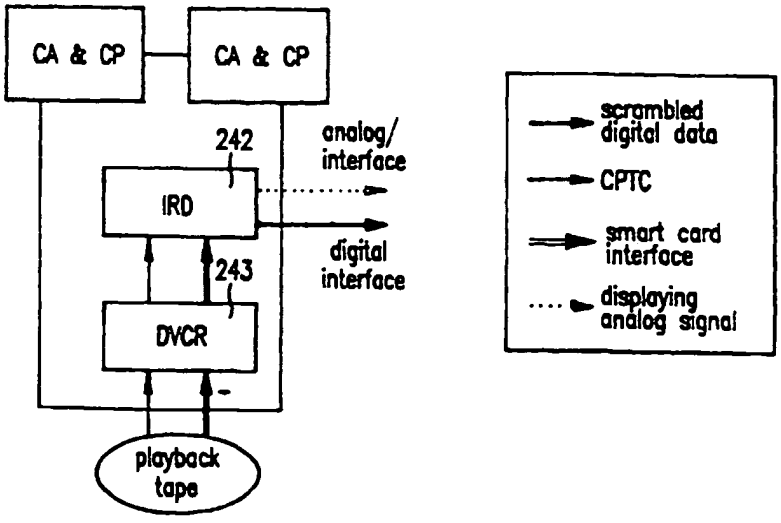


FIG.20

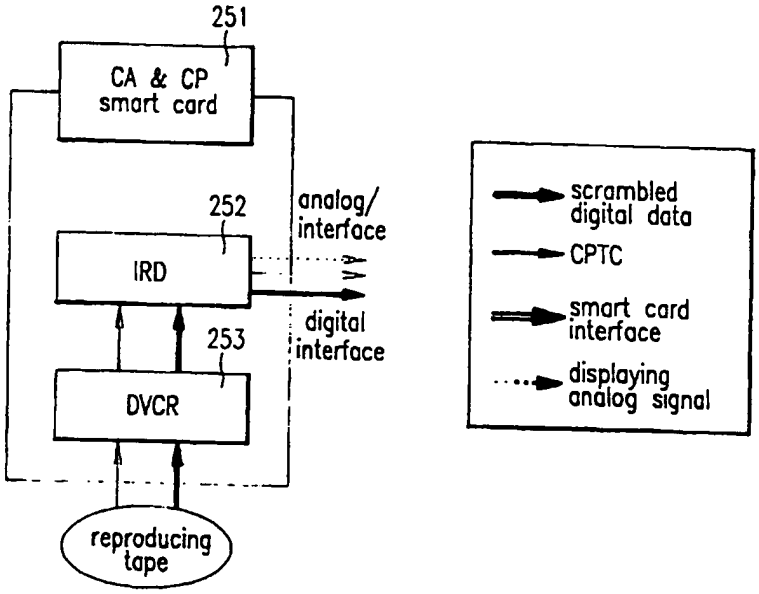


FIG.21

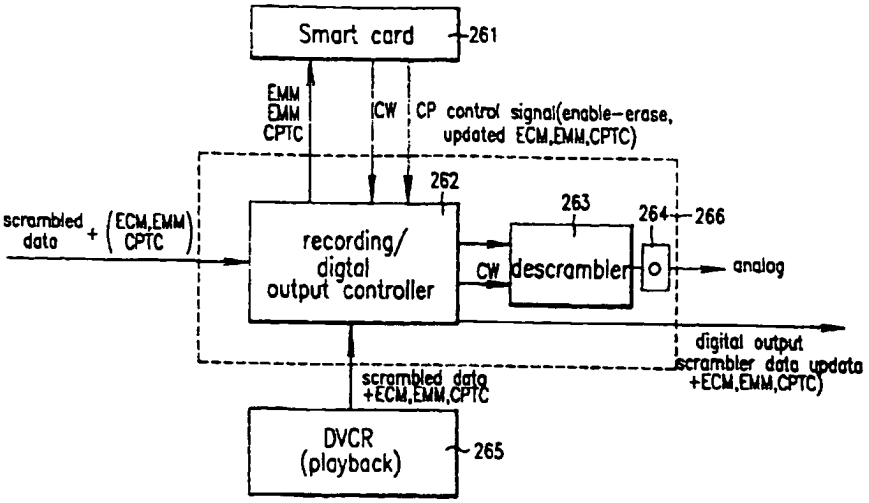
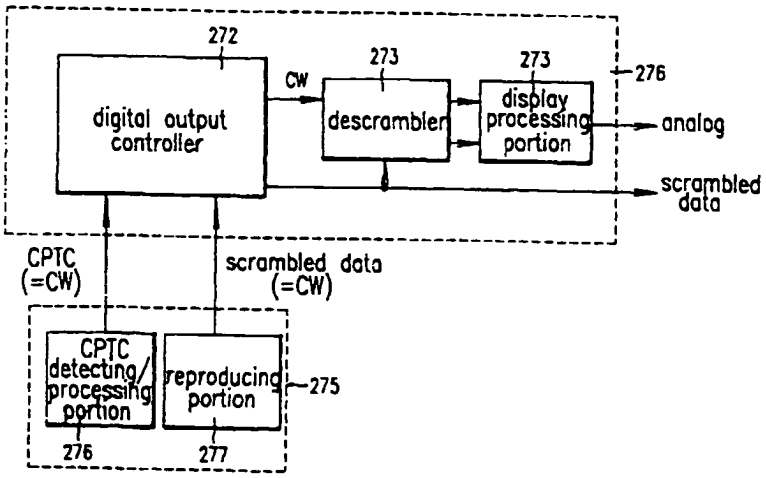
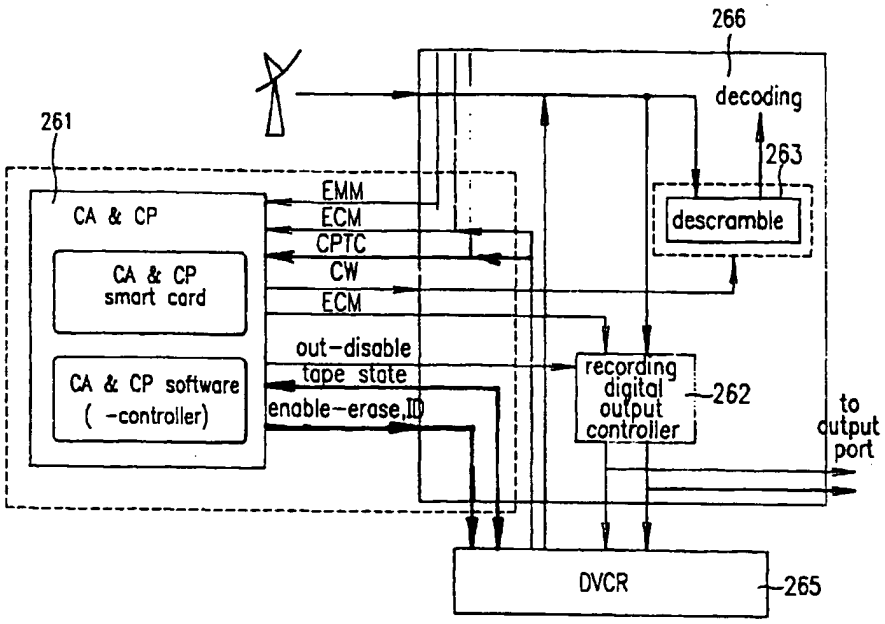


FIG.22



F I G.23



F I G.24

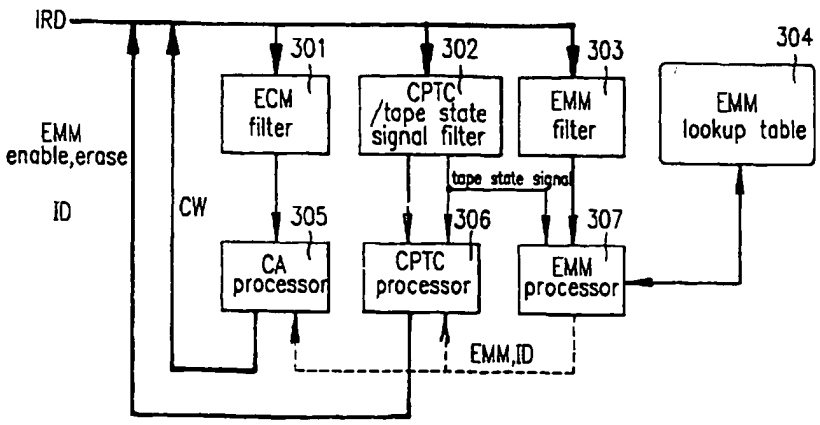


FIG.25

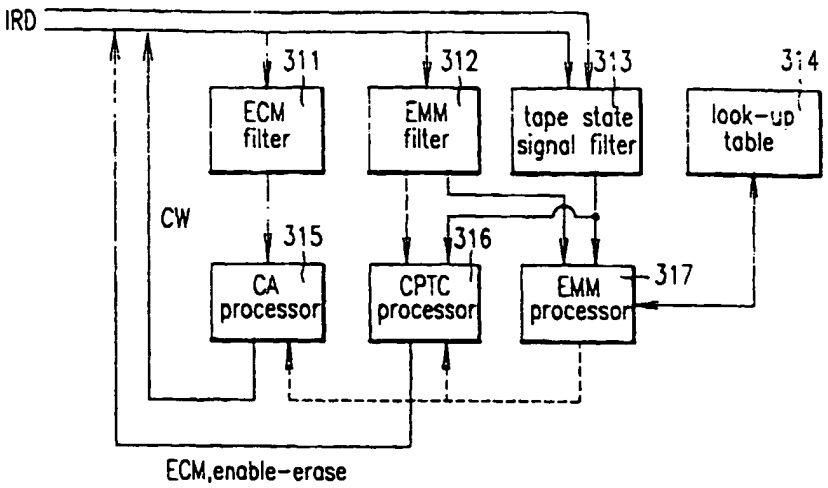
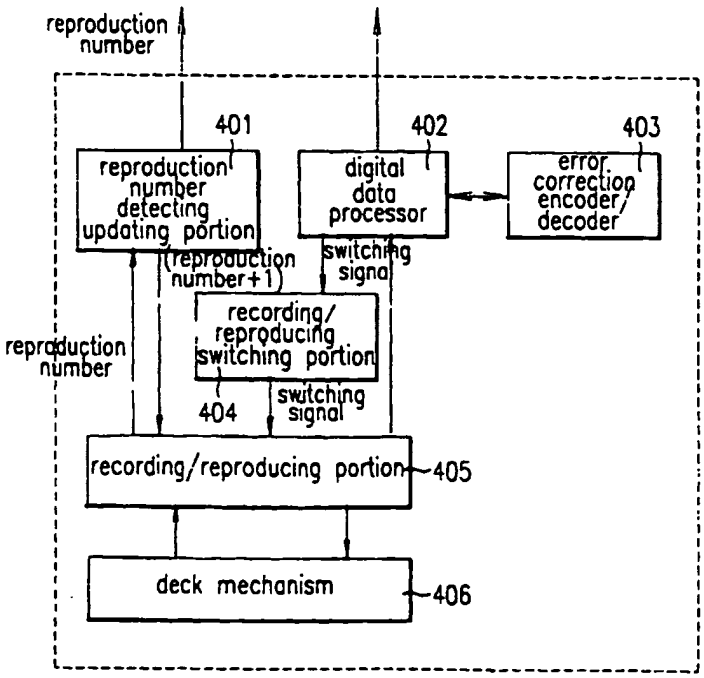


FIG.26



(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 715 244 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
05.06.1996 Bulletin 1996/23

(51) Int Cl.[6]: **G06F 1/00**

(21) Application number: **95308417.5**

(22) Date of filing: **23.11.1995**

(84) Designated Contracting States:
DE FR GB

(30) Priority: **23.11.1994 US 334041**

(71) Applicant: **XEROX CORPORATION**
Rochester New York 14644 (US)

(72) Inventor: **Stefik, Mark J.**
Woodside, California 94062 (US)

(74) Representative: **Goode, Ian Roy**
Rank Xerox Ltd
Patent Department
Parkway
Marlow Buckinghamshire SL7 1YL (GB)

(54) System for controlling the distribution and use of digital works utilizing a usage rights grammar

(57) A system for controlling use and distribution of digital works. The present invention allows the owner of a digital work to attach usage rights (1450) to their work. The usage rights define how the individual digital work may be used and distributed (1451). Instances of usage rights are defined using a flexible and extensible usage rights grammar. Conceptually, a right in the usage rights grammar is a label associated with a predetermined behavior and conditions to exercising the right. The behavior of a usage right is embodied in a predetermined set (1452) of usage transactions steps. The usage transaction steps further check all conditions (1453-1457) which must be satisfied before the right may be exercised. These usage transaction steps define a protocol for requesting the exercise of a right and the carrying out of a right.

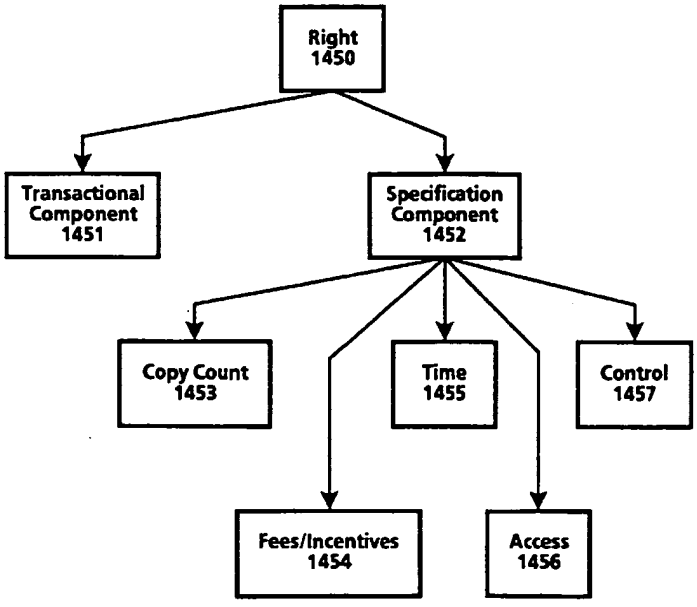


Fig.14

EP 0 715 244 A1

Printed by Jouve, 75001 PARIS (FR)

Description

The present invention relates to the field of distribution and usage rights enforcement for digitally encoded works.

A fundamental issue facing the publishing and information industries as they consider electronic publishing is how to prevent the unauthorized and unaccounted distribution or usage of electronically published materials. Electronically published materials are typically distributed in a digital form and recreated on a computer based system having the capability to recreate the materials. Audio and video recordings, software, books and multimedia works are all being electronically published. Companies in these industries receive royalties for each accounted for delivery of the materials, e.g. the sale of an audio CD at a retail outlet. Any unaccounted distribution of a work results in an unpaid royalty (e.g. copying the audio recording CD to another digital medium.)

The ease in which electronically published works can be "perfectly" reproduced and distributed is a major concern. The transmission of digital works over networks is commonplace. One such widely used network is the Internet. The Internet is a widespread network facility by which computer users in many universities, corporations and government entities communicate and trade ideas and information. Computer bulletin boards found on the Internet and commercial networks such as CompuServ and Prodigy allow for the posting and retrieving of digital information. Information services such as Dialog and LEXIS/NEXIS provide databases of current information on a wide variety of topics. Another factor which will exacerbate the situation is the development and expansion of the National Information Infrastructure (the NII). It is anticipated that, as the NII grows, the transmission of digital works over networks will increase many times over. It would be desirable to utilize the NII for distribution of digital works without the fear of widespread unauthorized copying.

The most straightforward way to curb unaccounted distribution is to prevent unauthorized copying and transmission. For existing materials that are distributed in digital form, various safeguards are used. In the case of software, copy protection schemes which limit the number of copies that can be made or which corrupt the output when copying is detected have been employed. Another scheme causes software to become disabled after a predetermined period of time has lapsed. A technique used for workstation based software is to require that a special hardware device must be present on the workstation in order for the software to run, e.g., see US-A-4,932,054 entitled "Method and Apparatus for Protecting Computer Software Utilizing Coded Filter Network in Conjunction with an Active Coded Hardware Device." Such devices are provided with the software and are commonly referred to as dongles.

Yet another scheme is to distribute software, but which requires a "key" to enable its use. This is employed in distribution schemes where "demos" of the software are provided on a medium along with the entire product. The demos can be freely used, but in order to use the actual product, the key must be purchased. These schemes do not hinder copying of the software once the key is initially purchased.

It is an object of the present invention to provide an improved system and method for controlling the use and distribution of digital works.

The invention accordingly provides a system and method as claimed in the accompanying claims.

A system for controlling use and distribution of digital works is disclosed. A digital work is any written, aural, graphical or video based work that has been translated to or created in a digital form, and which can be recreated using suitable rendering means such as software programs. The present invention allows the owner of a digital work to attach usage rights to their work. The usage rights define how the digital work may be used and distributed. These usage rights become part of the digital work and are always honored.

Instances of usage rights are defined using a flexible and extensible usage rights grammar. Conceptually, a right in the usage rights grammar is a label associated with a predetermined behavior and conditions to exercising the right. For example, a COPY right denotes that a copy of the digital work may be made. A condition to exercising the right is that the requester must pass certain security criteria. Conditions may also be attached to limit the right itself. For example, a LOAN right may be defined so as to limit the duration of which a work may be LOANed.

In the present invention a usage right is comprised of a right code along with the various conditions for exercising the right. Such conditions include a copy-count condition for limiting the number of times a right can be concurrently exercised (e.g. limit the number of copies on loan to some predetermined number), a security class condition for insuring that a repository has an appropriate level of security, access conditions for specifying access tests that must be passed, a time specification for indicating time based constraints for exercising a right and a fee specification for indicating usage fees for the exercise of a right. A digital work may have different versions of a right attached thereto. A version of a right will have the same right code as other versions, but the conditions (and typically the fees) would be different.

Digital works and their attached usage rights are stored in repositories. Digital works are transmitted between repositories. Repositories interact to exchange digital works according to a predetermined set of usage transactions steps. The behavior of a usage right is embodied in a predetermined set of usage transactions steps. The usage transaction steps further check all conditions which must be satisfied before the right may be exercised. These usage transaction steps define a protocol used by the repositories for requesting the exercise of a right and the carrying out of a right.

A system and method in accordance with the invention will now be described, by way of example, with reference to the accompanying drawings, in which:-

Figure 1 is a flowchart illustrating a simple instantiation of the operation of the currently preferred embodiment of the present invention.

Figure 2 is a block diagram illustrating the various repository types and the repository transaction flow between them in the currently preferred embodiment of the present invention.

Figure 3 is a block diagram of a repository coupled with a credit server in the currently preferred embodiment of the present invention.

Figures 4a and 4b are examples of rendering systems as may be utilized in the currently preferred embodiment of the present invention.

Figure 5 illustrates a contents file layout for a digital work as may be utilized in the currently preferred embodiment of the present invention.

Figure 6 illustrates a contents file layout for an individual digital work of the digital work of Figure 5 as may be utilized in the currently preferred embodiment of the present invention.

Figure 7 illustrates the components of a description block of the currently preferred embodiment of the present invention.

Figure 8 illustrates a description tree for the contents file layout of the digital work illustrated in Figure 5.

Figure 9 illustrates a portion of a description tree corresponding to the individual digital work illustrated in Figure 6.

Figure 10 illustrates a layout for the rights portion of a description block as may be utilized in the currently preferred embodiment of the present invention.

Figure 11 is a description tree wherein certain d-blocks have PRINT usage rights and is used to illustrate "strict" and "lenient" rules for resolving usage rights conflicts.

Figure 12 is a block diagram of the hardware components of a repository as are utilized in the currently preferred embodiment of the present invention.

Figure 13 is a block diagram of the functional (logical) components of a repository as are utilized in the currently preferred embodiment of the present invention.

Figure 14 is diagram illustrating the basic components of a usage right in the currently preferred embodiment of the present invention.

Figure 15 lists the usage rights grammar of the currently preferred embodiment of the present invention.

Figure 16 is a flowchart illustrating the steps of certificate delivery, hotlist checking and performance testing as performed in a registration transaction as may be performed in the currently preferred embodiment of the present invention.

Figure 17 is a flowchart illustrating the steps of session information exchange and clock synchronization as may be performed in the currently preferred embodiment of the present invention, after each repository in the registration transaction has successfully completed the steps described in Figure 16.

Figure 18 is a flowchart illustrating the basic flow for a usage transaction, including the common opening and closing step, as may be performed in the currently preferred embodiment of the present invention.

Figure 19 is a state diagram of server and client repositories in accordance with a transport protocol followed when moving a digital work from the server to the client repositories, as may be performed in the currently preferred embodiment of the present invention.

OVERVIEW

A system for controlling use and distribution of digital works is disclosed. The present invention is directed to supporting commercial transactions involving digital works.

Herein the terms "digital work", "work" and "content" refer to any work that has been reduced to a digital representation. This would include any audio, video, text, or multimedia work and any accompanying interpreter (e.g. software) that may be required for recreating the work. The term composite work refers to a digital work comprised of a collection of other digital works. The term "usage rights" or "rights" is a term which refers to rights granted to a recipient of a digital work. Generally, these rights define how a digital work can be used and if it can be further distributed. Each usage right may have one or more specified conditions which must be satisfied before the right may be exercised.

Figure 1 is a high level flowchart omitting various details but which demonstrates the basic operation of the present invention. Referring to Figure 1, a creator creates a digital work, step 101. The creator will then determine appropriate usage rights and fees, attach them to the digital work, and store them in Repository 1, step 102. The determination of appropriate usage rights and fees will depend on various economic factors. The digital work remains securely in Repository 1 until a request for access is received. The request for access begins with a session initiation by another repository. Here a Repository 2 initiates a session with Repository 1, step 103. As will be described in greater detail below, this session initiation includes steps which helps to insure that the respective repositories are trustworthy. As-

suming that a session can be established, Repository 2 may then request access to the Digital Work for a stated purpose, step 104. The purpose may be, for example, to print the digital work or to obtain a copy of the digital work. The purpose will correspond to a specific usage right. In any event, Repository 1 checks the usage rights associated with the digital work to determine if the access to the digital work may be granted, step 105. The check of the usage rights essentially involves a determination of whether a right associated with the access request has been attached to the digital work and if all conditions associated with the right are satisfied. If the access is denied, repository 1 terminates the session with an error message, step 106. If access is granted, repository 1 transmits the digital work to repository 2, step 107. Once the digital work has been transmitted to repository 2, repository 1 and 2 each generate billing information for the access which is transmitted to a credit server, step 108. Such double billing reporting is done to insure against attempts to circumvent the billing process.

Figure 2 illustrates the basic interactions between repository types in the present invention. As will become apparent from Figure 2, the various repository types will serve different functions. It is fundamental that repositories will share a core set of functionality which will enable secure and trusted communications. Referring to Figure 2, a repository 201 represents the general instance of a repository. The repository 201 has two modes of operation; a server mode and a requester mode. When in the server mode, the repository will be receiving and processing access requests to digital works. When in the requester mode, the repository will be initiating requests to access digital works. Repository 201 is general in the sense that its primary purpose is as an exchange medium for digital works. During the course of operation, the repository 201 may communicate with a plurality of other repositories, namely authorization repository 202, rendering repository 203 and master repository 204. Communication between repositories occurs utilizing a repository transaction protocol 205.

Communication with an authorization repository 202 may occur when a digital work being accessed has a condition requiring an authorization. Conceptually, an authorization is a digital certificate such that possession of the certificate is required to gain access to the digital work. An authorization is itself a digital work that can be moved between repositories and subjected to fees and usage rights conditions. An authorization may be required by both repositories involved in an access to a digital work.

Communication with a rendering repository 203 occurs in connection with the rendering of a digital work. As will be described in greater detail below, a rendering repository is coupled with a rendering device (e.g. a printer device) to comprise a rendering system.

Communication with a master repository 205 occurs in connection with obtaining an identification certificate. Identification certificates are the means by which a repository is identified as "trustworthy". The use of identification certificates is described below with respect to the registration transaction.

Figure 3 illustrates the repository 201 coupled to a credit server 301. The credit server 301 is a device which accumulates billing information for the repository 201. The credit server 301 communicates with repository 201 via billing transactions 302 to record billing transactions. Billing transactions are reported to a billing clearinghouse 303 by the credit server 301 on a periodic basis. The credit server 301 communicates to the billing clearinghouse 303 via clearinghouse transactions 304. The clearinghouse transactions 304 enable a secure and encrypted transmission of information to the billing clearinghouse 303.

RENDERING SYSTEMS

A rendering system is generally defined as a system comprising a repository and a rendering device which can render a digital work into its desired form. Examples of a rendering system may be a computer system, a digital audio system, or a printer. A rendering system has the same security features as a repository. The coupling of a rendering repository with the rendering device may occur in a manner suitable for the type of rendering device.

Figure 4a illustrates a printer as an example of a rendering system. Referring to Figure 4, printer system 401 has contained therein a printer repository 402 and a print device 403. It should be noted that the the dashed line defining printer system 401 defines a secure system boundary. Communications within the boundary are assumed to be secure. Depending on the security level, the boundary also represents a barrier intended to provide physical integrity. The printer repository 402 is an instantiation of the rendering repository 205 of Figure 2. The printer repository 402 will in some instances contain an ephemeral copy of a digital work which remains until it is printed out by the print engine 403. In other instances, the printer repository 402 may contain digital works such as fonts, which will remain and can be billed based on use. This design assures that all communication lines between printers and printing devices are encrypted, unless they are within a physically secure boundary. This design feature eliminates a potential "fault" point through which the digital work could be improperly obtained. The printer device 403 represents the printer components used to create the printed output.

Also illustrated in Figure 4a is the repository 404. The repository 404 is coupled to the printer repository 402. The repository 404 represents an external repository which contains digital works.

Figure 4b is an example of a computer system as a rendering system. A computer system may constitute a "multi-

function" device since it may execute digital works (e.g. software programs) and display digital works (e.g. a digitized photograph). Logically, each rendering device can be viewed as having its own repository, although only one physical repository is needed. Referring to Figure 4b, a computer system 410 has contained therein a display/execution repository 411. The display/execution repository 411 is coupled to display device, 412 and execution device 413. The dashed box surrounding the computer system 410 represents a security boundary within which communications are assumed to be secure. The display/execution repository 411 is further coupled to a credit server 414 to report any fees to be billed for access to a digital work and a repository 415 for accessing digital works stored therein.

STRUCTURE OF DIGITAL WORKS

Usage rights are attached directly to digital works. Thus, it is important to understand the structure of a digital work. The structure of a digital work, in particular composite digital works, may be naturally organized into an acyclic structure such as a hierarchy. For example, a magazine has various articles and photographs which may have been created and are owned by different persons. Each of the articles and photographs may represent a node in a hierarchical structure. Consequently, controls, i.e. usage rights, may be placed on each node by the creator. By enabling control and fee billing to be associated with each node, a creator of a work can be assured that the rights and fees are not circumvented.

In the currently preferred embodiment, the file information for a digital work is divided into two files: a "contents" file and a "description tree" file. From the perspective of a repository, the "contents" file is a stream of addressable bytes whose format depends completely on the interpreter used to play, display or print the digital work. The description tree file makes it possible to examine the rights and fees for a work without reference to the content of the digital work. It should be noted that the term description tree as used herein refers to any type of acyclic structure used to represent the relationship between the various components of a digital work.

Figure 5 illustrates the layout of a contents file. Referring to Figure 5, a digital work is comprised of story A 510, advertisement 511, story B 512 and story C 513. It is assumed that the digital work is stored starting at a relative address of 0. Each of the parts of the digital work are stored linearly so that story A 510 is stored at approximately addresses 0-30,000, advertisement 511 at addresses 30,001-40,000, story B 512 at addresses 40,001-60,000 and story C 513 at addresses 60,001-85K. The detail of story A 510 is illustrated in Figure 6. Referring to Figure 6, the story A 510 is further broken down to show text 614 stored at address 0-1500, soldier photo 615 at addresses 1501-10,000, graphics 616 stored at addresses 10,001-25,000 and sidebar 617 stored address 25,001-30,000. Note that the data in the contents file may be compressed (for saving storage) or encrypted (for security).

From Figures 5 and 6 it is readily observed that a digital work can be represented by its component parts as a hierarchy. The description tree for a digital work is comprised of a set of related descriptor blocks (d-blocks). The contents of each d-block is described with respect to Figure 7. Referring to Figure 7, a d-block 700 includes an identifier 701 which is a unique identifier for the work in the repository, a starting address 702 providing the start address of the first byte of the work, a length 703 giving the number of bytes in the work, a rights portion 704 wherein the granted usage rights and their status data are maintained, a parent pointer 705 for pointing to a parent d-block and child pointers 706 for pointing to the child d-blocks. In the currently preferred embodiment, the identifier 701 has two parts. The first part is a unique number assigned to the repository upon manufacture. The second part is a unique number assigned to the work upon creation. The rights portion 704 will contain a data structure, such as a look-up table, wherein the various information associated with a right is maintained. The information required by the respective usage rights is described in more detail below. D-blocks form a strict hierarchy. The top d-block of a work has no parent; all other d-blocks have one parent. The relationship of usage rights between parent and child d-blocks and how conflicts are resolved is described below.

A special type of d-block is a "shell" d-block. A shell d-block adds no new content beyond the content of its parts. A shell d-block is used to add rights and fee information, typically by distributors of digital works.

Figure 8 illustrates a description tree for the digital work of Figure 5. Referring to Figure 8, a top d-block 820 for the digital work points to the various stories and advertisements contained therein. Here, the top d-block 820 points to d-block 821 (representing story A 510), d-block 822 (representing the advertisement 511), d-block 823 (representing story B 512) and and d-block 824 (representing story C 513).

The portion of the description tree for Story A 510 is illustrated in Figure 9. D-block 925 represents text 614, d-block 926 represents photo 615, d-block 927 represents graphics 616 by and d-block 928 represents sidebar 617.

The rights portion 704 of a descriptor block is further illustrated in Figure 10. Figure 10 illustrates a structure which is repeated in the rights portion 704 for each right. Referring to Figure 10, each right will have a right code field 1050 and status information field 1052. The right code field 1050 will contain a unique code assigned to a right. The status information field 1052 will contain information relating to the state of a right and the digital work. Such information is indicated below in Table 1. The rights as stored in the rights portion 704 may typically be in numerical order based on the right code.

TABLE 1

DIGITAL WORK STATE INFORMATION		
Property	**Value**	**Use**
Copies-in-Use	Number	A counter of the number of copies of a work that are in use. Incremented when another copy is used; decremented when use is completed.
Loan-Period	Time-Units	Indicator of the maximum number of time-units that a document can be loaned out
Loaner-Copy	Boolean	Indicator that the current work is a loaned out copy of an authorized digital work.
Remaining-Time	Time-Units	Indicator of the remaining time of use on a metered document right.
Document-Descr	String	A string containing various identifying information about a document. The exact format of this is not specified, but it can include information such as a publisher name, author name, ISBN number, and so on.
Revenue-Owner	RO-Descr	A handle identifying a revenue owner for a digital work. This is used for reporting usage fees.
Publication-Date	Date-Descr	The date that the digital work was published.
History-list	History-Rec	A list of events recording the repostories and dates for operations that copy, transfer, backup, or restore a digital work.

The approach for representing digtal works by separating description data from content assumes that parts of a file are contiguous but takes no position on the actual representation of content. In particular, it is neutral to the question of whether content representation may take an object oriented approach. It would be natural to represent content as objects. In principle, it may be convenient to have content objects that include the billing structure and rights information that is represented in the d-blocks. Such variations in the design of the representation are possible and are viable alternatives but may introduce processing overhead, e.g. the interpretation of the objects.

Digital works are stored in a repository as part of a hierarchical file system. Folders (also termed directories and sub-directories) contain the digital works as well as other folders. Digital works and folders in a folder are ordered in alphabetical order. The digital works are typed to reflect how the files are used. Usage rights can be attached to folders so that the folder itself is treated as a digital work. Access to the folder would then be handled in the same fashion as any other digital work As will be described in more detail below, the contents of the folder are subject to their own rights. Moreover, file management rights may be attached to the folder which define how folder contents can be managed.

ATTACHING USAGE RIGHTS TO A DIGITAL WORK

It is fundamental to the present invention that the usage rights are treated as part of the digital work. As the digital work is distributed, the scope of the granted usage rights will remain the same or may be narrowed. For example, when a digital work is transferred from a document server to a repository, the usage rights may include the right to loan a copy for a predetermined period of time (called the original rights). When the repository loans out a copy of the digital work, the usage rights in the loaner copy (called the next set of rights) could be set to prohibit any further rights to loan out the copy. The basic idea is that one cannot grant more rights than they have.

The attachment of usage rights into a digital work may occur in a variety of ways. If the usage rights will be the same for an entire digital work, they could be attached when the digital work is processed for deposit in the digital work server. In the case of a digital work having different usage rights for the various components, this can be done as the digital work is being created. An authoring tool or digital work assembling tool could be utilized which provides for an automated process of attaching the usage rights.

As will be described below, when a digital work is copied, transferred or loaned, a "next set of rights" can be specified. The"next set of rights" will be attached to the digital work as it is transported.

Resolving Conflicting Rights

Because each part of a digital work may have its own usage rights, there will be instances where the rights of a "contained part" are different from its parent or container part. As a result, conflict rules must be established to dictate when and how a right may be exercised. The hierarchical structure of a digital work facilitates the enforcement of such

rules. A "strict" rule would be as follows: a right for a part in a digital work is sanctioned if and only if it is sanctioned for the part, for ancestor d-blocks containing the part and for all descendent d-blocks. By sanctioned, it is meant that (1) each of the respective parts must have the right, and (2) any conditions for exercising the right are satisfied.

It also possible to implement the present invention using a more lenient rule. In the more lenient rule, access to the part may be enabled to the descendent parts which have the right, but access is denied to the descendents which do not.

An example of applying both the strict rule and lenient is illustrated with reference to Figure 11. Referring to Figure 11, a root d-block 1101 has child d-blocks 1102-1105. In this case, root d-block represents a magazine, and each of the child d-blocks 1102-1105 represent articles in the magazine. Suppose that a request is made to PRINT the digital work represented by root d-block 1101 wherein the strict rule is followed. The rights for the root d-block 1101 and child d-blocks 1102-1105 are then examined. Root d-block 1101 and child d-blocks 1102 and 1105 have been granted PRINT rights. Child d-block 1103 has not been granted PRINT rights and child d-block 1104 has PRINT rights conditioned on payment of a usage fee.

Under the strict rule the PRINT right cannot be exercised because the child d-block does not have the PRINT right. Under the lenient rule, the result would be different. The digital works represented by child d-blocks 1102 and 1105 could be printed and the digital work represented by d-block 1104 could be printed so long as the usage fee is paid. Only the digital work represented by d-block 1103 could not be printed. This same result would be accomplished under the strict rule if the requests were directed to each of the individual digital works.

The present invention supports various combinations of allowing and disallowing access. Moreover, as will be described below, the usage rights grammar permits the owner of a digital work to specify if constraints may be imposed on the work by a container part. The manner in which digital works may be sanctioned because of usage rights conflicts would be implementation specific and would depend on the nature of the digital works.

REPOSITORIES

In the description of Figure 2, it was indicated that repositories come in various forms. All repositories provide a core set of services for the transmission of digital works. The manner in which digital works are exchanged is the basis for all transaction between repositories. The various repository types differ in the ultimate functions that they perform. Repositories may be devices themselves, or they may be incorporated into other systems. An example is the rendering repository 203 of Figure 2.

A repository will have associated with it a repository identifier. Typically, the repository identifier would be a unique number assigned to the repository at the time of manufacture. Each repository will also be classified as being in a particular security class. Certain communications and transactions may be conditioned on a repository being in a particular security class. The various security classes are described in greater detail below.

As a prerequisite to operation, a repository will require possession of an identification certificate. Identification certificates are encrypted to prevent forgery and are issued by a Master repository. A master repository plays the role of an authorization agent to enable repositories to receive digital works. Identification certificates must be updated on a periodic basis. Identification certificates are described in greater detail below with respect to the registration transaction.

A repository has both a hardware and functional embodiment. The functional embodiment is typically software executing on the hardware embodiment. Alternatively, the functional embodiment may be embedded in the hardware embodiment such as an Application Specific Integrated Circuit (ASIC) chip.

The hardware embodiment of a repository will be enclosed in a secure housing which if compromised, may cause the repository to be disabled. The basic components of the hardware embodiment of a repository are described with reference to Figure 12. Referring to Figure 12, a repository is comprised of a processing means 1200, storage system 1207, clock 1205 and external interface 1206. The processing means 1200 is comprised of a processor element 1201 and processor memory 1202. The processing means 1201 provides controller, repository transaction and usage rights transaction functions for the repository. Various functions in the operation of the repository such as decryption and/or decompression of digital works and transaction messages are also performed by the processing means 1200. The processor element 1201 may be a microprocessor or other suitable computing component. The processor memory 1202 would typically be further comprised of Read Only Memories (ROM) and Random Access Memories (RAM). Such memories would contain the software instructions utilized by the processor element 1201 in performing the functions of the repository.

The storage system 1207 is further comprised of descriptor storage 1203 and content storage 1204. The description tree storage 1203 will store the description tree for the digital work and the content storage will store the associated content. The description tree storage 1203 and content storage 1204 need not be of the same type of storage medium, nor are they necessarily on the same physical device. So for example, the descriptor storage 1203 may be stored on a solid state storage (for rapid retrieval of the description tree information), while the content storage 1204 may be on

a high capacity storage such as an optical disk.

The clock 1205 is used to time-stamp various time based conditions for usage rights or for metering usage fees which may be associated with the digital works. The clock 1205 will have an uninterruptable power supply, e.g. a battery, in order to maintain the integrity of the time-stamps. The external interface means 1206 provides for the signal connection to other repositories and to a credit server. The external interface means 1206 provides for the exchange of signals via such standard interfaces such as RS-232 or Personal Computer Manufacturers Card Industry Association (PCMCIA) standards, or FDDI. The external interface means 1206 may also provide network connectivity.

The functional embodiment of a repository is described with reference to Figure 13. Referring to Figure 13, the functional embodiment is comprised of an operating system 1301, core repository services 1302, usage transaction handlers 1303, repository specific functions, 1304 and a user interface 1305. The operating system 1301 is specific to the repository and would typically depend on the type of processor being used. The operating system 1301 would also provide the basic services for controlling and interfacing between the basic components of the repository.

The core repository services 1302 comprise a set of functions required by each and every repository. The core repository services 1302 include the session initiation transactions which are defined in greater detail below. This set of services also includes a generic ticket agent which is used to "punch" a digital ticket and a generic authorization server for processing authorization specifications. Digital tickets and authorizations are specific mechanisms for controlling the distribution and use of digital works and are described in more detail below. Note that coupled to the core repository services are a plurality of identification certificates 1306. The identification certificates 1306 are required to enable the use of the repository.

The usage transactions handlers 1303 comprise functionality for processing access requests to digital works and for billing fees based on access. The usage transactions supported will be different for each repository type. For example, it may not be necessary for some repositories to handle access requests for digital works.

The repository specific functionality 1304 comprises functionality that is unique to a repository. For example, the master repository has special functionality for issuing digital certificates and maintaining encryption keys. The repository specific functionality 1304 would include the user interface implementation for the repository.

Repository Security Classes

For some digital works the losses caused by any individual instance of unauthorized copying is insignificant and the chief economic concern lies in assuring the convenience of access and low-overhead billing. In such cases, simple and inexpensive handheld repositories and network-based workstations may be suitable repositories, even though the measures and guarantees of security are modest.

At the other extreme, some digital works such as a digital copy of a first run movie or a bearer bond or stock certificate would be of very high value so that it is prudent to employ caution and fairly elaborate security measures to ensure that they are not copied or forged. A repository suitable for holding such a digital work could have elaborate measures for ensuring physical integrity and for verifying authorization before use.

By arranging a universal protocol, all kinds of repositories can communicate with each other in principle. However, creators of some works will want to specify that their works will only be transferred to repositories whose level of security is high enough. For this reason, document repositories have a ranking system for classes and levels of security. The security classes in the currently preferred embodiment are described in Table 2.

TABLE 2

REPOSITORY SECURITY LEVELS	
Level	Description of Security
0	Open system. Document transmission is unencrypted. No digital certificate is required for identification. The security of the system depends mostly on user honesty, since only modest knowledge may be needed to circumvent the security measures. The repository has no provisions for preventing unauthorized programs from running and accessing or copying files. The system does not prevent the use of removable storage and does not encrypt stored files.
1	Minimal security. Like the previous class except that stored files are minimally encrypted, including ones on removable storage.
2	Basic security. Like the previous class except that special tools and knowledge are required to compromise the programming, the contents of the repository, or the state of the clock. All digital communications are encrypted. A digital certificate is provided as identification. Medium level encryption is used. Repository identification number is unforgeable.

TABLE 2 (continued)

REPOSITORY SECURITY LEVELS	
Level	Description of Security
3	General security. Like the previous class plus the requirement of special tools are needed to compromise the physical integrity of the repository and that modest encryption is used on all transmissions. Password protection is required to use the local user interface. The digital clock system cannot be reset without authorization. No works would be stored on removable storage. When executing works as programs, it runs them in their own address space and does not give them direct access to any file storage or other memory containing system code or works. They can access works only through the transmission transaction protocol.
4	Like the previous class except that high level encryption is used on all communications. Sensors are used to record attempts at physical and electronic tampering. After such tampering, the repository will not perform other transactions until it has reported such tampering to a designated server.
5	Like the previous class except that if the physical or digital attempts at tampering exceed some preset threshold that threaten the physical integrity of the repository or the integrity of digital and cryptographic barriers, then the repository will save only document description records of history but will erase or destroy any digital identifiers that could be misused if released to an unscrupulous party. It also modifies any certificates of authenticity to indicate that the physical system has been compromised. It also erases the contents of designated documents.
6	Like the previous class except that the repository will attempt wireless communication to report tampering and will employ noisy alarms.
10	This would correspond to a very high level of security. This server would maintain constant communications to remote security systems reporting transactions, sensor readings, and attempts to circumvent security.

The characterization of security levels described in Table 2 is not intended to be fixed. More important is the idea of having different security levels for different repositories. It is anticipated that new security classes and requirements will evolve according to social situations and changes in technology.

Repository User Interface

A user interface is broadly defined as the mechanism by which a user interacts with a repository in order to invoke transactions to gain access to a digital work, or exercise usage rights. As described above, a repository may be embodied in various forms. The user interface for a repository will differ depending on the particular embodiment. The user interface may be a graphical user interface having icons representing the digital works and the various transactions that may be performed. The user interface may be a generated dialog in which a user is prompted for information.

The user interface itself need not be part of the repository. As a repository may be embedded in some other device, the user interface may merely be a part of the device in which the repository is embedded. For example, the repository could be embedded in a "card" that is inserted into an available slot in a computer system. The user interface may be a combination of a display, keyboard, cursor control device and software executing on the computer system.

At a minimum, the user interface must permit a user to input information such as access requests and alpha numeric data and provide feedback as to transaction status. The user interface will then cause the repository to initiate the suitable transactions to service the request. Other facets of a particular user interface will depend on the functionality that a repository will provide.

CREDIT SERVERS

In the present invention, fees may be associated with the exercise of a right. The requirement for payment of fees is described with each version of a usage right in the usage rights language. The recording and reporting of such fees is performed by the credit server. One of the capabilities enabled by associating fees with rights is the possibility of supporting a wide range of charging models. The simplest model, used by conventional software, is that there is a single fee at the time of purchase, after which the purchaser obtains unlimited rights to use the work as often and for as long as he or she wants. Alternative models, include metered use and variable fees. A single work can have different fees for different uses. For example, viewing a photograph on a display could have different fees than making a hardcopy

or including it in a newly created work. A key to these alternative charging models is to have a low overhead means of establishing fees and accounting for credit on these transactions.

A credit server is a computational system that reliably authorizes and records these transactions so that fees are billed and paid. The credit server reports fees to a billing clearinghouse. The billing clearinghouse manages the financial transactions as they occur. As a result, bills may be generated and accounts reconciled. Preferably, the credit server would store the fee transactions and periodically communicate via a network with the billing clearinghouse for reconciliation. In such an embodiment, communications with the billing clearinghouse would be encrypted for integrity and security reasons. In another embodiment, the credit server acts as a "debit card" where transactions occur in "real-time" against a user account.

A credit server is comprised of memory, a processing means, a clock, and interface means for coupling to a repository and a financial institution (e.g. a modem). The credit server will also need to have security and authentication functionality. These elements are essentially the same elements as those of a repository. Thus, a single device can be both a repository and a credit server, provided that it has the appropriate processing elements for carrying out the corresponding functions and protocols. Typically, however, a credit server would be a card-sized system in the possession of the owner of the credit. The credit server is coupled to a repository and would interact via financial transactions as described below. Interactions with a financial institution may occur via protocols established by the financial institutions themselves.

In the currently preferred embodiment credit servers associated with both the server and the repository report the financial transaction to the billing clearinghouse. For example, when a digital work is copied by one repository to another for a fee, credit servers coupled to each of the repositories will report the transaction to the billing clearinghouse. This is desirable in that it insures that a transaction will be accounted for in the event of some break in the communication between a credit server and the billing clearinghouse. However, some implementations may embody only a single credit server reporting the transaction to minimize transaction processing at the risk of losing some transactions.

USAGE RIGHTS LANGUAGE

The present invention uses statements in a high level "usage rights language" to define rights associated with digital works and their parts. Usage rights statements are interpreted by repositories and are used to determine what transactions can be successfully carried out for a digital work and also to determine parameters for those transactions. For example, sentences in the language determine whether a given digital work can be copied, when and how it can be used, and what fees (if any) are to be charged for that use. Once the usage rights statements are generated, they are encoded in a suitable form for accessing during the processing of transactions.

Defining usage rights in terms of a language in combination with the hierarchical representation of a digital work enables the support of a wide variety of distribution and fee schemes. An example is the ability to attach multiple versions of a right to a work. So a creator may attach a PRINT right to make 5 copies for $10.00 and a PRINT right to make unlimited copies for $100.00. A purchaser may then choose which option best fits his needs. Another example is that rights and fees are additive. So in the case of a composite work, the rights and fees of each of the components works is used in determining the rights and fees for the work as a whole.

The basic contents of a right are illustrated in Figure 14. Referring to Figure 14, a right 1450 has a transactional component 1451 and a specifications component 1452. A right 1450 has a label (e.g. COPY or PRINT) which indicates the use or distribution privileges that are embodied by the right. The transactional component 1451 corresponds to a particular way in which a digital work may be used or distributed. The transactional component 1451 is typically embodied in software instructions in a repository which implement the use or distribution privileges for the right. The specifications components 1452 are used to specify conditions which must be satisfied prior to the right being exercised or to designate various transaction related parameters. In the currently preferred embodiment, these specifications include copy count 1453, Fees and Incentives 1454, Time 1455, Access and Security 1456 and Control 1457. Each of these specifications will be described in greater detail below with respect to the language grammar elements.

The usage rights language is based on the grammar described below. A grammar is a convenient means for defining valid sequence of symbols for a language. In describing the grammar the notation "[a|b|c]" is used to indicate distinct choices among alternatives. In this example, a sentence can have either an "a", "b" or "c". It must include exactly one of them. The braces {} are used to indicate optional items. Note that brackets, bars and braces are used to describe the language of usage rights sentences but do not appear in actual sentences in the language.

In contrast, parentheses are part of the usage rights language. Parentheses are used to group items together in lists. The notation (x*) is used to indicate a variable length list, that is, a list containing one or more items of type x. The notation $(x)^*$ is used to indicate a variable number of lists containing x.

Keywords in the grammar are words followed by colons. Keywords are a common and very special case in the language. They are often used to indicate a single value, typically an identifier. In many cases, the keyword and the parameter are entirely optional. When a keyword is given, it often takes a single identifier as its value. In some cases,

the keyword takes a list of identifiers.

In the usage rights language, time is specified in an hours:minutes:seconds (or hh:mm:ss) representation. Time zone indicators, e.g. PDT for Pacific Daylight Time, may also be specified. Dates are represented as year/ month/day (or YYYY/MMM/DD). Note that these time and date representations may specify moments in time or units of time Money units are specified in terms of dollars.

Finally, in the usage rights language, various "things" will need to interact with each other. For example, an instance of a usage right may specify a bank account, a digital ticke etc.. Such things need to be identified and are specified herein using the suffix "-ID."

The Usage Rights Grammar is listed in its entirety in Figure 15 and is described below.

Grammar element 1501 **"Digital Work Rights:= (Rights*)"** define the digital wor rights as a set of rights. The set of rights attached to a digital work define how that digital work may be transferred, used, performed or played. A set of rights will attach to the entire digital work and in the case of compound digital works, each of the components of the digita work. The usage rights of components of a digital may be different.

Grammar element 1502 **"Right: = (Right-Code {Copy-Count} {Control-Spec} {Time-Spec} {Access-Spec} {Fee-Spec})"** enumerates the content of a right. Each usage right must specify a right code. Each right may also optionally specify conditions which must be satisfied before the right can be exercised. These conditions are copy count, control, time, access and fee conditions. In the currently preferred embodiment, for the optional elements, the following defaults apply: copy count equals 1, no time limit on the use of the right, no access tests or a security level required to use the right and no fee is required. These conditions will each be described in greater detail below.

It is important to note that a digital work may have multiple versions of a right, each having the same right code. The multiple version would provide alternative conditions and fees for accessing the digital work.

Grammar element 1503 **"Right-Code : = Render-Code I Transport-Code I File-Management-CodeI Derivative-Works- Code Configuration-Code"** distinguishes each of the specific rights into a particular right type (although each right is identified by distinct right codes). In this way, the grammar provides a catalog of possible rights that can be associated with parts of digital works. In the following, rights are divided into categories for convenience in describing them.

Grammar element 1504 **"Render-Code : = [Play:{Player:Player-ID}I Print: {Printer: Printer-ID}]"** lists a category of rights all involving the making of ephemeral, transitory, or non-digital copies of the digital work. After use the copies are erased.

- Play A process of rendering or performing a digital work on some processor. This includes such things as playing digital movies, playing digital music, playing a video game, running a computer program, or displaying a document on a display.
- Print To render the work in a medium that is not further protected by usage rights, such as printing on paper.

Grammar element 1505 **"Transport-Code : = [Copy I Transfer I Loan (Remaining-Rights: Next-Set-of-Rights)] {(Next-Copy-Rights: Next-Set of Rights)}"** lists a category of rights involving the making of persistent, usable copies of the digital work on other repositories. The optional Next-Copy-Rights determine the rights on the work after it is transported. If this is not specified, then the rights on the transported copy are the same as on the original. The optional Remaining-Rights specify the rights that remain with a digital work when it is loaned out. If this is not specified, then the default is that no rights can be exercised when it is loaned out.

- Copy Make a new copy of a work
- Transfer Moving a work from one repository to another.
- Loan Temporarily loaning a copy to another repository for a specified period of time.

Grammar element 1506 **"File-Management-Code: = Backup {Back-Up-Copy-Rights: Next-Set -of Rights}I Restore I Delete I Folder I Directory {Name:Hide-Local I Hide - Remote}{Parts:Hide-Local I Hide-Remote}"** lists a category of rights involving operations for file management, such as the making of backup copies to protect the copy owner against catastrophic equipment failure.

Many software licenses and also copyright law give a copy owner the right to make backup copies to protect against catastrophic failure of equipment. However, the making of uncontrolled backup copies is inherently at odds with the ability to control usage, since an uncontrolled backup copy can be kept and then restored even after the authorized copy was sold.

The File management rights enable the making and restoring of backup copies in a way that respects usage rights, honoring the requirements of both the copy owner and the rights grantor and revenue owner. Backup copies of work descriptions (including usage rights and fee data) can be sent under appropriate protocol and usage rights control to other document repositories of sufficiently high security. Further rights permit organization of digital works into folders

which themselves are treated as digital works and whose contents may be "hidden" from a party seeking to determine the contents of a repository.

- Backup To make a backup copy of a digital work as protection against media failure.
- Restore To restore a backup copy of a digital work.
- Delete To delete or erase a copy of a digital work.
- Folder To create and name folders, and to move files and folders between folders.
- Directory . To hide a folder or its.contents.

Grammar element 1507 **"Derivative-Works-Code: [Extract | Embed | Edit {Process: Process-ID}] {Next-Copy-Rights : Next-Set-of Rights}"** lists a category of rights involving the use of a digital work to create new works.

- Extract To remove a portion of a work, for the purposes of creating a new work.
- Embed To include a work in an existing work.
- Edit To alter a digital work by copying, selecting and modifying portions of an existing digital work.

Grammar element 1508 **"Configuration-Code: = Install | Uninstall"** lists a category of rights for installing and uninstalling software on a repository (typically a rendering repository.) This would typically occur for the installation of a new type of player within the rendering repository.

- Install: To install new software on a repository.
- Uninstall: To remove existing software from a repository.

Grammar element 1509 **"Next-Set-of-Rights: = {(Add: Set-Of-Rights)} {(Delete: Set-Of-Rights)} {(Replace: Set-Of-Rights)} {(Keep: Set-Of-Rights)}"** defines how rights are carried forward for a copy of a digital work. If the Next-Copy-Rights is not specified, the rights for the next copy are the same as those of the current copy. Otherwise, the set of rights for the next copy can be specified. Versions of rights after Add: are added to the current set of rights. Rights after Delete: are deleted from the current set of rights. If only right codes are listed after Delete:, then all versions of rights with those codes are deleted. Versions of rights after Replace: subsume all versions of rights of the same type in the current set of rights.

If Remaining-Rights is not specified, then there are no rights for the original after all Loan copies are loaned out. If Remaining-Rights is specified, then the Keep: token can be used to simplify the expression of what rights to keep behind. A list of right codes following keep means that all of the versions of those listed rights are kept in the remaining copy. This specification can be overridden by subsequent Delete: or Replace: specifications.

Copy Count Specification

For various transactions, it may be desirable to provide some limit as to the number of "copies" of the work which may be exercised simultaneously for the right. For example, it may be desirable to limit the number of copies of a digital work that may be loaned out at a time or viewed at a time.

Grammar element 1510 **"Copy-Count : = (Copies: positive-integer | 0 | unlimited)"** provides a condition which defines the number of "copies" of a work subject to the right . A copy count can be 0, a fixed number, or unlimited. The copy-count is associated with each right, as opposed to there being just a single copy-count for the digital work. The Copy-Count for a right is decremented each time that a right is exercised. When the Copy-Count equals zero, the right can no longer be exercised. If the Copy-Count is not specified, the default is one.

Control Specification

Rights and fees depend in general on rights granted by the creator as well as further restrictions imposed by later distributors. Control specifications deal with interactions between the creators and their distributors governing the imposition of further restrictions and fees. For example, a distributor of a digital work may not want an end consumer of a digital work to add fees or otherwise profit by commercially exploiting the purchased digital work.

Grammar element 1511 **"Control-Spec : = (Control: {Restrictable | Unrestrictable} {Unchargeable | Chargeable})"** provides a condition to specify the effect of usage rights and fees of parents on the exercise of the right. A digital work is restrictable if higher level d-blocks can impose further restrictions (time specifications and access specifications) on the right. It is unrestrictable if no further restrictions can be imposed. The default setting is restrictable. A right is unchargeable if no more fees can be imposed on the use of the right. It is chargeable if more fees can be imposed. The default is chargeable.

Time Specification

It is often desirable to assign a start date or specify some duration as to when a right may be exercised. Grammar element 1512 **"Time-Spec : = ({Fixed-Interval | Sliding-Interval | Meter-Time} Until: Expiration-Date)"** provides for specification of time conditions on the exercise of a right. Rights may be granted for a specified time. Different kinds of time specifications are appropriate for different kinds of rights. Some rights may be exercised during a fixed and predetermined duration. Some rights may be exercised for an interval that starts the first time that the right is invoked by some transaction. Some rights may be exercised or are charged according to some kind of metered time, which may be split into separate intervals. For example, a right to view a picture for an hour might be split into six ten minute viewings or four fifteen minute viewings or twenty three minute viewings.

The terms "time" and "date" are used synonymously to refer to a moment in time. There are several kinds of time specifications. Each specification represents some limitation on the times over which the usage right applies. The Expiration-Date specifies the moment at which the usage right ends. For example, if the Expiration-Date is "Jan 1, 1995," then the right ends at the first moment of 1995. If the Expiration-Date is specified as "forever", then the rights are interpreted as continuing without end. If only an expiration date is given, then the right can be exercised as often as desired until the expiration date.

Grammar element 1513 **"Fixed-Interval : = From: Start-Time"** is used to define a predetermined interval that runs from the start time to the expiration date.

Grammar element 1514 **"Sliding-Interval : = Interval: Use-Duration"** is used to define an indeterminate (or "open") start time. It sets limits on a continuous period of time over which the contents are accessible. The period starts on the first access and ends after the duration has passed or the expiration date is reached, whichever comes first. For example, if the right gives 10 hours of continuous access, the use-duration would begin when the first access was made and end 10 hours later.

Grammar element 1515 **"Meter-Time: = Time-Remaining: Remaining-Use"** is used to define a "meter time," that is, a measure of the time that the right is actually exercised. It differs from the Sliding-Interval specification in that the time that the digital work is in use need not be continuous. For example, if the rights guarantee three days of access, those days could be spread out over a month. With this specification, the rights can be exercised until the meter time is exhausted or the expiration date is reached, whichever comes first.

Remaining-Use: = Time-Unit
Start-Time: = Time-Unit
Use-Duration: = Time-Unit
All of the time specifications include time-unit specifications in their ultimate instantiation.

Security Class and Authorization Specification

The present invention provides for various security mechanisms to be introduced into a distribution or use scheme. Grammar element 1516 **"Access-Spec : ({SC: Security-Class} {Authorization: Authorization-ID*} {Other-Authorization: Authorization-ID*} {Ticket: Ticket-ID})"** provides a means for restricting access and transmission. Access specifications can specify a required security class for a repository to exercise a right or a required authorization test that must be satisfied.

The keyword **"SC:"** is used to specify a minimum security level for the repositories involved in the access. If **"SC: "** is not specified, the lowest security level is acceptable.

The optional **"Authorization:"** keyword is used to specify required authorizations on the same repository as the work. The optional **"Other-Authorization:"** keyword is used to specify required authorizations on the other repository in the transaction.

The optional **"Ticket:"** keyword specifies the identity of a ticket required for the transaction. A transaction involving digital tickets must locate an appropriate digital ticket agent who can "punch" or otherwise validate the ticket before the transaction can proceed. Tickets are described in greater detail below.

In a transaction involving a repository and a document server, some usage rights may require that the repository have a particular authorization, that the server have some authorization, or that both repositories have (possibly different) authorizations. Authorizations themselves are digital works (hereinafter referred to as an authorization object) that can be moved between repositories in the same manner as other digital works. Their copying and transferring is subject to the same rights and fees as other digital works. A repository is said to have an authorization if that authorization object is contained within the repository.

In some cases, an authorization may be required from a source other than the document server and repository. An authorization object referenced by an Authorization-ID can contain digital address information to be used to set up a communications link between a repository and the authorization source. These are analogous to phone numbers.

For such access tests, the communication would need to be established and authorization obtained before the right could be exercised.

For one-time usage rights, a variant on this scheme is to have a digital ticket. A ticket is presented to a digital ticket agent, whose type is specified on the ticket. In the simplest case, a certified generic ticket agent, available on all repositories, is available to "punch" the ticket. In other cases, the ticket may contain addressing information for locating a "special" ticket agent. Once a ticket has been punched, it cannot be used again for the same kind of transaction (unless it is unpunched or refreshed in the manner described below.) Punching includes marking the ticket with a timestamp of the date and time it was used. Tickets are digital works and can be copied or transferred between repositories according to their usage rights.

In the currently preferred embodiment, a "punched" ticket becomes "unpunched" or "refreshed" when it is copied or extracted. The Copy and Extract operations save the date and time as a property of the digital ticket. When a ticket agent is given a ticket, it can simply check whether the digital copy was made after the last time that it was punched. Of course, the digital ticket must have the copy or extract usage rights attached thereto.

The capability to unpunch a ticket is inportant in the following cases:

- A digital work is circulated at low cost with a limitation that it can be used only once.
- A digital work is circulated with a ticket that can be used once to give discounts on purchases of other works.
- A digital work is circulated with a ticket (included in the purchase price and possibly embedded in the work) that can be used for a future upgrade.

In each of these cases, if a paid copy is made of the digital work (including the ticket) the new owner would expect to get a fresh (unpunched) ticket, whether the copy seller has used the work or not. In contrast, loaning a work or simply transferring it to another repository should not revitalize the ticket.

Usage Fees and Incentives Specification

The billing for use of a digital work is fundamental to a commercial distribution system. Grammar Element 1517 **"Fee-Spec: = {Scheduled-Discount} Regular-Fee-Spec | Scheduled-Fee-Spec | Markup-Spec"** provides a range of options for billing for the use of digital works.

A key feature of this approach is the development of low-overhead billing for transactions in potentially small amounts. Thus, it becomes feasible to collect fees of only a few cents each for thousands of transactions.

The grammar differentiates between uses where the charge is per use from those where it is metered by the time unit. Transactions can support fees that the user pays for using a digital work as well as incentives paid by the right grantor to users to induce them to use or distribute the digital work.

The optional scheduled discount refers to the rest of the fee specification--discounting it by a percentage over time. If it is not specified, then there is no scheduled discount. Regular fee specifications are constant over time. Scheduled fee specifications give a schedule of dates over which the fee specifications change. Markup specifications are used in d-blocks for adding a percentage to the fees already being charged.

Grammar Element 1518 **"Scheduled-Discount: = (Scheduled-Discount: (Time-Spec Percentage)*)"** A Scheduled-Discount is a essentially a scheduled modifier of any other fee specification for this version of the right of the digital work. (It does not refer to children or parent digital works or to other versions of rights.). It is a list of pairs of times and percentages. The most recent time in the list that has not yet passed at the time of the transaction is the one in effect. The percentage gives the discount percentage. For example, the number 10 refers to a 10% discount.

Grammar Element 1519 **"Regular-Fee-Spec : = ({Fee: | Incentive:} [Per-Use-Spec | Metered-Rate-Spec | Best-Price-Spec | Call-For-Price-Spec] {Min: Money-Unit Per: Time-Spec}{Max: Money-Unit Per: Time-Spec} To: Account-ID)"** provides for several kinds of fee specifications.

Fees are paid by the copy-owner/user to the revenue-owner if **Fee:** is specified. Incentives are paid by the revenue-owner to the user if **Incentive:** is specified. If the **Min:** specification is given, then there is a minimum fee to be charged per time-spec unit for its use. If the **Max:** specification is given, then there is a maximum fee to be charged per time-spec for its use. When **Fee:** is specified, **Account-ID** identifies the account to which the fee is to be paid. When **Incentive:** is specified, **Account-ID** identifies the account from which the fee is to be paid.

Grammar element 1520 **"Per-Use-Spec: = Per-Use: Money-unit"** defines a simple fee to be paid every time the right is exercised, regardless of how much time the transaction takes.

Grammar element 1521 **"Metered-Rate-Spec : = Metered: Money-Unit Per: Time-Spec"** defines a metered-rate fee paid according to how long the right is exercised. Thus, the time it takes to complete the transaction determines the fee.

Grammar element 1522 **"Best-Price-Spec : = Best-Price: Money-unit Max: Money-unit"** is used to specify a best-price that is determined when the account is settled. This specification is to accommodate special deals, rebates,

and pricing that depends on information that is not available to the repository. All fee specifications can be combined with tickets or authorizations that could indicate that the consumer is a wholesaler or that he is a preferred customer, or that the seller be authorized in some way. The amount of money in the **Max:** field is the maximum amount that the use will cost. This is the amount that is tentatively debited from the credit server. However, when the transaction is ultimately reconciled, any excess amount will be returned to the consumer in a separate transaction.

Grammar element 1523 **"Call-For-Price-Spec: = Call-For-Price "** is similar to a **"Best-Price-Spec"** in that it is intended to accommodate cases where prices are dynamic. A **Call-For-Price Spec** requires a communication with a dealer to determine the price. This option cannot be exercised if the repository cannot communicate with a dealer at the time that the right is exercised. It is based on a secure transaction whereby the dealer names a price to exercise the right and passes along a deal certificate which is referenced or included in the billing process.

Grammar element 1524 **"Scheduled-Fee-Spec: = (Schedule: (Time-Spec Regular-Fee-Spec)*)"** is used to provide a schedule of dates over which the fee specifications change. The fee specification with the most recent date not in the future is the one that is in effect. This is similar to but more general than the scheduled discount. It is more general, because it provides a means to vary the fee agreement for each time period.

Grammar element 1525 **"Markup-Spec: = Markup: percentage To: Account-ID"** is provided for adding a percentage to the fees already being charged. For example, a 5% markup means that a fee of 5% of cumulative fee so far will be allocated to the distributor. A markup specification can be applied to all of the other kinds of fee specifications. It is typically used in a shell provided by a distributor. It refers to fees associated with d-blocks that are parts of the current d-block. This might be a convenient specification for use in taxes, or in distributor overhead.

REPOSITORY TRANSACTIONS

When a user requests access to a digital work, the repository will initiate various transactions. The combination of transactions invoked will depend on the specifications assigned for a usage right. There are three basic types of transactions, Session Initiation Transactions, Financial Transactions and Usage Transactions. Generally, session initiation transactions are initiated first to establish a valid session. When a valid session is established, transactions corresponding to the various usage rights are invoked. Finally, request specific transactions are performed.

Transactions occur between two repositories (one acting as a server), between a repository and a document playback platform (e.g. for executing or viewing), between a repository and a credit server or between a repository and an authorization server. When transactions occur between more than one repository, it is assumed that there is a reliable communication channel between the repositories. For example, this could be a TCP/IP channel or any other commercially available channel that has built-in capabilities for detecting and correcting transmission errors. However, it is not assumed that the communication channel is secure. Provisions for security and privacy are part of the requirements for specifying and implementing repositories and thus form the need for various transactions.

Message Transmission

Transactions require that there be some communication between repositories. Communication between repositories occurs in units termed as messages. Because the communication line is assumed to be unsecure, all communications with repositories that are above the lowest security class are encrypted utilizing a public key encryption technique. Public key encryption is a well known technique in the encryption arts. The term key refers to a numeric code that is used with encryption and decryption algorithms. Keys come in pairs, where "writing keys" are used to encrypt data and "checking keys" are used to decrypt data. Both writing and checking keys may be public or private. Public keys are those that are distributed to others. Private keys are maintained in confidence.

Key management and security is instrumental in the success of a public key encryption system. In the currently preferred embodiment, one or more master repositories maintain the keys and create the identification certificates used by the repositories.

When a sending repository transmits a message to a receiving repository, the sending repository encrypts all of its data using the public writing key of the receiving repository. The sending repository includes its name, the name of the receiving repository, a session identifier such as a nonce (described below), and a message counter in each message.

In this way, the communication can only be read (to a high probability) by the receiving repository, which holds the private checking key for decryption. The auxiliary data is used to guard against various replay attacks to security. If messages ever arrive with the wrong counter or an old nonce, the repositories can assume that someone is interfering with communication and the transaction terminated.

The respective public keys for the repositories to be used for encryption are obtained in the registration transaction described below.

Session Initiation Transactions

A usage transaction is carried out in a session between repositories. For usage transactions involving more than one repository, or for financial transactions between a repository and a credit server, a registration transaction is performed. A second transaction termed a login transaction, may also be needed to initiate the session. The goal of the registration transaction is to establish a secure channel between two repositories who know each others identities. As it is assumed that the communication channel between the repositories is reliable but not secure, there is a risk that a non-repository may mimic the protocol in order to gain illegitimate access to a repository.

The registration transaction between two repositories is described with respect to Figures 16 and 17. The steps described are from the perspective of a "repository-1" registering its identity with a "repository-2". The registration must be symmetrical so the same set of steps will be repeated for repository-2 registering its identity with repository-1. Referring to Figure 16, repository-1 first generates an encrypted registration identifier, step 1601 and then generates a registration message, step 1602. A registration message is comprised of an identifier of a master repository, the identification certificate for the repository-1 and an encrypted random registration identifier. The identification certificate is encrypted by the master repository in its private key and attests to the fact that the repository (here repository-1) is a bona fide repository. The identification certificate also contains a public key for the repository, the repository security level and a timestamp (indicating a time after which the certificate is no longer valid.) The registration identifier is a number generated by the repository for this registration. The registration identifier is unique to the session and is encrypted in repository-1's private key. The registration identifier is used to improve security of authentication by detecting certain kinds of communications based attacks. Repository-1 then transmits the registration message to repository-2, step 1603.

Upon receiving the registration message, repository-2 determines if it has the needed public key for the master repository, step 1604. If repository-2 does not have the needed public key to decrypt the identification certificate, the registration transaction terminates in an error, step 1618.

Assuming that repository-2 has the proper public key the identification certificate is decrypted, step 1605. Repository-2 saves the encrypted registration identifier, step 1606, and extracts the repository identifier, step 1607. The extracted repository identifier is checked against a "hotlist" of compromised document repositories, step 1608. In the currently preferred embodiment, each repository will contain "hotlists" of compromised repositories. If the repository is on the "hotlist', the registration transaction terminates in an error per step 1618. Repositories can be removed from the hotlist when their certificates expire, so that the list does not need to grow without bound. Also, by keeping a short list of hotlist certificates that it has previously received, a repository can avoid the work of actually going through the list. These lists would be encrypted by a master repository. A minor variation on the approach to improve efficiency would have the repositories first exchange lists of names of hotlist certificates, ultimately exchanging only those lists that they had not previously received. The "hotlists" are maintained and distributed by Master repositories.

Note that rather than terminating in error, the transaction could request that another registration message be sent based on an identification certificate created by another master repository. This may be repeated until a satisfactory identification certificate is found, or it is determined that trust cannot be established.

Assuming that the repository is not on the hotlist, the repository identification needs to be verified. In other words, repository-2 needs to validate that the repository on the other end is really repository-1. This is termed performance testing and is performed in order to avoid invalid access to the repository via a counterfeit repository replaying a recording of a prior session initiation between repository-1 and repository-2. Performance testing is initiated by repository-2 generating a performance message, step 1609. The performance message consists of a nonce, the names of the respective repositories, the time and the registration identifier received from repository-1. A nonce is a generated message based on some random and variable information (e.g. the time or the temperature.) The nonce is used to check whether repository-1 can actually exhibit correct encrypting of a message using the private keys it claims to have, on a message that it has never seen before. The performance message is encrypted using the public key specified in the registration message of repository-1. The performance message is transmitted to repository-1, step 1610, where it is decrypted by repository-1 using its private key, step 1611. Repository-1 then checks to make sure that the names of the two repositories are correct, step 1612, that the time is accurate, step 1613 and that the registration identifier corresponds to the one it sent, step 1614. If any of these tests fails, the transaction is terminated per step 1616. Assuming that the tests are passed, repository-1 transmits the nonce to repository-2 in the clear, step 1615. Repository-2 then compares the received nonce to the original nonce, step 1617. If they are not identical, the registration transaction terminates in an error per step 1618. If they are the same, the registration transaction has successfully completed.

At this point, assuming that the transaction has not terminated, the repositories exchange messages containing session keys to be used in all communications during the session and synchronize their clocks. Figure 17 illustrates the session information exchange and clock synchronization steps (again from the perspective of repository-1.) Referring to Figure 17, repository-1 creates a session key pair, step 1701. A first key is kept private and is used by repository-1 to encrypt messages. The second key is a public key used by repository-2 to decrypt messages. The

second key is encrypted using the public key of repository-2, step 1702 and is sent to repository-2, step 1703. Upon receipt, repository-2 decrypts the second key, step 1704. The second key is used to decrypt messages in subsequent communications. When each repository has completed this step, they are both convinced that the other repository is bona fide and that they are communicating with the original. Each repository has given the other a key to be used in decrypting further communications during the session. Since that key is itself transmitted in the public key of the receiving repository only it will be able to decrypt the key which is used to decrypt subsequent messages.

After the session information is exchanged, the repositories must synchronize their clocks. Clock synchronization is used by the repositories to establish an agreed upon time base for the financial records of their mutual transactions. Referring back to Figure 17, repository-2 initiates clock synchronization by generating a time stamp exchange message, step 1705, and transmits it to repository- 1, step 1706. Upon receipt, repository-1 generates its own time stamp message, step 1707 and transmits it back to repository-2, step 1708. Repository-2 notes the current time, step 1709 and stores the time received from repository-1, step 1710. The current time is compared to the time received from repository-1, step 1711. The difference is then checked to see if it exceeds a predetermined tolerance (e.g. one minute), step 1712. If it does, repository-2 terminates the transaction as this may indicate tampering with the repository, step 1713. If not repository-2 computes an adjusted time delta, step 1714. The adjusted time delta is the difference between the clock time of repository-2 and the average of the times from repository-1 and repository-2.

To achieve greater accuracy, repository-2 can request the time again up to a fixed number of times (e.g. five times), repeat the clock synchronization steps, and average the results.

A second session initiation transaction is a Login transaction. The Login transaction is used to check the authenticity of a user requesting a transaction. A Login transaction is particularly prudent for the authorization of financial transactions that will be charged to a credit server. The Login transaction involves an interaction between the user at a user interface and the credit server associated with a repository. The information exchanged here is a login string supplied by the repository/credit server to identify itself to the user, and a Personal Identification Number (PIN) provided by the user to identify himself to the credit server. In the event that the user is accessing a credit server on a repository different from the one on which the user interface resides, exchange of the information would be encrypted using the public and private keys of the respective repositories.

Billing Transactions

Billing Transactions are concerned with monetary transactions with a credit server. Billing Transactions are carried out when all other conditions are satisfied and a usage fee is required for granting the request. For the most part, billing transactions are well understood in the state of the art. These transactions are between a repository and a credit server, or between a credit server and a billing clearinghouse. Briefly, the required transactions include the following:

- Registration and LOGIN transactions by which the repository and user establish their bona fides to a credit server. These transactions would be entirely internal in cases where the repository and credit server are implemented as a single system.
- Registration and LOGIN transactions, by which a credit server establishes its bona fides to a billing clearinghouse.
- An Assign-fee transaction to assign a charge. The information in this transaction would include a transaction identifier, the identities of the repositories in the transaction, and a list of charges from the parts of the digital work. If there has been any unusual event in the transaction such as an interruption of communications, that information is included as well.
- A Begin-charges transaction to assign a charge. This transaction is much the same as an assign-fee transaction except that it is used for metered use. It includes the same information as the assign-fee transaction as well as the usage fee information. The credit-server is then responsible for running a clock.
- An End-charges transaction to end a charge for metered use. (In a variation on this approach, the repositories would exchange periodic charge information for each block of time.)
- A report-charges transaction between a personal credit server and a billing clearinghouse. This transaction is invoked at least once per billing period. It is used to pass along information about charges. On debit and credit cards, this transaction would also be used to update balance information and credit limits as needed.

All billing transactions are given a transaction ID and are reported to the credit severs by both the server and the client. This reduces possible loss of billing information if one of the parties to a transaction loses a banking card and provides a check against tampering with the system.

Usage Transactions

After the session initiation transactions have been completed, the usage request may then be processed. To sim-

plify the description of the steps carried out in processing a usage request, the term requester is used to refer to a repository in the requester mode which is initiating a request, and the term server is used to refer to a repository in the server mode and which contains the desired digital work. In many cases such as requests to print or view a work, the requester and server may be the same device and the transactions described in the following would be entirely internal. In such instances, certain transaction steps, such as the registration transaction, need not be performed.

There are some common steps that are part of the semantics of all of the usage rights transactions. These steps are referred to as the common transaction steps. There are two sets -- the "opening" steps and the "closing" steps. For simplicity, these are listed here rather than repeating them in the descriptions of all of the usage rights transactions.

Transactions can refer to a part of a digital work, a complete digital work, or a Digital work containing other digital works. Although not described in detail herein, a transaction may even refer to a folder comprised of a plurality of digital works. The term "work" is used to refer to what ever portion or set of digital works is being accessed.

Many of the steps here involve determining if certain conditions are satisfied. Recall that each usage right may have one or more conditions which must be satisfied before the right can be exercised. Digital works have parts and parts have parts. Different parts can have different rights and fees. Thus, it is necessary to verify that the requirements are met for ALL of the parts that are involved in a transaction For brevity, when reference is made to checking whether the rights exist and conditions for exercising are satisfied, it is meant that all such checking takes place for each of the relevant parts of the work.

Figure 18 illustrates the initial common opening and closing steps for a transaction. At this point it is assumed that registration has occurred and that a "trusted" session is in place. General tests are tests on usage rights associated with the folder containing the work or some containing folder higher in the file system hierarchy. These tests correspond to requirements imposed on the work as a consequence of its being on the particular repository, as opposed to being attached to the work itself. Referring to Figure 18, prior to initiating a usage transaction, the requester performs any general tests that are required before the right associated with the transaction can be exercised, step, 1801. For example, install, uninstall and delete rights may be implemented to require that a requester have an authorization certificate before the right can be exercised. Another example is the requirement that a digital ticket be present and punched before a digital work may be copied to a requester. If any of the general tests fail, the transaction is not initiated, step, 1802. Assuming that such required tests are passed, upon receiving the usage request, the server generates a transaction identifier that is used in records or reports of the transaction, step 1803. The server then checks whether the digital work has been granted the right corresponding to the requested transaction, step 1804. If the digital work has not been granted the right corresponding to the request, the transaction terminates, step 1805. If the digital work has been granted the requested right, the server then determines if the various conditions for exercising the right are satisfied. Time based conditions are examined, step 1806. These conditions are checked by examining the time specification for the the version of the right. If any of the conditions are not satisfied, the transaction terminates per step 1805.

Assuming that the time based conditions are satisfied, the server checks security and access conditions, step 1807. Such security and access conditions are satisfied if: 1) the requester is at the specified security class, or a higher security class, 2) the server satisfies any specified authorization test and 3) the requester satisfies any specified authorization tests and has any required digital tickets. If any of the conditions are not satisfied, the transaction terminates per step 1805.

Assuming that the security and access conditions are all satisfied, the server checks the copy count condition, step 1808. If the copy count equals zero, then the transaction cannot be completed and the transaction terminates per step 1805.

Assuming that the copy count does not equal zero, the server checks if the copies in use for the requested right is greater than or equal to any copy count for the requested right (or relevant parts), step 1809. If the copies in use is greater than or equal to the copy count, this indicates that usage rights for the version of the transaction have been exhausted. Accordingly, the server terminates the transaction, step 1805. If the copy count is less than the copies in use for the transaction the transaction can continue, and the copies in use would be incremented by the number of digital works requested in the transaction, step 1810.

The server then checks if the digital work has a "Loan" access right, step 1811. The "Loan" access right is a special case since remaining rights may be present even though all copies are loaned out. If the digital work has the "Loan" access right, a check is made to see if all copies have been loaned out, step 1812. The number of copies that could be loaned is the sum of the Copy-Counts for all of the versions of the loan right of the digital work. For a composite work, the relevant figure is the minimal such sum of each of the components of the composite work. If all copies have been loaned out, the remaining rights are determined, step 1813. The remaining-rights is determined from the remaining rights specifications from the versions of the Loan right. If there is only one version of the Loan right, then the determination is simple. The remaining rights are the ones specified in that version of the Loan right, or none if Remaining-Rights: is not specified. If there are multiple versions of the Loan right and all copies of all of the versions are loaned out, then the remaining rights is taken as the minimum set (intersection) of remaining rights across all of the versions of the loan right. The server then determines if the requested right is in the set of remaining rights,. step 1814. If the

requested right is not in the set of remaining rights, the server terminates the transaction, step 1805.

If Loan is not a usage right for the digital work or if all copies have not been loaned out or the requested right is in the set of remaining rights, fee conditions for the right are then checked, step 1815. This will initiate various financial transactions between the repository and associated credit server. Further, any metering of usage of a digital work will commence. If any financial transaction fails, the transaction terminates per step 1805.

It should be noted that the order in which the conditions are checked need not follow the order of steps 1806-1815.

At this point, right specific steps are now performed and are represented here as step 1816. The right specific steps are described in greater detail below.

The common closing transaction steps are now performed. Each of the closing transaction steps are performed by the server after a successful completion of a transaction. Referring back to Figure 18, the copies in use value for the requested right is decremented by the number of copies involved in the transaction, step 1817. Next, if the right had a metered usage fee specification, the server subtracts the elapsed time from the Remaining-Use-Time associated with the right for every part involved in the transaction, step 1818. Finally, if there are fee specifications associated with the right, the server initiates End-Charge financial transaction to confirm billing, step 1819.

Transmission Protocol

An important area to consider is the transmission of the digital work from the server to the requester. The transmission protocol described herein refers to events occurring after a valid session has been created. The transmission protocol must handle the case of disruption in the communications between the repositories. It is assumed that interference such as injecting noise on the communication channel can be detected by the integrity checks (e.g., parity, checksum, etc.) that are built into the transport protocol and are not discussed in detail herein.

The underlying goal in the transmission protocol is to preclude certain failure modes, such as malicious or accidental interference on the communications channel. Suppose, for example, that a user pulls a card with the credit server at a specific time near the end of a transaction. There should not be a vulnerable time at which "pulling the card" causes the repositories to fail to correctly account for the number of copies of the work that have been created. Restated, there should be no time at which a party can break a connection as a means to avoid payment after using a digital work.

If a transaction is interrupted (and fails), both repositories restore the digital works and accounts to their state prior to the failure, modulo records of the failure itself.

Figure 19 is a state diagram showing steps in the process of transmitting information during a transaction. Each box represents a state of a repository in either the server mode (above the central dotted line 1901) or in the requester mode (below the dotted line 1901). Solid arrows stand for transitions between states. Dashed arrows stand for message communications between the repositories. A dashed message arrow pointing to a solid transition arrow is interpreted as meaning that the transition takes place when the message is received. Unlabeled transition arrows take place unconditionally. Other labels on state transition arrows describe conditions that trigger the transition.

Referring now to Figure 19, the server is initially in a state 1902 where a new transaction is initiated via start message 1903. This message includes transaction information including a transaction identifier and a count of the blocks of data to be transferred. The requester, initially in a wait state 1904 then enters a data wait state 1905.

The server enters a data transmit state 1906 and transmits a block of data 1907 and then enters a wait for acknowledgement state 1908. As the data is received, the requester enters a data receive state 1909 and when the data blocks are completely received it enters an acknowledgement state 1910 and transmits an Acknowledgement message 1911 to the server.

If there are more blocks to send, the server waits until receiving an Acknowledgement message from the requester. When an Acknowledgement message is received it sends the next block to the requester and again waits for acknowledgement. The requester also repeats the same cycle of states.

If the server detects a communications failure before sending the last block, it enters a cancellation state 1912 wherein the transaction is cancelled. Similarly, if the requester detects a communications failure before receiving the last block it enters a cancellation state 1913.

If there are no more blocks to send, the server commits to the transaction and waits for the final Acknowledgement in state 1914. If there is a communications failure before the server receives the final Acknowledgement message, it still commits to the transaction but includes a report about the event to its credit server in state 1915. This report serves two purposes. It will help legitimize any claims by a user of having been billed for receiving digital works that were not completely received. Also it helps to identify repositories and communications lines that have suspicious patterns of use and interruption. The server then enters its completion state 1916.

On the requester side, when there are no more blocks to receive, the requester commits to the transaction in state 1917. If the requester detects a communications failure at this state, it reports the failure to its credit server in state 1918, but still commits to the transaction. When it has committed, it sends an acknowledgement message to the server. The server then enters its completion state 1919.

The key property is that both the server and the requester cancel a transaction if it is interrupted before all of the data blocks are delivered, and commits to it if all of the data blocks have been delivered.

There is a possibility that the server will have sent all of the data blocks (and committed) but the requester will not have received all of them and will cancel the transaction. In this case, both repositories will presumably detect a communications failure and report it to their credit server. This case will probably be rare since it depends on very precise timing of the communications failure. The only consequence will be that the user at the requester repository may want to request a refund from the credit services -- and the case for that refund will be documented by reports by both repositories.

To prevent loss of data, the server should not delete any transferred digital work until receiving the final acknowledgement from the requester. But it also should not use the file. A well known way to deal with this situation is called "two-phase commit" or 2PC.

Two-phase commit works as follows. The first phase works the same as the method described above. The server sends all of the data to the requester. Both repositories mark the transaction (and appropriate files) as uncommitted. The server sends a ready-to-commit message to the requester. The requester sends back an acknowledgement. The server then commits and sends the requester a commit message. When the requester receives the commit message, it commits the file.

If there is a communication failure or other crash, the requester must check back with the server to determine the status of the transaction. The server has the last word on this. The requester may have received all of the data, but if it did not get the final message, it has not committed. The server can go ahead and delete files (except for transaction records) once it commits, since the files are known to have been fully transmitted before starting the 2PC cycle.

There are variations known in the art which can be used to achieve the same effect. For example, the server could use an additional level of encryption when transmitting a work to a client. Only after the client sends a message acknowledging receipt does it send the key. The client then agrees to pay for the digital work. The point of this variation is that it provides a clear audit trail that the client received the work. For trusted systems, however, this variation adds a level of encryption for no real gain in accountability.

The transaction for specific usage rights are now discussed.

The Copy Transaction

A Copy transaction is a request to make one or more independent copies of the work with the same or lesser usage rights. Copy differs from the extraction right discussed later in that it refers to entire digital works or entire folders containing digital works. A copy operation cannot be used to remove a portion of a digital work.

- The requester sends the server a message to initiate the Copy Transaction. This message indicates the work to be copied, the version of the copy right to be used for the transaction, the destination address information (location in a folder) for placing the work, the file data for the work (including its size), and the number of copies requested.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the client according to the transmission protocol. If a Next-Set-Of-Rights has been provided in the version of the right, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted. In any event, the Copy-Count field for the copy of the digital work being sent right is set to the number-of-copies requested.
- The requester records the work contents, data, and usage rights and stores the work. It records the date and time that the copy was made in the properties of the digital work.
- The repositories perform the common closing transaction steps.

The Transfer Transaction

A Transfer transaction is a request to move copies of the work with the same or lesser usage rights to another repository. In contrast with a copy transaction, this results in removing the work copies from the server.

- The requester sends the server a message to initiate the Transfer Transaction. This message indicates the work to be transferred, the version of the transfer right to be used in the transaction, the destination address information for placing the work, the file data for the work, and the number of copies involved.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted. In either case, the Copy-Count field for the transmitted rights are set to the number-of-copies requested.

- The requester records the work contents, data, and usage rights and stores the work.
- The server decrements its copy count by the number of copies involved in the transaction.
- The repositories perform the common closing transaction steps.
- If the number of copies remaining in the server is now zero, it erases the digital work from its memory.

The Loan Transaction

A loan transaction is a mechanism for loaning copies of a digital work. The maximum duration of the loan is determined by an internal parameter of the digital work. Works are automatically returned after a predetermined time period.

- The requester sends the server a message to initiate the Transfer Transaction. This message indicates the work to be loaned, the version of the loan right to be used in the transaction, the destination address information for placing the work, the number of copies involved, the file data for the work, and the period of the loan.
- The server checks the validity of the requested loan period, and ends with an error if the period is not valid. Loans for a loaned copy cannot extend beyond the period of the original loan to the server.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, the rights of the original are transmitted, as modified to reflect the loan period.
- The requester records the digital work contents, data, usage rights, and loan period and stores the work.
- The server updates the usage rights information in the digital work to reflect the number of copies loaned out.
- The repositories perform the common closing transaction steps.
- The server updates the usage rights data for the digital work. This may preclude use of the work until it is returned from the loan. The user on the requester platform can now use the transferred copies of the digital work. A user accessing the original repository cannot use the digital work , unless there are copies remaining. What happens next depends on the order of events in time.

Case 1. If the time of the loan period is not yet exhausted and the requester sends the repository a Return message.

- The return message includes the requester identification, and the transaction ID.
- The server decrements the copies-in-use field by the number of copies that were returned. (If the number of digital works returned is greater than the number actually borrowed, this is treated as an error.) This step may now make the work available at the server for other users.
- The requester deactivates its copies and removes the contents from its memory.

Case 2. If the time of the loan period is exhausted and the requester has not yet sent a Return message.

- The server decrements the copies-in-use field by the number digital works that were borrowed.
- The requester automatically deactivates its copies of the digital work. It terminates all current uses and erases the digital work copies from memory. One question is why a requester would ever return a work earlier than the period of the loan, since it would be returned automatically anyway. One reason for early return is that there may be a metered fee which determines the cost of the loan. Returning early may reduce that fee.

The Play Transaction

A play transaction is a request to use the contents of a work. Typically, to "play" a work is to send the digital work through some kind of transducer, such as a speaker or a display device. The request implies the intention that the contents will not be communicated digitally to any other system. For example, they will not be sent to a printer, recorded on any digital medium, retained after the transaction or sent to another repository.

This term "play" is natural for examples like playing music, playing a movie, or playing a video game. The general form of play means that a "player" is used to use the digital work. However, the term play covers all media and kinds of recordings. Thus one would "play" a digital work, meaning, to render it for reading, or play a computer program, meaning to execute it. For a digital ticket the player would be a digital ticket agent.

- The requester sends the server a message to initiate the play transaction. This message indicates the work to be

played, the version of the play right to be used in the transaction, the identity of the player being used, and the file data for the work.

- The server checks the validity of the player identification and the compatibility of the player identification with the player specification in the right. It ends with an error if these are not satisfactory.
- The repositories perform the common opening transaction steps.
- The server and requester read and write the blocks of data as requested by the player according to the transmission protocol. The requester plays the work contents, using the player.
- When the player is finished, the player and the requester remove the contents from their memory.
- The repositories perform the common closing transaction steps.

The Print Transaction

A Print transaction is a request to obtain the contents of a work for the purpose of rendering them on a "printer." We use the term "printer" to include the common case of writing with ink on paper. However, the key aspect of "printing" in our use of the term is that it makes a copy of the digital work in a place outside of the protection of usage rights. As with all rights, this may require particular authorization certificates.

Once a digital work is printed, the publisher and user are bound by whatever copyright laws are in effect. However, printing moves the contents outside the control of repositories. For example, absent any other enforcement mechanisms, once a digital work is printed on paper, it can be copied on ordinary photocopying machines without intervention by a repository to collect usage fees. If the printer to a digital disk is permitted, then that digital copy is outside of the control of usage rights. Both the creator and the user know this, although the creator does not necessarily give tacit consent to such copying, which may violate copyright laws.

- The requester sends the server a message to initiate a Print transaction. This message indicates the work to be played, the identity of the printer being used, the file data for the work, and the number of copies in the request.
- The server checks the validity of the printer identification and the compatibility of the printer identification with the printer specification in the right. It ends with an error if these are not satisfactory.
- The repositories perform the common opening transaction steps.
- The server transmits blocks of data according to the transmission protocol.
- The requester prints the work contents, using the printer.
- When the printer is finished, the printer and the requester remove the contents from their memory.
- The repositories perform the common closing transaction steps.

The Backup Transaction

A Backup transaction is a request to make a backup copy of a digital work, as a protection against media failure. In the context of repositories, secure backup copies differ from other copies in three ways: (1) they are made under the control of a Backup transaction rather than a Copy transaction, (2) they do not count as regular copies, and (3) they are not usable as regular copies. Generally, backup copies are encrypted.

Although backup copies may be transferred or copied, depending on their assigned rights, the only way to make them useful for playing, printing or embedding is to restore them.

The output of a Backup operation is both an encrypted data file that contains the contents and description of a work, and a restoration file with an encryption key for restoring the encrypted contents. In many cases, the encrypted data file would have rights for " printing" it to a disk outside of the protection system, relying just on its encryption for security. Such files could be stored anywhere that was physically safe and convenient. The restoration file would be held in the repository. This file is necessary for the restoration of a backup copy. It may have rights for transfer between repositories.

- The requester sends the server a message to initiate a backup transaction. This message indicates the work to be backed up, the version of the backup right to be used in the transaction, the destination address information for placing the backup copy, the file data for the work.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, a set of default rights for backup files of the original are transmitted by the server.
- The requester records the work contents, data, and usage rights. It then creates a one-time key and encrypts the contents file. It saves the key information in a restoration file.
- The repositories perform the common closing transaction steps.

In some cases, it is convenient to be able to archive the large, encrypted contents file to secure offline storage, such as a magneto-optical storage system or magnetic tape. This creation of a non-repository archive file is as secure as the encryption process. Such non-repository archive storage is considered a form of "printing" and is controlled by a print right with a specified "archive-printer." An archive-printer device is programmed to save the encrypted contents file (but not the description file) offline in such a way that it can be retrieved.

The Restore Transaction

A Restore transaction is a request to convert an encrypted backup copy of a digital work into a usable copy. A restore operation is intended to be used to compensate for catastrophic media failure. Like all usage rights, restoration rights can include fees and access tests including authorization checks.

- The requester sends the server a message to initiate a Restore transaction. This message indicates the work to be restored, the version of the restore right for the transaction, the destination address information for placing the work, and the file data for the work.
- The server verifies that the contents file is available (i.e. a digital work corresponding to the request has been backed-up.) If it is not, it ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The server retrieves the key from the restoration file. It decrypts the work contents, data, and usage rights.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the work. Otherwise, a set of default rights for backup files of the original are transmitted by the server.
- The requester stores the digital work.
- The repositories perform the common closing transaction steps.

The Delete Transaction

A Delete transaction deletes a digital work or a number of copies of a digital work from a repository. Practically all digital works would have delete rights.

- The requester sends the server a message to initiate a delete transaction. This message indicates the work to be deleted , the version of the delete right for the transaction.
- The repositories perform the common opening transaction steps.
- The server deletes the file, erasing it from the file system.
- The repositories perform the common closing transaction steps.

The Directory Transaction

A Directory transaction is a request for information about folders, digital works, and their parts. This amounts to roughly the same idea as protection codes in a conventional file system like TENEX, except that it is generalized to the full power of the access specifications of the usage rights language.

The Directory transaction has the important role of passing along descriptions of the rights and fees associated with a digital work. When a user wants to exercise a right, the user interface of his repository implicitly makes a directory request to determine the versions of the right that are available. Typically these are presented to the user -- such as with different choices of billing for exercising a right. Thus, many directory transactions are invisible to the user and are exercised as part of the normal process of exercising all rights.

- The requester sends the server a message to initiate a Directory transaction. This message indicates the file or folder that is the root of the directory request and the version of the di rectory right used for the transaction.
- The server verifies that the information is accessible to the requester. In particular, it does not return the names of any files that have a HIDE-NAME status in their directory specifications, and it does not return the parts of any folders or files that have HIDE-PARTS in their specification. If the information is not accessible, the server ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The server sends the requested data to the requester according to the transmission protocol.
- The requester records the data.
- The repositories perform the common closing transaction steps.

The Folder Transaction

A Folder transaction is a request to create or rename a folder, or to move a work between folders. Together with Directory rights, Folder rights control the degree to which organization of a repository can be accessed or modified from another repository.

- The requester sends the server a message to initiate a Folder transaction. This message indicates the folder that is the root of the folder request, the version of the folder right for the transaction, an operation, and data. The operation can be one of create, rename, and move file. The data are the specifications required for the operation, such as a specification of a folder or digital work and a name.
- The repositories perform the common opening transaction steps.
- The server performs the requested operation -- creating a folder, renaming a folder, or moving a work between folders.
- The repositories perform the common closing transaction steps.

The Extract Transaction

A extract transaction is a request to copy a part of a digital work and to create a new work containing it. The extraction operation differs from copying in that it can be used to separate a part of a digital work from d-blocks or shells that place additional restrictions or fees on it. The extraction operation differs from the edit operation in that it does not change the contents of a work, only its embedding in d-blocks. Extraction creates a new digital work.

- The requester sends the server a message to initiate an Extract transaction. This message indicates the part of the work to be extracted, the version of the extract right to be used in the transaction, the destination address information for placing the part as a new work, the file data for the work, and the number of copies involved.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the new work. Otherwise, the rights of the original are transmitted. The Copy-Count field for this right is set to the number-of-copies requested.
- The requester records the contents, data, and usage rights and stores the work. It records the date and time that new work was made in the properties of the work.
- The repositories perform the common closing transaction steps.

The Embed Transaction

An embed transaction is a request to make a digital work become a part of another digital work or to add a shell d-block to enable the adding of fees by a distributor of the work.

- The requester sends the server a message to initiate an Embed transaction. This message indicates the work to be embedded, the version of the embed right to be used in the transaction, the destination address information for placing the part as a a work, the file data for the work, and the number of copies involved.
- The server checks the control specifications for all of the rights in the part and the destination. If they are incompatible, the server ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The server transmits the requested contents and data to the requester according to the transmission protocol. If a Next-Set-Of-Rights has been provided, those rights are transmitted as the rights for the new work. Otherwise, the rights of the original are transmitted. The Copy-Count field for this right is set to the number-of-copies requested.
- The requester records the contents, data, and usage rights and embeds the work in the destination file.
- The repositories perform the common closing transaction steps.

The Edit Transaction

An Edit transaction is a request to make a new digital work by copying, selecting and modifying portions of an existing digital work. This operation can actually change the contents of a digital work. The kinds of changes that are permitted depend on the process being used. Like the extraction operation, edit operates on portions of a digital work. In contrast with the extract operation, edit does not affect the rights or location of the work. It only changes the contents. The kinds of changes permitted are determined by the type specification of the processor specified in the rights. In the currently preferred embodiment, an edit transaction changes the work itself and does not make a new work. However,

it would be a reasonable variation to cause a new copy of the work to be made.

- The requester sends the server a message to initiate an Edit transaction. This message indicates the work to be edited, the version of the edit right to be used in the transaction, the file data for the work (including its size), the process-ID for the process, and the number of copies involved.
- The server checks the compatibility of the process-ID to be used by the requester against any process-ID specification in the right. If they are incompatible, it ends the transaction with an error.
- The repositories perform the common opening transaction steps.
- The requester uses the process to change the contents of the digital work as desired. (For example, it can select and duplicate parts of it; combine it with other information; or compute functions based on the information. This can amount to editing text, music, or pictures or taking whatever other steps are useful in creating a derivative work.)
- The repositories perform the common closing transaction steps.

The edit transaction is used to cover a wide range of kinds of works. The category describes a process that takes as its input any portion of a digital work and then modifies the input in some way. For example, for text, a process for editing the text would require edit rights. A process for "summarizing" or counting words in the text would also be considered editing. For a music file, processing could involve changing the pitch or tempo, or adding reverberations, or any other audio effect. For digital video works, anything which alters the image would require edit rights. Examples would be colorizing, scaling, extracting still photos, selecting and combining frames into story boards, sharpening with signal processing, and so on.

Some creators may want to protect the authenticity of their works by limiting the kinds of processes that can be performed on them. If there are no edit rights, then no processing is allowed at all. A processor identifier can be included to specify what kind of process is allowed. If no process identifier is specified, then arbitrary processors can be used. For an example of a specific process, a photographer may want to allow use of his photograph but may not want it to be colorized. A musician may want to allow extraction of portions of his work but not changing of the tonality.

Authorization Transactions

There are many ways that authorization transactions can be defined. In the following, our preferred way is to simply define them in terms of other transactions that we already need for repositories. Thus, it is convenient sometimes to speak of "authorization transactions," but they are actually made up of other transactions that repositories already have.

A usage right can specify an authorization-ID, which identifies an authorization object (a digital work in a file of a standard format) that the repository must have and which it must process. The authorization is given to the generic authorization (or ticket) server of the repository which begins to interpret the authorization.

As described earlier, the authorization contains a server identifier, which may just be the generic authorization server or it may be another server. When a remote authorization server is required, it must contain a digital address. It may also contain a digital certificate.

If a remote authorization server is required, then the authorization process first performs the following steps:

- The generic authorization server attempts to set up the communications channel. (If the channel cannot be set up, then authorization fails with an error.)
- When the channel is set up, it performs a registration process with the remote repository. (If registration fails, then the authorization fails with an error.)
- When registration is complete, the generic authorization server invokes a "Play" transaction with the remote repository, supplying the authorization document as the digital work to be played, and the remote authorization server (a program) as the "player." (If the player cannot be found or has some other error, then the authorization fails with an error.)
- The authorization server then "plays" the authorization. This involves decrypting it using either the public key of the master repository that issued the certificate or the session key from the repository that transmitted it. The authorization server then performs various tests. These tests vary according to the authorization server. They include such steps as checking issue and validity dates of the authorization and checking any hot-lists of known invalid authorizations. The authorization server may require carrying out any other transactions on the repository as well, such as checking directories, getting some person to supply a password, or playing some other digital work. It may also invoke some special process for checking information about locations or recent events. The "script" for such steps is contained within the authorization server.
- If all of the required steps are completed satisfactorily, the authorization server completes the transaction normally, signaling that authorization is granted.

The Install Transaction

An Install transaction is a request to install a digital work as runnable software on a repository. In a typical case, the requester repository is a rendering repository and the software would be a new kind or new version of a player. Also in a typical case, the software would be copied to file system of the requester repository before it is installed.

- The requester sends the server an Install message. This message indicates the work to be installed, the version of the Install right being invoked, and the file data for the work (including its size).
- The repositories perform the common opening transaction steps.
- The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error.
- The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step certifies the software.)
- The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.)
- The requester retrieves the instructions in the compatibility-checking script and follows them. If the software is not compatible with the repository, the installation transaction ends with an error. (This step checks platform compatibility.)
- The requester retrieves the instructions in the installation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error. Note that the installation process puts the runnable software in a place in the repository where it is no longer accessible as a work for exercising any usage rights other than the execution of the software as part of repository operations in carrying out other transactions.
- The repositories perform the common closing transaction steps.

The Uninstall Transaction

An Uninstall transaction is a request to remove software from a repository. Since uncontrolled or incorrect removal of software from a repository could compromise its behavioral integrity, this step is controlled.

- The requester sends the server an Uninstall message. This message indicates the work to be uninstalled, the version of the Uninstall right being invoked, and the file data for the work (including its size).
- The repositories perform the common opening transaction steps.
- The requester extracts a copy of the digital certificate for the software. If the certificate cannot be found or the master repository for the certificate is not known to the requester, the transaction ends with an error.
- The requester checks whether the software is installed. If the software is not installed, the transaction ends with an error.
- The requester decrypts the digital certificate using the public key of the master repository, recording the identity of the supplier and creator, a key for decrypting the software, the compatibility information, and a tamper-checking code. (This step authenticates the certification of the software, including the script for uninstalling it.)
- The requester decrypts the software using the key from the certificate and computes a check code on it using a 1-way hash function. If the check-code does not match the tamper-checking code from the certificate, the installation transaction ends with an error. (This step assures that the contents of the software, including the various scripts, have not been tampered with.)
- The requester retrieves the instructions in the uninstallation script and follows them. If there is an error in this process (such as insufficient resources), then the transaction ends with an error.
- The repositories perform the common closing transaction steps.

Claims

1. A distribution system for distributing digital works, said digital works having one or more usage rights attached thereto, said distribution system comprising:

   a grammar for creating instances of usage rights indicating a manner by which a possessor of an associated digital work may transport said associated digital work;

means for creating usage rights from said grammar;
means for attaching created usage rights to a digital work;
a requester repository for accessing digital works, said requester repository having means for generating usage transactions, each said usage transaction specifying a usage right;
a server repository for storing digital works with attached created usage rights, said server repository having means for processing usage transactions from said requester repository to determine if access to a digital work may be granted.

2. The distribution system as recited in Claim 1 wherein said grammar further specifies a default plurality of conditions for an instance of a usage right, wherein said one or more conditions must be satisfied before said usage right may be exercised.

3. The distribution system as recited in Claim 2 wherein said means for creating usage rights from said grammar is further comprised of means for changing said default plurality of conditions for an instance of a usage right.

4. The distribution system as recited in Claim 1 wherein said digital work is a software program.

5. The distribution system as recited in Claim 1 wherein said grammar is further for creating a first version of a usage right having a first set of conditions and a second version of said usage right having a second set of conditions.

6. A computer based system for controlling distribution and use of digital works comprising:

a usage rights grammar for creating instances of usages rights which define how a digital work may be used or distributed, said usage rights grammar comprising a first plurality of grammar elements for defining transport usage rights and a second plurality of grammar elements for defining rendering usage rights;
means for attaching usage rights to digital works;
a plurality of repositories for storing and exchanging digital works, each of said plurality of repositories comprising :
means for storing digital works and their attached usage rights;
transaction processing means having a requester mode of operation for requesting access to a requested digital work, said request specifying a usage right, and a server mode of operation for processing requests to access said requested digital work based on said usage right specified in said request and the usage rights attached to said requested digital work; and
a coupling means for coupling to another of said plurality of repositories across a communications medium.

7. The computer based system for controlling distribution and use of digital works as recited in Claim 6 wherein said first plurality of grammar elements is comprised of:

a loan grammar element for enabling a digital work to be loaned to another repository;
a copy grammar element for enabling a copy of a digital work to be made and transported to another repository; and
a transfer grammar element for enabling a digital work to be transferred to another repository.

8. The computer based system for controlling distribution and use of digital works as recited in Claim 6 or Claim 7 wherein said second plurality of grammar elements is comprised of:

a play grammar element for enabling a digital work to be rendered on a specified class of player device; and
a print grammar element for enabling a digital work to be printed on a specified class of printer device.

9. The computer based system for controlling distribution and use of digital works as recited in any one of Claims 6 to 8 wherein said grammar comprises one or more further pluralities of grammar elements, for defining file management usage rights, for enabling a digital work to be used in the creation of a new digital work,for enabling the secure installation and uninstallation of digital works comprising of software programs, or for providing a set of creator specified conditions which must be satisfied for each instantiation of a usage right defined by a grammar element.

10. A method for controlling distribution and use of digital works comprising the steps of:

a) creating a set of usage rights from a usage rights grammar, each of said usage rights defining a specific instance of how a digital work may be used or distributed, each of said usage rights specifying one or more conditions which must be satisfied in order for said usage right to be exercised;
b) attaching said set of usage rights to a digital work;
c) storing said digital work and its attached usage rights in a first repository;
d) a second repository initiating a request to access said digital work in said first repository, said request specifying a usage right;
e) said first repository receiving said request from said second repository;
f) said first repository determining if said specified usage right is attached to said digital work;
g) said first repository denying access to said digital work if said identified usage right is not attached to said digital work;
h) if said identified usage right is attached to said digital work, said first repository determining if conditions specified by said usage right are satisfied;
i) if said conditions are not satisfied, said first repository denying access to said digital work;
j) if said conditions are satisfied, said first repository transmitting said digital work to said second repository.

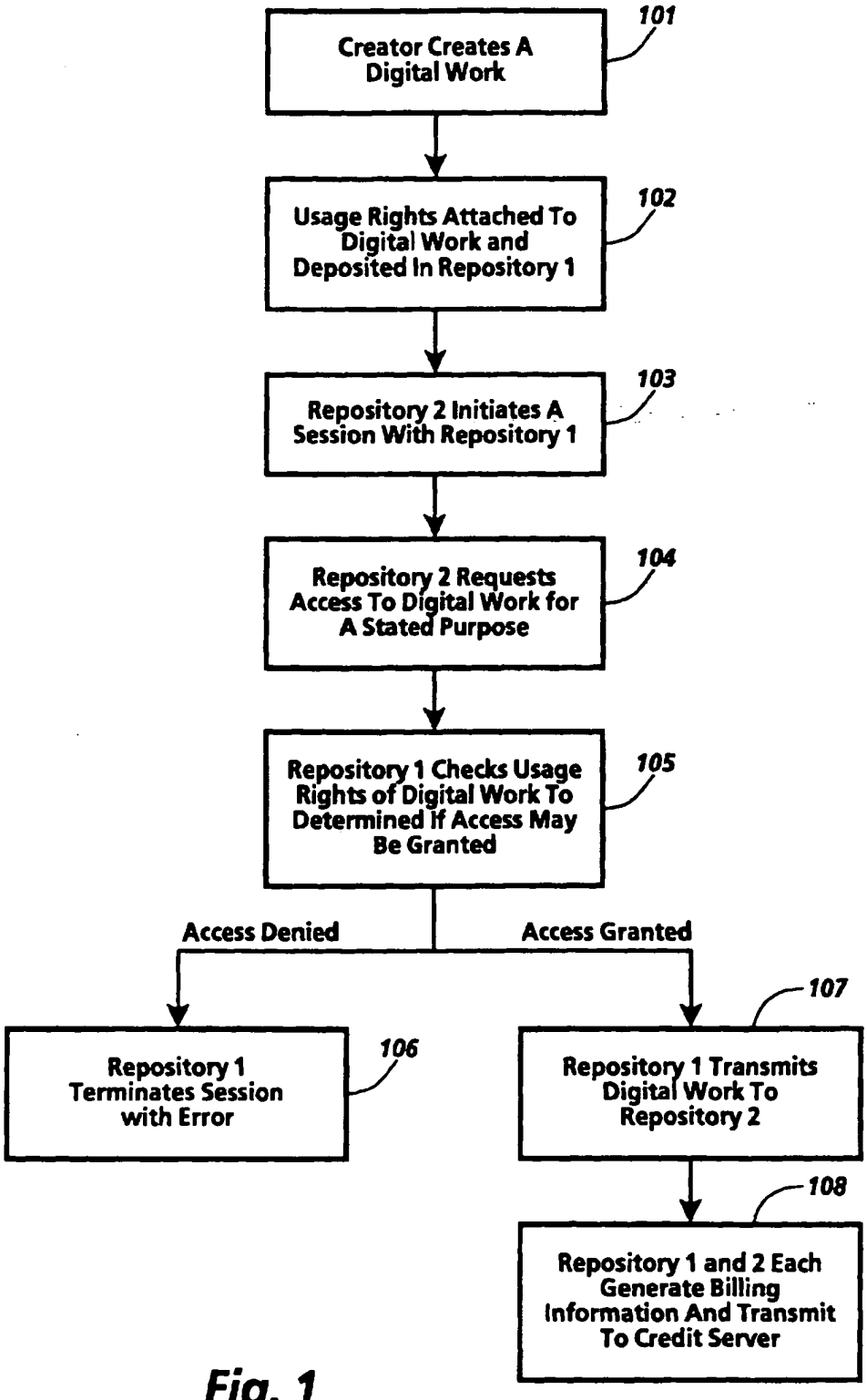


Fig. 1

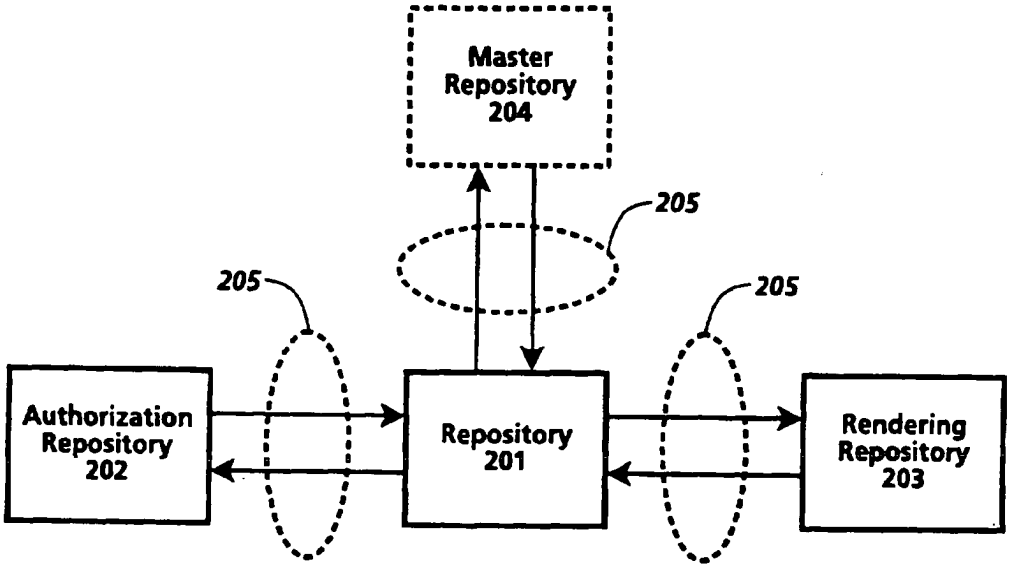


Fig. 2

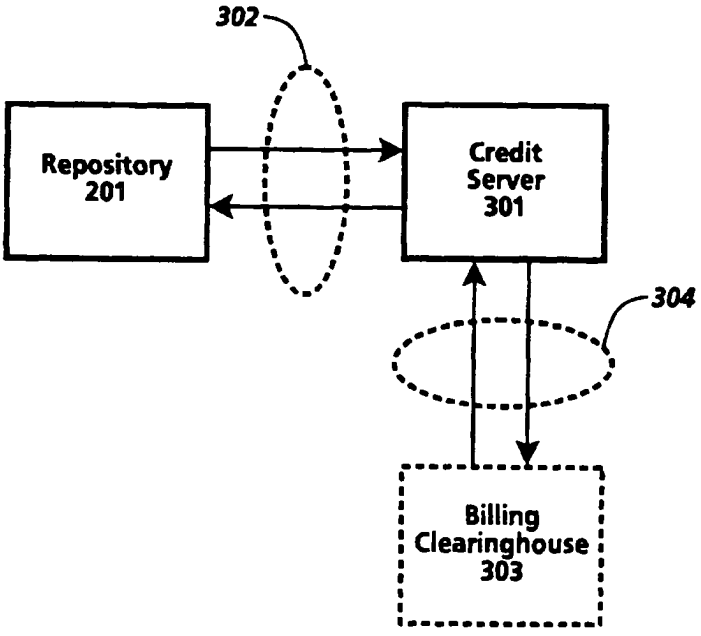


Fig. 3

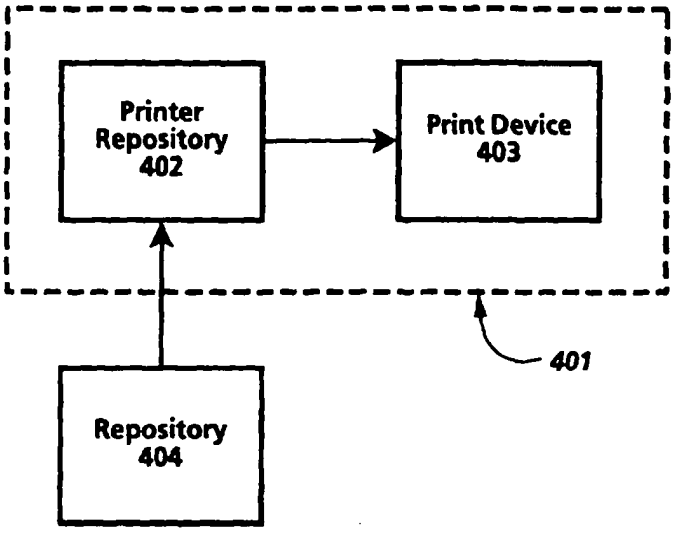


Fig. 4a

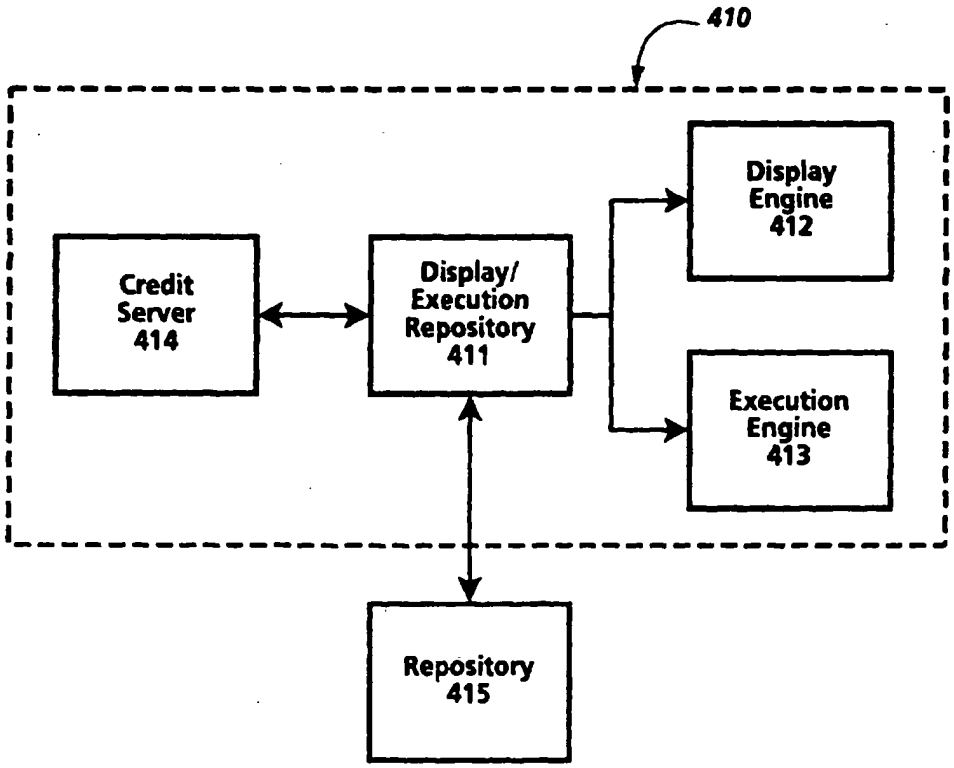


Fig. 4b

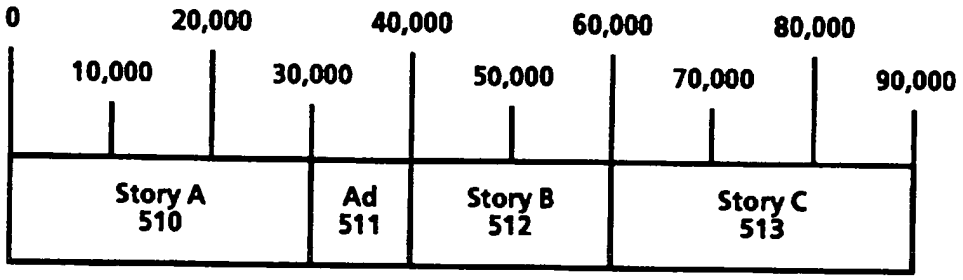


Fig. 5

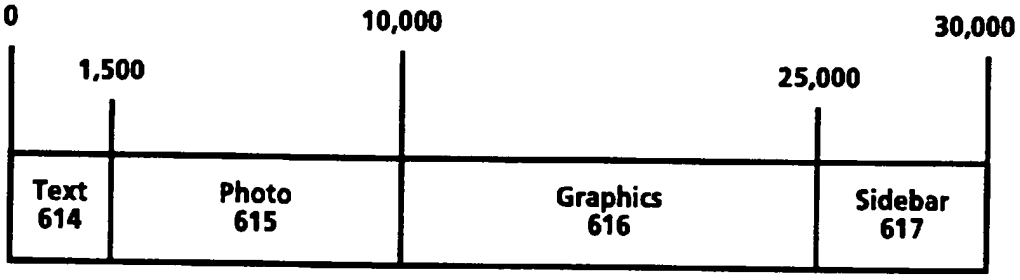


Fig. 6

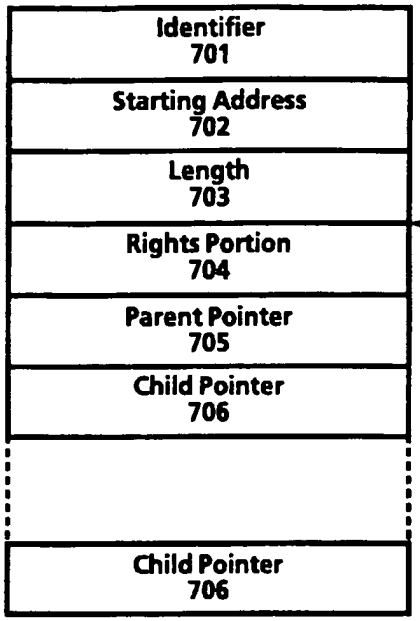


Fig. 7

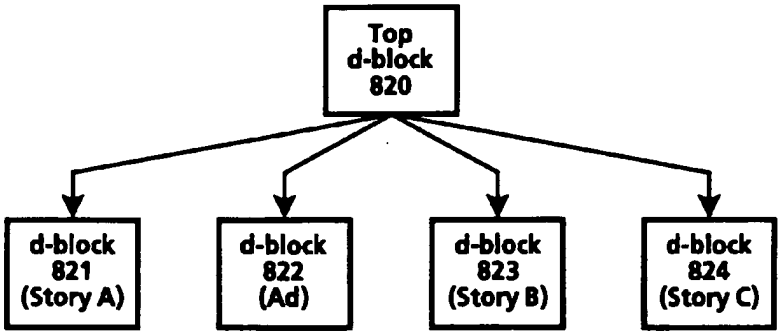


Fig. 8

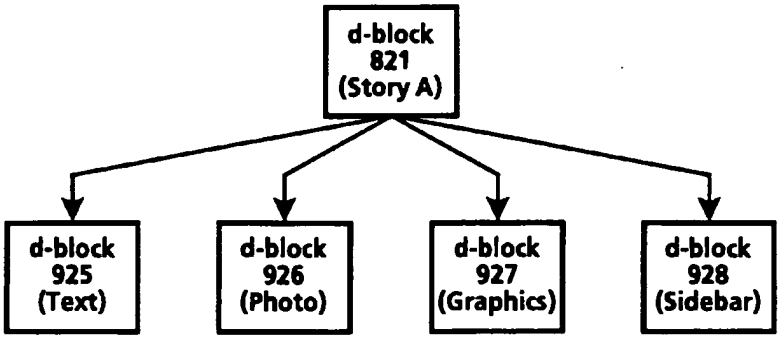


Fig. 9

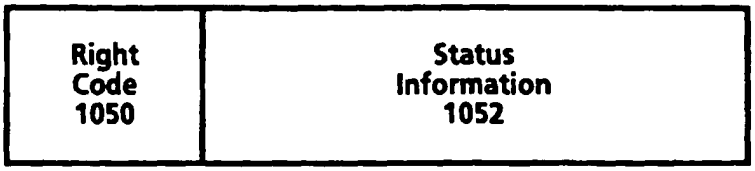


Fig.10

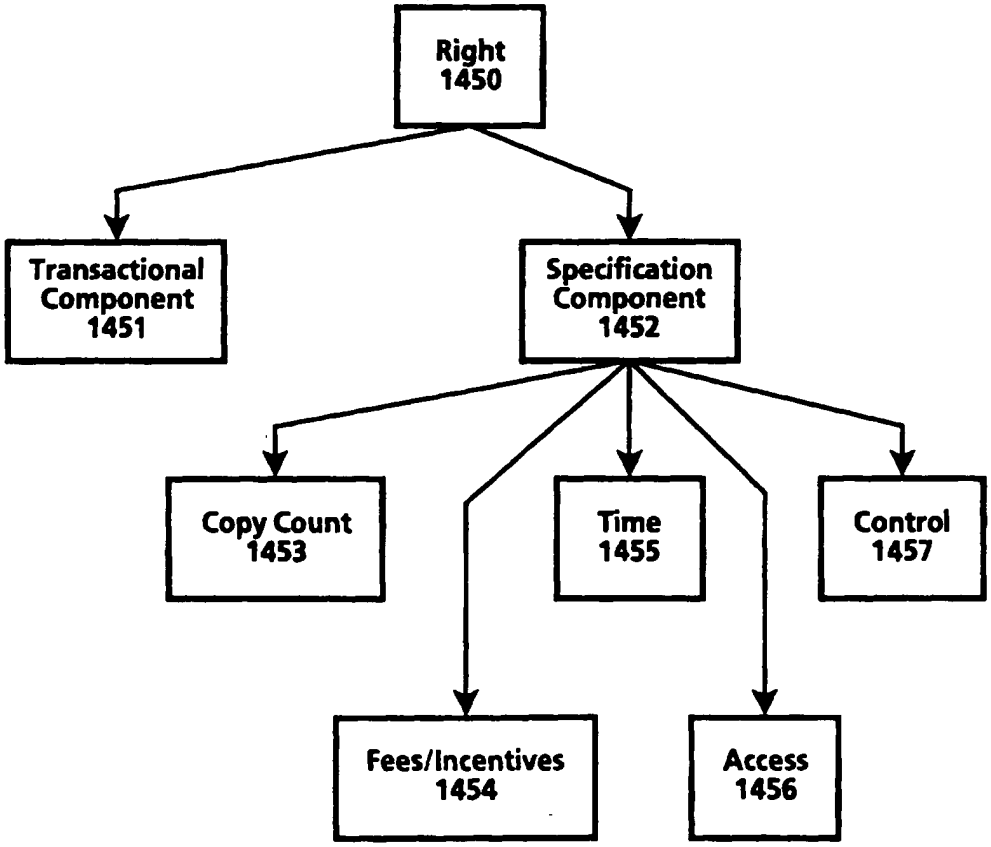


Fig.14

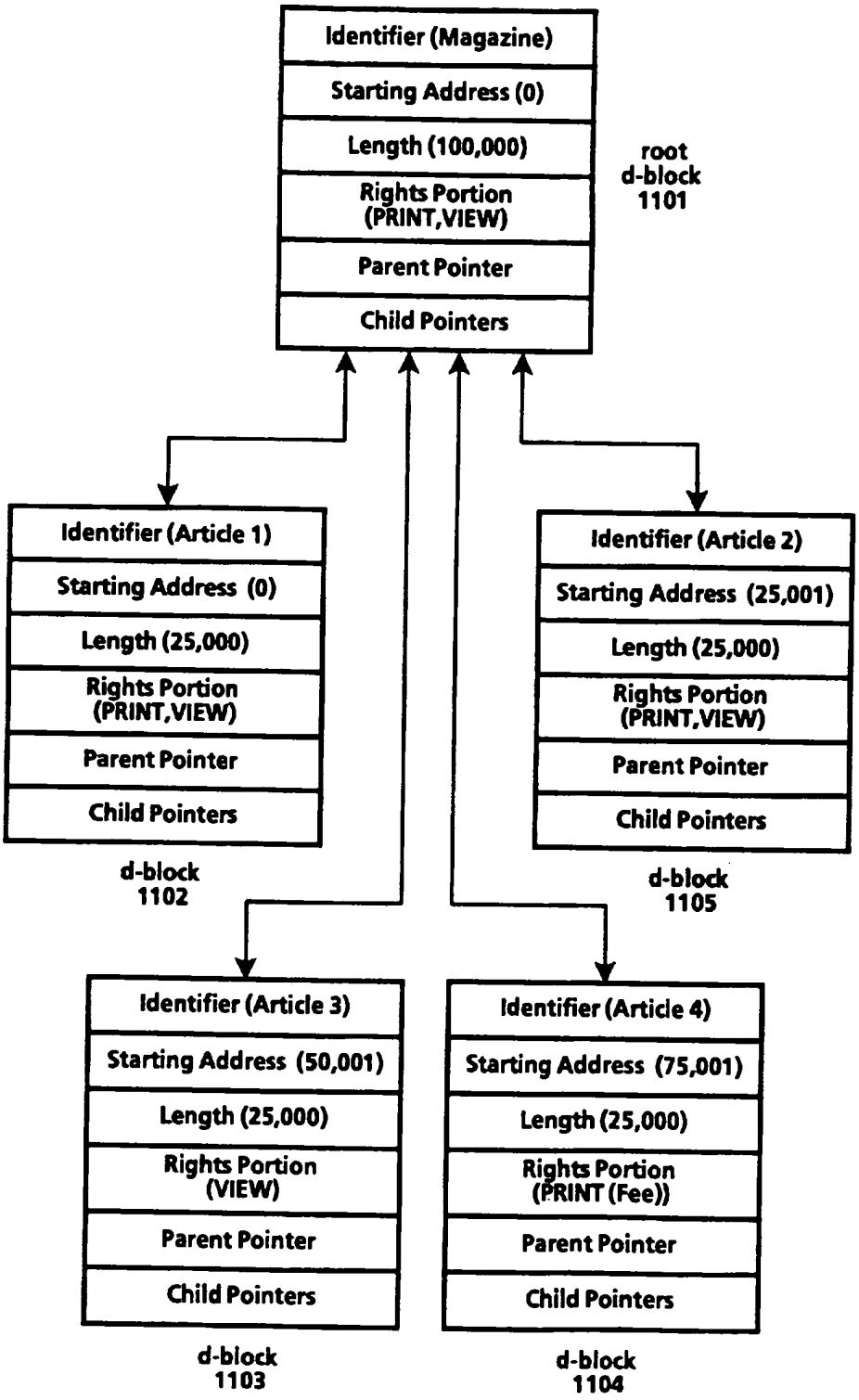


Fig.11

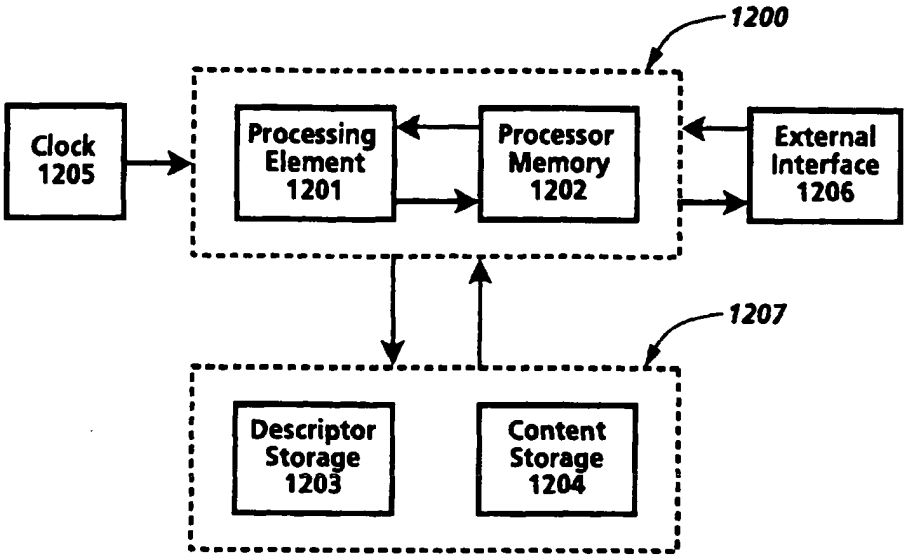


Fig.12

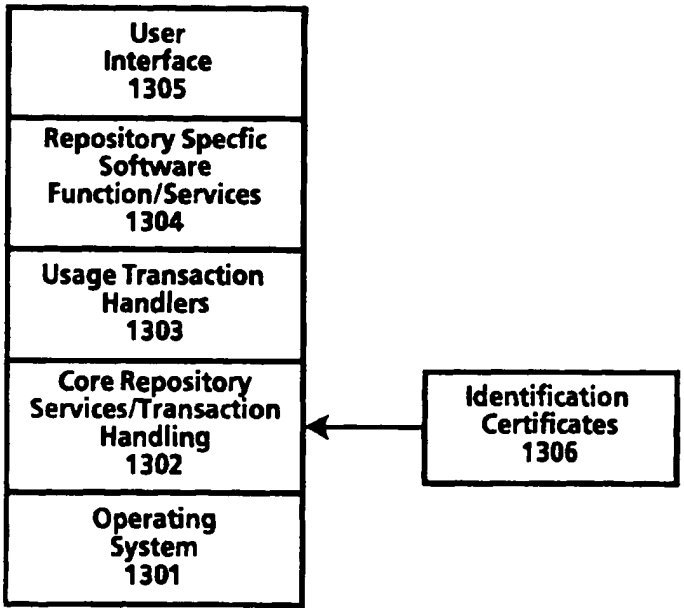


Fig.13

1501 ~ **Digital Work Rights**: = (Rights*)
1502 ~ **Right** : = (Right-Code {Copy-Count} {Control-Spec} {Time-Spec } {Access-Spec} {Fee-Spec})
1503 ~ **Right-Code** : = Render-Code | Transport-Code | File-Management-Code| Derivative-Works- Code | Configuration-Code
1504 ~ **Render-Code** : = [Play : {Player: Player-ID} | Print: {Printer: Printer-ID}]
1505 ~ **Transport-Code** : = [Copy | Transfer | Loan {Remaining-Rights: Next-Set-of-Rights}]{(Next-Copy-Rights: Next-Set-of-Rights)}
1506 ~ **File-Management-Code** : = Backup {Back-Up-Copy-Rights: Next-Set-of-Rights} | Restore | Delete | Folder | Directory {Name: Hide-Local | Hide-Remote} {Parts: Hide-Local | Hide-Remote}
1507 ~ **Derivative-Works-Code** : = [Extract | Embed | Edit{Process: Process-ID}] {Next-Copy-Rights : Next-Set-of Rights}
1508 ~ **Configuration-Code** : = Install | Uninstall
1509 ~ **Next-Set-of-Rights** : = {(Add: Set-Of-Rights)} {(Delete: Set-Of-Rights)} {(Replace: Set-Of-Rights)}{(Keep: Set-Of-Rights)}
1510 ~ **Copy-Count** : = (Copies:positive-integer | 0 | Unlimited)
1511 ~ **Control-Spec** : = (Control: {Restrictable | Unrestrictable} {Unchargeable | Chargeable})
1512 ~ **Time-Spec** : = ({Fixed-Interval | Sliding-Interval | Meter-Time} Until: Expiration-Date)
1513 ~ **Fixed-Interval** : = From: Start-Time
1514 ~ **Sliding-Interval** : = Interval: Use-Duration
1515 ~ **Meter-Time**: = Time-Remaining: Remaining-Use
1516 ~ **Access-Spec** : = ({SC: Security-Class} {Authorization: Authorization-ID*} {Other-Authorization: Authorization-ID*} {Ticket: Ticket-ID})
1517 ~ **Fee-Spec**: = {Scheduled-Discount} Regular-Fee-Spec | Scheduled-Fee-Spec | Markup-Spec
1518 ~ **Scheduled-Discount**: = Scheduled-Discount: (Scheduled-Discount: (Time-Spec Percentage)*)
1519 ~ **Regular-Fee-Spec** : = ({Fee: | Incentive: } [Per-Use-Spec | Metered-Rate-Spec | Best-Price-Spec | Call-For-Price-Spec] {Min: Money-Unit Per: Time-Spec}{Max: Money-Unit Per: Time-Spec} To: Account-ID)
1520 ~ **Per-Use-Spec**: = Per-Use: Money-unit
1521 ~ **Metered-Rate-Spec** : = Metered: Money-Unit Per: Time-Spec
1522 ~ **Best-Price-Spec** : = Best-Price: Money-unit Max: Money-unit
1523 ~ **Call-For-Price-Spec** : = Call-For -Price
1524 ~ **Scheduled-Fee-Spec**: = (Schedule: (Time-Spec Regular-Fee-Spec)*)
1525 ~ **Markup-Spec**: = Markup: percentage To: Account-ID

Fig.15

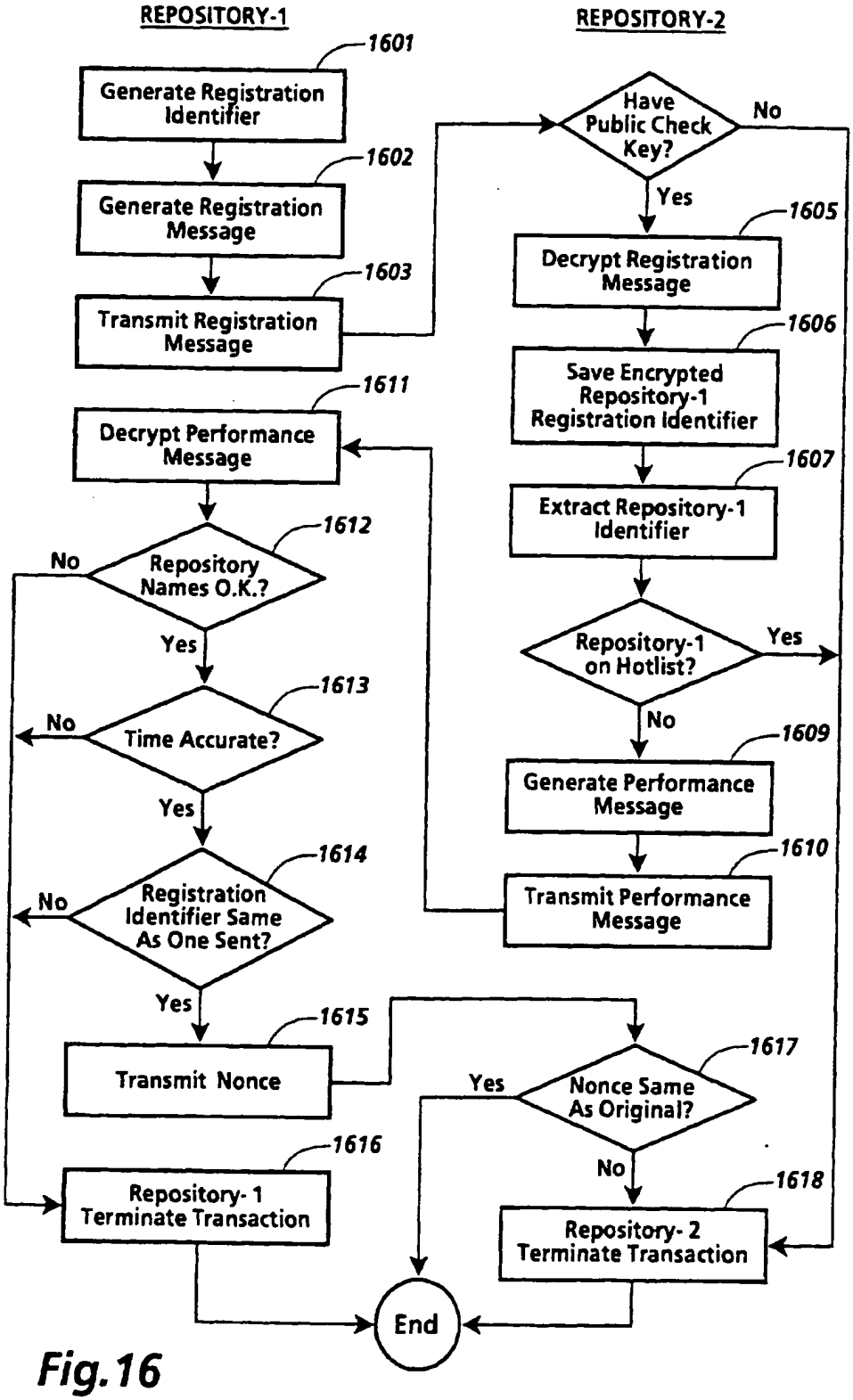


Fig.16

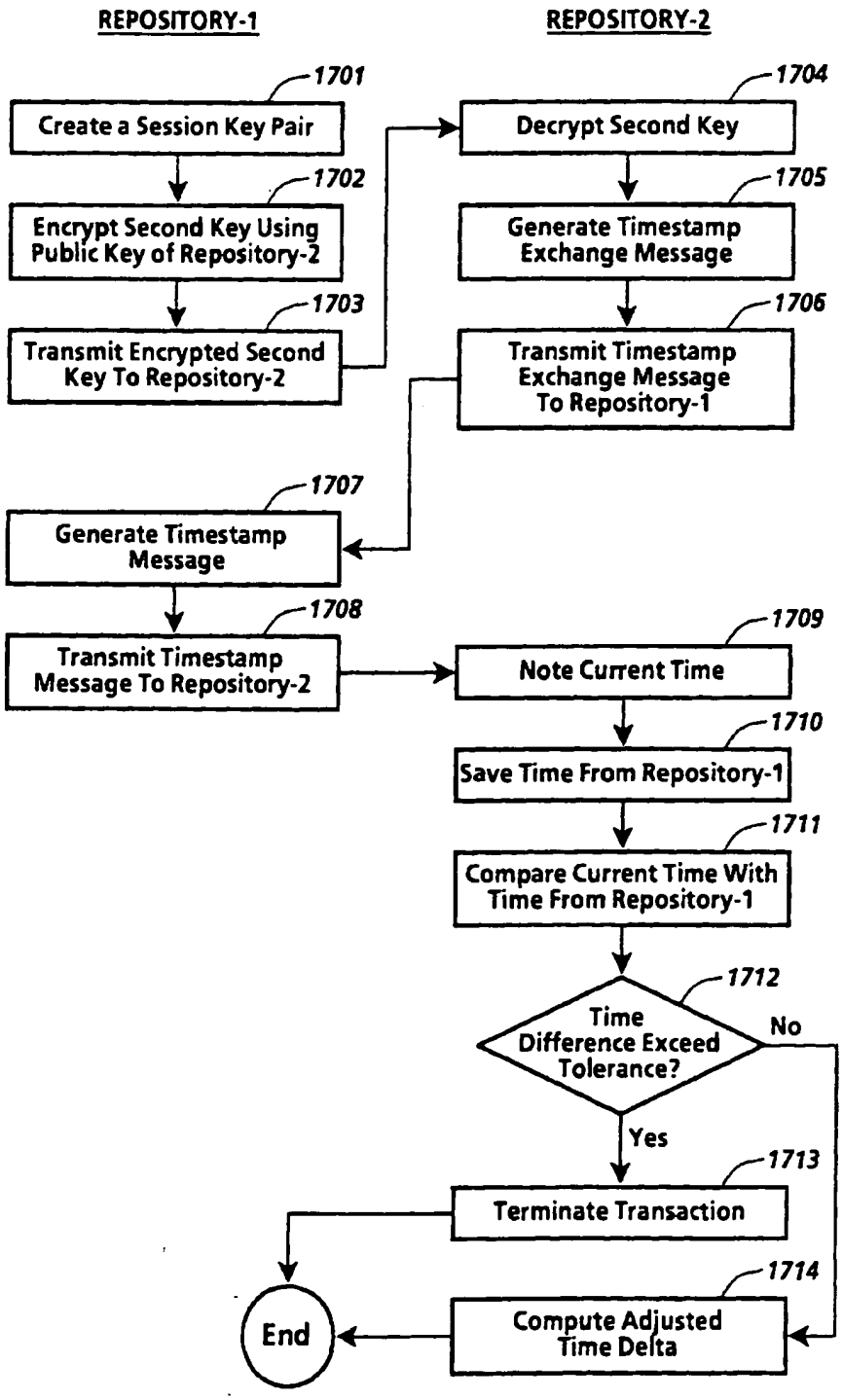


Fig.17

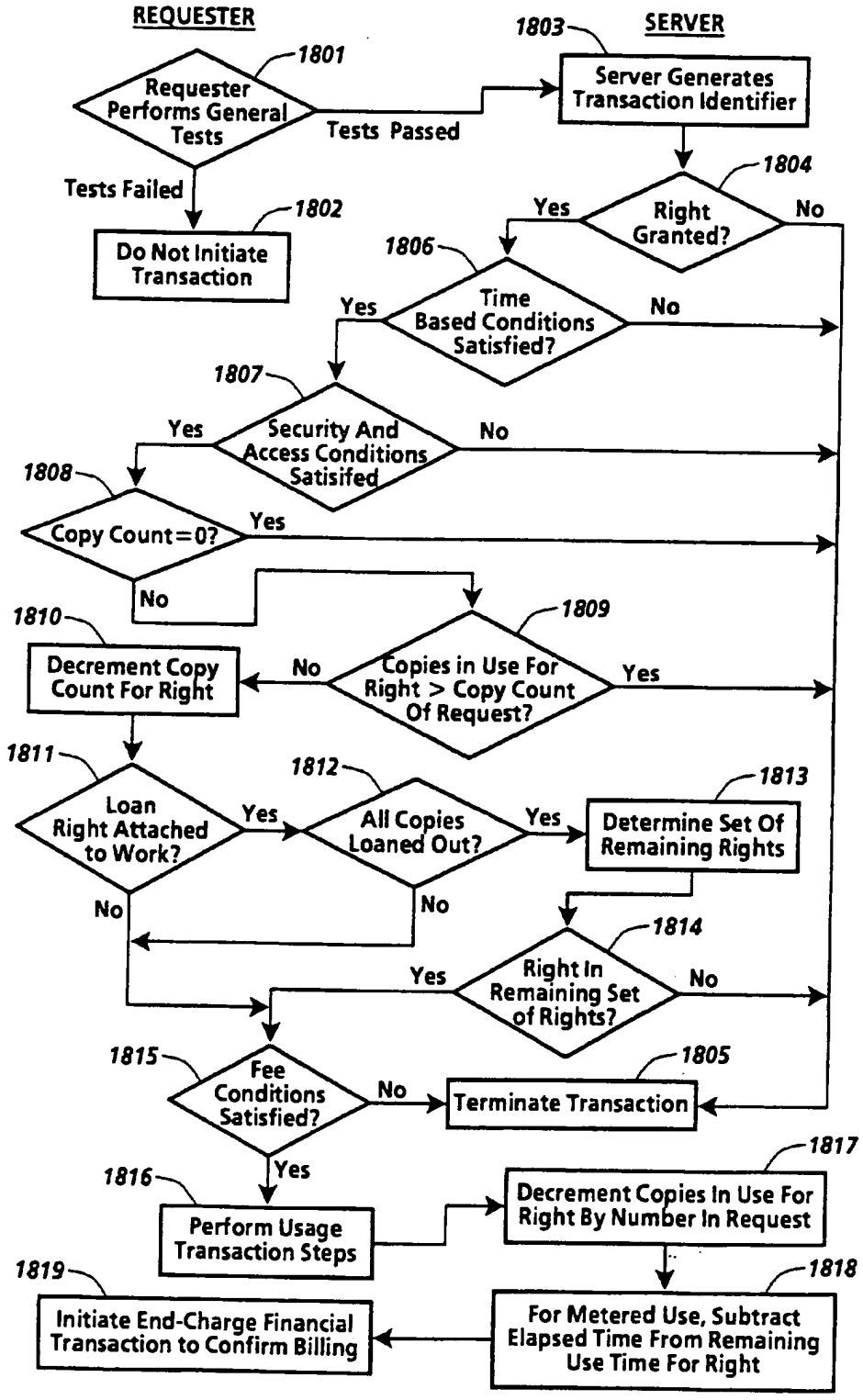


Fig.18

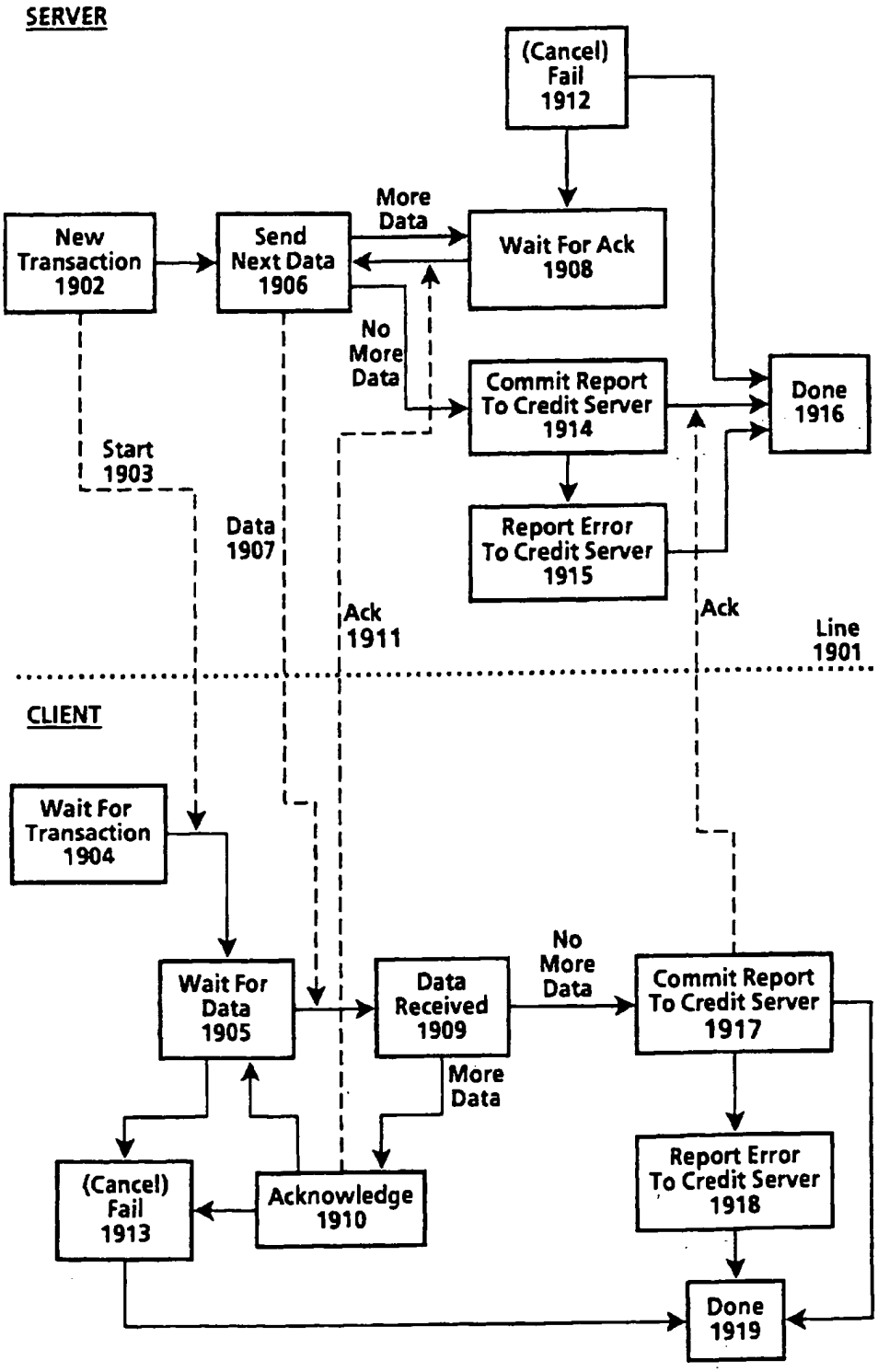


Fig.19

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 95 30 8417

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
A	WO-A-92 20022 (DIGITAL EQUIPMENT CORP.) * page 45, line 10 - page 80, line 19; figures 1-43 *	1,6,10	G06F1/00
A	US-A-5 291 596 (MIITA) * the whole document *	1,6,10	
A	GB-A-2 236 604 (SUN MICROSYSTEMS INC) * page 9, line 11 - page 20, line 15 *	1,6,10	

TECHNICAL FIELDS SEARCHED (Int.Cl.6)

G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	1 April 1996	Moens, R

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

(12) UK Patent Application (19) GB (11) 2 236 604 (13) A

(43) Date of A publication 10.04.1991

(21) Application No 9009655.3

(22) Date of filing 30.04.1990

(30) Priority data
(31) 415884 (32) 02.10.1989 (33) US

(51) INT CL[5]
G06F 1/00

(52) UK CL (Edition K)
G4A AAP

(56) Documents cited
EP 0002390 A1 WO 88/02202 A1

(58) Field of search
UK CL (Edition K) G4A AAP
INT CL[5] G06F 1/00 12/14
Online database: WPI

(71) Applicant
Sun Microsystems Inc

(Incorporated in the USA - Delaware)

2550 Garcia Avenue, Mountain View, California 94043, United States of America

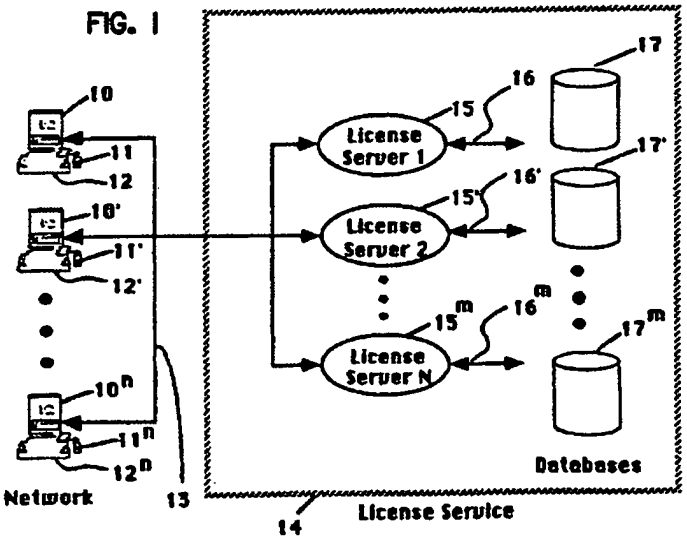
(72) Inventor
John Richard Corbin

(74) Agent and/or Address for Service
Potts Kerr and Co
15 Hamilton Square, Birkenhead, Merseyside, L41 6BR, United Kingdom

(54) **Protecting against the unauthorised use of software in a computer network**

(57) The present invention provides to a software application the verification and licence check out functions which are normally performed by a licence server. The encrypted licence information is contained in a licence token, and is stored in a database 17 controlled by the licence server 15. In contrast to the prior art where the server either grants or denies the request after verifying the user's credentials, the server in the preferred embodiment of the present invention finds the correct licence token for the software application and transmits the token to a licencing library. A licence access module attached to the application decodes the token. Routines in the licencing library coupled to the software application verify the licence information before issuing the licence and updating the token. The access module then encodes the updated token before returning it to the server. Because the verification and issuing function of a token are performed by a software application, the application rather than the server becomes the point of attack by unauthorised users. Reverse engineering the access module is less rewarding than attacking the server because the module reveals the contents of a small fraction of a database of licences.

FIG. 1



At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

GB 2 236 604 A

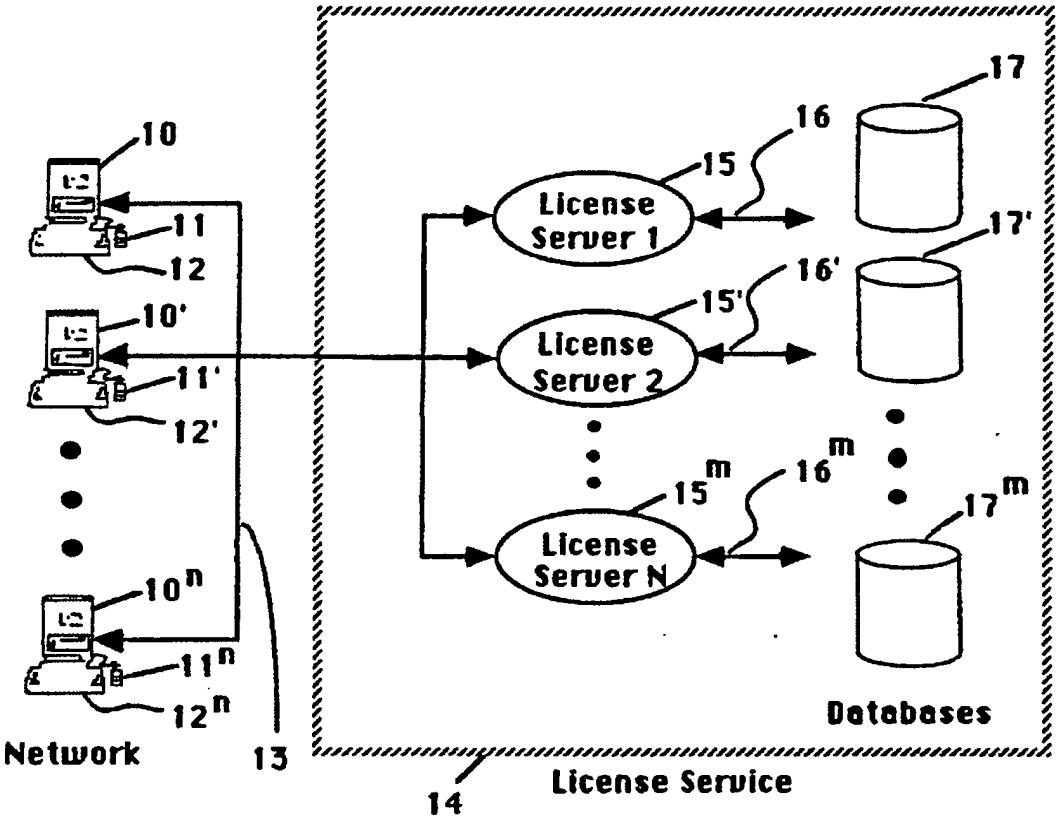

10
11
12
10'
11'
12'
10^n
11^n
12^n
Network
13
14
License Service
License Server 1
License Server 2
License Server N
15
15'
15^m
16
16'
16^m
17
17'
17^m
Databases

FIG. 1

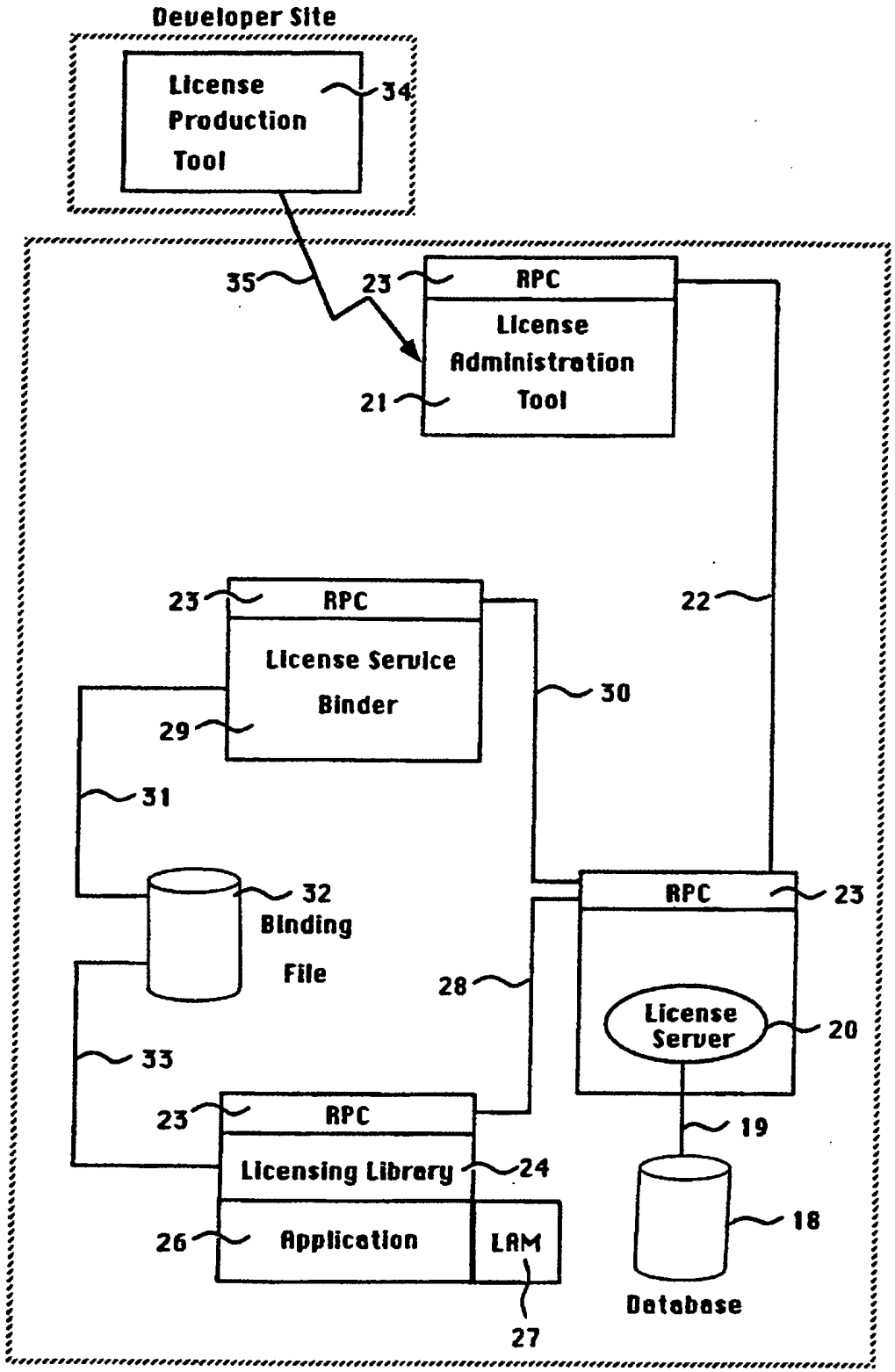

Developer Site
License Production Tool
34
35
23
RPC
License Administration Tool
21
22
23
RPC
License Service Binder
29
30
31
32
Binding File
RPC
23
28
License Server
20
33
23
RPC
Licensing Library
24
19
26
Application
LAM
27
18
Database
Customer Site

FIG. 2

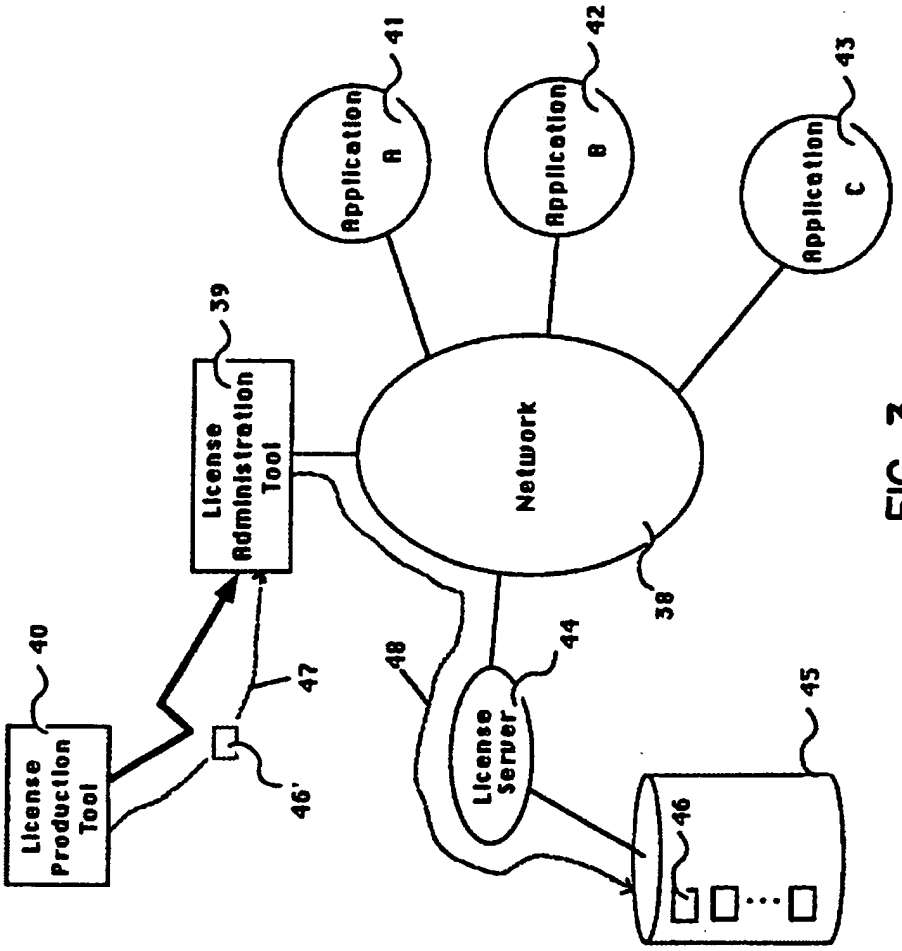

Application A
41
Application B
42
Application C
43
License Administration Tool
39
Network
38
License Production Tool
40
47
46'
48
License Server
44
45
46

FIG. 3

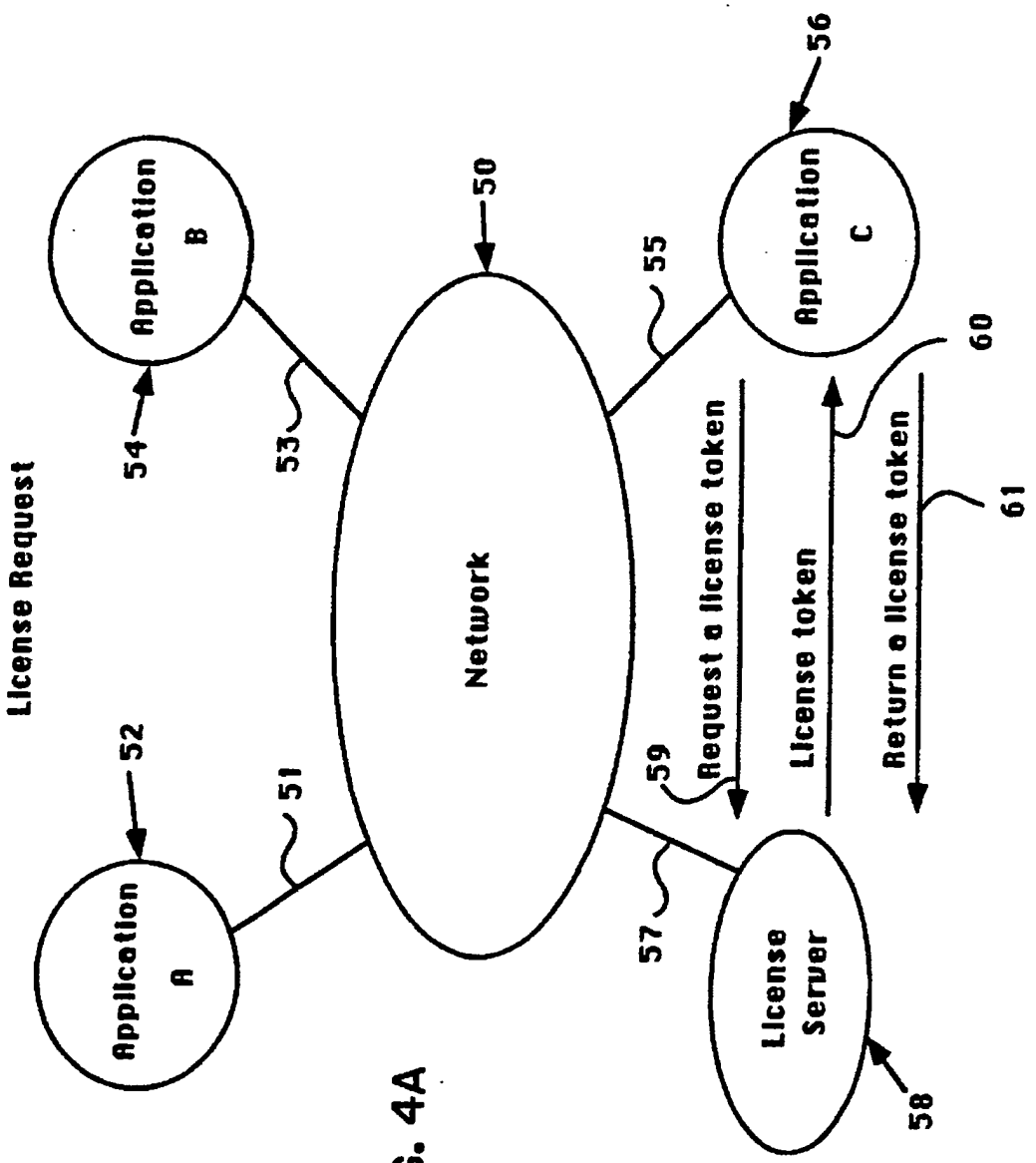

License Request
Application A
Application B
Application C
Network
License Server
50
51
52
53
54
55
56
57
58
59
60
61
Request a license token
License token
Return a license token

FIG. 4A

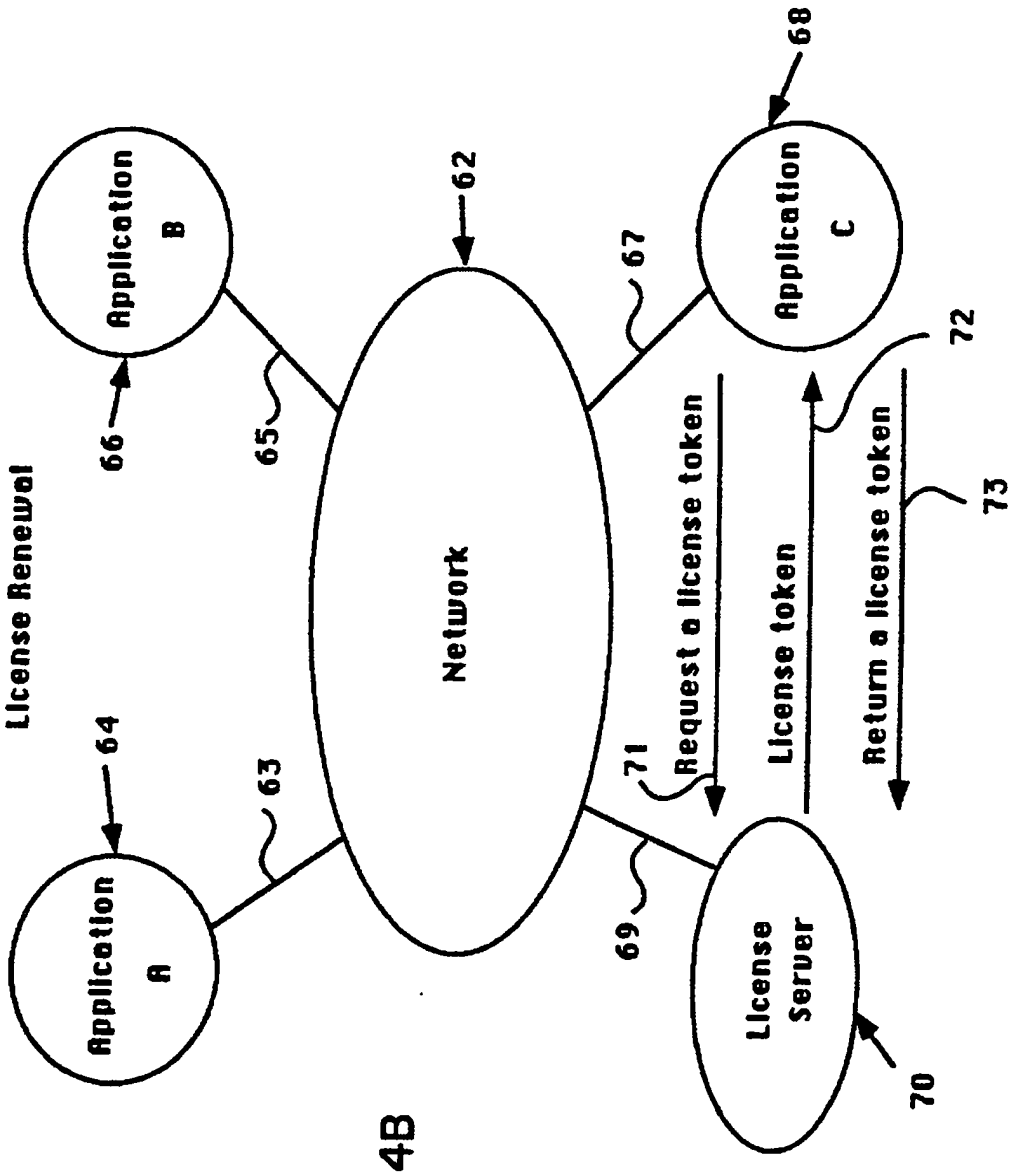

License Renewal
Application A
Application B
Application C
Network
License Server
64
63
66
65
62
67
68
69
70
71
Request a license token
License token
72
Return a license token
73

FIG. 4B

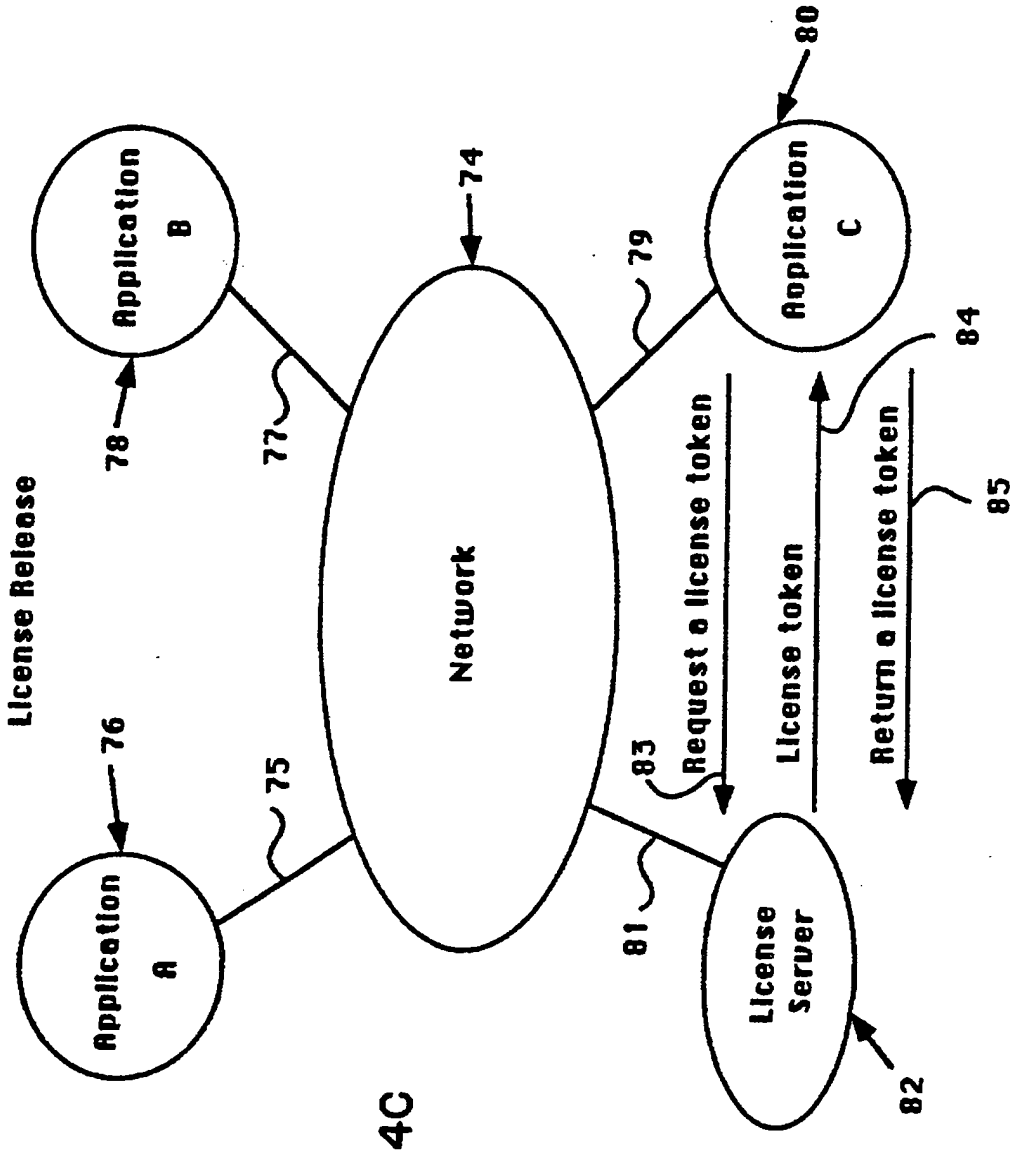

Application B
78
77
74
79
Application C
80
License Release
Network
84
Request a license token
83
License token
Return a license token
85
76
75
Application A
81
License Server
82

FIG. 4C

METHOD FOR PROTECTING AGAINST THE UNAUTHORIZED USE OF SOFTWARE IN A COMPUTER NETWORK ENVIRONMENT

BACKGROUND OF THE INVENTION

1. FIELD OF THE INVENTION

The present invention relates to a method for protecting against the unauthorized use of a software application in a computer network environment.

2. ART BACKGROUND

A computer network is typically an interconnection of machines or agents over links or cables. The open access characteristics of a computer network presents opportunities for the unauthorized copying of software, thus eroding the licensing revenue potential of software developers. Traditionally, either the entire network must be licensed (commonly referred to as a site license), or each node where the software is run must be licensed (commonly referred to as a node license). A node refers to a single machine, agent or system in a computer network. A license is an authorization given by a software developer to a customer to use a software application in a specific manner.

A site license lets all users at a designated location or network use the software application, regardless of their position on the network. This flat-fee approach is an overkill for a low usage software application. A node license not only ties a software application to a particular machine in a network, but also is not cost effective for the infrequent use of a software application. See, for example, U.S. Patent No. 4,688,169. Furthermore, if new users of licensed nodes wish to use the software application, they are often required to purchase additional licenses.

An alternative to a site license or a node license is the concept of a concurrent usage license. A concurrent usage license restricts the number of users allowed to use a software application at any given time, regardless of their location on the network. Just as renters check out available copies of a

movie video from a video rental store, users on a network check out a software application from an agent on a first-come-first-serve basis. Thus, a concurrent usage license charges a fee for the use of a software application proportional to its actual use.

Methods to license a software application for concurrent use in a network environment are currently offered by Highland Software, Inc. and Apollo Computer, Inc. See, M. Olson and P. Levine, "Concurrent Access Licensing", *Unix Review*, September 1988, Vol. 6, No. 9. In general, the license for a software application is stored in a database controlled by a license server. A license server is a program that not only stores the license, but also verifies the user's credentials before checking out the license to the authenticated user. To protect against the authorized use, these methods to license concurrent usage rely on secured communications such as public/private key encryption. Under public/private key encryption, each user of the system has two keys, one of which is generally known to the public, and the other which is private. The private transformation using the private key is related to the public one using the public key but the private key cannot be computationally determined from the public key. See Denning, D., *Cryptography and Data Security*, Addison-Wesley, 1982. The encryption key is hidden in the license server to encrypt the database of licenses. Well designed public/private key encryption schemes are difficult to crack, especially if the license server is located in a trusted environment. A trusted environment is one whose access is limited to users having the proper credentials. However, a license server is more likely to be located at a customer's site and hence in an hostile environment. It follows that the license server is vulnerable to sophisticated intruders. Once the private key is decrypted, all sensitive information on the license server such as licenses are compromised.

It is therefore an object of the present invention to provide a more secure method to protect against the unauthorized use of software in a concurrent use licensing environment.

SUMMARY OF THE INVENTION

The present invention provides to the software application the verification and license check out functions which are normally performed by a license server. The preferred embodiment of the present invention comprises a computer network including a plurality of agents running at least one license server and at least one software application. The license server controls a database of an agent containing the license information for the software application. The license information is contained in a license token, and is stored in the database controlled by the license server. The license token is a special bit pattern or packet which is encrypted by the software vendor of the application software. The software application communicates with the license server through a licensing library. The licensing library is a collection of library routines that the software application invokes to request or renew a license from the license server. Before a software application obtains a license, the license token must be decoded by a license access module. The license access module, which is linked with the software application and the licensing library is a program that decodes the license token from a vendor specific format to a licensing library format.

When an user wishes to run a software application, the licensing library invokes a call to request a license token from the license server. In contrast to the prior art where the license server either grants or denies the request after verifying the user's credentials, the license server in the preferred embodiment of the present invention finds the correct license token for the software application and transmits the license token to the licensing library. The license access module attached to the licensing library decodes the licensing token. Routines in the licensing library coupled to the software application verify the license information before checking out the license and updating the license token. The license access module encodes the updated license token before returning it to the license server.

Because the verification and check out function of a license token are performed by a software application, the software application rather than the license server becomes the point of attack by unauthorized users. Reverse engineering the license access module is less rewarding than attacking the license server because the license access module reveals the contents of a fraction of a database of licenses. By the time most attackers crack the license access module, the software vendors would most likely introduce newer versions of the software application and new license access modules for them. Thus the present invention provides a more secure method for protecting against the unauthorized use of a software application in a computer network environment without modifying the underlying computer network.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 illustrates a network environment employing the present invention.

Figure 2 describes the architecture of a network licensing scheme employing the preferred embodiment of the present invention.

Figure 3 describes the installation of a license token in the preferred embodiment of the present invention.

Figure 4a illustrates the use of a license token to request a license from a license server in the preferred embodiment of the present invention.

Figure 4b illustrates the use of a license token to renew a license from a license server in the preferred embodiment of the present invention.

Figure 4c illustrates the use of a license token to release a license from a license server in the preferred embodiment of the present invention.

NOTATION AND NOMENCLATURE

The detailed description that follows is presented largely in terms of algorithms and symbolic representations of operations on data bits and data structures within a computer memory. These algorithmic descriptions and representations are the means used by those skilled in the data processing arts to most effectively convey the substance of their work to others skilled in the art.

An algorithm is here, and generally, conceived to be a self-consistent sequence of steps leading to a desired result. These steps are those requiring physical manipulation of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It proves convenient at times, principally for reasons of common usage, to refer to these signals as bit patterns, values, elements, symbols, characters, data packages, or the like. It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities.

Further, the manipulations performed are often referred to in terms, such as adding or comparing, that are commonly associated with mental operations performed by a human operator. No such capability of a human operator is necessary, or desirable in most cases, in any of the operations described herein that form part of the present invention; the operations are machine operations. Useful machines for performing the operations of the present invention include general purpose digital computers or other similar devices. In all cases there should be borne in mind the distinction between the method of operations in operating a computer and the method of computation itself. The present invention relates to method steps for operating a computer in processing electrical or other (e.g. mechanical, chemical) physical signals to generate other desired physical signals.

The present invention also relates to an apparatus for performing these operations. This apparatus may be specially constructed for the required purposes, or it may comprise a general purpose computer as selectively activated or reconfigured by a computer program stored in the computer. The algorithms presented herein are not inherently related to any particular computer or other apparatus. In particular, various general purpose machines may be used with programs written in accordance with the teachings herein, or it may prove more convenient to construct a more specialized apparatus to perform the required method steps. The required structure for a variety of these machines will appear from the description given below.

DETAILED DESCRIPTION OF THE INVENTION

The following detailed description is divided into several sections. The first of these sections describes a general network environment for accessing a database of licensed software programs. Subsequent sections discuss the details of a method for protecting against the unauthorized use of a software application.

I. General Network Environment

Referring to **Figure 1**, computer network environment comprises a plurality of data processing devices identified generally by numerals **10** through 10^n (illustrated as **10**, **10'** and 10^n). These data processing devices may include terminals, personal computers, workstations, minicomputer, mainframes and even supercomputers. For the purposes of this Specification, all data processing devices which are coupled to the present invention's network are collectively referred to as "agents". It should be understood that the agents may be manufactured by different vendors and may also use different operating systems such as MS-DOS, UNIX, OS/2, MAC OS and others. Particular examples of suitable agents include machines manufactured by Sun Microsystems, Inc., Mountain View, Calif. Each of the agents has an input device such as a keyboard **11**, **11'** and 11^n or a mouse **12**, **12'** and 12^n. As shown, agents **10** through 10^n (illustrated as **10**, **10'** and 10^n) are interconnected for data transfer to one another by a common cable **13**. It will be appreciated by one skilled in the art that the common cable **13** may comprise any shared media, such as coaxial cable, fiber optics, radio channel and the like. Furthermore, the network resulting from the interconnection of the cable **13** and agents **10** through 10^n (illustrated as **10**, **10'** and 10^n) may assume a variety of topologies, such as ring, star, bus, and may also include a collection of smaller networks linked by gateways or bridges.

Referring again to Figure 1 is a license service 14. The license service 14 is a resource shared by every agent connected to the network. In the preferred embodiment of the present invention, the license service 14 comprises license servers 15 through 15^m (illustrated as 15, 15' and 15^m) and databases 17 through 17^m (illustrated as 17, 17' and 17^m), where m is less than or equal to n. A license server is a program that runs on an agent with a memory storage capability. Each license server 15 (illustrated as 15, 15' and 15^m) communicates with a database 17 stored in memory on the agent over an interface 16 (illustrated as 16, 16' and 16^m). As will be described in detail below, the database 17 stores licensing information for various software applications which are purchased and authorized to run in the computer network environment. The license server is not limited to run on a specific agent, but can operate on any agent including the agent on which the user is to operate the application. Thus, any agent connected to the network may function as a license server as well as a device on which a user may operate application software. As will be described below, the license server does not perform verification of licenses of application software; rather the license server is passive and provides storing, locking, logging, and crash recovering function for the application software.

Figure 2 illustrates the architecture of a network licensing scheme of the present invention. The architecture comprises a database 18, database interface 19, license server 20, licensing library 24, License access module 27, license administration tool 21, license service binder 29, and license production tool 34.

The database 18 stores licensing information and application usage data. Preferably the database 18 comprises a plurality of records which contain the following information:

Database Element	Description
Unique Key Table	Keys for all other tables
Vendor Table	Vendor's ID and name
Product Table	Product number and name
Version Table	Version number and date
License Table	License #, exp date, total units
License Token Table	Stores encoded license token
Unit Group Table	A group's allocation of license
Group List Table	Name of the group
Allowed Users Table	Credentials of allowed users
Current License Use Table	Applications using a license
Lock Table	Locked records in database
Authorized administrator Table	Login names of administrators
License Operation Log Table	Administrator's log information
License Usage Log Table	Request handle plus Client Log
License Queue Log Table	License wait queue
Application Message Log Table	Application specific messages

A database interface 19 provides communication between the license server 20 and the database 18 in order to prevent concurrent access to the same database record by multiple users which can cause the data in the record to become corrupted. Thus, only the owner of the lock can read from and write to the locked record during the usage of the application.

The license server 20 operates on an agent and interfaces the database 18 to license administration tool 21, licensing library 24 and license service binder 29. The license server 20 communicates with the license administration tool 21, licensing library 24 and license service binder 29 via an interface 23. Preferably the interface 23 is a remote procedure call

mechanism which permits a process operating on one device or agent connected to the network to request a resource or service from a remote device or agent connected to the network. See A. Birrell and B. Nelson, "Implementing Remote Procedure Calls," *ACM Transaction on Computer Systems*, February 1984, Vol. 2, No. 1.

Multiple license servers may reside on multiple agents. Preferably the license server 20 operates in a background mode of the agent such that its operation is transparent to a user of that agent. More particularly, as will be described below, the license server 20 provides the following functions: 1) servicing the requests from the licensing library 24 for license token; (2) maintaining a wait queue for requests to the database 18 when no licensing units are available; (3) generating `locks` for exclusive access to database 18; and (4) providing access to information in the database 18.

The licensing library 24 is a set of library routines which enable the application 26 to request licensing service from the license server 20. Upon receiving the request for service from the licensing library 24, the license server 20 retrieves a license token from the database 18 and transmits it to the licensing library 24. The licensing library 24 is linked with the application 26 and communicates with the license server 20 over a path 28 with, preferably, a remote procedure call mechanism 23. Among the major library calls in the licensing library 24 is the application's request for a license from the license server 20. Other important library calls include the request to renew and to release a license. The use of the license token to accomplish the request for the various licensing service will be described in detail below.

The license access module (LAM) 27 is prepared by the software vendor 24 to decode the license token. Once decoded, the application 26 via routines in the licensing library verifies the licensing information in the license token and determines whether a license may be checked out. The LAM 27

also encodes the license token before the application returns it to the database 18 via license server 20. The license access module 27 is described in further detail below.

The license administration tool 21 is utilized by the network administrator to perform administrative functions relevant to the concurrent usage of a software application. The license administration tool 21 may run on any agent connected to the computer network. The license administration tool 21 is primarily used to install the license token into the database 18 through the license server 20. The functionality of the license administration tool 21 includes: (1) starting or terminating a license server, (2) accessing a database controlled by a license server; and (3) generating and printing reports on license usage.

The application 26 may not access the database 18 directly; rather, the request for a license is made through the licensing library 24 to the license server 20 over a path 28. Most network licensing schemes employ secured communication between the licensing library 24 and the license server 20. In contrast, the present invention uses the license access module (LAM) 27 the license library 24 and a plurality of license tokens to protect against the unauthorized use of software application in a computer network.

Referring once again to Figure 2, a license service binder 29 is shown coupled to the license server 20 over a path 30. The license service binder 29 is invoked by means known in the art, such as a network service program. The license service binder 29 locates all agents that are designated as servers on the network, and keeps track of which server is servicing which application. The license service binder 29 contacts each server on its table of available servers and requests a list of products it serves. Finally the license service binder 29 writes the contents of the table of available license servers and the list of products into a binding file 32 over a path 31. In Figure 2, the binding file 32 is coupled to the licensing library 24 over a path 33. The application 26

queries the binding file 32 to see which license server can service its request for a license.

A license production tool 34 is used by the software vendor to create a license token for transmittal to the network administrator. Receiving the license token, the network administrator installs it with the license administration tool 21 into the database 18 through license server 20.

II. License Token

Referring to Figure 3, the creation of a license token in a computer network employing the preferred embodiment of the present invention will be described. A computer network 38 is shown coupled with a license administration tool 39 and a single license server 44. The license server 44 communicates with a database 45. Applications 41, 42, and 43 are shown requesting licensing service from the license server 44. When a customer purchases a license for an application, such as a CAD/CAM program for its research and development department, the software vendor creates a license token with a license production tool, and delivers the license token to the customer's network administrator. A license token is a special bit pattern or packet representing a license to use a software application. The network administrator installs the license token 46 into the database of the license server using the license administration tool 39. Unlike the token used in a token ring which is passed from agent to agent, a license token in the preferred embodiment of the present invention is passed only between a license server and a licensing library for a predetermined amount of time. The predetermined amount of time corresponds to the time the license token is checked out of the license server. Currently, the license token is checked out to an application for no more than ten seconds, and the license token is returned as quickly as possible to the issuing license server. The license token 46 contains information encrypted in the vendor's format such as ,vendor identification, product and version numbers as well as the number of license units purchased

for the license token. A license unit corresponds to the license weighting for an agent connected to the computer network. For example, powerful workstations could require more license units to use a software application than an average personal computer.

The software vendor produces a license token using a license production tool 40. A path 47 illustrates how a license token 46' makes its way to a license administration tool 39 at the customer's site. There, the system administrator installs the license token 46' as license token 46 into the license database 45 of the license server 44. A path 48 indicates the transfer of the license token 46' from the license administration tool 39 to the license server 44 and into the database 45 as license token 46. The license server 44 is now ready to entertain requests from applications 41, 42, and 43 for a license to use the application corresponding to token 46 as well as other applications represented in its database 45.

It should be understood that each network may have a plurality of license servers and each license server may have in its database a plurality of license tokens for a variety of software applications. Referring again to Figure 3, if application A 41 requests and checks out the license token 46 for less than ten seconds, applications B and C 42, 43 would be unable to check out the license token 46 if their requests were made during the same time application 41 is checking out a license from the license token 46 because of the locking mechanism provided by database interface 19. Thus, to achieve concurrent license usage in network 38, it is preferred that the network administrator installs more than one license server. To minimize the task of recovering from license server crashes, it is also preferred that the system administrator spreads the license units for any one application among a plurality of strategically located license servers. For instance, if a network has four license servers, the network administrator may want to allocate the twenty license units for a particular popular application among four license tokens with

five license units for each license token. In the event one license server crashes or the license token is checked out, the other three license servers may provide licensing service to other applications.

Figure 4a illustrates the use of a license token to request for a license. As shown, a network 50 is provided, and is coupled to Applications 52, 54 and 56 respectively. Application 56 succeeded in requesting a license token from the license server 58 in step 59. The license token is transmitted to application 56 in step 60. When done, Application 56 returns the license token to the license server 58 in step 61. Aside from the license request function performed with the license token as shown in Figure 4a, the license token is also used in other critical stages of the licensing process. For example, an user may wish to run an application beyond the initial allocated time. As shown in Figure 4b, Application 68 makes a license renewal request 71 from the license server 70 with license token 72. Similarly, in Figure 4c the user makes a license release request 83 when the application no longer needs the license units. As such, the user updates the license token 84 by returning the updated license token to the license server 82 in step 85.

III. License Access Module

In Figure 2, a license access module (LAM) 27 is linked with the application 26 and the licensing library 24 to form the executable code that software vendors ship to the customers. The license access module 27 decodes and encodes the encrypted license token as it is being passed between the license server and the licensing library 24. Thus the level of security of an application from unauthorized use depends heavily upon how secure the license access module is.

Conventional network licensing schemes use public/private key encryption to encode sensitive information. Such a scheme is effective if the license server is in a trusted environment. However, the customer has the

same access to any agent in a network, including the license server. The security of the licensing scheme can be compromised by a user who decrypts the license server's private key. Once the unauthorized user determines the server's private key, he can decrypt all sensitive information on the license server. Should all license servers use the same key, as is frequently done, then all the security of the applications served by all the license servers will be compromised.

The license access module 27 first translates a license token from a vendor specific format to a format usable by the licensing library 24. The license access module accomplishes the translation in two modules. One module translates or decodes a license token from a vendor specific format to a licensing library format. The second module translates or encodes the updated license token from the licensing library format to the vendor specific format. The second module is invoked anytime the licensing library updates the information in a license token.

Upon receiving the license token in the licensing library format, the licensing library invokes routines which verify the correctness of the license by reviewing the following license information stored in the token: (1) flag, (2) maintenance contract date, (3) host name and domain, (4) product name, (5) host id number, (6) license serial number, and (7) expiration date of license. This is compared to the information maintained by the application. If the information matches, the license is verified. After completing the verification process, a routine in the licensing library is initiated which checks out the license by decrementing the license units in license token by the number of licensing units being checked out.

The decoding and encoding routines allow software vendors to implement their own security mechanism to protect their licenses from unauthorized use even though they reside at the customer's site.

Below is an example of a sample application using the licensing library and the license access module written in C language:

```
#define LIC_RENEWAL_TIME (60)              /set renewal time for this session/
#define EST_LIC_RENEWAL_TIME (LIC_RENEWAL_TIME x .9)

NL_vendor_id NL_Vendor_id = 1223;                  /set vendor #/
NL_prod_num NL_Prod_num = "02"                     /set product #/
NL_version NL_Version = ( 12/20/88, "1.0" );       /set version id #/

--
      status = NL_init (vendor_id, NULL, &job_id); /initialize license service/
      if (status != NL_NO_ERROR)      /accept job id if no error/
            {
            fprintf (stderr, "nl_init failed - error =
            %d\n", status );                       /error message if error and
                                                   return/
            return;
            }
      units = 3;
      code_funcs.encode_p = nl_encode;   /pointer to encode function/
      code_funcs.decode_p = nl_decode;   /pointer to decode function/
      if (signal (SIGALRM), alarm_intr ) == (void *) -1)      /set alarm if no
                                                              error/
            {
            perror ("Cannot set SIGALRM"); /otherwise, error message/
            return;
            }
      status = NL_request (job_id, NL_Prod_num,  /request a license/
      &NL_Version,
      units, LIC_RENEWAL_TIME, NL_L2_SRCH,
      &code_funcs, NULL,
      &req_handle, NULL, &app_info);

      if (status != NL_NO_ERROR)              /no error, license checked
            {                                 out from license server/
            fprintf (stderr, "nl_request failed - error =
            %d\n", status);                   /otherwise, error message/
            return;
            }

      /*
       * We got a license                     /license request successful/
       */

      alarm (EST_LIC_RENEWAL_TIME);           /set alarm for license renewal
                                               time/
--    Application Runs                        /runs application/
--
      status = NL_release (req_handle);       /request to release a license/
      if (status != NL_NO_ERROR)
            {
            fprintf (stderr, "nl_release failed - error =       /otherwise, error
```

```
                %d\n", status);                          messages/
            return;
            }

    int
    alarm_intr ()
            {

    status = NL_confirm (req_handle,      /renew licensing unit with
    LIC_RENEWAL_TIME, NULL);               licensing server/
    /*
     * Verify vendor private information
    */
            }

    If (status!= NL_NO_ERROR)
    fprintf (stderr, "nl_confirm failed - error =      /otherwise, error
    %\n", status);                                     message/
            {
                    puts ("license renewed")           /successful license
            }                                          renewal/
```

The sample application given above is accompanied by self-explanatory annotation to the right margin of the codes. Of particular interest are code_func.encode_p and code_func.decode_p. Encode_p and decode_p are pointers to the software vendor's encode and decode routines, respectively. Taking the pointers in the code_func variable, the licensing library can use the pointers to invoke the decoding and encoding routines in the license access module. The three major licensing library routines, request for a license (NL_request), release a license (NL_release) and renew a license (NL_confirm) invoke the decoding and encoding routines. For example of a license access module, see Appendix 1.

In implementing the license access module, the license server becomes merely a repository for license tokens. The licensing library coupled to the application performs the procedure of authenticating the license token prior to granting a license and therefore access to run the application.

Because the level of security of the system is dictated by the license access module, the software vendors are free to make the license access module as simple or as complex as they desire. In particular, they are free to

adopt any of the encryption schemes as part of their encryption routines. If the security mechanism is broken, and the encryption known to others, then the software vendors can easily remedy the situation by releasing a new version of the product with a new license access module.

While the present invention has been particularly described with reference to Figures 1-4 as well as Appendix 1, and with emphasis on certain language in implementing a method to protect against the unauthorized use of software application in a computer network environment, it should be understood that they are for illustration only and should not be taken as limitation upon the invention. In addition, it is clear that the method of the present invention has utility in any application run in a computer network environment. It is contemplated that many changes and modifications may be made, by one skilled in the art, without departing from the spirit and scope of the invention disclosed above.

CLAIMS

1. In a computer network environment including a plurality of software applications licensed to run on at least one network of agents, said applications located on said agents wherein use of the application on a particular agent is permitted upon the grant of a license, said license being requested by a user from said agent of said applications, a system for protecting against the unauthorized use of said applications comprising:

license token means for storing licensing information of said applications; license server means connected to said agents for communicating with said applications, said license server means having a database which stores said license token means, said license server means further retrieving said license token means from said database upon a request for a license by said applications, said license server means further transmitting said license token means to said applications;

license access means connected to said agents for decoding and encoding said license token means from said license server means, said license access means being integrated with said applications, said license access means receiving said license token means from said license server means; and

licensing library means connected to said agents for verifying said decoded license token means before access to said license is granted, said licensing library means being integrated with said applications.

2. The system as defined in claim 1, wherein each said license token means containing licensing information for at least one version of each said applications.

3. The system as defined in claim 1, wherein the contents of said license token means is encrypted.

4. The system as defined in claim 1, wherein said license token means is passed between said license server means and said licensing library means for a predetermined time period.

5. The license token means as defined in claim 4, wherein during said predetermined time period, only one said applications may check out one said license token means.

6. The system as defined in claim 1, wherein said license server means receives said request for a license from said applications, said license server searches in said database for a license token means storing the license requested by said application before retrieving said license token means.

7. The system as defined in claim 1, wherein said license access means decodes the contents of said license token means before said licensing library means verifies said license token means.

8. The system as defined in claim 1, wherein said license access means encodes said license token means after said licensing library verifies said license token means and prior to returning said license token means to said license server means.

9. The system as defined in claim 1, wherein said licensing library verifies said license token means by

comparing the licensing information stored in said license token means with the licensing information maintained by said application.

10. The system as defined in claim 1, wherein said licensing library means checks out said license of said application in response to a positive comparison of the license information.

11. The licensing library means as defined in claim 10, wherein said license for said application being checked out after said licensing library verifies said license token means.

12. In a computer network environment including a plurality of software applications licensed to run on at least one network of agents, said applications located on said agents wherein use of the application on a particular agent is permitted upon the grant of a license, said license being requested by a user from said agent of said applications, a system for protecting against the unauthorized use of said applications comprising:

license token means for storing licensing information of said applications;

license server means connected to said agents for communicating with said applications, said license server means having a database which stores said license token means, said license server means further retrieving said license token means from said database upon a request for a license by said applications, said license server means further transmitting said license token means to said applications;

license access means connected to said application and accessible from said agents for decoding and encoding said license token means from said license server means, said license access means being integrated with said applications;

licensing library means connected to said application and accessible from said agents for verifying said decoded license token means before access to said license is granted, said licensing library means being integrated with said applications; and

license binding means connected to said license server means and to said licensing library means for constructing a binding file, said binding file informing said licensing library means which of said license server means may grant a license to said application.

13. The system as defined in claim 12, wherein said licensing library means are located on the same agents as said applications.

14. The system as defined in claim 12, wherein said license sever means are located on the same agents as said licensing library means.

15. The system as defined in claim 12, wherein each said license token means contains licensing information for at least one version of each of said applications.

16. The system as defined in claim 12, wherein the contents of said license means is encrypted.

17. The system as defined in claim 12, wherein said license token means is passed between said license server

means and said licensing library means for a predetermined time period.

18. The license token means as defined in claim 17, wherein, during said predetermined time period, only one of said applications may check out one said license token means.

19. The system as defined in claim 12, wherein said license server means further transmit said license token means to said licensing library means.

20. The system as defined in claim 12, wherein said license access means decodes the contents of said license token means before said licensing library means verifies said license token means.

21. The system as defined in claim 12, wherein said license access means encodes said license token means after said licensing library verifies said license token means and prior to returning said license token means to said license server means.

22. The system as defined in claim 12, wherein said license binding means constructs said binding file by contracting each said license server means to request for a list of applications it serves, said binding file containing said list of applications available from said license server means.

23. In a computer network environment including a plurality of software applications licensed to run on at least one network of agents, said applications located on

said agents wherein use of the application on a particular agent is permitted upon the grant of a license, said license being requested by a user from said agent of said applications, a system for protecting against the unauthorized use of said applications substantially as hereinbefore described with reference to the accompanying drawings.

Published 1981 at The Patent Office, State House, 66/71 High Holborn, London WC1R 4TP. Further copies may be obtained from Sales Branch, Unit 6, Nine Mile Point, Cwmfelinfach, Cross Keys, Newport, NP1 7HZ. Printed by Multiplex techniques ltd, St Mary Cray, Kent.

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 840 194 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
06.05.1998 Bulletin 1998/19

(51) Int. Cl.[6]: G06F 1/00

(21) Application number: 97108754.9

(22) Date of filing: 02.06.1997

(84) Designated Contracting States:
DE FR GB
Designated Extension States:
AL LT LV RO SI

(30) Priority: 29.10.1996 JP 286345/96

(71) Applicant:
MATSUSHITA ELECTRIC INDUSTRIAL CO., LTD.
Kadoma-shi Osaka (JP)

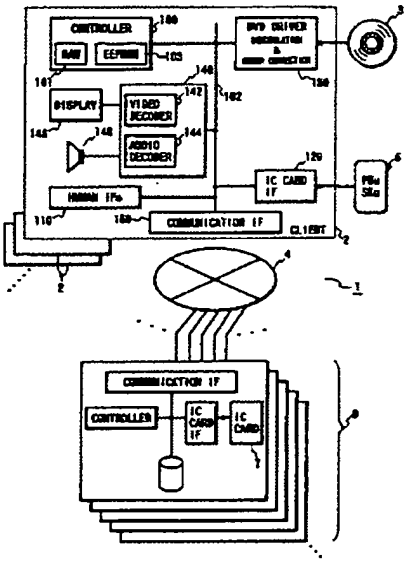
(72) Inventors:
- Uranaka, Sachiko
Tokyo (JP)
- Kiyono, Masaki
Kamakura-shi, Kanagawa-ken (JP)

(74) Representative:
Pellmann, Hans-Bernd, Dipl.-Ing. et al
Patentanwaltsbüro
Tiedtke-Bühling-Kinne & Partner
Bavariaring 4
80336 München (DE)

(54) System and method for controlling the use of a package of distributed application software

(57) A system for permitting only an authentic user to play a desired application contained in a distributed application package in one of predetermined operation, e.g., free play mode, charged mode, limit-attached play mode, etc. The system comprises a client for playing an application under the control of a server connected with the client through a communication network. The application package (the volume) includes a distribution descriptor which contains mode codes assigned to the volume and the applications of the volume. The data of distribution descriptor is decided and stored in the descriptor at the time of distribution of the volume. This feature makes the system flexible. There is also disclosed a system operatable without communicating with a server.

FIG. 1



EP 0 840 194 A2

Description

BACKGROUND OF THE INVENTION

1. Field of the Invention

The invention generally relates to a security system and, more specifically, to a method and system for permitting an authentic user to use charged information which has been distributed via package or transmission media while charging and controlling the use of distributed charged information.

2. Description of the Prior Art

In order to use charged information such as music, movies, games, etc. provided by information providers that provide various programs of such charged information, a user has generally to take two steps. In the first step (or obtaining step), the user obtains a desired program from one of the information providers by purchasing a package media such as an FD (floppy disc), an optical disc (e.g., CD-ROM (compact disc read only memory) and DVD (digital versatile disc or video disc)), etc. on which the desired program is recorded (off-line distribution or obtaining) or by down loading the desired program from the server computer of an information provider through a predetermined procedure (on-line distribution or obtaining). In case of the on-line obtaining, the user may either play the program while obtaining it (i.e., the two steps are executed in parallel) or store the program while obtaining it in the first step and execute the program later as the second step (or using step). In case of the off-line obtaining, in the second step the user loads the obtained recording media into an appropriate device and directly plays (or executes) the program or once stores the program into the memory of the device and then plays the program.

Japanese Patent unexamined publication No. Hei7-295674 (1995) discloses a security system for use in the second or using step for a CD-ROM. In this system, the user can use encrypted information which is recorded together with a public key of a toll center (a center public key) on a CD-ROM by encrypting with the center public key and sending a code of desired program included in the information and a user-generated key to the information provider and by decrypting the information with an encryption key which has been encrypted with the user-generated key and sent by the information provider. However, the identity of the user is not verified, permitting a mala fide user who have obtained other person's CD-ROM to use it. Further, the center public key is pressed together with the encrypted information on the CD-ROM. This makes it difficult to change the center public key. Also, this causes different providers who probably want to use different center public keys to force the CD-ROM manufacturer to use different masters (or stampers) in pressing the CD-ROMs.

Japanese Patent unexamined publication No. Hei7-288519 (1995) discloses a security system for use in both the first and second steps. However, this system is only applicable to a system in which charged information is distributed on line.

Japanese Patent unexamined publication No. Hei8-54951 (1996) discloses a system in which the quantity of used software is monitored, and further software use by the user is impeded if the quantity exceeds a predetermined quantity. Since a dedicated hardware is necessary for impeding of software use, this system is only suitable for the use in a server in a on-line distribution system.

There is also a system for permitting a user to use, only for a trial period, software which has been distributed with data defining the trial period. In this system, a mala fide user may make the software reusable by installing the software again or setting the user system clock for a past time.

There are these and other programs in the art. It is an object of the invention to provide a system for permitting only an authentic user (a user who have legally obtained charged information either on line or off line from an information provider) to use the charged information without any limitation, charging for each time of its use, or within the tolerance of a use-limiting factor (e.g., the quantity used, the days elapsed since the day of its purchase or the current date) according to the type of the charged information.

SUMMARY OF THE INVENTION

According to the principles of the invention, it is assumed that charged information or an application package is distributed, either via package (or recording) media or via transmission media, together with at least control information such as a media title and a media code, etc. However, an illustrative embodiment will be described mainly in conjunction with charged information recorded on and distributed by means of the DVD.

For any type of charged information, charged information has been encrypted with a key and recorded on a DVD when obtained by a user. If distributed charged information to be played is of the limitlessly playable type, the charged information processing is achieved in the following way: the key is first obtained in a user public key-encrypted form from

the DVD on which the key has been recorded at the time of selling the DVD; the user public key-encrypted key is decrypted with a user secret key stored in a IC card into a decrypted key; and the encrypted charged information is decrypted with the decrypted key and consumed (that is, played or executed). The user-public key-encrypted key may be obtained on line from the server serving the client (device).

If distributed charged information to be played is of the usage-sensitive charging type, the user is charged for each time of using the information. In this case, prior to processing the charged information, the client double-encrypts and sends a user's credit card number to one of the to 11 servers of the provider of the information; the server adds an amount (e.g., play time or duration) used associated with the information to the value in a total amount (software meter) field in a volume data table, and sends the updated total amount value to the client; and the client displays the updated total amount. Then the client starts the charged information processing.

If distributed charged information to be played is of the limit-attached type, that is, the use of the information is to be limited by the tolerance of a certain limiting factor concerning the information consumption, then the client is permitted to consume the charged information only if the use-limiting factor is within the preset limit. In case of this type of charged information, prior to processing the charged information, the client sends the identifier (ID) code of a user specified application which is recorded on the DVD to the server; on receiving the ID code the sever tests if the use-limiting factor associated with the user specified application is within the preset limit; if not, then the server informs the client of the test result, and the client displays the test result; if the test was successful, then the server updates the meter (or integrated value) of the use-limiting factor and sends the updated value to the client; and in response to the reception of the updated value the client displays the updated value. Then the client starts the charged information processing.

BRIEF DESCRIPTION OF THE DRAWING

Further objects and advantages of the present invention will be apparent from the following description of the preferred embodiments of the invention as illustrated in the accompanying drawings. In the drawing,

FIG. 1 is a block diagram showing an arrangement of a system for permitting a user to use a distributed application package on the terms of use of the package with a higher security according to a first illustrative embodiment of the invention;

FIG. 2 is a diagram showing an exemplary structure of an application (or a charged information) package recorded on a DVD used in the inventive system;

FIGs. 3 and 4 are diagrams showing, in a detailed form, exemplary data structures of the volume descriptor 22 and the distribution descriptor 23, respectively;

FIG. 5 is a flow chart of a volume control program for playing the application(s) recorded on the DVD according to the principle of the invention;

FIG. 6A is a diagram showing an exemplary structure of a volume data table stored in a server shown in FIG. 1;

FIG. 6B is a diagram showing an exemplary structure of a application data table stored in a server 8;

FIG. 7 is a diagram showing a structure of a server table 75 stored in the EEPROM 103 of the client 2;

FIGs. 8A and 8B are flow charts of initial routines executed interactively by the client 2 and the server 8, respectively, at the beginning of the processes 650, 700 and 800.

FIG. 9 is a flow chart showing a procedure of a free play process shown as step 650 in FIG. 5, wherein connecting adjacent blocks by two flow lines indicates that each block is executed interactively by a client and an associated server;

FIGs. 10A and 10B are flow charts jointly showing a procedure formed of exemplary expected play time informing routines interactively executed;

FIGs. 11A and 11B are flow charts jointly showing a procedure formed of exemplary timed play and metered usage report routines interactively executed for playing an application while timing the duration and displaying a timed play duration after the play;

FIGs. 12A and 12B are flow charts jointly showing a procedure formed of exemplary timed application-play subroutines interactively executed for playing the application while timing the duration;

FIGs. 13A and 13B are flow charts jointly showing a procedure formed of alternative timed application-play subroutines interactively executed in which timing of play time is achieved with a timer in the client;

FIG. 14 is a flow chart of an exemplary application play subroutine called in steps 612 and 622 of FIGs. 12A and 13A, respectively, and executed by the controller 100;

FIG. 15 is a flow chart showing a procedure of a charged play process 700 shown as step 700 in FIG. 5,

FIGs. 16A and 16B are flow charts jointly showing a procedure formed of exemplary expected charge informing routines interactively executed;

FIGs. 17A and 17B are flow charts jointly showing a procedure formed of routines interactively executed in block 650 of FIG. 15;

FIGs. 18A and 18B are flow charts jointly showing a procedure formed of exemplary timed play and metered charge report routines interactively executed for playing an application while timing the duration and displaying a charge and a total amount of charges after the play;

FIG. 19 is a flow chart showing a procedure interactively executed by the client 2 and the server 8 in the operation block 800 of FIG. 5, wherein blocks connected with two flow lines indicates that operation of the blocks is done by the two elements 2 and 8;

FIGs. 20A and 20B are a key-encrypting key table and a user's public key table, respectively, stored in the server; and

FIG. 20C is a flow chart of a process for obtaining the application encrypting key K_v from the server 8;

FIG. 21 is a block diagram of an exemplary decipherer-built-in IC card IF according to the invention;

FIG. 22 is a diagram showing a K_v decoder used in place of the K_v decoder 126 of FIG. 21 in a system 1 using the cryptosystem of FIG. 20C;

FIG. 23 is a diagram for explaining the meanings of the terms-of-use (TOU) codes and the corresponding limit values;

FIG. 24 is a block diagram showing an arrangement of a system for playing a distributed application package on the terms of use of the package without communicating with any server according to a second illustrative embodiment of the invention;

FIG. 25 is a flow chart schematically showing an exemplary control program executed by the controller 100a shown in FIG. 24;

FIGs. 26 and 27 are flow charts showing an operation of a free play mode shown in step 650a of FIG. 25 in a detailed form and a further detailed form, respectively; and

FIG. 28 is a flow chart showing an operation of a limit-attached play mode shown in step 800a of FIG. 25.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

For the sake of better understanding of the following description, it will be useful to define some terms to be used.

Charged information provided by an information provider may be distributed off-line (in off-line distribution) or on-line (in on-line distribution). In off-line distribution, the charged information is recorded on package media or recording media, and distributed through the sales network of the provider, that is, sold at stores in the sales network. The package media include all sorts of portable recording media such as various types of magnetic discs, a variety of optical memory discs (e.g., CD, CD-ROM, DVD), and magnetic tapes and cartridges. In on-line distribution, the charged information is transmitted via transmission media from the servers at the service points of the provider and the distributors aligned with the provider to the client device (e.g., PC (personal computer)) of the user who requested the charged information, and stored in a recording media of the client (device). The transmission media include any telecommunication channels which permit data communication between the servers and the client device. The package media and the transmission media are hereinafter referred to en bloc as "distribution media".

The charged information may be any type of software such as music, movies, games, etc. which are each referred to as an "application" without discrimination. The distribution unit of charged information is referred to as a "charged information package" or an "application package". There may be included one or more applications in an application package.

The present invention relates to a system for permitting a user to use a distributed application package on the terms of use of the package with a higher security.

Embodiment I

For the purpose of simplicity, a first illustrative embodiment will be described in which package media, among other things, DVDs are used as distribution media.

FIG. 1 is a block diagram showing an arrangement of a system for permitting a user to use the application(s) recorded on a DVD on the terms of use of the DVD with a higher security according to the first illustrative embodiment of the invention. In FIG. 1, the system 1 comprises a client or DVD player 2 which plays a DVD 3, a telecommunication network 4, and a server 8 at a toll center of the provider 6 which provides the application package of the DVD 3.

FIG. 2 is a diagram showing an exemplary structure of an application (or a charged information) package 20 recorded on the DVD 3 used in the inventive system 1. In FIG. 2, the application package 20 comprises at least one application 21, a volume (or package) descriptor 22 comprising data concerning the application package 20, and a distribution descriptor 23 comprising data which is determined mainly at the time of, e.g., distribution or sales after the pressing of the DVD 3. (The volume descriptor 22 and the distribution descriptor 23 constitutes the volume control data of the volume 20.) In this embodiment it is assumed that a volume (or package) control program which controls the use of the application package 20 in cooperation with the server 8 is included in and distributed with the application package

20. Thus, the application package 20 further comprises the package control program 24 suited for the terms of use of the package 20. The application(s) 21, the volume descriptor 22 and the package (or volume) control program 24 are recorded in the data area of the DVD 3 at the time of manufacturing the DVD 3, while the distribution descriptor 23 is recorded in the burst cutting area at the time of, e.g., sales of the DVD 3.

FIGs. 3 and 4 are diagrams showing, in a detailed form, exemplary data structures of the volume descriptor 22 and the distribution descriptor 23, respectively. In FIG. 3, the volume descriptor 22 at least contains a volume identifier (VID_v) 25 which the title of the application package 20 is probably used for and which is the same as the application identifier if the package or volume 20 contains only one application; a provider identifier 26; volume creation date and time 27 which may be used for the base point by which volume expiration data and time as described later is determined; and volume effective date and time 28 indicative of date and time until which the volume 20 is available. If the volume 20 contains more than one applications, the volume descriptor 22 further contains application identifiers (AID_a's) 29.

In FIG. 4, the distribution descriptor 23 comprises the fields of: a volume issue number (NO_{v-i}) 30 which contains a serial number given to each of the distributed application packages of an identical volume identifier (volume ID or title) VID_v in the order of distribution; a server public key (PK_s) 31 the data of which is given by the server 6 at a toll center of the provider 6; a PK_u (user-public-key)-encrypted application-encrypting key (K_v) 32; and sales date and time 33. The key PK_s 31 field contains a key which has been used in encrypting each application 21 in the package 20 and which has been encrypted with a user public key (PK_u) of the user who has legally obtained the package 20. Appropriate data are recorded in all of the fields 30 through 34 at the time of distribution of the package 20, i.e., at the time of sales of the DVD 3 in this embodiment.

The distribution descriptor 23 further comprises the field 34 of terms-of-use code (mode code) plus limit value for the volume (the volume limit value field) and, for each of the application IDs 29, the fields 35 of terms-of-use code plus limit value for the application ID 29 (application limit value field). If terms of use are set only to the volume 20, there is no need of the field 35. If terms of use are set to each application, the field is empty.

FIG. 23 is a diagram for explaining the meanings of the terms-of-use (TOU) codes and the corresponding limit values. In FIG. 23, the terms-of-use code may be, e.g., one byte in length. The higher digit (X) of the TOU code indicates the target to which the terms of use is applied as shown in table 36. That is, higher digits of 0, 1, 2,... indicate that the TOU codes beginning with those digits are for the entire volume, application 1, application 2 and so on. The lower digit (Y) of the above mentioned terms-of-use code indicates the terms of use of the package 20 or the application 21 to which the code is set, and is directly followed by a corresponding limit value as shown in table 37 of FIG. 23. Specifically, the terms-of-use code (or TOU code)of 00H means, for example, that the volume 20 is usable freely after distribution. The value '31H' means, for example, that the application 3 to which the TOU code is set can be used by paying per unit of play duration. The lower digit of 2H or more means that the volume 20 or the application to which the TOU code is set can be used freely until the corresponding limit value are reached, which disables further use. As seen from the table, the use-limiting factors determined by the TOU codes whose lower digits are 2H to 5H are the current date and time, the expiration date and time, the amount of used period, and the access count, respectively.

Since the data of the distribution descriptor 23 can be set as described above, this provides both the providers and the users with more flexibility than conventional system can provide.

Again in FIG. 1, the DVD player 2 comprises a controller 100 for controlling the entire DVD player 2; data bus 102 connected with the not-shown CPU (central processing unit), not-shown ROM (read-only memory), RAM (random access memory) 101, and EEPROM (electrically erasable programmable ROM) 103 included in the controller 100; human interfaces (IFs) 110 including input devices such as a keyboard, a voice recognition device, a mouse, a remote controller, etc. ; an IC card interface (IF) 120 for connecting the bus 102 with the ROM (not shown) in a IC card 5; a DVD driver 130 for reading out the data recorded on the DVD 3 and for demodulating and error-correcting the read data; a video and audio output IF 140 for receiving a MPEG 2 bit stream and outputting a video and audio output signals; a display device 146; a loudspeaker 148, and a communication IF 150 for communicating through the public telecommunication network 4. The IC card 5 stores a user's password PW_u and a user's secret key SK_u which corresponds to the user's public key PK_u mentioned in conjunction with the PK_u-encrypted AP-encrypting key (K_v) contained in the field 32 of the distribution descriptor 23 recorded in the burst cutting area of the DVD 3. The video and audio output IF 140 includes a MPEG 2 video decoder 142 and a MPEG 2 audio decoder 144.

As for obtaining the DVD 3, there may be some ways. If one is to buy a DVD 3, e.g., at some book store or through mail order, he or she has to have the PK_u-encrypted version of an application-encrypting key (K_v) recorded in the burst cutting area of the desired DVD 3 by notifying his or her public key PK_u which corresponds to his or her secret key SK_u stored in the IC card 5. If one is a member of a DVD distribution service, he or she can obtain a DVD with a PK_u-encrypted AP-encrypting key recorded without notifying the PK_u each time of obtaining because he or she must have notified the PK_u when he or she applied for the service.

In operation, the user first sets a desired DVD 3 in the DVD driver 130 of the DVD player 2, and issues a start command to the DVD player 2 through an appropriate human IF 110. In response to a receipt of the start command, the

controller 100 reads the volume control program 24 from the data area of the DVD 3 through the DVD driver 130 while loading the read program 24 into the RAM 101 of the controller 100, and then executes the volume control program 24.

FIG. 5 is a flow chart of the volume control program 24 for playing the application(s) 21 recorded on the DVD 3 according to the principle of the invention. In FIG. 5, the controller 100 first checks the AID1 field to see if the volume 20 contains a single application in step 500. If not, then the controller 100 displays the application IDs in the field 29 and prompts the user to select a desired one of the applications in step 502, and waits for the selection in step 504. If any application is selected in step 504, the controller 100 registers the application ID of the application as the application to be played in step 506 and proceeds to step 508 to check the field 35 of the terms-of-use (TOU) code plus limit value for the selected application to see if the field is empty. If so, the controller 100 proceeds to step 510 to read the volume limit field 34.

On the other hand, if the test result is YES in step 500, then the controller 100 registers the volume ID as the application to be played in step 512, and reads the volume limit value 34 in step 510.

If the step 510 is completed or the test result of step 508 is NO, then the controller 100 checks the terms-of-use (TOU) code to see if the lower digit of the TOU code is 0 in step 514. If so, then the controller 100 plays an application free of charge in step 650, and otherwise makes another check to see if the lower digit of the TOU code is 1 in step 516. if so, the controller 100 plays an application in a usage-sensitive charging in step 700, and otherwise (if the lower digit of the TOU code is 2 or more) play an application only when the software meter of a use-limiting factor is under a preset value in step 800. On completing any of the steps or processes 650 through 800, the controller 100 ends the program 24. Thus, the DVD player 2 plays a program specified by the user according to the terms of use determined by the TOU code which has been set to either the application package or the specified application.

The processes 650, 700 and 800 are executed interactively with an associated server 8. The servers 8 need various data for executing these processes, and store such data in the form of tables.

FIG. 6A is a diagram showing an exemplary structure of a volume data table stored in a server 8. In FIG. 6A, Each of the records of the volume data table 60 comprises volume ID (VID_v) and issue No. (NO_{v-i}) fields. The combination of VID_v and NO_{v-i} serves as the user ID of the user of the application package 20 or the DVD 3. For this reason, the table 60 has, for the members or subscribers of DVD distribution service or the like, personal data fields which contains, for example, a member ID, a name, an address, etc. Each record further comprises a volume minute meter field ($VM\text{-}METER_{v-i}$) containing a software meter of play duration in minute which is attached to (or associated with) the volume 20; a volume charge meter ($VC\text{-}METER_{v-i}$) containing a software charge meter which is attached to the volume 20; a limit value (LV_{v-i}) containing a limit value associated with the TOU code (e.g., the effective date and time, the allowable expiration date and time, the allowable access, etc.); a limit value meter ($LV\text{-}METER_{v-i}$); an application ID (AID_{v-i-a}) field containing the title of the application; an application minute meter ($AM\text{-}METER_{v-i-a}$) field containing a software meter of play duration in minute which is attached to the application of AID_{v-i-a}; an application charge meter ($AC\text{-}METER_{v-i-a}$) field for a software meter of play duration in minute which is attached to the application of AID_{v-i-a}; a limit value (LV_{v-i-a}) containing a limit value associated with the TOU code; and a limit value meter ($LV\text{-}METER_{v-i}$).

FIG. 6B is a diagram showing an exemplary structure of a application data table stored in a server 8. In FIG. 6B, the application data table 70 comprises the fields of, for example, an application code ($ACODE_n$), an application title (AID_n), a duration (D), a rate-per-access (RATE/ACCESS), an access count, a minute meter, etc. The duration is a period of time what it takes to play the application. The rate per access is a charge for a play of the whole application, which is used for informing the user of an expected play duration prior to a play. The rate per unit time is a charge for a unit time of play, which is used for the calculation of a charge for an actually timed play duration. The access count and minute meter fields contains the number of accesses to the application and a total amount of play time, which are not necessary for the present invention but will be used in statistical calculations for the analysis of, e.g., the tastes.

FIG. 7 is a diagram showing a structure of a server table 75 stored in the EEPROM 103 of the client 2. In FIG. 7, the fields of the table 75 comprises a server public key (PK_s), a server ID (SID_s), a server network address ($SADD_s$), etc. this table 75 is used for associating the sever public key (PK_s) contained in the distribution descriptor 23 recorded in the burst cutting area of the DVD with the ID and the network address.

Play an Application Free of Charge

The initial routines of the processes 650, 700 and 800 are the same.

FIGs. 8A and 8B are flow charts of initial routines 80a and 80b which are executed interactively by the client 2 and the server 8, respectively, at the beginning of the processes 650, 700 and 800. In FIG. 8, the controller 100 of the client or the DVD 2, in step 82, sends a service request with the network address $CADD_c$ of the client or DVD 2, the TOU code plus limit value, the volume ID (VID_v), the issue number (NO_{v-i}), the application ID (AID_{v-i-a}), and other data to the associated server 8 the ID of which is SID_s (SID_s is obtained from the table 75 in FIG. 7 by using the public key recorded on the DVD 3), and in step 92 waits for a response from the server (SID_s) 8. If there is a response from the server (SID_s), the client 2 proceeds to the next step through a circle with "A" therein.

On the other hand, in FIG. 8B, the server 8 of SID_s receives the message from the client 2, that is, the service request and the accompanying data and stores date in a predetermined location for subsequent use in step 84. Then, the server 8 searches the table 60 for a record which contains VID_v and NO_{v-i} in the volume ID and issue No. fields thereof, respectively in step 86. If the search is unsuccessful, then the server 8 adds the record for VID_v and NO_{v-i} and fills relevant fields with AID_{v-i-a} and a limit value, if any, in the table 60 in step 88, and proceeds to step 90. Also, if the search in step 86 is successful, the server 9 proceeds to step 90, where the server 8 selects a routine to execute next according to the value of the TOU code and enters the selected routine through a circle with "B" therein. In this case, if the TOU code = x0H (x: an arbitrary HEX number, the letter H in the last position indicates that the preceding number is in hexadecimal), then a routine for playing an application free of charge is selected. If the TOU code = x1H, then a routine for playing an application in usage-sensitive charging is selected. If the TOU code ≥ x2H, then a routine is selected which plays an application only if the software meter of a use-limiting factor is under a preset value.

FIG. 9 is a flow chart showing a procedure of a free play process shown as step 650 in FIG. 5, wherein connecting adjacent blocks by two flow lines indicates that each block is executed interactively by a client of $CADD_c$ and an associated server SID_s as shown in detail later. If the TOU code is 0 in step 514 of FIG. 5, then the server ($CADD_c$) enters the free play process 650 as shown in FIG. 9, and the client and the server (SID_s) execute the initial routine 80 in block 660. In block 670, they executes an expected play time informing routine, that is, displays an expected play time before playing an specified application. In block 680, they execute an application play and metered play time report routine. Since the routine 80 has been detailed in FIG. 8, the expected play time informing routine and the application play and metered play time report routine will be detailed in the following.

FIGs. 10A and 10B are flow charts jointly showing a procedure formed of exemplary expected play time informing routines 97a and 97b interactively executed by the client 2 and the associated server 8, respectively. In FIG. 10B, the server 8 retrieves the duration (D_n) of the application of AID_{v-i-a} from the table 70 in a well known manner in step 91. In the next step 92, the server 8 calculates an expected total amount of play time according to the value of the TOU code. Specifically, if the TOU code is 0xH, then the client adds the duration (D_n) and the value of the $VM\text{-}METER_{v-i}$ field of the record identified by VID_v and NO_{v-i} in the table 60. If the TOU code is axH (a: the application number of the specified application in the volume), then the client adds the duration (D_n) and the value of the $AM\text{-}METER_{v-i-a}$ field of the record identified by VID_v, NO_{v-i}, and AID_{v-i-a} in the table 60. Then the server 8 sends the result to the client whose network address is $CADD_c$ in step 93, and ends the process.

On the other hand in FIG. 10A, the client 2 receives the incoming message or the value of the updated meter in step 94. In the next step 95, the value is displayed as the total amount of usage. Then the client 2 ends the process.

In updating a relevant meter, a predetermined value of duration has been used in the just described routines of FIG. 10 (a preset value metering system). This arrangement is suited mainly for such applications as it takes a constant time to play, and will not cause a problem unless the user discontinues the play. From this point of view, it is preferable to actually measure the playing time in metering (a timed value metering system). However, it is also noted that the preset value metering system is useful in informing the user of expected play time prior to an actual playing.

FIGs. 11A and 11B are flow charts jointly showing a procedure formed of exemplary timed play and metered usage report routines 675a and 675b interactively executed by the client and the server, respectively, for playing an application while timing the duration and displaying a timed play duration after the play. In the routine 675, the client and the server call a timed application-play subroutine for playing the application while timing the duration (play time) in step 200.

Then the server 8 proceeds to step 210, where the client updates a relevant meter according to the TOU code in the same manner as in step 92 of FIG. 10B. Specifically, if the TOU code is 0xH, then the play time is added to the value of the $VM\text{-}METER_{v-i}$ field of the record identified by VID_v and NO_{v-i} in the table 60. If the TOU code is axH (a: the application number of the specified application in the volume), then the play time is added to the value of the $AM\text{-}METER_{v-i-a}$ field of the record identified by VID_v, NO_{v-i}, and AID_{v-i-a} in the table 60. Then the server 8 sends the play time and the value of the updated meter (i.e., the total amount of play time) to the client whose network address is $CADD_c$ in step 212, and ends the process.

On the other hand, the client 2, after step 200, make a test to see if there is a response from the server of SID_s in step 214. This step is repeated until the client 2 receives a call from the server 8, when the client 2 receives the incoming message or the value of the updated meter in step 216. In the next step 218, the client 2 displays the play time and the total amount of play time, and then ends the routine 675.

FIGs. 12A and 12B are flow charts jointly showing a procedure formed of exemplary timed application-play subroutines 205a and 205b executed by the client 2 and the server 8, respectively, for playing the application while timing the duration. The server 8 of SID_s waits for a notice in step 611 to see if the client has started playing the application. On the other hand, the client 2 of $CADD_c$ informs the server of a start of play in step 610 and immediately call an application play subroutine in step 612. This, causes the server 8 to start a timer in step 613, and waits for a notice of a stop of play from the client 2 in step 615. On completing the step 612, the client informs the server 8 of the stop of play in step 614. In response to this notice, the server 8 stops and reads the timer as the play time in step 617. After steps 614 and 617, the client and the server return.

Though the above described arrangement has used a timer of the server, it may be possible to use a timer of the client.

FIGs. 13A and 13B are flow charts jointly showing a procedure formed of alternative timed application-play subroutines 205ac and 205bc interactively executed by the client 2 and the server 8, respectively, in which timing of play time is achieved with a timer in the client. In the alterative subroutine 205a, the client 2 starts a timer in step 620, calls an application play routine in step 622, stops the timer in step 624, sends the play time to the server 8 in step 626, and then returns. On the other hand, the server 8, on entering the subroutine 295b, waits for a call from the client of $CADD_c$ in step 621. If there is a call from the client 2, then the server 8 receives the play time in step 623 and then returns.

However, the arrangement of FIG. 13 has a possibility of permitting a mala fide user to manipulate the timer of the client 2. From this point of view, the arrangement shown in FIG. 12 is preferable to that of FIG. 13.

FIG. 14 is a flow chart of an exemplary application play subroutine called in steps 612 and 622 of FIGs. 12A and 13A, respectively, and executed by the controller 100.

Prior to the description of the flow chart, we define some notation concerning encryption and decryption. If encrypting X with a key EK according to an encrypting algorithm e yields Y, then it is expressed as:

$$e(EK, X) = Y.$$

Similarly, if decrypting Y with a key DK according to a decrypting algorithm d yields Z, then it is expressed as:

$$d(DK, Y) = Z.$$

Assuming that the algorithms e and d and the keys EK and DK correspond each other, that is, $d(DK, Y) = X$, it follows that

$$d(DK, e(EK, X)) = X.$$

Returning now to FIG. 14, the controller 100 read the PK_u-encrypted application-encrypting (AP-encrypting) key (K_v) or $e1(PK_u, K_v)$ from the filed 32 of the distribution descriptor 23 of the DVD in step 602. Here,

$$v = 1, 2, \ldots V,$$

where V is the number of kinds of the application package. This indicates that different application-encrypting keys K1 through K_v is assigned to respective kinds of applications, that is, volume VID1 through VID_v.

In the next step 604, the user secret key SK_u is read from the IC card 5. In the next step 606, the PK_u-encrypted AP-encrypting key $e1(PK_u, K_v)$ is decrypted with the user secret key SK_u to obtain the application encrypting key K_v. Then in the next step 608, the K_v-encrypted application (AP), i.e., $e(K_v, AP)$ which is recorded on the DVD 3 is decrypted with the obtained AP-encrypting key K_v to obtain $d(K_v, e(K_v, AP)) = AP$, while passing the obtained application data to the video and audio output IF 140. The obtained application data has the form of an MPEG 2 bit stream. The video and audio output IF 140 converts the MPEG 2 bit stream of the application data into video and audio output signals through MPEG 2 video and audio decoding. The video and audio output signals are applied to the display device 146 and the loudspeaker 148, respectively.

Play an Application in Usage-sensitive Charging system

FIG. 15 is a flow chart showing a procedure of a charged play process 700 shown as step 700 in FIG. 5, wherein connecting adjacent blocks by two flow lines indicates that each block is executed interactively by a client of $CADD_c$ and an associated server of SID_s. In FIG. 15, the client 2 enters the process 700 via step 516 of FIG. 5 and proceeds to block 630, where the client 2 and the associated server 8 execute the initial routine 80. In the next block 640, the client 2 displays an expected charge and a total amount of charges received from the server 8, and let the user decide whether to play the desired application.

FIGs. 16A and 16B are flow charts jointly showing a procedure formed of exemplary expected charge informing routines 640a and 640b interactively executed by the client 2 and the associated server 8, respectively. The routines 640a and 640b are very similar to the routine 97 except that in the routine 640, the DURATION (D_n) or "play time" has been replaced with RATE PER ACCESS and "charge"; between steps 92a and 93a, there has been added a step 641 of the server generating and storing a pseudo random number R in a memory location R'; in step 93a, the server sends the pseudo random number R as well; between steps 94 and 95a there has been added a step 643 of the client storing the received pseudo random number R in a memory location R" for subsequent use. The replacement of DURATION (D_n) with RATE PER ACCESS is achieved by accessing a RATE PER ACCESS field 74 instead of a DURATION field

73 in table 70. Further, in the routine 640 there have been added the following steps: in step 644 following the step 96a, the client 2 makes a check to see if the user decides to play the application; if not, the client 2 sends a quit message to the server of $SADD_s$ in step 645, and ends the routine 640; on the other hand, in step 642 following the step 93a, the server 8 of SID_s waits for a call from the client 2 of $CADD_c$; on receiving a call from the client, the server makes another check in step 646 to see if what has been received is a quit message; if so, the client ends the routine 640; and if the user decided to play the application in step 644, which means that what the server has received is not a quit message but an encrypted credit card number as seen from the description below, then the client 2 and the server 8 proceed to the step 650 of FIG. 15.

In the next block 650, the server 8 obtains a user's credit card number (CCNOu) through the client 2 keeping the security of the card number as shown in FIGs. 17A and 17B. In step 647, the client 2 encrypts the credit card number of the user which has been input by the user through a human IF 110 with a key, i.e., the pseudo random number R which has been stored in a memory location R' in step 643 of FIG. 16A to obtain e2(R, CCNOu). In the next step 648, the client 2 further encrypts R + e2(R, CCNOu) with another key or a server public key read from the distribution descriptor 23 recorded in the burst cutting area of the DVD to obtain

$$e1(PK_s, R + e2(R, CCNOu)).$$

In the next step 649, the client 2 sends the encrypted data to the server 8. Through step 646 of FIG. 16B, the server proceeds to step 650, where the server 8 finds that what was received from the client $CADD_c$ is encrypted data. In the next step 651, the server 8 reads a server secret key SK_s from an IC card 7. in the next step, the server 8 decrypts the received encrypted data with the server secret key SK_s as follows;

$$d1(SK_s, \text{encrypted data}) = d1(SK_s, e1(PK_s, R + e2(R, CCNOu)) = R + e2(R, CCNOu).$$

In step 653, the server 8 makes a check to see if the just obtained pseudo random number R coincides with the random number R which has been stored in a memory location R' of the server. if so, the server 8 sends an enable message to the client of $CADD_c$, and in step 655 decrypts e2(R, CCNOu) with the pseudo random number R to obtain the user's credit card number CCNOu. On the other hand, in response to a reception of the enable message in step 657, the client 2 exits from the process. After step 655, the server also exits from the process. If the result is NO in step 653, then the server 8 sends a disable message to the client in step 656, and ends the process. In response to a reception of the disable message in step 657, then the client displays a message to this effect in step 658, and then ends the process.

After operation of block 650, the client 2 waits, in step 663, for a report from the server on whether the credit card for the transmitted card number (CCNOu) is valid or not, while the server 8 refers to the credit company associated with the card number in step 661 to see if the credit card is valid. if not, the server 8 informs the client 2 of the invalidity of the credit card in step 662, and ends the process. If the card is valid in step 661, the server 8 informs the client of the validity in step 667. If the client 2 receives a report from the server in step 663, the client makes another check in step 664 to see if the report indicates the validity of the card. If not, the client display a message to indicate the invalidity in step 665, and ends the process. If the report indicates the validity in step 664, which means the completion of step 667, then the client 2 and the server 8 proceed to the next block 670.

In step 670, the client 2 and the server 8 execute timed play and metered charge report routine. FIGs. 18A and 18B are flow charts jointly showing a procedure formed of routines 675ac and 675bc interactively executed for playing an application while timing the duration and displaying a charge and a total amount of charges after the play. In FIG. 18, the routines 675ac and 675bc are identical to the routine 675a and 675b in FIGs. 11A and 11B except that "time" has been replaced with "charge", and accordingly VM-METER and AM-METER have been replaced with VC-METER and AC-METER.

The operation, in the client 2, of playing an application on usage-sensitive charging is completed by block 675 of FIG. 15 or step 218a of FIG. 18A. After step 212a, the server 8 charges the play to the credit card number CCNOu obtained in step 655 of FIG. 17B in step 680. This completes the whole of the charged application play process of FIG. 15.

In this process, only information on charge is given to the user. It is very easy to provide information on both time and charge by adding steps 91 through 93 and 95 to the routines 640b and 640a, and by adding steps 210 and 218 to the routines 675bc and 675ac.

As described above, expected time and/or charge are (is) displayed before playing a user specified application. This is helpful for the user to decide whether to play the application. Additionally, charging is done based on the actually timed play duration. This makes the charging reasonable.

In the above description, the arrangement is such that the user has to input his or her credit card number CCNOu each time he or she wants to play an application. However, instead of doing this, the credit card number CCNOu may be stored in non-volatile memory or EEPROM 103 in a PW_u-encrypted form. In this case, CCNOu is obtained by decrypting PW_u-encrypted CCNOu (e.g., $e(PW_u, CCNOu)$) with a password entered by the user. That is, $d(\text{entered password}, e(PW_u, CCNOu)) = CCNOu$.

Permit the Play Within a Preset Limit

FIG. 19 is a flow chart showing a procedure interactively executed by the client 2 and the server 8 in the operation block 800 of FIG. 5, wherein blocks connected with two flow lines indicates that operation of the blocks is done by the two elements 2 and 8. In this case, it is assumed that a preset limit is recorded in or on the application package and is transmitted from client 2 to server each time of play. On entering the process 800 via step 516 of FIG. 5, the client 2 proceeds to step 801, where the client 2 and the server 8 executes the initial routines 80. It is noted that in routine 80b, if there is a record for VID_v and NO_{v-i}, then the limit value (LV_{v-i}) field of the table 60 of FIG. 6A contains the limit value transmitted from the client 2, otherwise, the received limit value is stored in the LV_{v-i} field when the record for VID_v and NO_{v-i} is added in step 88.

In step 810, the server 8 makes a check if a meter associated with the TOU code received from the client 2 is under the limit value. This check is made by comparing an LV field and LV-meter field associated with the TOU code in table 60. If the value of the LV-meter is equal to or greater than the LV field value, then the server returns an over limit massage to the client 2 in step 820. If not, the server 8 returns an underlimit message to the client 2 in step 822, and proceeds to step 828. If the client 2 receives the overlimit message in step 824, then the client 2 displays a message to this effect. If not, the client 2 proceeds to the step 828.

Since the expected play time informing routines 97a and 97b and the application play subroutine 600 has been described above, the description of steps 828 and 830 are omitted.

According to this feature of the invention, it is possible to limit the use of charged information. This feature is especially useful in case when a user who have paid in advance for the use of the application package is permitted to use the application package within a limit value.

Though it has been assumed that the limit values are included in the application package, the limit values may be kept in the servers of the provider or distributer from the beginning. In this case, the limit values are fixed. However, if limit values are permitted to be set and recorded in the application package at the time of distribution or sales, the limit values are advantageously set according to an amount paid.

As is apparent from the foregoing, as a limit value, any use-limiting factors will do that can be measured in quantity. Such limit values are, for example, the effective date and time, the allowable expiration date and time, the maximum amount of play time, the allowable access count.

It is also possible to combine this feature with a charged application play feature. That is, an arrangement may be such that the user is permitted to use an application package on usage-sensitive charging only if the value of an LV-meter associated with the TOU is under the value of the corresponding LV or the value recorded in a field 33 or 34 of the distribution descriptor 23.

Modification I

In the above embodiment, applications, if more than one, in one volume are encrypted by an identical application encrypting key K_v. However, the applications APa in one volume may be encrypted with respective AP-encrypting keys K_a, where a lower case "a" following AP and K is a serial number assigned to each application ID. In this case, each of the AP-encrypting keys K_a are encrypted with the user public key PK_u, and stored in the PK_u-encrypted AP-encrypting key (K_a) fields 32a in the distribution descriptor 23.

Modification II

It has been assumed that the user of the DVD 3 is limited to the purchaser thereof who have had the PK_u-encrypted AP-encrypting key (K_v) recorded on the DVD 3. However, the system may be so arranged that predetermined people, e.g., family members FM_1, FM_2,..., FM_N of the purchaser can use the DVD (N is the number of the family members). One of the ways to realize this is to encrypt the AP-encrypting key K_v with a public key PK_{u-n} of each member FMn (n = 1, 2,...,N) to obtain $e1(PK_{u-1}, K_v)$, $e1(PK_{u-2}, K_v)$,..., $e1(PK_{u-n}, K_v)$ and to record them in the PK_{u-n}-encrypted AP-encrypting key $e1(PK_{u-n}, K_v)$ fields 32 of the distribution descriptor 23 at the time of purchase of the DVD.

Modification III: K_v Retrieval From Server

In the above description, the AP-encrypting key K_v has been recorded in a PK_u-encrypted form on the DVD 3. However, the AP-encrypting key K_v may be managed by the server 8 and transmitted to the client or the DVD player 2 in response to a request issued from the DVD player 2 each time of use of the DVD 3. In this case, there is no need of providing the distribution descriptor 23 with the PK_u-encrypted AP-encrypting key field 32. Instead each of the servers has to store an AP-encrypting key table (or K_v table) and a PK_u table (shown in FIGs. 20A and 20B) in the hard disc. As shown in FIG. 20A, the K_v table a volume ID (VID_v) field (as the entry of record) and an AP-encrypting key (K_v) field in

each record. In FIG. 20B, each record of the PK_u table comprises a volume ID (VID_v) field (as the entry of record), a volume issue number (NO_{v-i}) field and a PK_u field (Successive same values in the first field are shown by showing only the first appearing one). Further, the process (or step) 610 of obtaining the AP-encrypting key K_v, that is, a group of the steps 602, 604 and 606 in the application play routine 600, has to be replaced with a process of FIG. 20C.

FIG. 20C is a flow chart of a process in which the client DVD player 2 obtains the application encrypting key K_v from the server 8. In step 616, the server 8 retrieves a key K_v from the K_v table by using VID_v. In the next step 618, the key K_v is encrypted with an arbitrary number used only in the current process, e.g., a pseudo random number R to obtain $e2(R, K_v)$. In the next step 620, the server 8 retrieves a key PK_u from the PK_u table by reading the PK_u field of the record which contains VID_v and NO_{v-i} in the VID_v and NO_{v-i} fields, respectively. In the next step 622, $R + e2(R, K_v)$ is encrypted with the retrieved key PK_u to obtain a double encrypted AP-encrypting key

$$e1(PK_u, R + e2(R, K_v)),$$

which is returned to the client with a client network address $CADD_c$ in the next step 624.

On the other hand, the controller 100 of the client 2 waits for a response from the server 8 of SID_s in step 626. If there is any response from the server 8 of SID_s in step 626, then the client DVD 3 receives the data $e1(PK_u, R + e2(R, K_v))$ from the server 8 in step 628. In the next step 630, the received data is decrypted with the user secret key SK_u read from the IC card 5. Specifically, the following calculation is done.

$$d1(SK_u, e1(PK_u, R + e2(R, K_v))) \Longrightarrow R + e2(R, K_v)$$

In the next step 632, $e2(R, K_v)$ is decrypted with the obtained pseudo random number R. Specifically, the following calculation is done.

$$d2(R, e2(R, K_v)) \Longrightarrow K_v$$

Thereafter, the controller 100 proceeds to the step 608 of FIG. 14.

In this modification, the applications APa in one volume may be encrypted with respective AP-encrypting keys K_a. In this case, the K_v table has to be replaced with K_a table in which each record comprises an application ID (AID_a) field and an AP-encrypting key (K_a) field. Further in step 612, the controller 100 of the DVD player 2 has to also send the application ID of the application to be played to the server.

Also in this modification, the system may be, again, so arranged that predetermined people, e.g., family members FM_1, FM_2,..., FM_N of the purchaser can use the DVD (N is the number of the family members). In this case, for each member FMn (n = 1, 2,...,N), the server 8 has to use the member's own public key PK_{u-n} in encrypting the AP-encrypting key K_v. One way to realize this is to issue a volume issue number NO_{v-n} to each member FMn at the time of sales of the DVD, provide the non-volatile memory (not shown) of the DVD player 2 with a table for associating the user's password PWn with the volume issue number NO_{v-n}, send the volume issue number (NO_{v-n}) associated with the user's password in step 612, and use not the PK_u table but a PK_{u-n} table in which each of the records has the following fields:

$$VID_v, NO_{v-n}, PK_{u-n}.$$

Another way is to issue and record not only a volume issue number NO_{v-i} but also family member numbers FMNn for all members at the time of sales of the DVD, provide the non-volatile memory (not shown) of the DVD player 2 with a table for associating the user's password PWn with the corresponding family member number FMNn, send the volume issue number (NO_{v-i}) and the family member number FMNn associated with the user's password in step 612, and use another PK_{u-n} table in which each of the records has the following fields:

$$VID_v, NO_{v-i}, FMNn, PK_{u-n}.$$

In the process of FIG. 20C, the server 8 may be authenticated by means of a public-key cryptosystem using a pair of server secret and public keys (SK_s, PK_s). In this case, the server 8 signs the double-encrypted AP-encrypting key

$$e1(PK_u, R + e2(R, K_v))$$

with a signing key or the server secret key SK_s after step 622. While the client or DVD player 2 tests the signature by the server 8 with a test key or the server public key PK_s contained in the PK_s field 31 of the distribution descriptor 23 recorded in the burst cutting area of the DVD 2 before step 630.

However, even if just described authentication of the server 8 is omitted, an attacker will never go to any greater length than a steal of TOU code plus limit value, a volume ID VID_v, a volume issue number NO_{v-i}, and the client network address $CADD_c$. This is not a serious problem.

In the process of FIG. 20C, a pseudo random number R has been used as a pseudo variable which takes a different value each time of execution of the process. However, as the pseudo variable, any thing will do if the result of encryption with it takes a different value each time of execution of the process.

Modification IV

In the first illustrative embodiment, the decryption of application is achieved by software. For this purpose, the controller 100 has to read the user secret key SK_u from the IC card 5 through the bus 102, which leaves the possibility of permitting a breaker to easily steal the user secret key SK_u through the bus 102. In order to prevent this, the process

achieved by the steps 604 through 608 may be realized by hardware as shown in FIG. 21, which is a block diagram of an exemplary decipherer-built-in IC card IF. In FIG. 21, the decipherer-built-in IC card IF 120a comprises an IC card receptacle 121 and a printed wiring board 122 extending from and fixed with the receptacle 121. An IC 123 is mounted on the printed wiring board 122. The IC 123 comprises a memory IF 125 which usually connects the memory of the IC card 5 with the bus 102 and, in response to an instruction from the controller 100, reads and passes the key SK_u to the next stage; a K_v decoder 126 for receiving the key Sk_u and encrypting $e1(PK_u, K_v)$ with the key SK_u to yield K_v; and an AP decoder 127 for receiving the key K_v and encrypting $e(K_v, AP)$ to yield application data (AP). The printed wiring board 122 portion may be preferably molded together with the IC card receptacle 121 portion so as to make the whole a single body. By doing this, leaking of the user secret key SK_u can be prevented.

This modification can be also applied to a system 1 using the cryptosystem of FIG. 20C. In this case, the K_v decoder 126 of FIG. 21 has to be replaced with a K_v decoder 126a as shown in FIG. 22. In FIG. 22, the K_v decoder 126a decrypts the input data, $e1(PK_u, R + e2(R, K_v))$, from the bus 102 by using the user secret key SK_u passed by the memory IF 125 to obtain $R + e2(R, K_v)$, while decrypting the obtained data $e2(R, K_v)$ with the obtained random number R and outputting the key K_v.

Embodiment II

FIG. 24 is a block diagram showing an arrangement of a system capable of playing a distributed application package, e.g., a DVD on the terms of use of the DVD without communicating with any server according to a second illustrative embodiment of the invention. In FIG. 24, the system 1a is identical to the client 2 of FIG. 1 except that the communication IF 150 has been eliminated because of no need of communication with a server and the controller 100 has been replaced with a controller 100a. In the controller 100a, a not-shown ROM for storing a control program as described later and the EEPROM 103 have been also replaced with a new ROM (not shown) and an EEPROM 103a. In order to play a role of the server 8, the system 1a has to have table 60 of FIG. 6A in any non-volatile memory, e.g., the EEPROM 103a and an application duration (play time) for each application as defined in table 70 of FIG. 6B has to be included in the control data of each application package.

FIG. 25 schematically shows an exemplary control program executed by the controller 100a shown in FIG. 24. The control program of FIG. 25 is also identical to that of FIG. 5 except that the decision step 516 and the step 700 has been eliminated because the limit-attached play mode is not supported by the system 1a in this embodiment, and the steps 650 and 800 are replaced with steps 650a and 800a. Accordingly, operation after step 514 will be described in the following.

If the lower digit of the terms-of-use (TOU) code is 0 in the decision step 514, then in step 650a the controller 100a plays, in the free play mode, the application stored in the selected application in step 506 or 512 and ends the operation. It should be noted that since the system 1a does not have the charged play mode, the lower digit of the TOU code is defined as follows.

Higher digit of terms-of-use code (Hexadecimal)	Corresponding limit value	Play mode
0	None	Free play mode
2	Effective date and time	Limit-attached play mode
3	Allowable expiration date and time	
4	Maximum amount of used period	
5	Allowable access count	
:	:	
:	:	

Accordingly, if the lower digit of the TOU code is not 0 in the decision step 514, then in step 800a the controller 100a plays, in the limit-attached play mode, the application stored in the selected application in step 506 or 512 and ends the operation.

FIGs. 26 and 27 show an operation of a free play mode shown in step 650a of FIG. 25 in a detailed form and a further detailed form, respectively. In FIG. 26, the controller 100a executes an initial routine 80a in step 660a, in step 670a executes an expected play time informing routine, and in step 680a executes an application play and metered play time report routine.

As shown in FIG. 27, in the initial routine 80c, the controller 100a searches the table 60 for a record which contains VID_v and NO_{v-i} in the volume ID and issue No. fields thereof, respectively in step 86. If the search is unsuccessful, then the controller 100a adds the record for VID_v and NO_{v-i} and fills relevant fields with AID_{v-i-a} and a limit value, if any, in the table 60 in step 88, and proceeds to step 90. Also, if the search in step 86 is successful, the server 9 proceeds to step 90, where the controller 100a selects a routine to execute next according to the value of the TOU code and enters the selected routine. In this case, if the TOU code = x0H (x: an arbitrary HEX number, the letter H in the last position indicates that the preceding number is in hexadecimal), then a routine for playing an application free of charge is selected. If the TOU code ≥ x1H, then a routine is selected which plays an application only if the software meter of a use-limiting factor is under a preset value.

The expected play time informing routine 670a is identical to the routines 97 (FIG. 10) minus communication steps 93 and 94, comprising the above described steps 91, 92 and 95. Similarly, it is seen from FIGs. 11 and 13A that the above described steps 620, 622, 624, 210 and 218 are executed in this order in the timed play and metered usage report routine 680a. In this way, the system 1a permits the user to play the application stored in the selected application (steps 506 and 512 of FIG. 25) free of charge.

FIG. 28 is a flow chart showing an operation of a limit-attached play mode shown in step 800a of FIG. 25. Since this operation is very similar to that of FIG. 19, only the flow is briefly described, omitting the details of each step. In FIG. 28, controller 100a first makes a check if a meter associated with the TOU code has reached the limit value obtained with the TOU code. If so, then the server returns an overlimit massage to controller 100a in step 820. Otherwise, the controller 100a proceeds to the expected play time informing routine 828a (= 670a), where the controller 100a executes the above described steps 91, 92 and 95, and then calls the application play subroutine 600 in step 830, thereby completing the operation. Since the application play subroutine 600 has been detailed above, further description is omitted. In this way, the system 1a permits the user to play the application stored in the selected application (steps 506 and 512 of FIG. 25) only if the limit value associated with the TOU code assigned to the volume or the user-specified application has not been reached.

According to the second embodiment, the system 1a can operate in either of the free play mode and the limit-attached play mode without the need of communication with a server. For this, the system 1a may be made portable.

Modifications

In the above description, the illustrative embodiment has been described in conjunction with the DVD. The same discussion can be applied to such package media as permit write once or more.

Further, the present invention is also applicable to application packages distributed via transmission media. In this case, the distributed application packages are stored in a bulk storage in the user's device. An application package comprises one or more application and application control data, that is, an application descriptor and distribution descriptor. One volume is stored as a file. Since a plurality of application package may be stored in a single storage, each application package does not have to contain a control program. One control program, which may be distributed via either package or transmission media, is enough for one user device. The folder or directory in which the application packages are stored is set for a user specified one in the control program when the control program is installed. The data to be recorded in the distribution descriptor is included in the application package by the provider according to the information given by the user.

As described above, one who is permitted to use an application package is limited to an owner of the IC card which stores a user secret key SK_u corresponding to the user public key PK_u used for encryption of the AP-encrypting key K_v in the application package. For this, even if someone has unjustly obtained an application package, for example, by copying the whole volume from the DVD on which the volume is recorded, he or she can not use it without the IC card of the owner of the DVD. Thus the inventive system can prevent unjust use of an application package (DVD in this case) by any other person than the regular owner of the application package.

Also, the inventive system is so arranged that most part of the application package is recorded by pressing in manufacturing process of the DVDs, whereas at least a part of the volume control data (i.e., the distribution descriptor) can be determined at the time of, e.g., distribution of each of the DVDs after the manufacturing process. This makes the system flexible because control data can be easily changed without changing the stamper.

In the initial routines 80a and 80b in FIG. 8A and 8B, the data transmitted with the service request may be encrypted in the same manner as in case of the transmission of user's credit card number shown in FIG. 17. However, in case of the initial routines, there are a plurality of data. These data may be encrypted in the following way.

If the data to be encrypted are D1, D2,... then they are first encrypted with a key R as follows:

e2(R, D1), e2(R, D2),...

Then further encryption is made with a server public key PK_s as follows:

e1(PK_s, R + e2(R, D1) + e2(R, D2),....).

In the process of FIG. 17, the user may be authenticated by means of a public-key cryptosystem using a pair of

user secret and public keys (SK_u, PK_u). In this case, the client 2 signs the double-encrypted credit card number

e1(PK_s, R + e2(R, CCNOu))

with a signing key or the user secret key SK_u after step 648. While the server tests the signature by the client 2 with a test key or the user public key PK_u before step 650.

Instead of storing a single server public key in the distribution descriptor 23, a plurality of server public keys or all the server public keys may be recorded. By doing this, it is possible, for example, to setting a different charge depending on the server public key which the user have selected by appropriately combining the tables 70 and 75.

Also, application packages with an identical volume ID can have different server public keys recorded. A plurality of toll center may be advantageously provided for application packages of the same title.

In order to prevent any use of IC card by other person than the owner of the IC card, it is possible to add, before the Sku reading step 604, the steps of prompting the user to enter a password through a human IF 110 and proceeding to step 604 only if the entered password coincides with the user password PW_u stored in the IC card.

Though the IC card 5 is used in the above embodiment, the IC card IF 120 may be replaced with a magnetic card reader to permitting the use of the magnetic card. Alternatively, the arrangement may be such that the user enters his or her password each time the user uses the DVD.

Instead of storing the user secret key SK_u in the IC card 5, the key Sku may be stored in non-volatile memory in a PW_u-encrypted form. In this case, the key SK_u is obtained by decrypting PW_u-encrypted SK_u with a password entered by the user.

The discussion of three preceding paragraphs are applied to the IC card used for storing the server secret key in the server. However, in this case the user has to be taken as the administrator of the toll server.

Many widely different embodiments of the present invention may be constructed without departing from the spirit and scope of the present invention. It should be understood that the present invention is not limited to the specific embodiment described in the specification, except as defined in the appended claims.

A system for permitting only an authentic user to play a desired application contained in a distributed application package in one of predetermined operation, e.g., free play mode, charged mode, limit-attached play mode, etc. The system comprises a client for playing an application under the control of a server connected with the client through a communication network. The application package (the volume) includes a distribution descriptor which contains mode codes assigned to the volume and the applications of the volume. The data of distribution descriptor is decided and stored in the descriptor at the time of distribution of the volume. This feature makes the system flexible. There is also disclosed a system operatable without communicating with a server.

Claims

1. An application package for use in a system for playing an application contained in the application package (the volume), the application package comprising:

   application data for at least one application; and
   volume control data for use in controlling said system, wherein said volume control data at least comprises:
   a volume ID for identifying the kind of said application package (said volume);
   an issue number assigned in order of issue to each of the volumes of said kind; and
   application IDs each assigned to one of said at least one application contained in said volume, and wherein:
   at least a part of said volume control data is to be added to said volume after the creation of said volume; and
   said at least a part of said volume control data includes said issue number.

2. An application package as defined in claim 1, wherein:

   said application data has been encrypted with an encrypting key; and
   said at least a part of said volume control data includes a user's public key-encrypted version of said encrypting key used.

3. An application package as defined in claim 1, wherein said at least a part of said volume control data includes mode codes which are assigned to said volume or said at least one application and each indicate a play mode associated with one of said volume or said at least one application to which the mode code is assigned.

4. A package media on which an application package as defined in claim 1 has been recorded.

5. A package media of a write-once type on which an application package as defined in claim 1 has been recorded.

6. A package media on which an application package as defined in claim 1 has been recorded wherein said at least a part of said volume control data is recorded in an area different from data area where said application data is recorded on the package media.

7. A method for sending data with a raised security from a first device to a second device through a public telecommunication network, comprising the steps of:

in said second device,

generating a pseudo random number;
transmitting said pseudo random number to said first device;

in said first device,

encrypting said data with said transmitted pseudo random number into encrypted data;
encrypting concatenated data consisting of said pseudo random number and said encrypted data with a public key of said second device into double-encrypted data;
sending said double-encrypted data to said second device; in said second device,
decrypting said double-encrypted data with a secret key of said second device which corresponds to said public key into decrypted data consisting of a decrypted random number portion and another decrypted portion; and
decrypting said another decrypted portion with said transmitted random number to obtain said data.

8. A method for sending a plurality of pieces of data with a raised security from a first device to a second device through a public telecommunication network, comprising the steps of:

in said second device,

generating a pseudo random number;
transmitting said pseudo random number to said first device;

in said first device,

encrypting each of said pieces of data with said transmitted pseudo random number into an encrypted piece of data;
encrypting concatenated data consisting of said pseudo random number and said encrypted pieces of data with a public key of said second device into double-encrypted data;
sending said double-encrypted data to said second device; in said second device,
decrypting said double-encrypted data with a secret key of said second device which corresponds to said public key into decrypted data consisting of a decrypted random number portion and said plurality of decrypted data portions; and
decrypting each of said decrypted portions with said transmitted random number to obtain said pieces of data.

9. A method as defined in claim 7 or 8, further comprising the steps, executed after said step of decrypting said double-encrypted data, of:

proceeding to a next step only if said decrypted random number portion coincides with said transmitted pseudo random number; and
said second device informing said first device of a failure in decryption if said decrypted random number portion does not coincide with said transmitted pseudo random number.

10. In a system provided with means for playing an application contained in an application package, a method for permitting a user to play an encrypting key-encrypted application contained in a distributed application package which further contains, as volume control data, a user's public key-encrypted encrypting key so encrypted as to be able to be decrypted with a secret key of the user into said encrypting key, the method comprising the steps of:

reading said user's public key-encrypted encrypting key from said distributed application package (said vol-

ume);
obtaining said secret key;
decrypting said user's public key-encrypted encrypting key with said secret key to obtain said encrypting key; and
decrypting said encrypting key-encrypted application with said obtained encrypting key into application data while passing said application data to said means for playing an application.

11. In a system comprising a client provided with means for playing an application contained in an application package and a server connected with the client through a communication network, a method for permitting a user to play one of encrypting key-encrypted applications contained in a distributed application package which further contains, as volume control data, a volume ID for identifying the kind of said distributed application package (said volume), an issue number issued to each volume of the kind in an issued order and application IDs, the method comprising the steps of:

said client reading said volume ID, said issue number and an application ID for said one of encrypting key-encrypted applications (said encrypting key-encrypted application) from said volume and sending to said server;

in said server,

retrieving said encrypting key by using said volume ID;
retrieving a public key of said user by using said volume ID and said issue number;
generating a pseudo random number;
double-encrypting said encrypting key with said pseudo random number and said public key into a double encrypted data;
sending said double-encrypted data to said client; in said client,
obtaining a secret key of said user which corresponds to said public key;
obtaining said encrypting key by decrypting said double-encrypted data with said secret key;
decrypting said encrypting key-encrypted application with said obtained encrypting key into application data while passing said application data to said means for playing an application.

12. A method as defined in claim 10 or 11, wherein said means for obtaining a secret key comprises means for reading said secret key from a portable memory of said user.

13. A method as defined in claim 12, wherein said portable memory is an IC card.

14. In a system comprising a client provided with means for playing an application package and a server connected with the client through a communication network for controlling the client, the application package (the volume) containing, as volume control data, a volume ID and an issue number issued to each of the volumes of said volume ID in an issued order, a method for controlling the amount of play time comprising the steps of:

said client sending said volume ID and said issue number to said server;
said server retrieving an expected play time associated with said volume ID and said issue number; and
said server adding said expected play time to the value of a total play time associated with said volume ID and said issue number.

15. In a system comprising a client provided with means for playing an application contained in an application package and a server connected with the client through a communication network for controlling the client, the application package (the volume) containing, as volume control data, a volume ID, an issue number issued to each of the volumes of said volume ID in an issued order and an application ID for the application, a method for controlling the amount of play time comprising the steps of:

said client sending said volume ID, said issue number and said application ID to said server;
said server retrieving an expected play time associated with said volume ID, said issue number and said application ID; and
said server adding said expected play time to the value of a total play time associated with said volume ID and said issue number.

16. In a system comprising a client provided with means for playing an application contained in an application package and a server connected with the client through a communication network for controlling the client, the application package (the volume) containing, as volume control data, a volume ID and an issue number issued to each of the volumes of said volume ID in an issued order, a method for controlling the amount of play time comprising the steps of:

said client and said server interactively measuring, as a measured play time, a play time of said application; and
said server adding said measured play time to the value of a total play time associated with said volume ID and said issue number.

17. A method as defined in claim 16, wherein said step of measuring a play time comprises the step of using a timer of said server.

18. A method as defined in claim 16, wherein said step of measuring a play time comprises the step of using a timer of said client.

19. In a system comprising a client for playing an application package and a server connected with the client through a communication network wherein the application package (the volume) comprises application data and control data and at least a part of the control data has been added to the volume after the creation of said volume, a method for sending desired data from one side of said client and said server to the other side, the method comprising the steps of:

including a secret key of said other side in said at lest a part of said control data;

in said other side,

generating a pseudo random number;
transmitting said pseudo random number to said one side;

in said one side,

encrypting said desired data with said transmitted pseudo random number into encrypted data;
encrypting concatenated data consisting of said pseudo random number and said encrypted data with said public key of said other side into double-encrypted data;
sending said double-encrypted data to said other side;

in said other side,

decrypting said double-encrypted data with a secret key of said other side which corresponds to said public key into decrypted data consisting of a decrypted random number portion and another decrypted portion; and
decrypting said another decrypted portion with said transmitted random number to obtain said desired data.

20. A method as defined in claim 19, wherein said generating a pseudo random number includes storing said pseudo random number in memory, and wherein the method further comprises the step, executed prior to said decrypting said another decrypted portion, of:

in response to a determination that said decrypted random number portion does not coincide with said pseudo random number stored in said means for storing said pseudo random number stored in said memory, informing said one side of a failure in decryption instead of passing the control to next means.

21. In a system comprising a client provided with means for playing an application contained in an application package and a server connected with the client through a communication network, a method for permitting a user to play an application contained in a distributed application package which further contains, as volume control data, a volume ID for identifying the kind of said distributed application package (said volume), an issue number issued to each volume of the kind in an issued order, and an application ID for said application, the method comprising the steps of:

proceeding to a next step only if the value of a meter field associated with said volume ID, said issue number and said application ID is under the value of a limit value field associated with said volume ID, said issue number and said application ID in a volume data table; and

displaying a message informing an overlimit on a display device of said client and quit the operation otherwise.

22. In a system comprising a client provided with means for playing an application contained in an application package and a server connected with the client through a communication network, a method for permitting a user to play an application contained in a distributed application package which further contains, as volume control data, a volume ID for identifying the kind of said distributed application package (said volume), an issue number issued to each volume of the kind in an issued order, an application ID for said application and a limit value for limiting the play of said application, the method comprising the steps of:

proceeding to a next step only if the value of a meter field associated with said volume ID, said issue number and said application ID in a volume data table is under said limit value; and

displaying a message informing an overlimit on a display device of said client and quit the operation otherwise.

23. A method as defined in claim 21, wherein said limit value is one of effective date and time, allowable expiration date and time, a maximum amount of play time, and an allowable access count.

24. A method as defined in any of claims 11, 15 and 16, wherein said step of said client sending to said server comprises the steps of:

said client encrypting at least one of said volume ID, said issue number and said application ID into encrypted data; and

said server decrypting said encrypted data.

25. A system for sending data with a raised security from a first device to a second device through a public telecommunication network, comprising:

means provided in said second device for generating a pseudo random number;
means provided in said second device for transmitting said pseudo random number to said first device;
means provided in said first device for encrypting said data with said transmitted pseudo random number into an encrypted data;
means provided in said first device for encrypting concatenated data consisting of said pseudo random number and said encrypted data with a public key of said second device into double-encrypted data;
means provided in said first device for sending said double-encrypted data to said second device;
means provided in said second device for decrypting said double-encrypted data with a secret key of said second device which corresponds to said public key into decrypted data consisting of a decrypted random number portion and another decrypted portion; and
means provided in said second device for decrypting said another decrypted portion with said transmitted random number to obtain said data.

26. A system for sending a plurality of pieces of data with a raised security from a first device to a second device through a public telecommunication network, comprising:

means provided in said second device for generating a pseudo random number;
means provided in said second device for transmitting said pseudo random number to said first device;
means provided in said first device for encrypting each of said pieces of data with said transmitted pseudo random number into an encrypted piece of data;
means provided in said first device for encrypting concatenated data consisting of said pseudo random number and said encrypted pieces of data with a public key of said second device into double-encrypted data;
means provided in said first device for sending said double-encrypted data to said second device;
means provided in said second device for decrypting said double-encrypted data with a secret key of said second device which corresponds to said public key into decrypted data consisting of a decrypted random number portion and said plurality of decrypted data portions; and
means provided in said second device for decrypting each of said decrypted portions with said transmitted random number to obtain said pieces of data.

27. A system as defined in claim 25 or 26, further comprising:

means, provided in said second device, activated prior to decrypting each of said decrypted portions and responsive to a determination that said decrypted random number portion does not coincide with said transmitted pseudo random number, for informing said first device of a failure in decryption instead of passing the control to next means.

28. A system for playing an encrypting key-encrypted application contained in a distributed application package which further contains, as volume control data, a user's public key-encrypted encrypting key so encrypted as to be able to be decrypted with a secret key of the user into said encrypting key, the system comprising:

means for reading said user's public key-encrypted encrypting key from said distributed application package (said volume);
means for obtaining said secret key;
means for decrypting said user's public key-encrypted encrypting key with said secret key to obtain said encrypting key;
means for decrypting said encrypting key-encrypted application with said obtained encrypting key to provide application data; and
means for using said application data for playing.

29. A system for permitting a user to play an encrypting key-encrypted application contained in a distributed application package which further contains, as volume control data, a volume ID for identifying the kind of said distributed application package (said volume), an issue number issued to each volume of the kind in an issued order and application IDs, the system comprising:

a client for playing an application by using application data; and
a server for controlling said client through a communication network, wherein said client comprises:
means for reading and sending said volume ID, said issue number and an application ID for said one of encrypting key-encrypted applications (said encrypting key-encrypted application) from said volume to said server, said server comprises:

means for retrieving said encrypting key by using said volume ID;
means for retrieving a public key of said user by using said volume ID and said issue number;
means for generating a pseudo random number;
means for double-encrypting said encrypting key with said pseudo random number and said public key into a double encrypted data; and
means for sending said double-encrypted data to said client, and said client comprises:
means for obtaining a secret key of said user which corresponds to said public key;
means for obtaining said encrypting key by decrypting said double-encrypted data with said secret key;
means for decrypting said encrypting key-encrypted application with said obtained encrypting key to provide application data; and
means for using said application data for playing.

30. A system as defined in claim 28 or 29, wherein said means for obtaining a secret key comprises means for reading said secret key from a portable memory of said user.

31. A system as defined in claim 30, wherein said portable memory is an IC card.

32. A system for permitting a user to play a distributed application package which further contains, as volume control data, a volume ID for identifying the kind of said distributed application package (said volume) and an issue number issued to each volume of the kind in an issued order, the system comprising:

a client for playing said distributed application package; and
a server for controlling said client through a communication network, wherein:
said client comprises means for sending said volume ID and said issue number to said server; and
said server comprises means for retrieving an expected play time associated with said volume ID and said issue number, and means for adding said expected play time to the value of a total play time associated with said volume ID and said issue number.

33. A system for permitting a user to play an application contained in a distributed application package which further contains, as volume control data, a volume ID for identifying the kind of said distributed application package (said volume), an issue number issued to each volume of the kind in an issued order and an application ID for the application, the system comprising:

a client for playing said application; and
a server for controlling said client through a communication network, wherein:
said client comprises means for sending said volume ID, said issue number and said application ID to said server; and
said server comprises means for retrieving an expected play time associated with said volume ID, said issue number and said application ID, and means for adding said expected play time to the value of a total play time associated with said volume ID and said issue number.

34. A system for permitting a user to play an application contained in a distributed application package which further contains, as volume control data, a volume ID for identifying the kind of said distributed application package (said volume), an issue number issued to each volume of the kind in an issued order and an application ID for the application, the system comprising:

a client for playing said application; and
a server for controlling said client through a communication network, wherein:
said client and said server comprise means for interactively measuring, as a measured play time, a play time of said application; and
said server further comprises means for adding said measured play time to the value of a total play time associated with said volume ID and said issue number.

35. A system as defined in claim 34, wherein said means for interactively measuring a play time comprises means for using a timer of said server.

36. A system as defined in claim 34, wherein said means for interactively measuring a play time comprises means for using a timer of said client.

37. A system for permitting a user to play an application package (the volume) comprising application data and control data wherein at least a part of the control data has been added to the volume after the creation of said volume, the system comprising:

a client for playing said volume; and
a server for controlling said client through a communication network, wherein said server comprises means for storing a secret key of said server and said at least a part of said control data includes a public key corresponding to said secret key, and wherein the system comprises:
means provided in said server for generating a pseudo random number;
means for storing said pseudo random number;
means provided in said server for transmitting said pseudo random number to said client;
means provided in said client for encrypting desired data with said transmitted pseudo random number into encrypted data;
means provided in said client for encrypting concatenated data consisting of said pseudo random number and said encrypted data with said public key into double-encrypted data;
means provided in said client for sending said double-encrypted data to said server;
means provided in said server for decrypting said double-encrypted data with said secret key into decrypted data consisting of a decrypted random number portion and another decrypted portion; and
means provided in said server for decrypting said another decrypted portion with said transmitted random number to obtain said desired data.

38. A system as defined in claim 37, further comprising:

means, provided in said server, activated prior to said decrypting said another decrypted portion and responsive to a determination that said decrypted random number portion does not coincide with said pseudo random number stored in said means for storing said pseudo random number, for informing said client of a failure in decryption instead of passing the control to next means.

39. A system for permitting a user to play an application contained in a distributed application package which further contains, as volume control data, a volume ID for identifying the kind of said distributed application package (said volume), an issue number issued to each volume of the kind in an issued order and application IDs, the system comprising:

a client for playing an application by using application data; and
a server for controlling said client through a communication network, wherein said client comprises:
means for reading and sending said volume ID, said issue number and an application ID for said one of encrypting key-encrypted applications (said encrypting key-encrypted application) from said volume to said server, said server comprises:
means for proceeding to next step only if the value of a meter field associated with said volume ID, said issue number and said application ID is under the value of a limit value field associated with said volume ID, said issue number and said application ID in a volume data table; and
means for causing said client to display a message informing an overlimit on a display device of said client and quit the operation otherwise.

40. A system for permitting a user to play an application contained in a distributed application package which further contains, as volume control data, a volume ID for identifying the kind of said distributed application package (said volume), an issue number issued to each volume of the kind in an issued order, application IDs and limit values associated with respective application IDs for limiting the play of respective applications, the system comprising:

a client for playing an application by using application data; and
a server for controlling said client through a communication network, wherein said client comprises:
means for reading and sending said volume ID, said issue number, an application ID for said one of encrypting key-encrypted applications (said encrypting key-encrypted application) and a limit value associated with said application ID from said volume to said server, and wherein said server comprises:
means for proceeding to a next step only if the value of a meter field associated with said volume ID, said issue number and said application ID in a volume data table is under said limit value; and
means for causing said client to display a message informing an overlimit on a display device of said client and quit the operation otherwise.

41. A system as defined in claim 39, wherein said limit value is one of effective date and time, allowable expiration date and time, a maximum amount of play time, and an allowable access count.

42. A system as defined in any of claims 29, 33 and 34, wherein said means for sending to said server comprises means for encrypting at least one of said volume ID, said issue number and said application ID.

43. A method for permitting an authentic user to play a desired one of the applications contained in a distributed application package in a system capable of playing an application, wherein said application package (said volume) contains volume control data including mode codes assigned to said volume and the applications of said volume, the method comprising the steps of:

deciding to use one of predetermined play modes specified by one of said mode codes associated with said desired application; and
playing said desired application in said specified play mode.

44. A method as defined in claim 43, wherein the method further comprises the step of including, in said mode codes, values indicative of a free play mode and at least one limit-attached play mode which correspond(s) to respective limit value(s) used for limiting usage.

45. A method as defined in claim 44, wherein said step of playing said desired application comprises the step of:

in response to a determination that said one of said mode codes associated with said desired application includes a value indicative of said free play mode, simply playing said desired application.

46. A method as defined in claim 44, wherein said step of playing said desired application comprises the step of:

in response to a determination that said one of said mode codes associated with the desired application

includes one of values indicative of said at least one limit-attached play mode, displaying a message to the effect that a limit value associated with said one of values has been reached instead of playing said desired application if said limit value has been reached.

47. A method as defined in claim 43, wherein said volume control data further includes a volume ID, an issue number and an application ID for each of said applications, and wherein said step of deciding to use one of predetermined play modes comprises the steps of:

obtaining said one of said mode codes associated with said desired application and corresponding limit value by using said application ID; and
comparing said one of said mode codes with a meter value associated with said volume ID, said issue number and said application ID.

48. A method as defined in claim 45, wherein each of said applications has been each encrypted with an encrypting key and said volume control data includes a user's public key-encrypted version of said encrypting key (a public key-encrypted version encrypting key), and wherein said step of simply playing said desired application comprises the steps of:

reading said user's public key-encrypted encrypting key from said volume;
obtaining a user's secret key which corresponds to said user's public key;
decrypting said user's public key-encrypted encrypting key with said user's secret key to obtain said encrypting key; and
decrypting said desired application with said obtained encrypting key.

49. A system for permitting an authentic user to play a desired one of the applications contained in a distributed application package, wherein said application package (said volume) contains volume control data including mode codes assigned to said volume and the applications of said volume, the system comprising:

means for deciding to use one of predetermined play modes specified by one of said mode codes associated with said desired application; and
means for playing said desired application in said specified play mode.

50. A system as defined in claim 49, wherein the system further comprises means for including, in said mode codes, values indicative of a free play mode and at least one limit-attached play mode which correspond(s) to respective limit value(s) used for limiting usage.

51. A system as defined in claim 50, wherein said means for playing said desired application comprises:

means, responsive to a determination that said one of said mode codes associated with said desired application includes a value indicative of said free play mode, for simply playing said desired application.

52. A system as defined in claim 50, wherein said means for playing said desired application comprises:

means, responsive to a determination that said one of said mode codes associated with the desired application includes one of values indicative of said at least one limit-attached play mode, for displaying a message to the effect that a limit value associated with said one of values has been reached instead of playing said desired application if said limit value has been reached.

53. A system as defined in claim 49, wherein said volume control data further includes a volume ID, an issue number and an application ID for each of said applications, and wherein said means for deciding to use one of predetermined play modes comprises:

means for obtaining said one of said mode codes associated with said desired application and corresponding limit value by using said application ID; and
means for comparing said one of said mode codes with a meter value associated with said volume ID, said issue number and said application ID.

54. A system as defined in claim 51, wherein each of said applications has been encrypted with an encrypting key and

said volume control data includes a user's public key-encrypted version of said encrypting key (a public key-encrypted version encrypting key), and wherein said means for simply playing said desired application comprises:

means for reading said user's public key-encrypted encrypting key from said volume;
means for obtaining a user's secret key which corresponds to said user's public key;
means for decrypting said user's public key-encrypted encrypting key with said user's secret key to obtain said encrypting key; and
means for decrypting said desired application with said obtained encrypting key.

55. A method for permitting an authentic user to play a desired one of the applications contained in a distributed application package in a system comprising a client capable of playing an application and a server connected with said client through a communication network, wherein said application package (hereinafter referred to as "said volume") contains volume control data including mode codes assigned to said volume and the applications of said volume, the method comprising the steps of:

said client deciding to use one of predetermined play modes specified by one of said mode codes associated with said desired application; and
playing said desired application in said specified play mode by means of cooperation between said client and said server.

56. A method as defined in claim 55, wherein the method further comprises the step of including, in each of said mode code, a value indicative of one of a free play mode, a charged play mode and at least one limit-attached play mode, wherein said volume control data further comprises a limit value associated with each of said at least one limit-attached play mode.

57. A method as defined in claim 55 or 56, wherein said volume control data further includes a volume ID, an issue number, and an application ID for each of said applications, and wherein said step of playing said desired application in said specified play mode includes an application play step of simply playing said specified application.

58. A method as defined in claim 57, wherein each of said applications contained in a distributed application package has been encrypted with an encrypting key and said volume control data includes a user's public key-encrypted version of said encrypting key (a public key-encrypted version encrypting key), and wherein said application play step comprising the steps of:

reading said user's public key-encrypted encrypting key from said volume;
obtaining a user's secret key which corresponds to said user's public key;
decrypting said user's public key-encrypted encrypting key with said user's secret key to obtain said encrypting key; and
decrypting said desired application with said obtained encrypting key.

59. A method as defined in claim 57, wherein each of said applications contained in a distributed application package has been encrypted with an encrypting key and said volume control data includes a user's public key-encrypted version of said encrypting key (a public key-encrypted version encrypting key), and wherein said application play step comprises the steps of:

in said server,

retrieving an encrypting key by using said volume ID;
retrieving a user's public key associated with said volume ID and said issue number;
double-encrypting said encrypting key with a pseudo random number and said user's public key into a double encrypted data;
sending said double-encrypted data to said client; in said client,
obtaining a user's secret key which corresponds to said user's public key;
obtaining said encrypting key by decrypting said double-encrypted data with said user's secret key;
decrypting said desired application with said obtained encrypting key.

60. A method as defined in claim 57, wherein said step of playing said desired application further comprises the steps, executed prior to said application play step, of:

said server retrieving an expected play time associated with said desired application; and
displaying said expected play time on a display device of said client.

61. A method as defined in claim 57, wherein said step of playing said desired application further comprises the steps of:

measuring, as a measured play time, a duration of said application play step;
adding said measured play time to a play time meter associated with said mode code to obtain a total amount of play time; and
displaying said measured play time and said total amount of play time on a display device of said client after said application play step.

62. A method as defined in claim 61, wherein said step of measuring a duration comprises the step of measuring said play time by using a timer of said server.

63. A method as defined in claim 61, wherein said step of measuring a duration comprises the step of measuring said play time using a timer of said client.

64. A method as defined in claim 57, wherein said step of deciding to use one of predetermined play modes comprises deciding to use said charged play mode if said one of said mode codes associated with said desired application includes a value indicative of said charged play mode, and wherein said step of playing said desired application comprises the steps of:

said client obtaining and sending a credit card number of said user to said server;
proceeding to a next step only if the credit card of said number is found to be valid from a reference to an associated credit company;
displaying, on a display device of said client, a charge for play decided based on a measurement of a duration of said application play step and a total amount of play charges after said application play step; and
said server charging said play to said credit card number.

65. A method as defined in claim 64, wherein said step of playing said desired application further comprises the steps, prior to said application play step, of:

displaying, prior to said application play step, an expected charge and an expected total amount of charges on said display device; and
letting the user decide whether to play said desired application.

66. A method as defined in claim 64, wherein said step of said client obtaining and sending a credit card number of said user to said server comprises the steps of:

in said server,

generating a pseudo random number;
storing said pseudo random number in memory;
transmitting said pseudo random number to said client;

in said client,

prompting said user to input said credit card number;
double-encrypting said credit card number first with said transmitted random number and then with a sever's public key included in said volume control data into a double-encrypted number;
sending said double-encrypted number to said server; in said server,
decrypting said double-encrypted number with a server's secret key into a decrypted random number and another decrypted data; and
decrypting said another decrypted data with said transmitted random number to obtain said credit card number.

67. A method as defined in claim 66, wherein said step of said client obtaining and sending a credit card number of said

user to said server further comprises the steps, executed prior to said step of decrypting said another encrypted data, of:

proceeding to a next step only if said decrypted random number coincides with said pseudo random number which has been stored in said memory; and
displaying a message informing a failure in decryption and quitting the operation otherwise.

68. A method as defined in claim 57 wherein said step of deciding to use one of predetermined play modes comprises deciding to use one of said at least one limit-attached play mode if said one of said mode codes associated with said desired application includes a value indicative of said one of said at least one limit-attached play mode, and wherein said step of playing said desired application comprises the step of:

in response to a determination that a meter value associated with said one of said mode codes associated with said desired application in a record identified by said volume ID, said issue number and an application ID of said desired application in a volume data table has reached a limit value associated with said mode code, displaying a message informing an overlimit on a display device of said client instead of executing said application play step.

69. A method as defined in claim 68, wherein said limit value is one of effective date and time, allowable expiration date and time, a maximum amount of play time, and an allowable access count.

70. A system for playing a distributed application package in one of predetermined play modes in concert with a server, wherein the application package contains a data set encrypted with an encrypting key (a K-encrypted data set) for each of at least one application and volume control data for use in controlling operation of the system and the server and the volume control data includes mode codes defining said play modes, the system comprising:

means for permitting a user to select one of said at least one application of said volume;
means for deciding to use one of said predetermined play modes associated with one of said mode codes assigned to said selected application; and
means for playing said selected application in said selected play mode in concert with said server.

71. A system as defined in claim 70, wherein each of said mode codes includes one of values for a free play mode, a charged play mode and at least one limit-attached play mode.

72. A system as defined in claim 70, wherein said volume control data further includes a volume ID, an issue number and an application ID for each of said applications, and wherein said means for playing said selected application in said selected play mode at least comprises:

means for setting said server for said selected play mode by sending to said server said volume ID, said issue number, and the application ID and said mode code associated with said selected application; and
application play means for simply playing said specified application.

73. A system as defined in claim 72, wherein said volume control data further includes a user's public key-encrypted encrypting key, and wherein said application play means comprises:

means for reading said user's public key-encrypted encrypting key from said volume;
means for obtaining a user's secret key which corresponds to said user's public key;
means for decrypting said user's public key-encrypted encrypting key with said user's secret key to obtain said encrypting key; and
means for decrypting the K-encrypted data set of said selected application with said obtained encrypting key.

74. A system as defined in claim 73, wherein means for decrypting said user's public key-encrypted encrypting key and said means for decrypting the K-encrypted data set are realized as an integrated circuit.

75. A system as defined in claim 72, wherein said application play means comprises:

means for receiving double-encrypted data from said server;
means for obtaining a user's secret key which corresponds to said user's public key;

means for obtaining said encrypting key by decrypting said double-encrypted data with said user's secret key; and

means for decrypting the K-encrypted data set of said selected application with said obtained encrypting key.

76. A system as defined in claim 75, wherein means for obtaining said encrypting key and said means for decrypting the K-encrypted data set are realized as an integrated circuit.

77. A system as defined in claim 74 or 76, wherein said integrated circuit is incorporated into said means for obtaining a user's secret key.

78. A system as defined in claim 73, wherein said means for deciding to use one comprises means for deciding to use a free play mode and wherein said means for playing said selected application further comprises: means, prior to said application play means, of:

means for receiving data from said server; and
displaying said data as an expected play time for said selected application.

79. A system as defined in claim 73, wherein said means for deciding to use one of said predetermined play modes comprises means for deciding to use a free play mode, and wherein said means for playing said selected application further comprises:

means for causing said server to obtain, as a measured play time, data of a operation period of said application play means;
means for receiving first and second data from said server; and
means for displaying, just after the completion of operation by said application play means, said first and second data as said measured play time and a total amount of play time. data as said measured play time and a total amount of play time.

80. A system as defined in claim 79, wherein said means for causing said server to obtain data of said operation period comprises means for informing said server of the start and the end of operation by said application play means to utilize a timer of said server.

81. A system as defined in claim 79, wherein said means for causing said server to obtain data of a operation period comprises:

means for measuring said operation period of said application play means; and
means for sending said operation period to said server for use in a calculation of said total amount of play time.

82. A system as defined in claim 72, wherein said means for deciding to use one comprises means for deciding to use a charged play mode and wherein said means for playing said selected application further comprises:

means for obtaining and sending a credit card number of said user to said server;
means responsive to a verification result of said credit card from said server for starting a next process only if said result is positive; and
means for displaying a charge for play decided based on a measured play time of said application play means and a total amount of play charges after operation of said application play means.

83. A system as defined in claim 82, wherein said means for playing said selected application further comprises:

means activated prior to operation of said application play means for displaying an expected charge and an expected total amount of charges and letting the user decide whether to play said selected application.

84. A system as defined in claim 82, wherein said volume control data of said distributed application package further includes a sever's public key, and wherein said means for obtaining and sending a credit card number of said user to said server comprises:

means for prompting said user to input said credit card number;
means for receiving a random number from said server;

means for obtaining said server's public key from said volume;
means for double-encrypting said credit card number first with said random number and then with said server's public key into a double-encrypted data;
sending said double-encrypted number to said server;

85. A system as defined in claim 84, wherein said means for said client obtaining and sending a credit card number of said user to said server further comprises:

means responsive to a positive result of random number check from said server for starting a next process; and
means responsive to a negative result of said random number check from said server for displaying a message indicative of a failure in said random number check and quitting the operation for said selected application.

86. A system as defined in claim 72, wherein:

said means for deciding to use one comprises means for deciding to use a limit-attached play mode; and
said sending to said server includes sending a limit value associated with said mode code, and wherein said means for playing said selected application further comprises:
means operative prior to operation of said application play means for receiving from said server a limit check result indicative of whether a limit value associated with said mode code has been reached; and
means responsive to an over limit case of said result for starting a next operation.

87. A system as defined in claim 86, wherein said limit value is one of effective date and time, allowable expiration date and time, a maximum amount of play time, and an allowable access count.

88. A system for controlling through a communication network a client device to play a distributed application package in one of predetermined play modes, wherein the application package contains a data set encrypted with an encrypting key (a K-encrypted data set) for each of at least one application and volume control data for use in controlling operation of the system and the client and the volume control data includes a volume ID, an issue number, an application ID for each of said applications, and a mode code for said volume or mode codes for said applications, the system comprising:

volume data table for storing, for each volume, said volume ID, said issue number, said mode code for said volume, and said application ID and said mode code for each of said applications;
means for receiving a service request, a volume ID, an issue number, an application ID and a mode code and other data from said client;
means for storing said received application ID, said received mode code and other data in appropriate fields of a record identified by said volume ID and said issue number;
means responsive to a determination that there is no record identified by said volume ID and said issue number in said volume data table for adding said record in said volume data table and storing said received application ID and mode code and said other data in relevant fields of said record; and
means operative on the basis of said received mode code for deciding to subsequently passing the control to means for supporting a play mode associated said received mode code.

89. A system as defined in claim 88, wherein said means for supporting a play mode at least comprises means for supporting application play means, of client, for simply playing an application identified by said received application ID, and wherein said means for supporting said application play means of said client comprises:

first means for associating a given volume ID with a corresponding encrypting key;
second means for associating both a given volume ID and issue number with a corresponding user's public key;
means for retrieving an encrypting key associated with said received volume ID from said first means;
means for retrieving a user's public key associated with said received volume ID and issue number from said second means;
means for double-encrypting said encrypting key with a pseudo random number and said user's public key into a double encrypted data; and
sending said double-encrypted data to said client.

90. A system as defined in claim 89, further comprising an application data table for storing data for each kind of appli-

cation, wherein said received mode code defines a free play mode, and wherein said means for supporting a play mode associated said received mode code comprises:

means, activated prior to an operation of said means for supporting application play means of said client, for retrieving an expected play time associated with said received application ID from said application data table; and
means for sending said expected play time to said client.

91. A system as defined in claim 89, wherein said received mode code defines a free play mode, and wherein said means for supporting a play mode associated said received mode code comprises:

means for measuring, as a measured play time, a duration of application play;
means for adding said measured play time to a play time meter associated with said received mode code in said volume data table to obtain a total amount of play time; and
means for sending said measured play time and said total amount of play time to said client.

92. A system as defined in claim 91, wherein said means for measuring a duration comprises:

means responsive to a notice of the start of operation by said application play means of said client for starting a timer; and
means responsive to a notice of the end of said operation for stopping said timer.

93. A system as defined in claim 91, wherein said means for measuring a duration comprises:

means for receiving a measured duration from said client.

94. A system as defined in claim 88, wherein said received mode code defines a charged play mode, and wherein said means for supporting a play mode associated said received mode code comprises:

means for receiving a credit card number of said user from said server;
means, responsive to a determination, from a verification of said credit card number, that said credit card number is not valid, for informing said client of invalidity and quitting the operation of said means for supporting a play mode;
means, responsive to a determination, from said verification of said credit card number, that said credit card number is valid, for informing said client of a validity and proceeding to a next operation; and
means for charging said play to said credit card number.

95. A system as defined in claim 94, wherein said means for supporting a play mode associated said received mode code further comprises:

means activated prior to operation of said application play means of said client for retrieving an expected charge from said application data table by using said received application ID;
means for calculating a sum of said expected charge and a value of a charge meter associated with said received volume ID or application ID depending on said received mode code;
means operative prior to operation of said application play means for sending said expected charge and said sum to said client; and
means responsive to a receipt of a message of quitting for quitting said means for supporting a play mode.

96. A system as defined in claim 94, wherein said means for receiving a credit card number of said user from said server comprises:

means for generating a pseudo random number;
means for storing said pseudo random number in memory;
means for transmitting said pseudo random number to said client;
means for waiting for a double-encrypted data from said client;
means for obtaining a server's secret key;
means for decrypting said double-encrypted number with said server's secret key into a decrypted random number and another decrypted data; and

means for decrypting said another encrypted data with said transmitted random number to obtain said credit card number.

97. A system as defined in claim 96, wherein said means for obtaining a user's secret key comprises means for reading said user's secret key from a portable memory of said user.

98. A system as defined in claim 96, wherein said means for receiving a credit card number of said user from said server further comprises:

means responsive to a determination, made prior to said decrypting said another, that said decrypted random number coincides with said pseudo random number which has been stored in said memory for sending an enable message to said client and proceeding to a next operation; and
means responsive to a determination, made prior to said decrypting said another, that said decrypted random number does not coincide with said pseudo random number which has been stored in said memory for sending a disable message to said client and quitting said supporting a play mode.

99. A system as defined in claim 88, wherein:

said received mode code defines a limit-attached play mode; and
means for receiving a service request further receives a limit value associated with said mode code, and wherein said means for supporting a play mode associated said received mode code comprises:
means for proceeding to a next operation only if the value of a software meter associated with said mode code in said volume data table is under said limit value; and
means for sending a message informing an over limit to said client and quitting the operation of said means for supporting a play mode associated said received mode code if the value of a software meter associated with said mode code in said volume data table is not under said limit value.

100. A system as defined in claim 99, wherein said limit value is one of effective date and time, allowable expiration date and time, a maximum amount of play time, and an allowable access count.

101. A system as defined in any of claims 54, 73 and 75, wherein said means for obtaining a user's secret key comprises means for reading said user's secret key from a portable memory of said user.

102. A system as defined in claim 28 or 29, wherein said means for obtaining said secret key comprises means for reading said user's secret key from a portable memory of said user.

103. A method as defined in any of claims 10, 11, 19, 21, 22 and 55, wherein said application package is recorded on a package media.

104. A method as defined in claim 103, wherein said package media is of a write-once type, and said client is a system capable of playing said package media of said write-once type.

105. An application package as defined in claim 1, wherein said package media is distributed to a purchaser thereof or a subscriber thereof via a transmission media.

106. A system as defined in any of claims 28, 29, 37, 39, 40, 70 and 88, wherein said application package is recorded on a package media.

107. A system as defined in claim 106, wherein said application package is recorded on a package media of a write-once type.

108. A system as defined in claim 106, wherein at least a part of said volume control data is recorded, after manufacturing said package media, in an area different from a data area where said at least one application is recorded.

109. A system as defined in claim 108, wherein said client is a system provided with means for playing said package media of said write-once type.

110. A system as defined in any of claims 28, 29, 37, 39, 40, 70 and 88, wherein said application package is recorded

on a DVD and at least a part of said volume control data is recorded, after manufacturing said package media, in a BCA (burst cutting area) of the DVD, and wherein said client is a system provided with means for playing said DVD.

111. A method as defined in any of claims 10, 11, 19, 21, 22, 43 and 55, wherein the application package has been distributed to a purchaser thereof or a subscriber via a transmission media and at least a part of said volume control data has been added to said application package after preparing said application package.

112. A system as defined in any of claims 28, 29, 37, 39, 40, 49, 70 and 88, wherein said application package has been distributed to a purchaser thereof or a subscriber thereof via a transmission media and at least a part of said volume control data has been added to said application package after preparing said application package.

FIG. 1

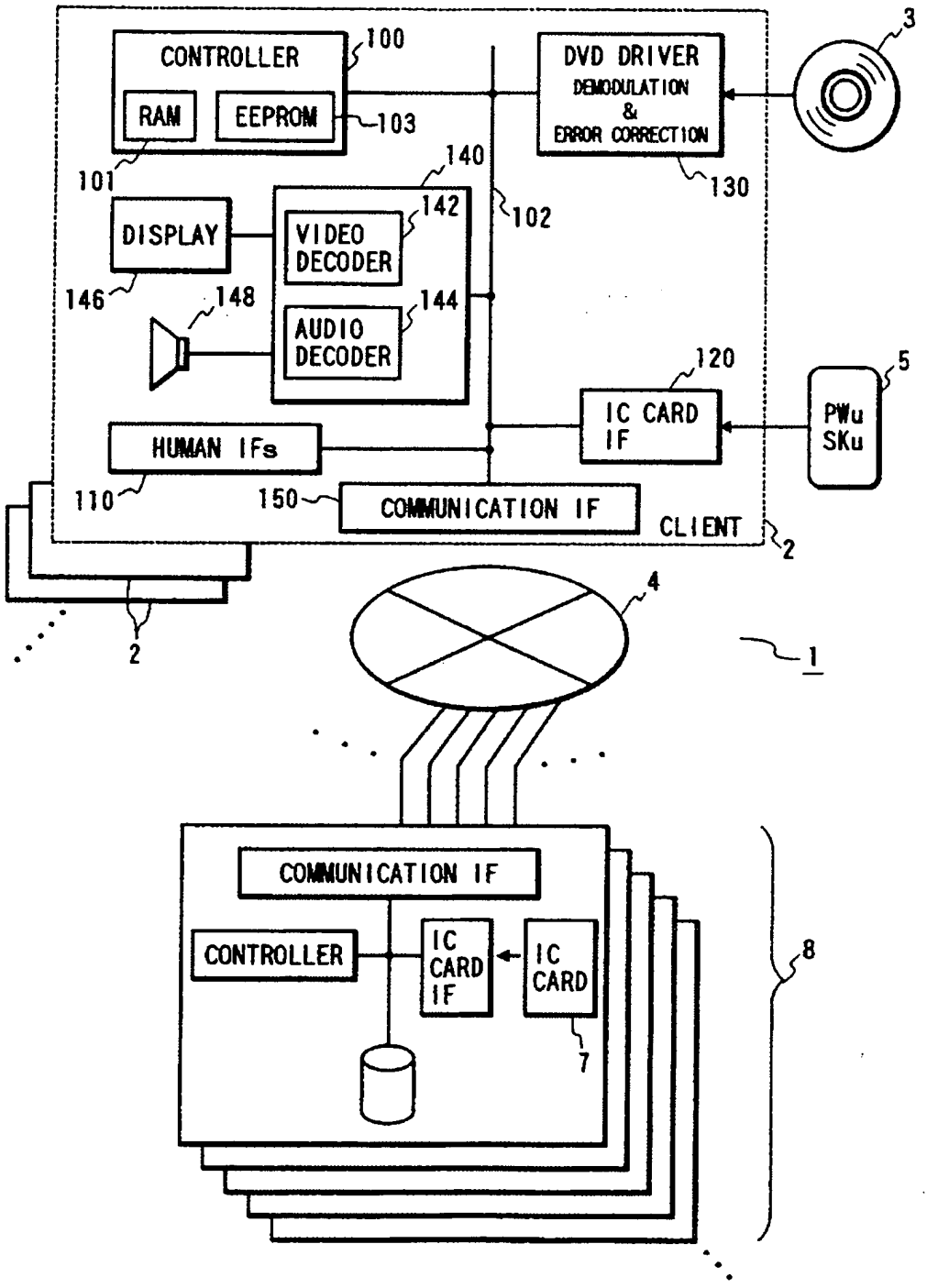


FIG. 2

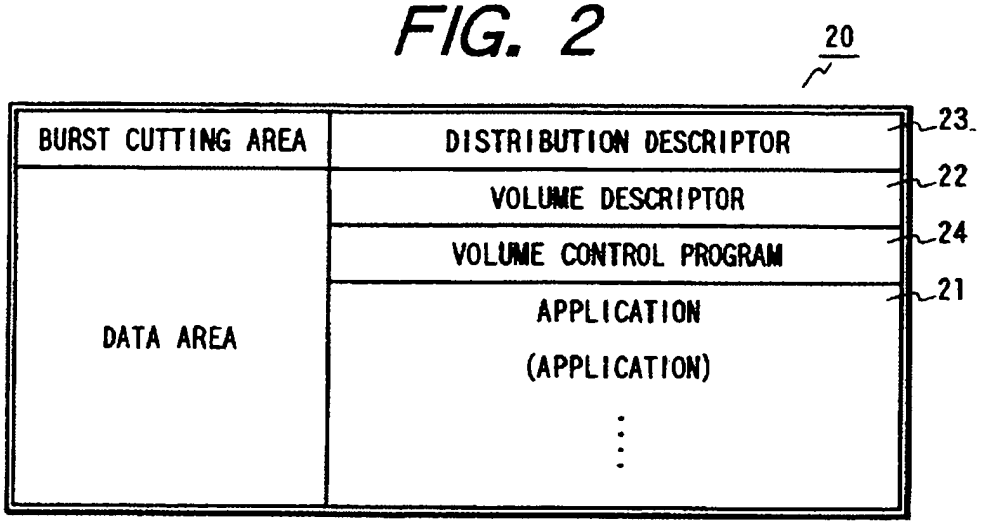


FIG. 3

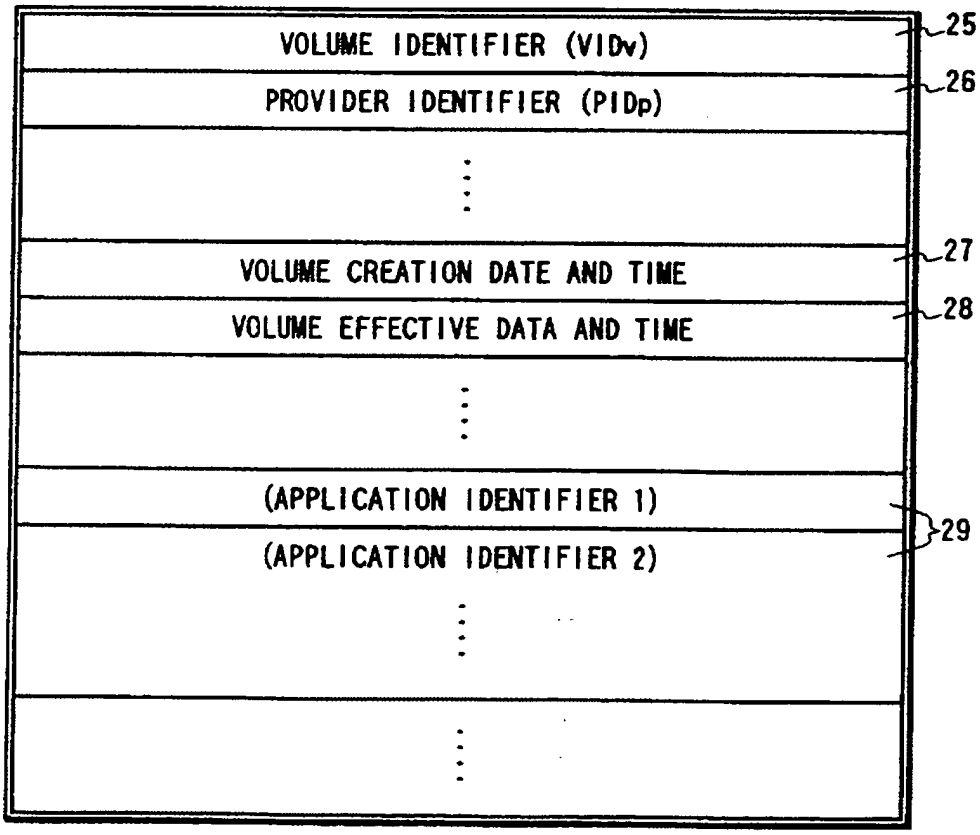


FIG. 4

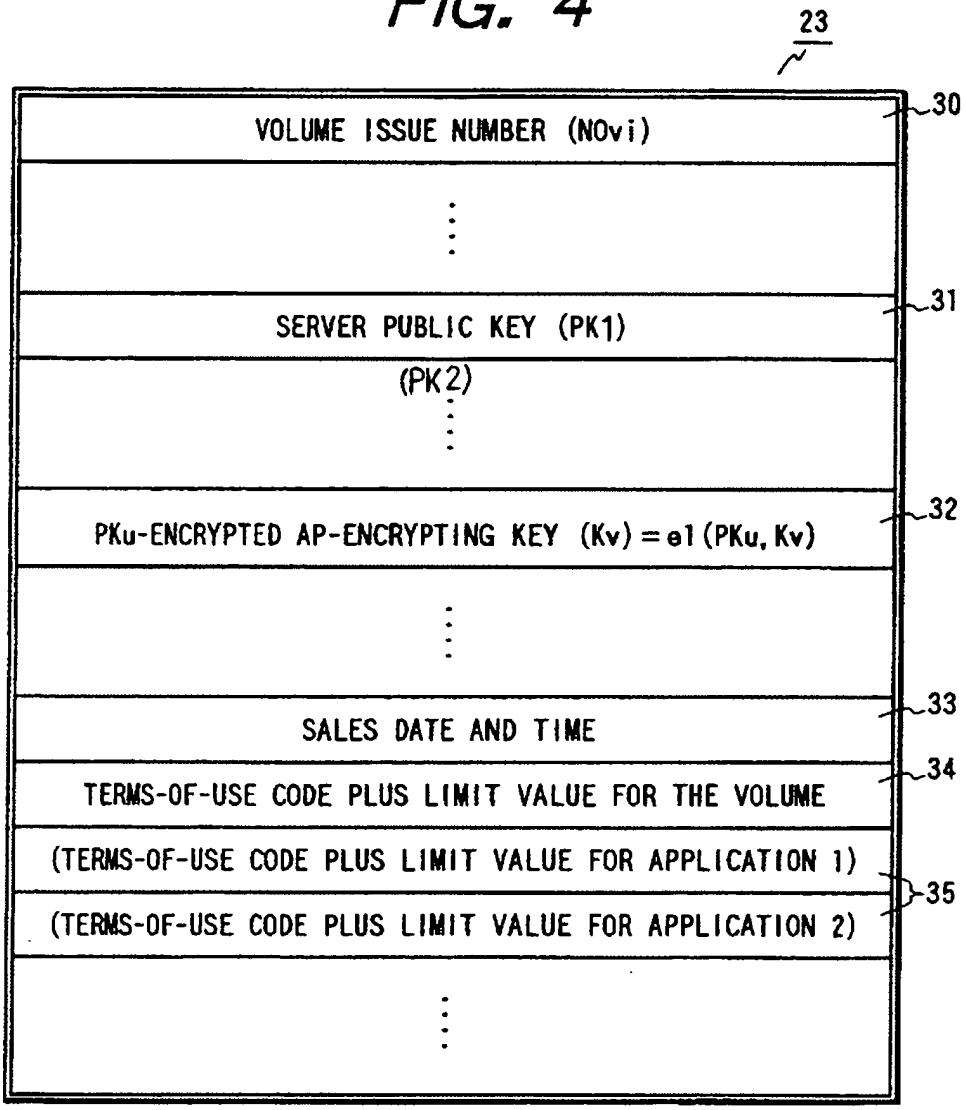


FIG. 5

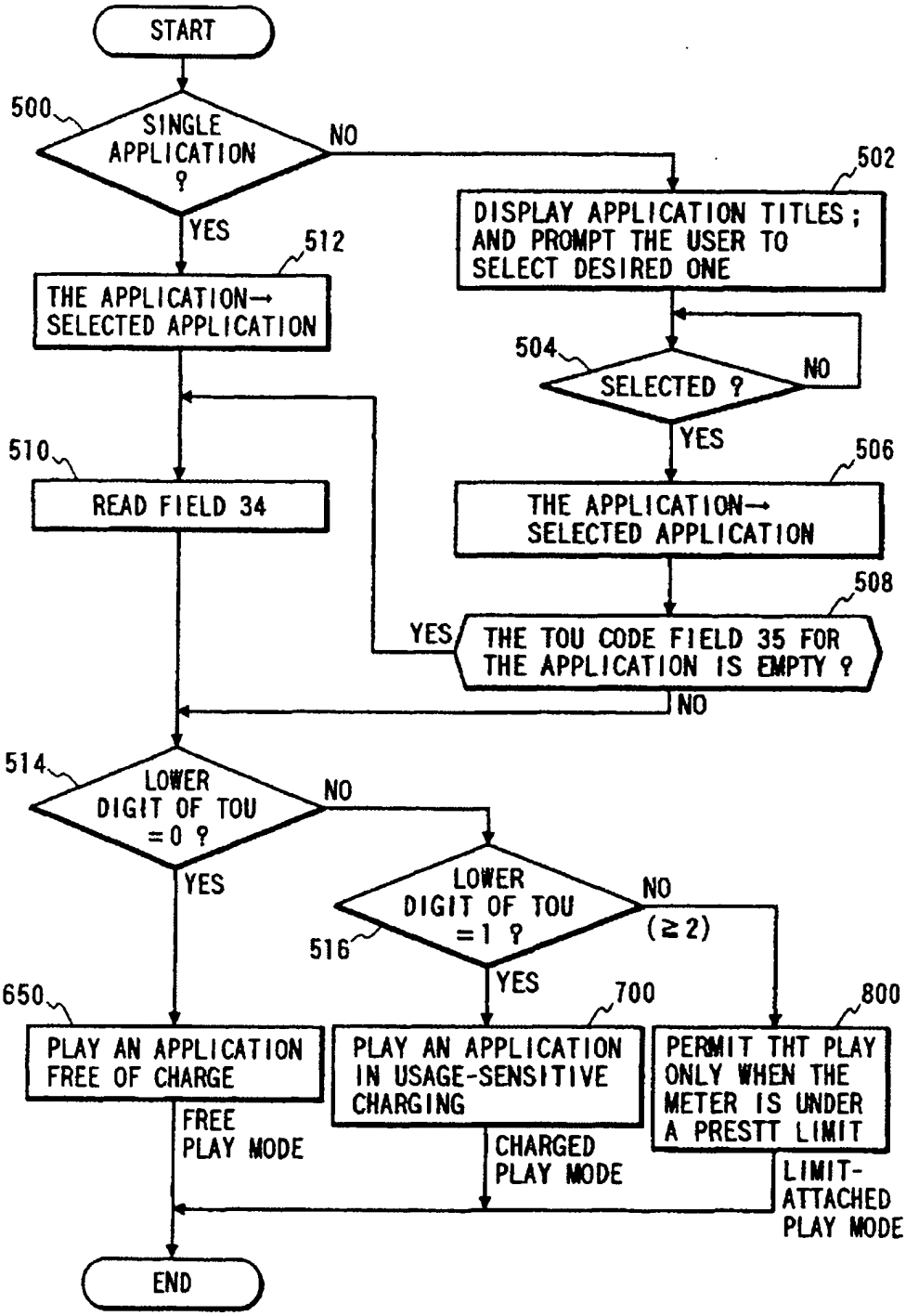


FIG. 6A

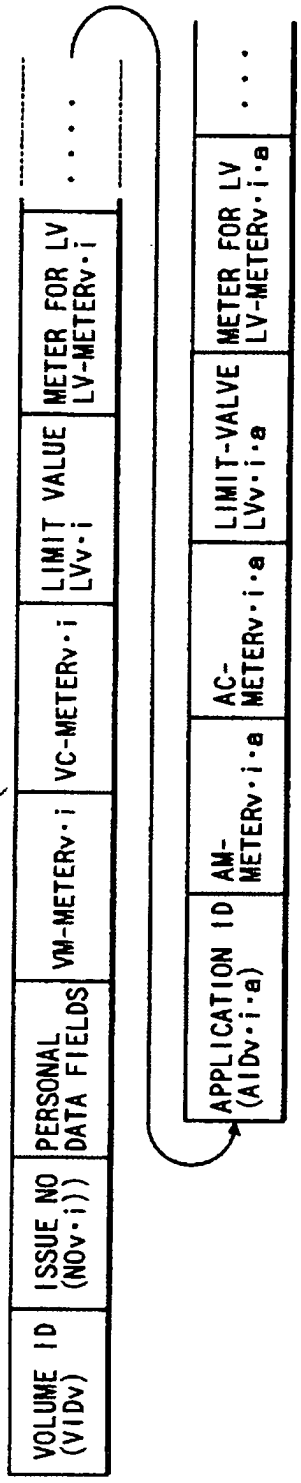


FIG. 6B

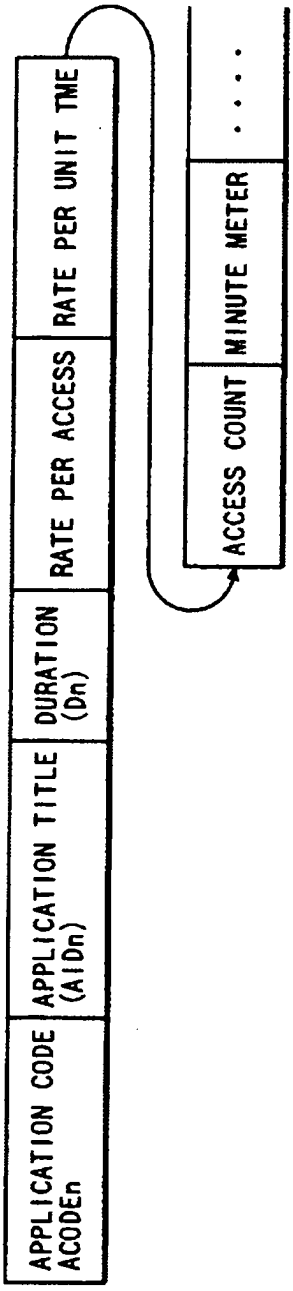


FIG. 7

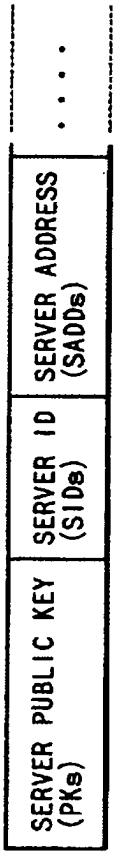


FIG. 8A

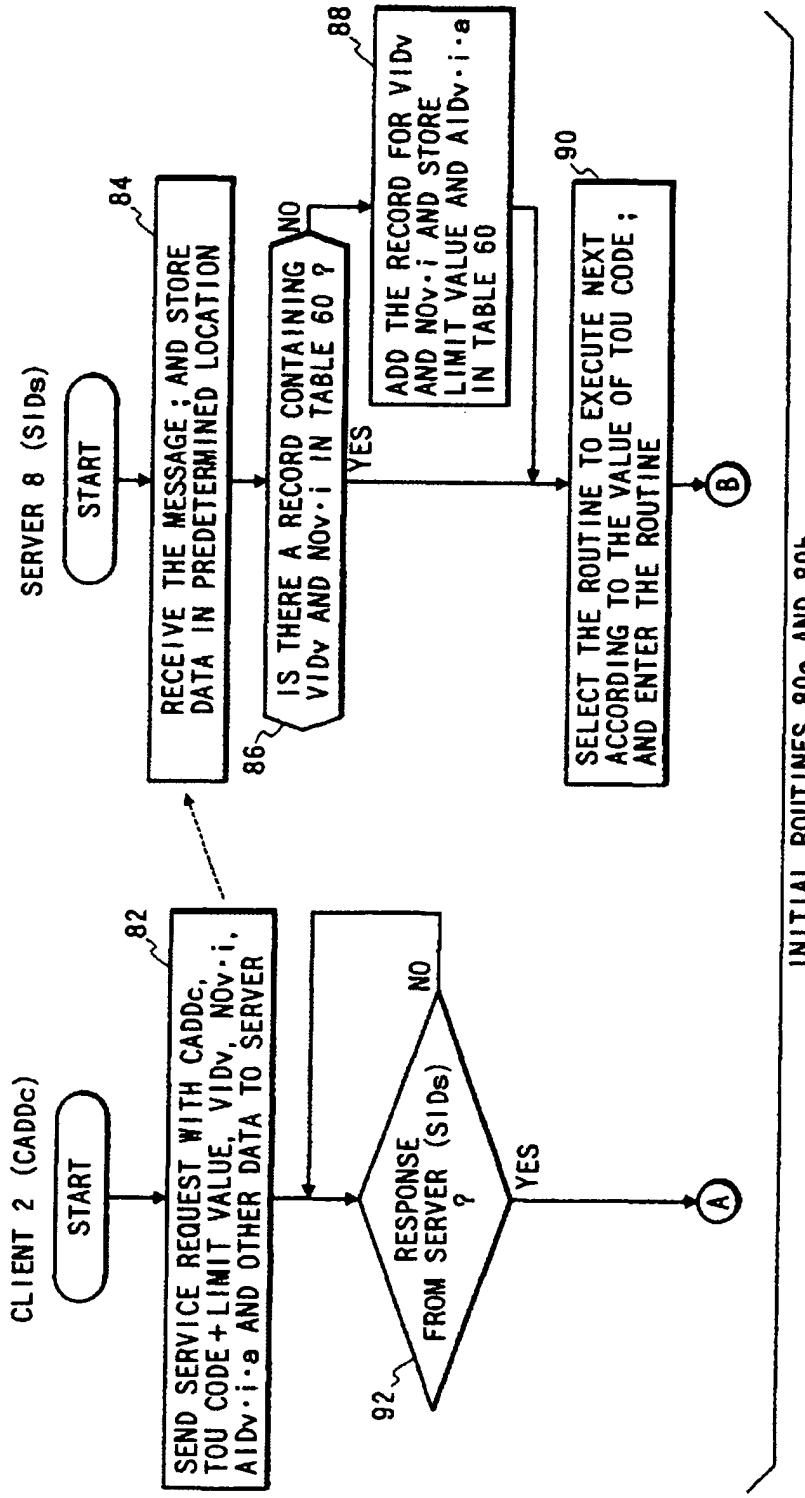


FIG. 8B

FIG. 9

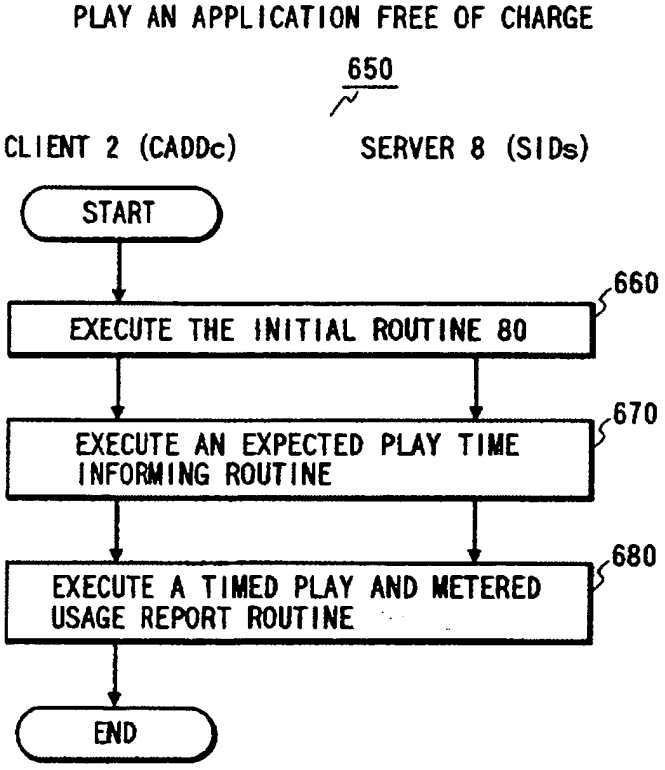


FIG. 10A

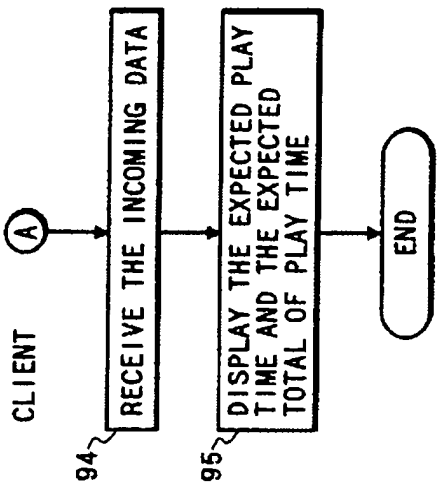


FIG. 10B

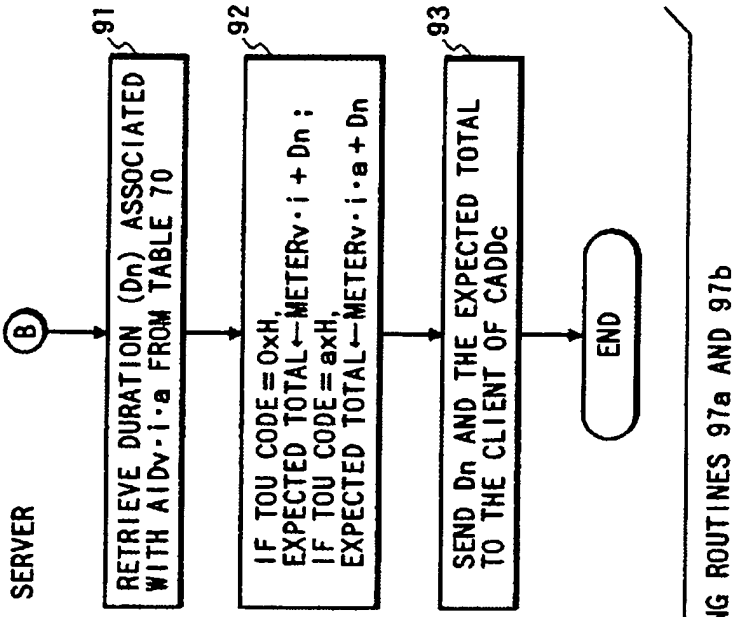


FIG. 11A

TIMED PLAY AND METERED USAGE REPORT ROUTINES 675a AND 675b

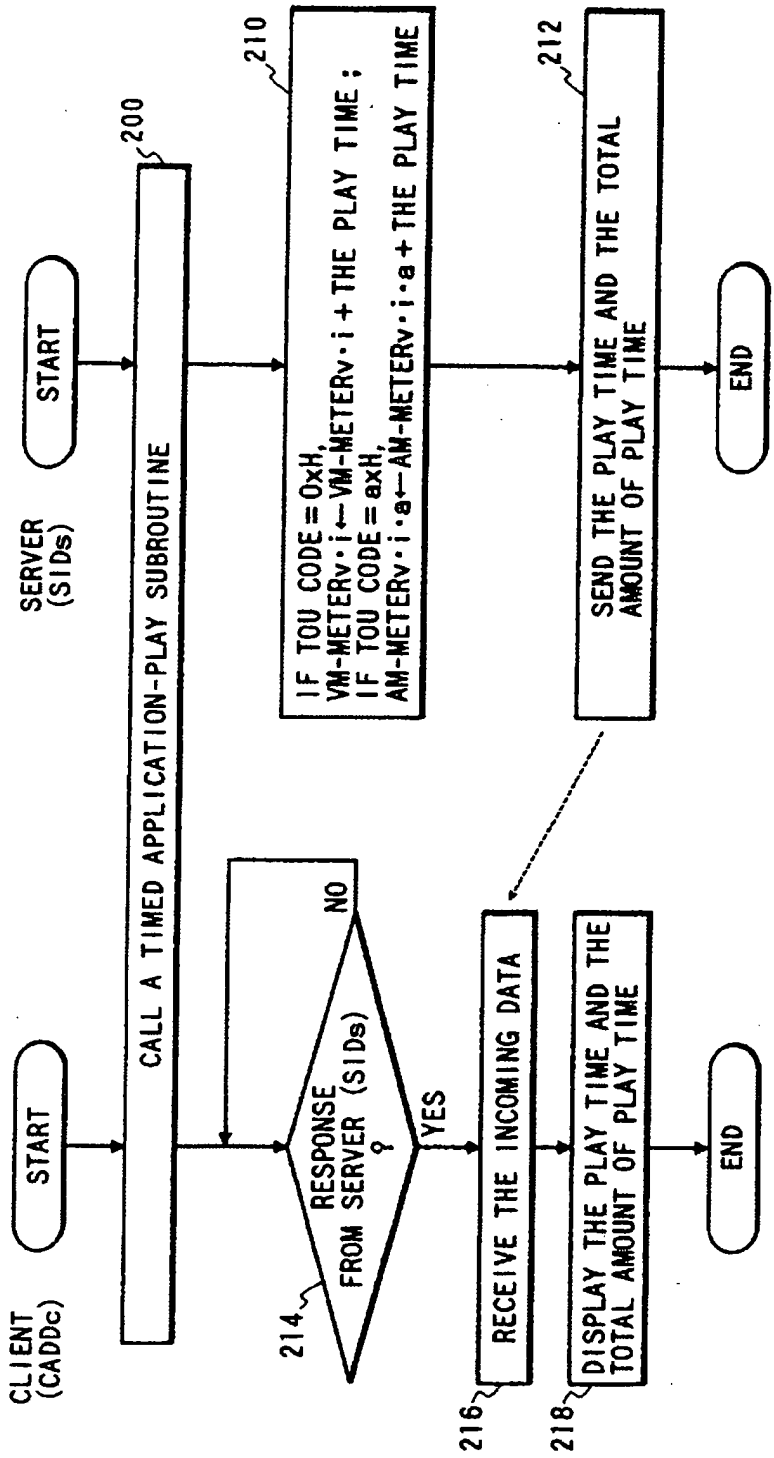


FIG. 11B

FIG. 12A

FIG. 12B

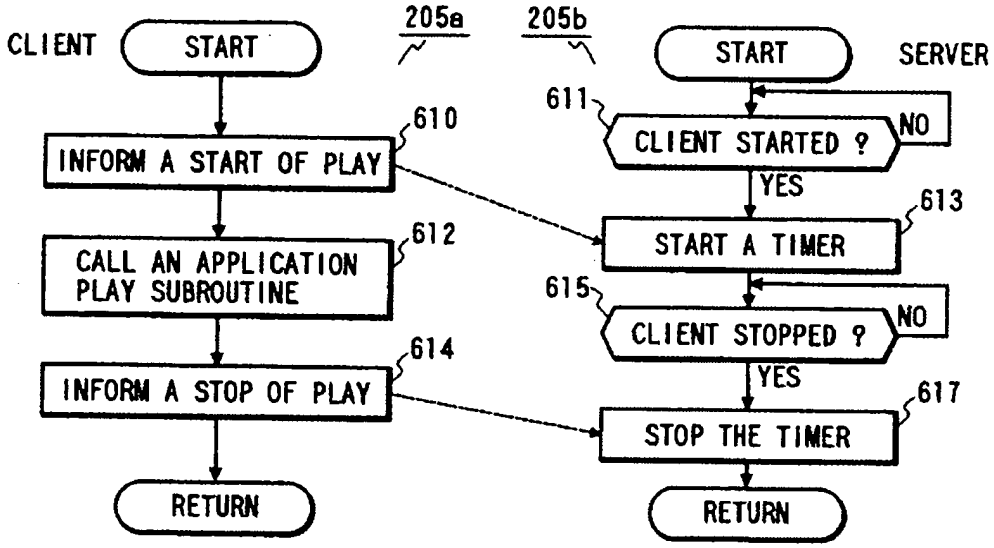


FIG. 13A

FIG. 13B

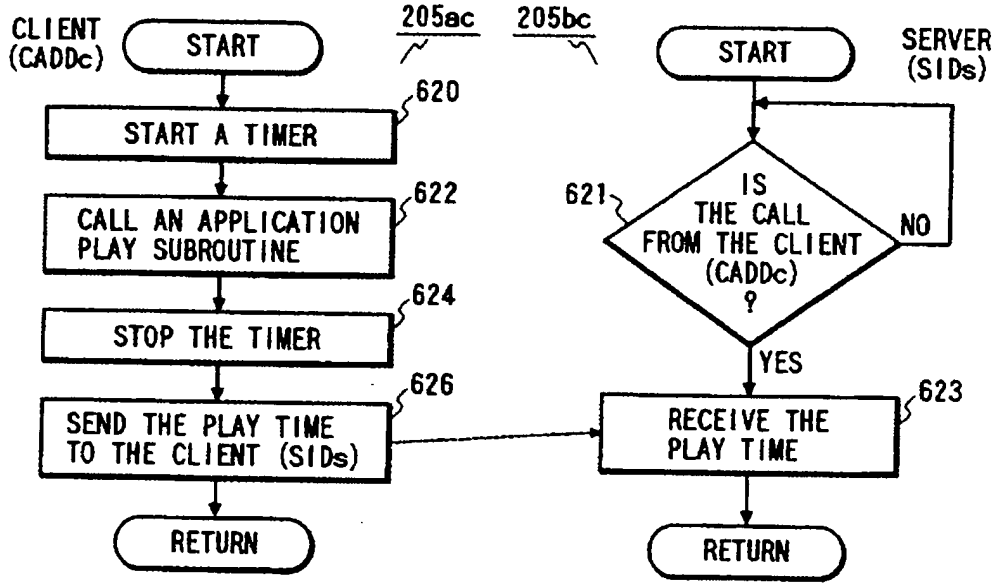


FIG. 14

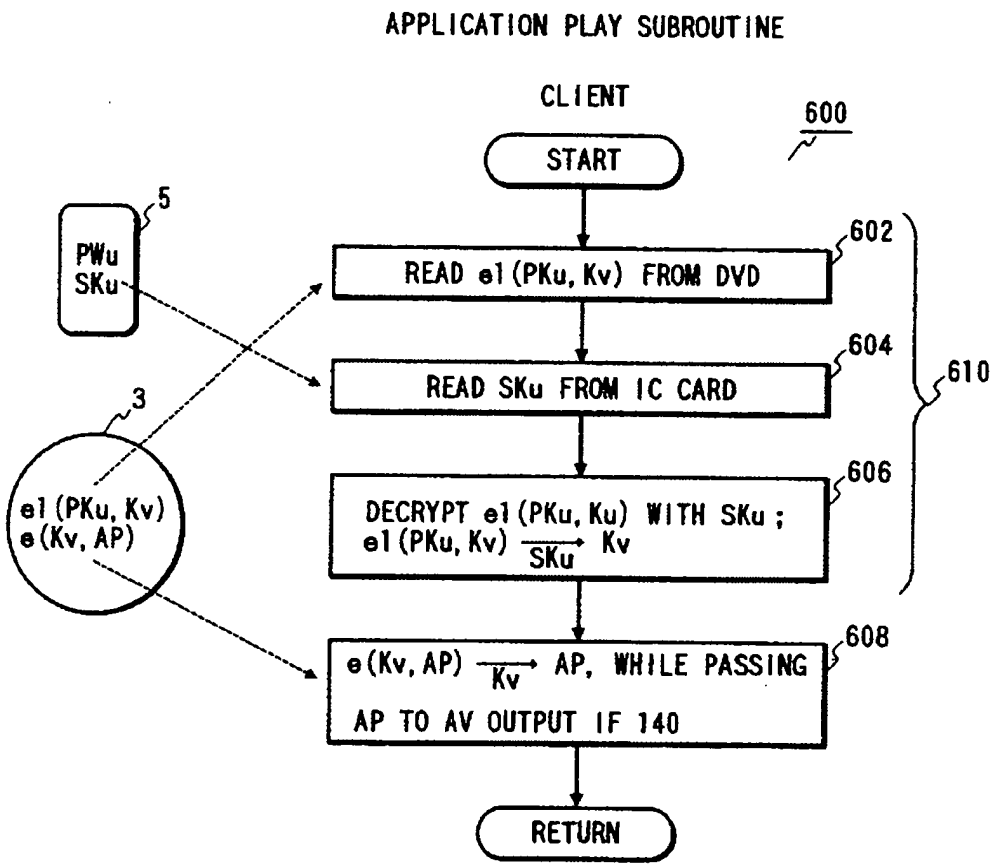


FIG. 15

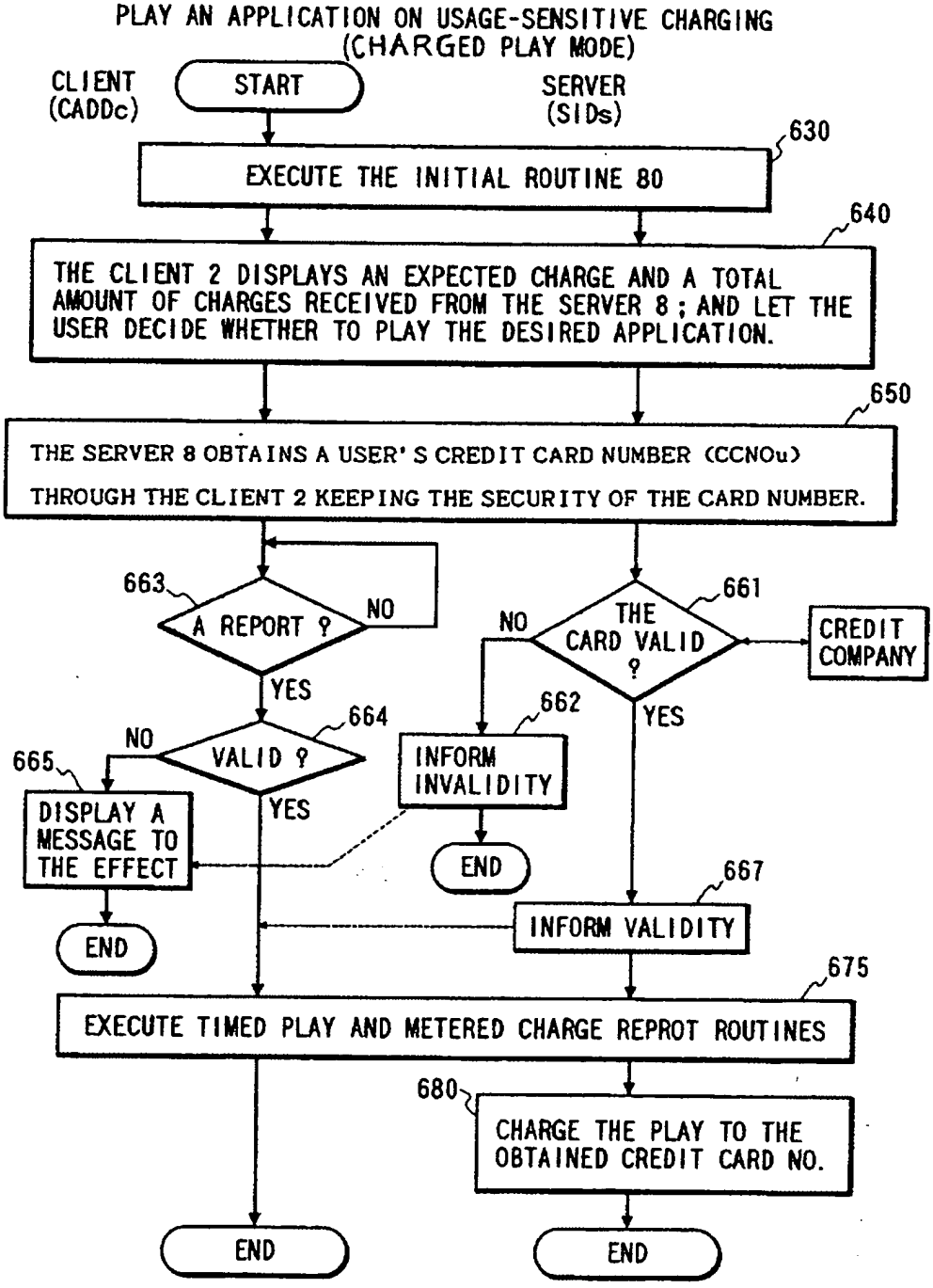


FIG. 16A

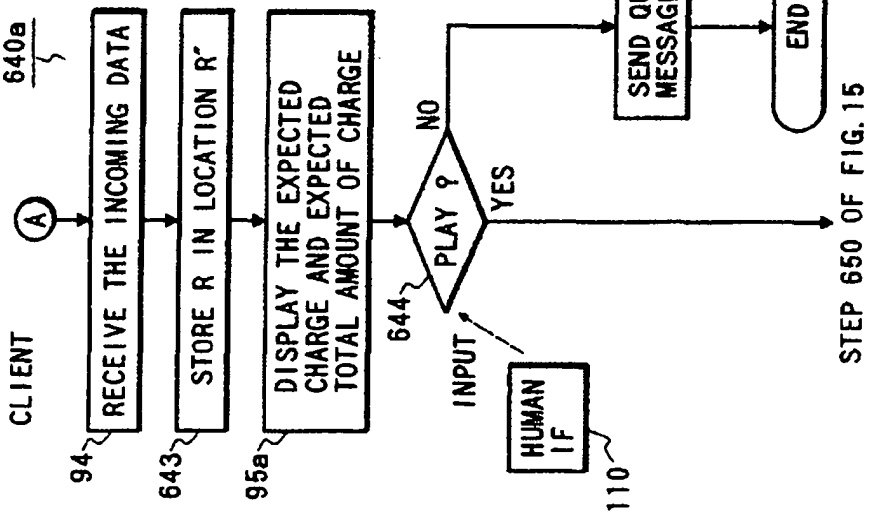


FIG. 16B

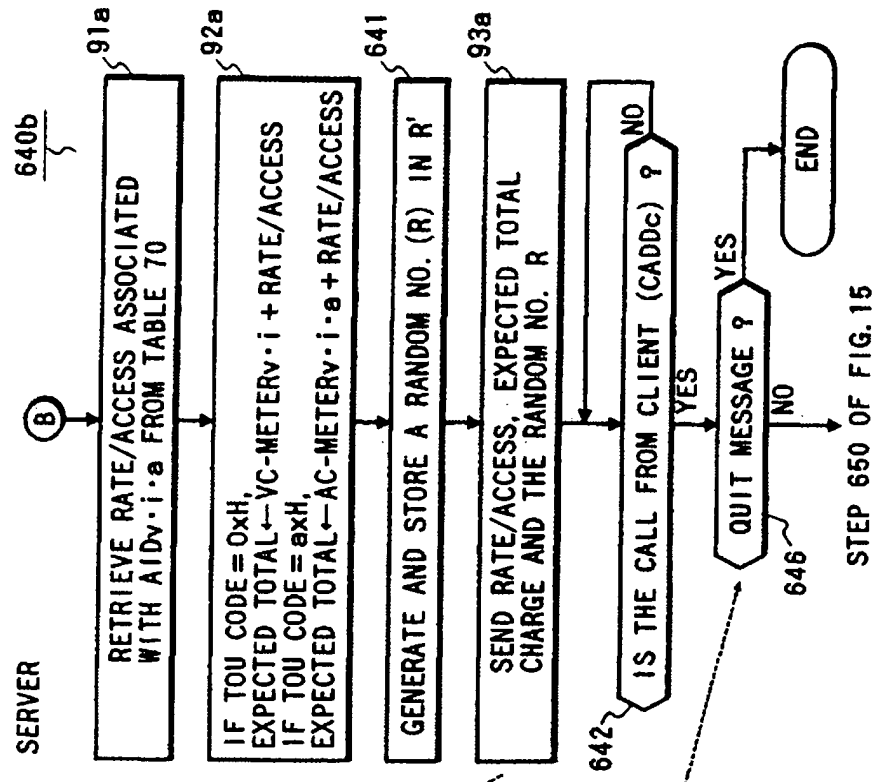


FIG. 17A

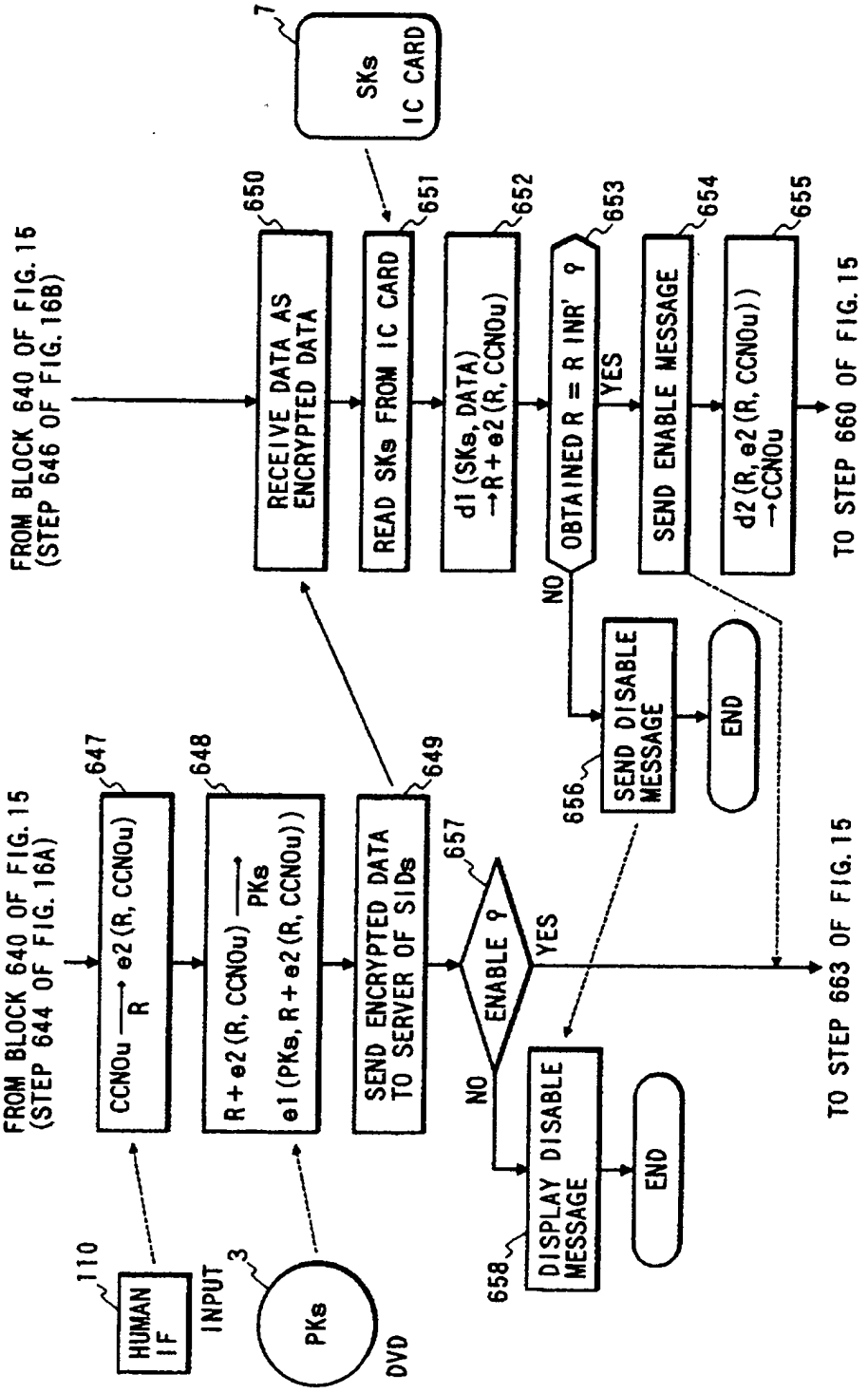


FIG. 17B

FIG. 18A

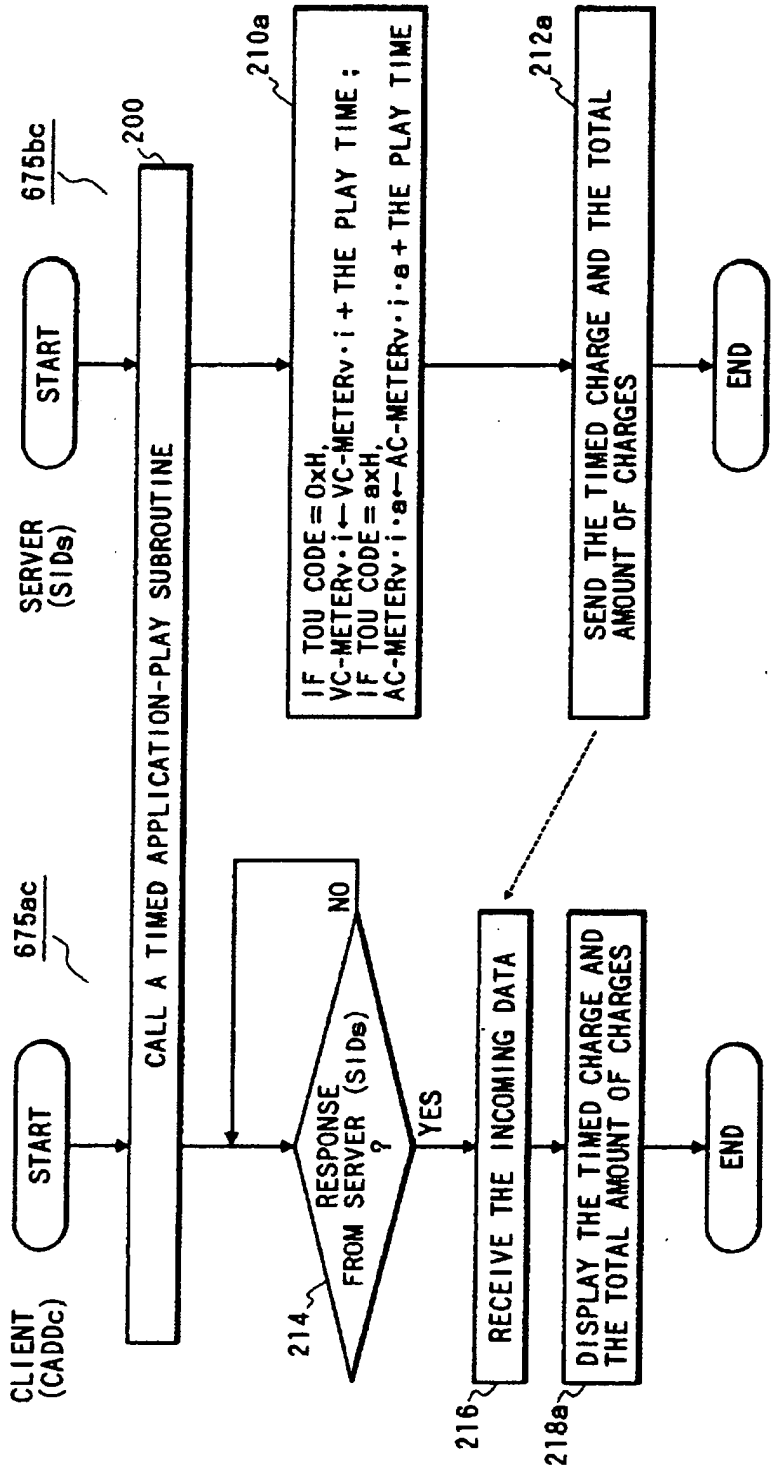


FIG. 18B

FIG. 19

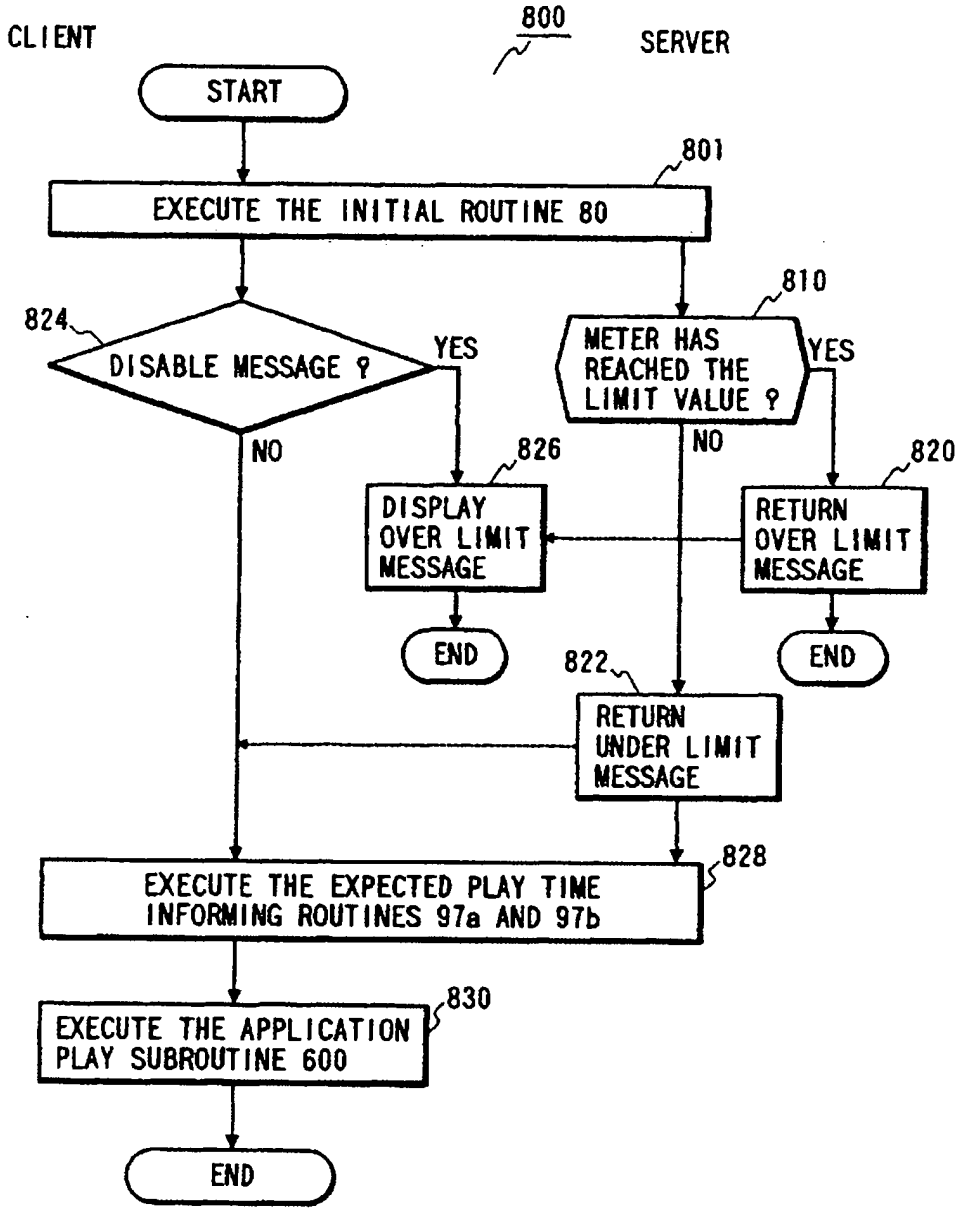


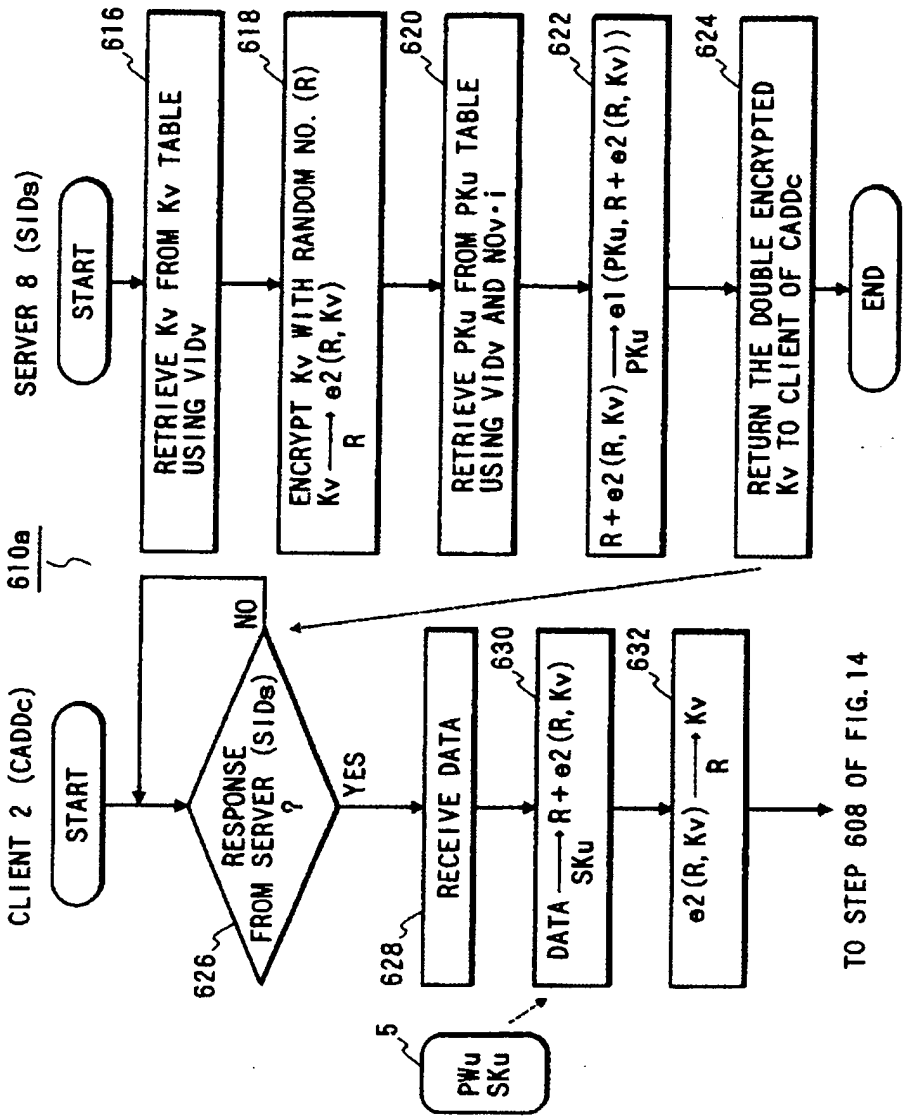
FIG. 20A

VIDv	Kv
VID1	K1
VID2	K2
...	...

FIG. 20B

VIDv	NOv·i	PKu
VID1	NO1·1	PK347020
	NO1·2	PK001031
	...	...
	NO1·365	PK314162
VID2	NO2·1	PK141421
.....	.....	.....
	NO2·77	PK789012
VID3	NO3·1	PK123456
.....	.....	.....

FIG. 20C



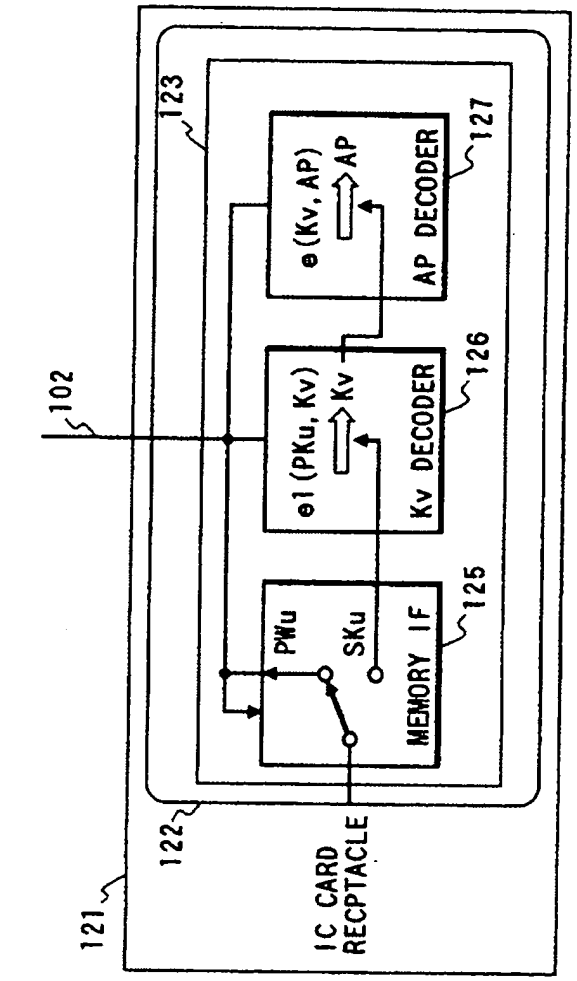
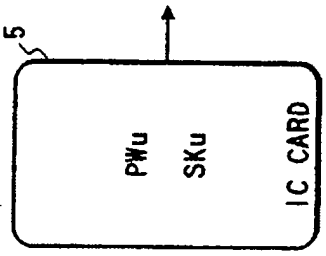


FIG. 21



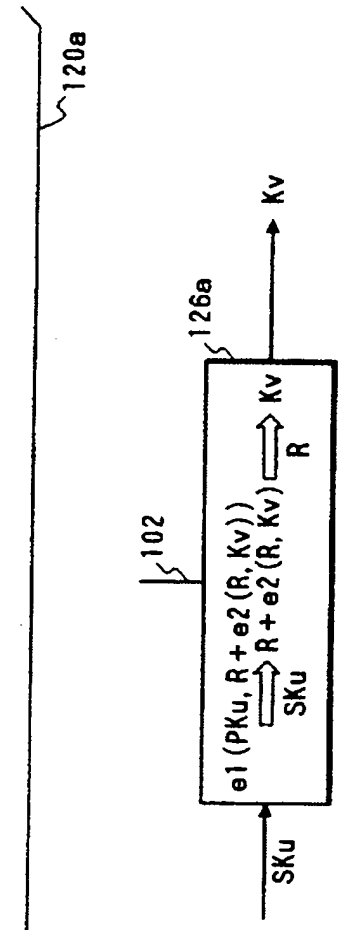


FIG. 22

FIG. 23

THE HIGHER DIGIT OF TERMS-OF-USE CODE (HEXADECIMAL)	THE TERMS-OF-USE CODE IS APPLIED TO :
0	THE ENTIRE VOLUME
1	APPLICATION 1
2	APPLICATION 2
⋮	⋮

36

XYH (X, Y = 1, 2, ···, F)

THE LOWER DIGIT OF TERMS-OF-USE CODE (HEXADECIMAL)	CORRESPONDING LIMIT VALUE	
0	NONE	(FREE)
1	NONE	(CHARGED)
2	THE EFFECTIVE DATE AND TIME	
3	THE ALLOWABLE EXPIRATION DATE AND TIME	
4	THE MAXIMUM AMOUNT OF USED PERIOD	
5	THE ALLOWABLE ACCESS COUNT	
⋮	⋮	

37

FIG. 24

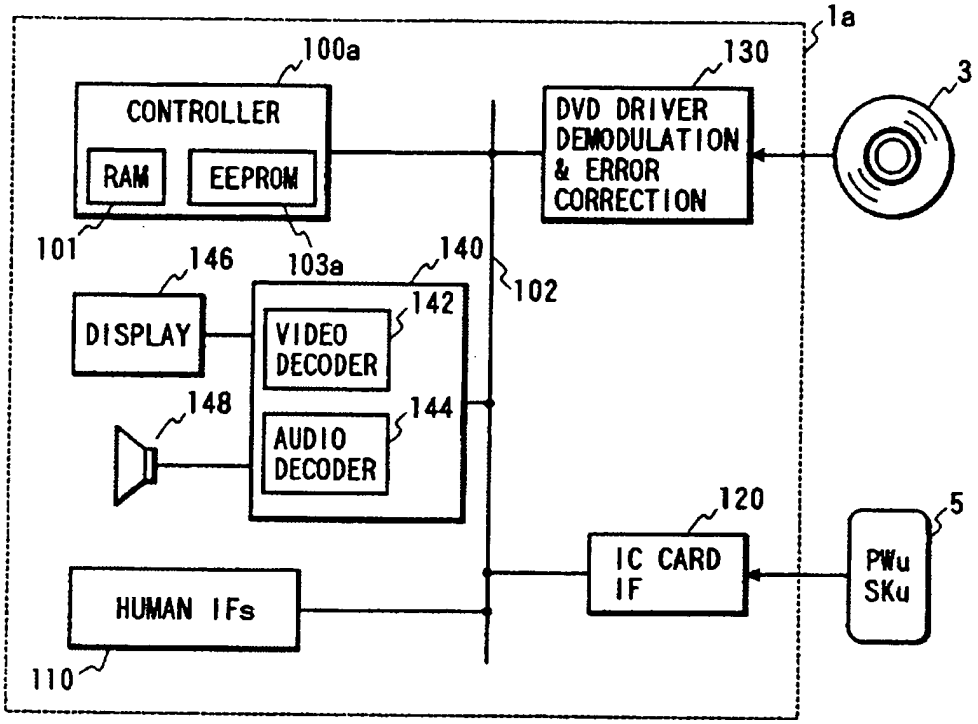


FIG. 26

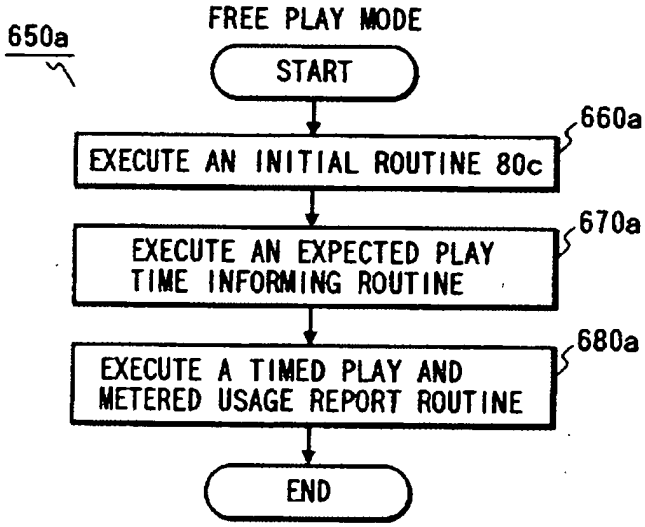


FIG. 25

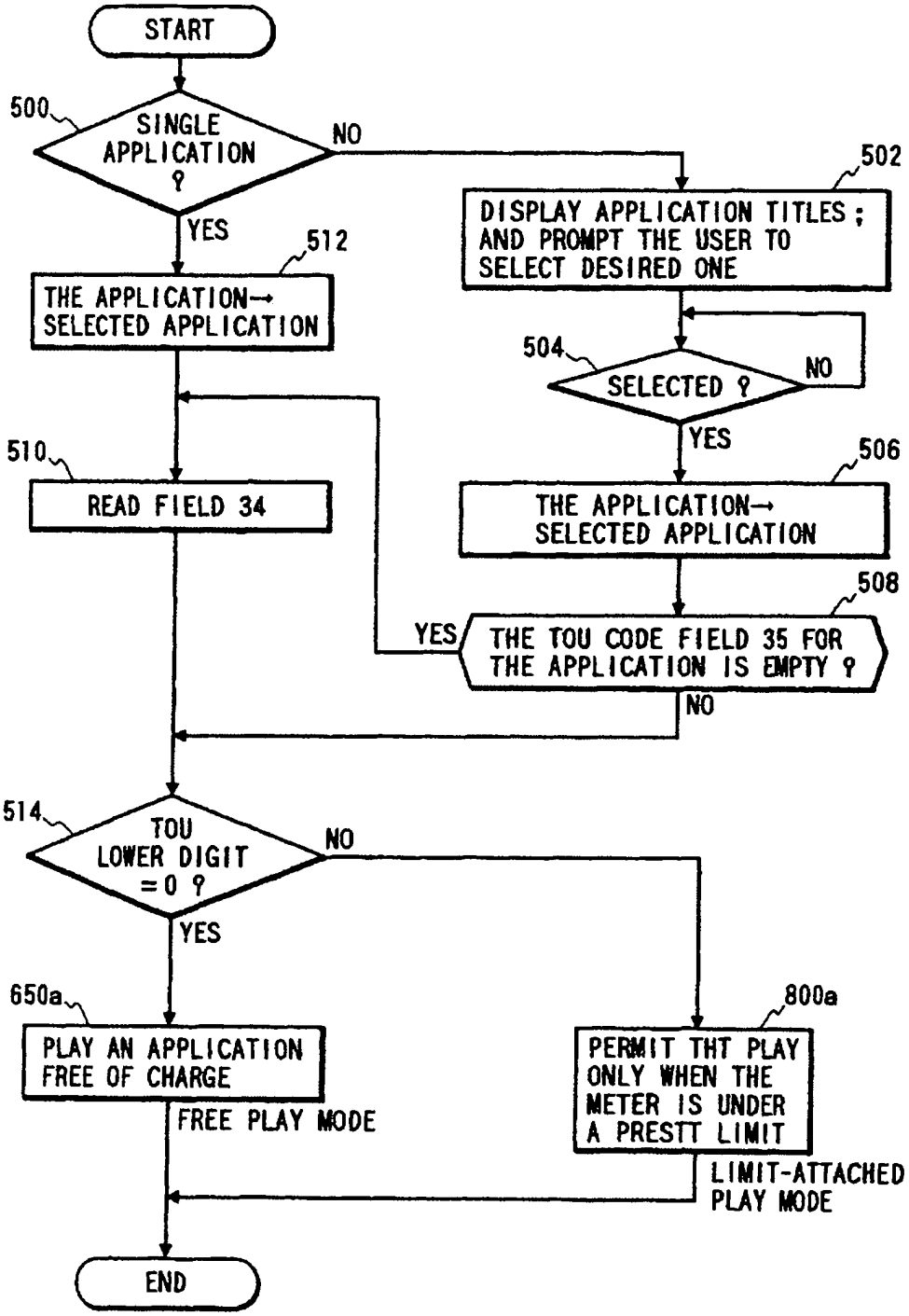


FIG. 27

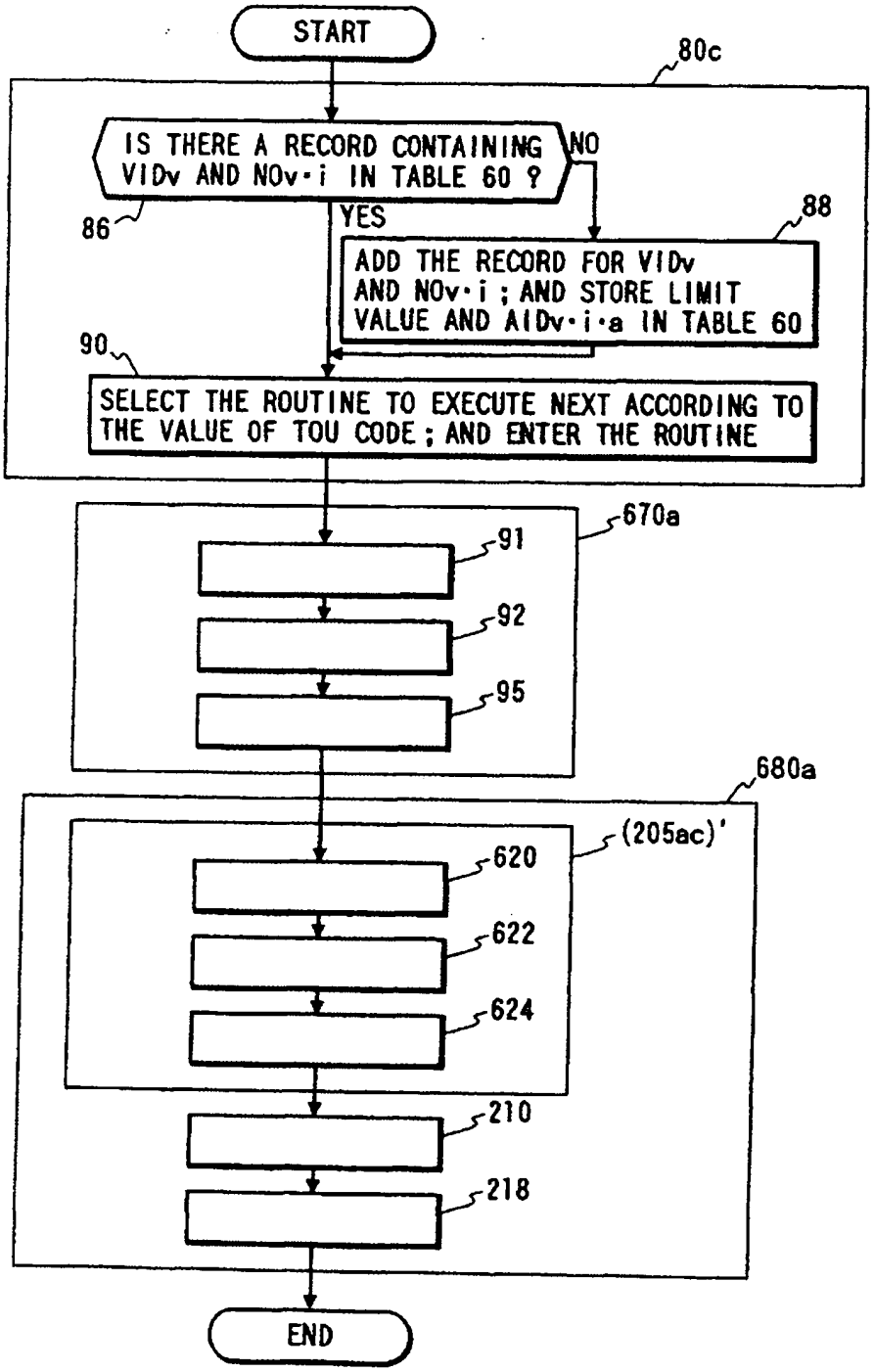
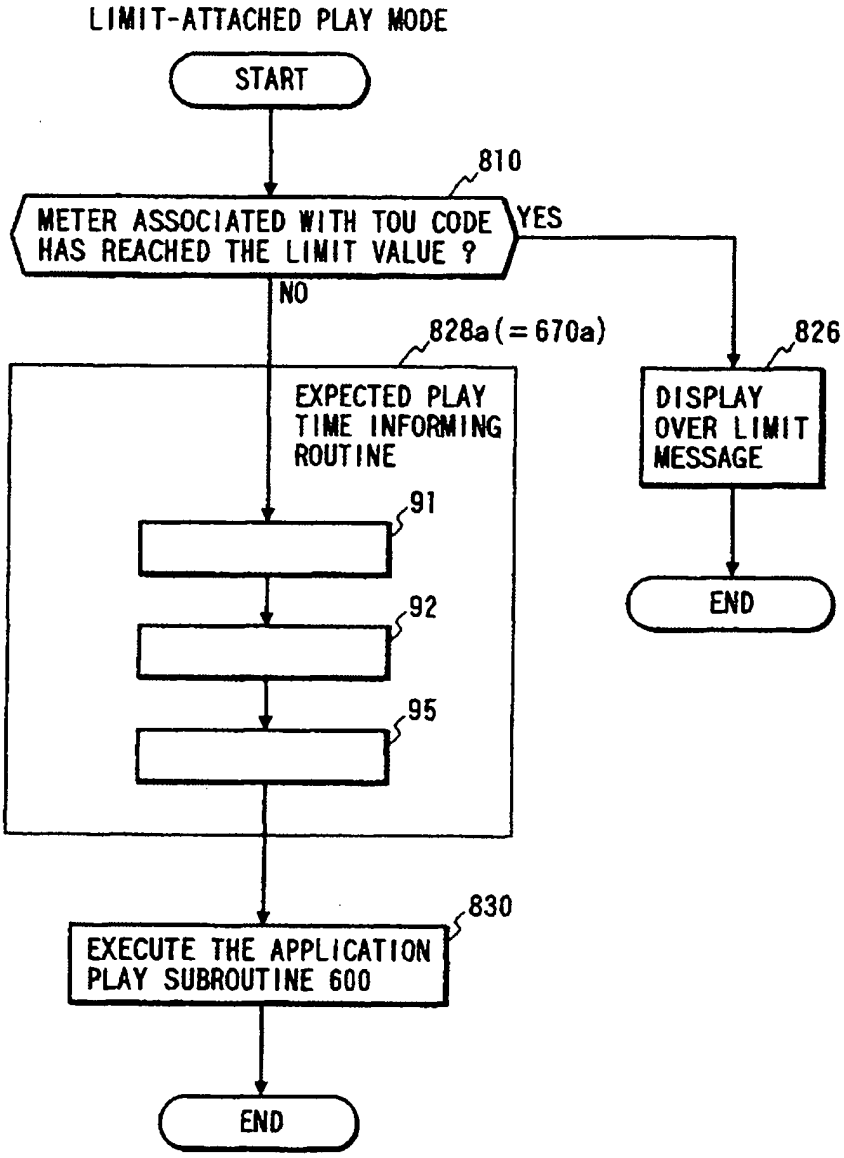


FIG. 28



(12) UK Patent Application (19) GB (11) 2 309 364 (13) A

(43) Date of A Publication 23.07.1997

(21) Application No 9700921.1

(22) Date of Filing 17.01.1997

(30) Priority Data
(31) 08588848 (32) 19.01.1996 (33) US

(51) INT CL[6]
H04L 9/30

(52) UK CL (Edition O)
H4P PDCSC
U1S S2204 S2208 S2209

(56) Documents Cited
EP 0328232 A2 WO 95/23468 A1

(58) Field of Search
UK CL (Edition O) H4P PDCSA PDCSC
INT CL[6] H04L 9/30 9/32
Online:WPI,INSPEC

(71) Applicant(s)
Northern Telecom Limited

(Incorporated in Canada - Quebec)

World Trade Center Of Montreal,
380 St Antoine Street West, 8th Floor, Montreal,
Quebec H2Y 3Y4, Canada

(72) Inventor(s)
David Allan
Liam Casey
Adrian Jones

(74) Agent and/or Address for Service
M C Dennis
Nortel Patents, London Road, HARLOW, Essex,
CM17 9NA, United Kingdom

(54) Public/private key encryption/decryption

(57) In a hybrid fiber-coax distribution network, communications between a central station and particular end stations are encrypted using a working key (WK) of a symmetric encryption scheme. The central station has a public and private key (PPK) of a PPK encryption scheme, and some of the end stations can also each have a respective PPK. To provide secure communications for each end station, if the end station has a PPK, then the respective WK is generated in the central station and communicated, encrypted using the end station's public key (PK), to the end station. Otherwise, the WK is generated in the end station and communicated, encrypted using the central station's PK, to the central station. An individual identifier for each end station, and a cryptographic signature at least for end stations not having a PPK, can be communicated to the central station for authentication of the end stations.

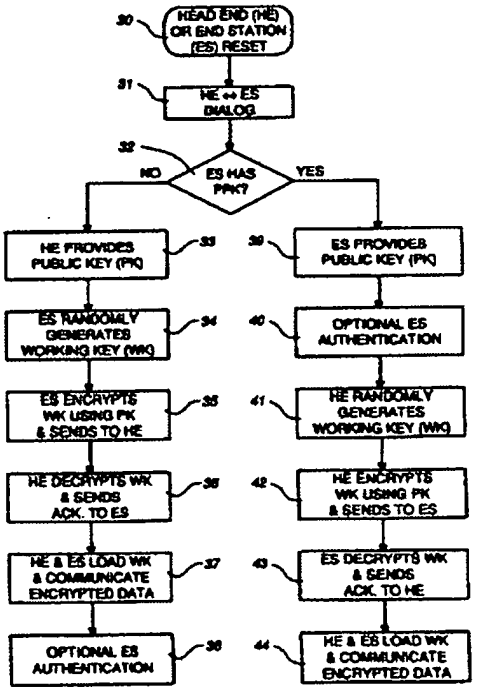


Fig. 2

GB 2 309 364 A

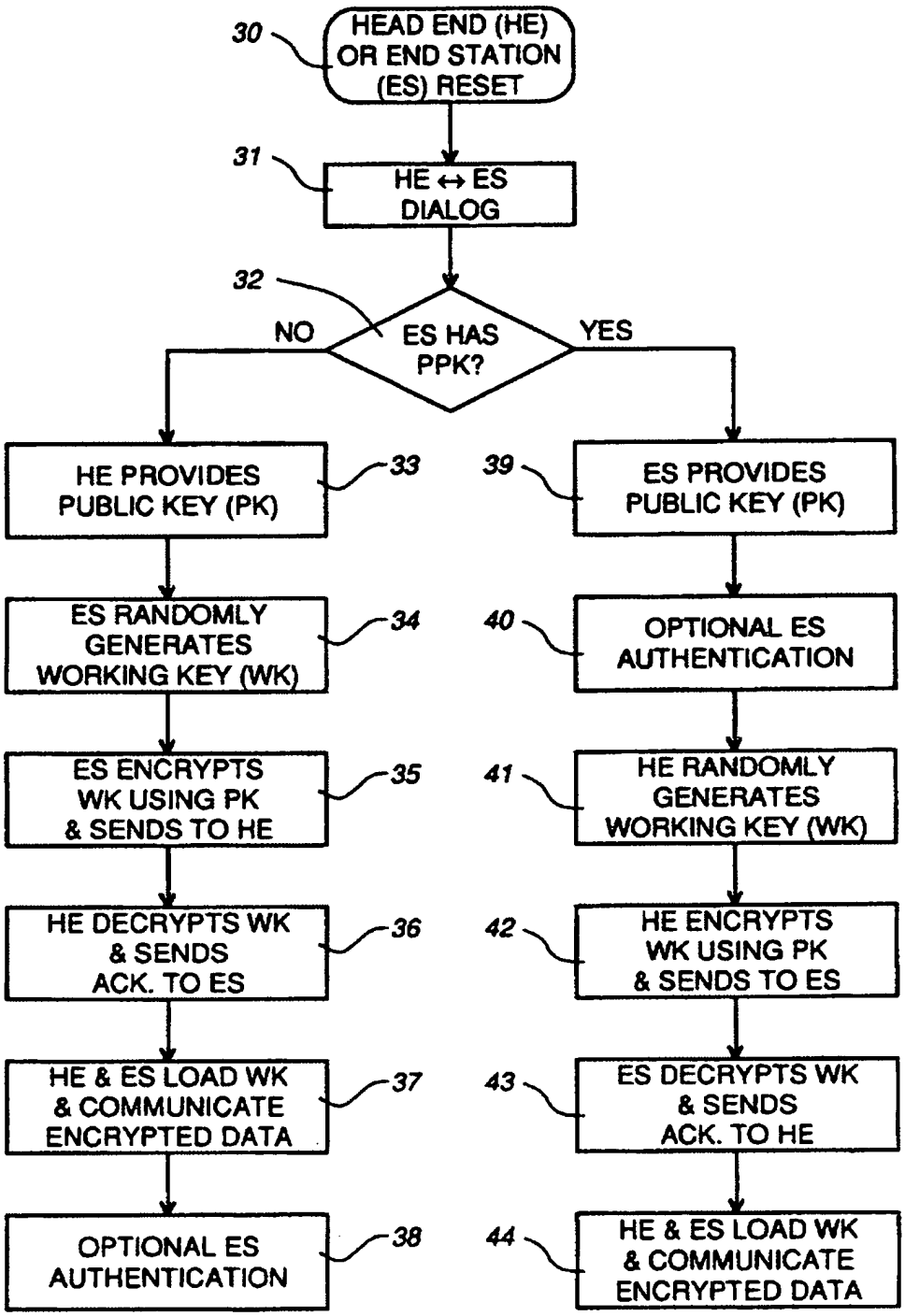

30
HEAD END (HE) OR END STATION (ES) RESET
31
HE ↔ ES DIALOG
32
ES HAS PPK?
NO
YES
HE PROVIDES PUBLIC KEY (PK)
33
ES RANDOMLY GENERATES WORKING KEY (WK)
34
ES ENCRYPTS WK USING PK & SENDS TO HE
35
HE DECRYPTS WK & SENDS ACK. TO ES
36
HE & ES LOAD WK & COMMUNICATE ENCRYPTED DATA
37
OPTIONAL ES AUTHENTICATION
38
39
ES PROVIDES PUBLIC KEY (PK)
40
OPTIONAL ES AUTHENTICATION
41
HE RANDOMLY GENERATES WORKING KEY (WK)
42
HE ENCRYPTS WK USING PK & SENDS TO ES
43
ES DECRYPTS WK & SENDS ACK. TO HE
44
HE & ES LOAD WK & COMMUNICATE ENCRYPTED DATA

Fig. 2

FACILITATING SECURE COMMUNICATIONS IN A DISTRIBUTION NETWORK

This invention relates to methods of facilitating secure communications in a distribution network, such as for example a coaxial cable or hybrid fiber-coax (HFC) network.

Background of the Invention

A distribution network, such as an HFC network in which data is communicated to subscriber end stations via optical fiber and coaxial distribution cables, is a point-to-multipoint network in which data addressed to and intended for any particular subscriber is also inevitably supplied via the network to other subscribers. If the data is not scrambled or encrypted, it can be easily monitored by these other subscribers, leading to a loss of subscriber privacy and a loss of revenues for data suppliers when the data (e.g. television programs) is supplied for a fee. Accordingly, it is important to provide a desired level of security in the data communications in a distribution network.

While various encryption and decryption schemes are known, these have a number of disadvantages associated with them in the environment of a distribution network. A significant factor in this respect is the cost and security of subscriber end stations. As a distribution network will contain large numbers of subscriber end stations, it is commercially necessary that the cost of each end station be kept relatively low. It is therefore desirable to avoid incorporating expensive security schemes in the subscriber end stations. However, subscriber end stations are also easily subject to theft, tampering, and duplication, so that complicated schemes have been considered necessary to provide adequate security.

For example, a security scheme can be implemented using an encryption key which can be stored in the subscriber end station. To prevent access to the encryption key, the store in the subscriber end station, and data lines to and from this store, must also be made physically secure. This leads to extra complexity and costs. Different subscribers may have differing security and privacy needs, which makes it desirable for the network to accommodate differing security schemes and end station costs.

A further security-related desirable aspect of a distribution network is an ability for authentication of subscriber end stations, typically using a unique end station identity which can be physically incorporated (e.g. hard wired) into the end station during manufacture.

Encryption schemes can be divided into those involving public and private keys (PPK) and those involving symmetric keys. In PPK schemes, a first station can distribute its public key, in accordance with which a second station can encrypt data and send the encrypted data to the first station, which decrypts the data using its private key. Because the private key is retained at the first station, and is not practically discoverable

by other parties, PPK schemes are considered to be secure. However, the encryption and decryption processes are relatively slow, so that such schemes are not practical for encryption of real-time high-speed data, such as television program signals, for which distribution networks are primarily intended.

In symmetric key schemes, a single key, referred to as a working key, is used by both of first and second stations to encrypt and decrypt data being communicated between the stations. The nature of the working key is such that encryption of real-time high-speed data, such as television program signals, is practical. However, these schemes require that the working key be present in both stations, and make it desirable for the working key to be periodically changed or updated. Thus symmetric key schemes require generation of a working key in one of the stations or in a third station referred to as a key distribution agent, and communication of the working key to the other station(s).

This communication itself presents a risk of the working key being insecure, and this risk increases with the frequency with which the working key is updated. It is also known to avoid this risk by using a PPK scheme for communication of a working key, and then to use the working key for data encryption.

An object of this invention is to provide a method of facilitating secure communications in a distribution network.

Summary of the Invention

One aspect of this invention provides a method of facilitating secure communications using encryption and decryption processes in a distribution network comprising a central station and a plurality of addressable end stations, in which communications from the central station addressed to and intended for a particular end station are delivered via the network to a plurality of end stations, wherein the central station has, and one or more of the end stations can each have, a respective public and private key (PPK) of a PPK encryption scheme, comprising the steps of:

(a) determining in communications between the central station and an end station whether the end station has a PPK, if so proceeding with step (b) and if not proceeding with step (c);

(b) at the central station, determining the public key (PK) of the end station, generating a working key (WK) for encryption of communications to the end station, encrypting the WK using the PK of the end station, and communicating the encrypted WK to the end station; at the end station, decrypting the WK using the private key of the end station; and proceeding with step (d);

(c) at the end station, determining the public key (PK) of the central station, generating a working key (WK) for encryption of communications to the central station, encrypting the WK using the PK of the central station, and communicating the encrypted WK to the central station; at the central station, decrypting the WK using the private key of the central

station; and proceeding with step (d);

(d) using the WK to encrypt at the central station, and to decrypt at the end station, communications from the central station to the end station.

Another aspect of this invention provides a method of facilitating secure communications in a distribution network comprising a central station and a plurality of addressable end stations, in which communications from the central station addressed to and intended for a particular end station are delivered via the network to a plurality of end stations, wherein the central station has a public and private key (PPK) of a PPK encryption scheme and each end station has an individual identity (ID) and an individual cryptographic signature encrypted using a private key of a predetermined PPK encryption scheme, comprising the steps of: communicating the ID of an end station to the central station; at the end station, generating a working key (WK) for encryption of communications between the end station and the central station and encrypting the WK using the public key of the central station; communicating the encrypted WK from the end station to the central station; at the central station, decrypting the encrypted WK using the private key of the central station; communicating the cryptographic signature of the end station to the central station; and at the central station, decrypting the cryptographic signature using a public key of the predetermined PPK scheme for authentication of the end station.

Brief Description of the Drawings

The invention will be further understood from the following description with reference to the accompanying drawings, in which:

Fig. 1 illustrates parts of a distribution network to which the invention is applied; and

Fig. 2 is a flow chart illustrating steps of a method for facilitating secure communications in the network in accordance with the invention.

Detailed Description

The invention is described below in the context of a hybrid fiber-coax (HFC) distribution network in which signals are distributed from a central station or head end (HE) to a large number of subscriber end stations (ES) via optical fibers and coaxial cables in known manner. An example of such a network is described in Warwick United States Patent No. 5,408,259 issued April 18, 1995 and entitled "Data Modulation Arrangement For Selectively Distributing Data". Typically in such a network digital data communications are provided between any ES and the HE using asynchronous transfer mode (ATM) cells which are communicated in both directions, i.e. downstream from the HE to the ES and upstream from the ES to the HE, using suitable modulation schemes and carrier frequencies outside the bands used for analog television signals also carried on

the coaxial cables. However, it is observed that the invention is equally applicable to other forms of distribution network.

Referring to Fig. 1, there is illustrated parts of a distribution network in which many end stations, only two of which are shown and are referenced 10 and 12, are connected via branched cables 14 of the distribution network to a head end 16, via which the end stations have access to a network (not shown) which for example supplies digital television program signals subscribed to by end station subscribers. The cables 14 can comprise both optical fiber and coaxial cables forming a hybrid fiber-coax arrangement, on which the digital signals can be communicated in known manner using ATM cells.

As can be appreciated from the illustration in Fig. 1, signals communicated by the head end 16 and intended for any particular end station will actually be delivered via the cables 14 to all of the end stations. For secure and/or private communication of the signals, the head end 16 includes an encryption engine 18 which encrypts the signals in accordance with a working key known only by the head end and the intended end station, which also includes an encryption engine 20 which decrypts the signals for use. These working keys are similarly used for communications in the opposite direction, from the end station to the head end 14. The working keys of this symmetric key encryption scheme are provided in the head end and the end station in a manner which is described in detail below.

The end stations 10 and 12 are of two types, with differing levels of security to enable different security needs of subscribers to be accommodated. The end station 12 represents a relatively secure end station, which includes its own public and private keys of a PPK encryption scheme. As explained in the introduction, such an end station has a relatively high complexity and cost, because of the need for secure storage of the keys and operation of the PPK encryption. Other end stations, which do not have their own public and private keys and accordingly can be provided at a much lower cost, are represented by the end station 10. The network as a whole may have an arbitrary mix of these two types of end station.

Each end station 10 or 12 also has an individual, unique identity number, which is stored (e.g. hard wired) into the ES during its manufacture. This is referred to as a global ID (identity). The global IDs of all of the end stations are stored in a database 22, which can be colocated with the head end 16 or separately from it and with which the head end 16 communicates via a path 24. The head end 16 also has its own public and private keys of a PPK encryption scheme.

Fig. 2 shows steps of a process which is followed in order to set up secure communications between the head end 16 and one of the end stations 10 or 12. This process takes place between the head end and the respective end station without involvement of any other node such as a central key distribution agent, and is described

below as being initiated in each case following any reset (e.g. following a power-up) of either the head end 16 or the respective end station. Consequently, the working key which is used for encrypting the communications between the head end and the end station is changed on any reset. However, the same process can alternatively or additionally be carried out on demand, and/or periodically to provide periodic changes of the working key. It is also observed that the encrypted communications take place between the encryption engines 18 in the head end 16 and 20 in the respective end station 10 or 12, and communications on the network access side of the head end 16 are not subject to the same encryption.

In Fig. 2, a block 30 represents a reset of the head end (HE) or end station (ES), in response to which, as shown by a block 31 in Fig. 2, a dialog or handshake is carried out between the HE and the ES to establish communications between them. These communications are effected using unencrypted ATM cells using addresses of the end station and the head end. As a part of this dialog, as shown by a block 32 in Fig. 2 the head end 16 interrogates the end station to determine whether or not the end station has its own public and private keys. If not, i.e. if the end station is an end station 10 as described above, then the process continues with successive blocks 33 to 38 in Fig. 2. If the interrogation establishes that the end station is an end station 12 having its own public and private keys, then the process instead continues with blocks 39 to 44 in Fig. 2.

In the former case of an end station 10, as shown by the block 33 the head end 16 communicates its public key (PK) to the end station 10; this communication can form part of the dialog block 31. The end station 10 randomly generates (block 34) a working key (WK) for communicating signals in a symmetric key encryption scheme, and encrypts (block 35) this working key in accordance with the supplied public key, sending the encrypted working key in a message to the head end 16. The head end 16 decrypts (block 36) the encrypted working key from this message in accordance with its private key, which is not known to others so that the communication of the working key from the end station 10 to the head end 16 is secure, and optionally but preferably sends an acknowledgement to the end station 10. As shown by the block 37, the head end 16 and the end station 10 then load their encryption engines 18 and 20 respectively with the working key, and thereafter (until this process is repeated, for example in response to a subsequent reset at either end) communications between them take place with data encrypted in accordance with the working key. An optional additional step represented by the block 38 provides for authentication of the end station 10 in a manner described below.

Conversely, in the latter case of an end station 12, as shown by the block 39 the end station 12 communicates its public key (PK) to the head end 16; this communication can form part of the dialog block 31. An optional authentication step for the end station

12 can be carried out by the head end 16 as represented by the block 40 in a manner described below. The head end 16 randomly generates (block 41) a working key (WK) for communicating signals in a symmetric key encryption scheme, and encrypts (block 42) this working key in accordance with the supplied public key of the end station 12, sending the encrypted working key in a message to the end station 12. The end station 12 decrypts (block 43) the encrypted working key from this message in accordance with its private key, which is not known to others so that the communication of the working key from the head end 16 to the end station 12 is secure, and optionally but preferably sends an acknowledgement to the head end 18. As shown by the block 44, the head end 16 and the end station 12 then load their encryption engines 18 and 20 respectively with the working key, and thereafter (until this process is repeated, for example in response to a subsequent reset at either end) communications between them take place with data encrypted in accordance with the working key.

It can be seen from the above description that, in the relatively secure but more expensive situation in which the end station 12 includes its own public and private keys, these are used for communicating a working key generated in the head end, whereas in the other case the end station 10 generates the working key and this is communicated to the head end using the latter's public key.

The optional step of authentication of the end station 12 in the block 40 as described above can make use of the global ID of the end station 12 together with data in the database 22, in which the public key of the end station 12 is stored in association with this global ID. As part of the dialog block 31, the end station communicates its global ID to the head end 16. In the step 40, therefore, the head end 16 can communicate via the path 24 with the database 22 to confirm that the public key which it has received from the end station 12 in the step 39 matches that stored in the database 22 for this end station's global ID, the subsequent steps 41 to 44 only being followed if this authentication step is successful.

Alternatively, or in addition, the optional end station authentication step of block 40 can comprise the steps of the head end sending an unencrypted message to the end station 12 with a request that it be cryptographically signed. In accordance with this request, the end station 12 produces a digest of the message using a known hashing function (thereby reducing the data to be encrypted), encrypts this digest in accordance with its private key, and sends the encrypted message digest to the head end 16. The head end 16 then decrypts this in accordance with the public key of the end station, retrieved from the database 22, to confirm the digest of its original message which the head end also produces using the hashing function.

It can be seen that, alternatively, the steps represented by the blocks 39 and 40 in Fig. 2 could be replaced by a single step in which the head end 16 determines the public

key of the end station 12 from the database 22 in accordance with the global ID of the end station 12 supplied in the dialog 31, without any authentication of the end station or any communication of the public key from the end station 12.

The above sequences provide a particularly strong or secure authentication of the end station 12. For the end station 10 which does not have its own public and private keys, a weaker but still valuable authentication can be provided as shown by the block 38. The authentication block 38 is shown in Fig. 2 as the final block in the process because this enables the exchange of data in the authentication process to be encrypted in accordance with the working key, but this authentication step could alternatively be provided anywhere else in the sequence of steps from the blocks 31 to 37.

For this optional authentication step, the end station 10 is manufactured (e.g. hard wired) with not only its global ID, but also a cryptographic signature. Conveniently, the end station 10 is manufactured with a certificate comprising data including the global ID of the end station and the public key of the manufacturer and a cryptographic signature comprising an encryption, in accordance with the private key of the manufacturer, of a digest of that data produced using a known hashing function. The public key of the manufacturer can also or instead be stored in the database 22. The optional end station authentication step of the block 38 comprises a communication of the cryptographic signature from the end station 10 to the head end 16 (as explained above this could be a part of the dialog 31 or any later step, but the encryption after the block 37 obstructs public observation in the network of cryptographic signatures). The head end 16 then confirms the authenticity of the end station 10 by decrypting the cryptographic signature using the manufacturer's public key, producing a digest from the same data (global ID and public key, both of which can be communicated in the dialog step 31 or later) and the known hashing function, and matching these.

This is a relatively weak authentication, in that identical copies of the end station 10, including duplicated data and cryptographic signatures, could operate at different times on the network without this being detected. However, simultaneous operation of two or more such duplicates would be detected by the fact that two or more end stations would be supplying the same global ID which is supposedly unique. Thus even such a weak authentication is valuable especially in detecting illicit large-scale duplication of end stations.

The processes in accordance with the invention as described above provide a number of significant advantages over known configurations. In particular, requirements for secure storage of public and private keys are minimized in the network as a whole, and eliminated for the end stations 10 which can accordingly be provided at relatively lower cost. At the same time, end stations 12 with greater security can be provided, and the head end 16 can operate simultaneously with both types of end station. This, combined

with optional authentication of the end stations as described above, enables different degrees of security to be easily provided in the network in accordance with service requirements.

Furthermore, renewal of the working keys at reset is simpler than providing time-based schedules for changing encryption keys, and key exchanges take place only between the head end and the end station which use the keys, thereby enhancing security compared with distribution of keys from a key distribution agent. In addition, all of the data flowing between the head end and any particular end station 10 or 12, between successive resets, can be encrypted using a single working key, thereby simplifying the encryption and decryption processes. However, it is observed that different working keys could be generated, communicated, and used in the same manner as described above for encrypting and decrypting different types of information, or different services, for a single end station 10 or 12.

Although particular embodiments of the invention have been described in detail, it should be appreciated that numerous modifications, variations, and adaptations may be made without departing from the scope of the invention as defined in the claims.

WHAT IS CLAIMED IS:

1. A method of facilitating secure communications using encryption and decryption processes in a distribution network comprising a central station and a plurality of addressable end stations, in which communications from the central station addressed to and intended for a particular end station are delivered via the network to a plurality of end stations, wherein the central station has, and one or more of the end stations can each have, a respective public and private key (PPK) of a PPK encryption scheme, comprising the steps of:

(a) determining in communications between the central station and an end station whether the end station has a PPK, if so proceeding with step (b) and if not proceeding with step (c);

(b) at the central station, determining the public key (PK) of the end station, generating a working key (WK) for encryption of communications to the end station, encrypting the WK using the PK of the end station, and communicating the encrypted WK to the end station; at the end station, decrypting the WK using the private key of the end station; and proceeding with step (d);

(c) at the end station, determining the public key (PK) of the central station, generating a working key (WK) for encryption of communications to the central station, encrypting the WK using the PK of the central station, and communicating the encrypted WK to the central station; at the central station, decrypting the WK using the private key of the central station; and proceeding with step (d);

(d) using the WK to encrypt at the central station, and to decrypt at the end station, communications from the central station to the end station.

2. A method as claimed in claim 1 wherein each end station has an individual identity (ID) and step (a) includes the step of communicating the ID of the end station to the central station.

3. A method as claimed in claim 2 wherein in step (b) the PK of the end station is determined by the central station from a database using the ID of the end station.

4. A method as claimed in claim 1, 2, or 3 wherein step (b) further comprises an end station authentication step comprising the steps of communicating an unencrypted message from the central station to the end station, producing an encrypted message at the end station using the private key of the end station, communicating the encrypted message to the central station, decrypting the message at the central station using the PK of the end station, and comparing the decrypted message with the original message.

5. A method as claimed in claim 4 wherein in step (b) the end station authentication step is carried out before the step of communicating the encrypted WK to the end station.

6. A method as claimed in any of claims 1 to 5 wherein in step (b) the PK of the end station is communicated to the central station from the end station.

7. A method as claimed in claims 2 and 6 wherein in step (b) the PK of the end station is verified by the central station from a database using the ID of the end station.

8. A method as claimed in any of claims 1 to 7 wherein a plurality of end stations which do not have a PPK each have an individual cryptographic signature encrypted using a private key of a predetermined PPK scheme, step (a) or (c) includes the step of communicating the cryptographic signature of the end station to the central station, and step (c) further comprises an end station authentication step comprising, at the central station, decrypting the cryptographic signature using a public key of the predetermined PPK scheme.

9. A method as claimed in claims 2 and 8 wherein the individual cryptographic signature comprises an encryption of data derived from the ID of the respective end station.

10. A method as claimed in claim 8 or 9 wherein the predetermined PPK scheme uses a private key and a public key of a source of the end station.

11. A method as claimed in claim 8, 9, or 10 wherein the cryptographic signature is communicated to the central station in step (c).

12. A method as claimed in claim 11 and including the steps of encrypting the cryptographic signature at the end station, and decrypting the encrypted cryptographic signature at the central station, using the WK.

13. A method as claimed in any of claims 1 to 12 and further comprising the step of using the WK to encrypt at the end station, and to decrypt at the central station, communications from the end station to the central station.

14. A method of facilitating secure communications in a distribution network comprising a central station and a plurality of addressable end stations, in which communications from the central station addressed to and intended for a particular end station are delivered via the network to a plurality of end stations, wherein the central station has a public and private key (PPK) of a PPK encryption scheme and each end station has an individual identity (ID) and an individual cryptographic signature encrypted using a private key of a predetermined PPK encryption scheme, comprising the steps of:

communicating the ID of an end station to the central station;

at the end station, generating a working key (WK) for encryption of communications between the end station and the central station and encrypting the WK

using the public key of the central station;

communicating the encrypted WK from the end station to the central station;

at the central station, decrypting the encrypted WK using the private key of the central station;

communicating the cryptographic signature of the end station to the central station; and

at the central station, decrypting the cryptographic signature using a public key of the predetermined PPK scheme for authentication of the end station.

15. A method as claimed in claim 14 wherein the individual cryptographic signature comprises an encryption of data derived from the ID of the respective end station.

16. A method as claimed in claim 14 or 15 wherein the predetermined PPK scheme uses a private key and a public key of a source of the end station.

17. A method as claimed in claim 14, 15, or 16 wherein the step of communicating the cryptographic signature of the end station to the central station comprises the steps of encrypting the cryptographic signature at the end station using the WK, communicating the encrypted cryptographic signature from the end station to the central station, and decrypting the encrypted cryptographic signature at the central station using the WK.

18. A method of facilitating secure communications in a distribution network, substantially as hereinbefore described with reference to Figs 1 and 2 of the accompanying drawings.

The Patent Office

12

Application No: GB 9700921.1
Claims searched: 1-13

Examiner: Mr B J Spear
Date of search: 19 March 1997

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.O): H4P (PDCSC)

Int Cl (Ed.6): H04L 9/30

Other: Online: WPI, INSPEC

Documents considered to be relevant:

Category	Identity of document and relevant passage	Relevant to claims
	NONE	

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

The Patent Office

Application No: GB 9700921.1
Claims searched: 14-17

Examiner: Mr B J Spear
Date of search: 21 May 1997

Patents Act 1977
Further Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.O): H4P (PDCSA)

Int Cl (Ed.6): H04L 9/32

Other: Online: WPI, INSPEC

Documents considered to be relevant:

Category	Identity of document and relevant passage	Relevant to claims
A	EP0328232A2 (Fischer)	-
A	WO 95/23468A1 (Merdan)	-

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

1 483 282

PATENT SPECIFICATION (11) 1 483 282

(21) Application No. 52131/74 (22) Filed 2 Dec. 1974
(31) Convention Application No. 7342706
(32) Filed 30 Nov. 1973 in
(33) France (FR)
(44) Complete Specification published 17 Aug. 1977
(51) INT CL² G06F 13/00
(52) Index at acceptance
G4A 10EX 13E 13M 17B4 17P 6G 6H 6X AP ND NR

(54) APPARATUS FOR PROTECTING THE INFORMATION IN AN VIRTUAL MEMORY SYSTEM IN PROGRAMMED DATA PROCESSING APPARATUS

(71) We, COMPAGNIE INTERNATIONALE POUR L'INFORMATIQUE CII-HONEYWELL-BULL (formerly Compagnie Honeywell-Bull), a French Body Corporate, of 94 Avenue Gambetta, Paris 75020, France, do hereby declare the invention, for which we pray that a patent may be granted to us, and the method by which it is to be performed, to be particularly described in and by the following statement:—

The present invention concerns apparatus for protecting the information in a virtual memory system in programmed data processing apparatus.

Several schemes have been utilized in the past in order to protect information. Some of them are detailed by Robert M. Graham in a paper entitled "Protection in an Information Processing Utility", published in CACM (May 1968).

This type of memory protection is inadequate for present day multiprogramming systems because there is no provision for gradations of privilege or gradations of accessability, and severely limits the control over access to information. There should be provisions for different access rights to the different types of information. A partial answer to this problems is found in the concept of a memory having a segment as the unit of information to which access is controlled (see Patent Application No. 21630/74, (Serial No. 1,465,344), filed on 15 May 1974). Varying degrees of access to each segment is possible by providing for different types of privileges attached to each segment such as master/slave, write/no-write and execute/non-execute. However, this method of protecting the privacy and integrity of information does not take into account the user of the information. Under this type of protection, privilege is not accorded the user but the information being protected. Hence a user if he has access at all to a segment has access similar to all other users who have access to the segment. David C. Evans and Jean Yves LeClerc in a paper entitled "Address Mapping and the Control of Access in an Interactive Computer," SJCC 1967, recognized the problem and attempted a solution. Evans and LeClerc said in that article p. 23, "The user of a computing system should be able to interact arbitrarily with the system, his own computing processes, and other users in a controlled manner. He should have access to a large information storage and retrieval system called the file system. The file system shoufd allow access by all users to information in a way which permits selectively controlled privacy and security of information. A user should be able to partition his computation into semi-independent tasks having controlled communication and interaction among tasks. Such capability should reduce the human effort required to construct, debug, and modify programs and should make possible increased reliability of programs. The system should not arbitrarily limit the use of input/output equipment or limit input/output programming by the user". Evans and LeClerc proposed conditioning access rights on the procedure-in-execution. The segment, under their proposal, is still the unit of information to which access is controlled; however, a segment's access control attributes are recorded substantially in a user-name versus procedure tables whose entries are the access modes. Such a solution, however, has serious drawbacks. For one, the construction and updating of each segment's table of access control attributes presents a formidable task. For another, too many uses of the segment and event occurrences must be foreseen. To overcome this problem access control by procedure-set was suggested. Under this suggestion, related procedures are grouped into "sets of procedures" and access rights to segments is based on the identity of the set to which the procedure seeking access

belongs. This method alleviated the problem of constructing and updating each segment's voluminous tables of access control attributes, but introduced the problem of determining to which set a given procedure belonged, particularly when a procedure was or could be a number of many sets. This ambiguity in defining sets, and the possible transitions between sets makes the implementation of access control based on "sets of procedures" extremely difficult.

To overcome the difficulties encountered with the "set" technique a ring concept was developed. The ring concept groups the sets of procedures into rings that can unambiguously be ordered by increasing power or level of privilege. By assigning a collection of sets to a collection of concentric rings, and assigning numbers to each ring with the smallest ring having the smallest number and each succeeding larger ring having a progressively greater number, different levels of privilege can then be unambiguously assigned to the user of a segment. Under this concept the innermost ring having the smallest number assigned to it has the greatest privilege. Hence it can be postulated that users in the lowest ring number can access information having higher ring numbers, but users in a higher ring number cannot access information having lower ring numbers or can access information in a lower ring number only in a specified manner. This palpable change of power or level of privilege with a change in rings is a concept which overcomes the objections associated to a change of sets.

Multics (*Multi*plexed *I*nformation and *C*omputing *S*ervice) is an operating system developed primarily by Massachusetts Institute of Technology, in cooperation with General Electric Co. and others which first utilized the ring theory of protection in software on a converted Honeywell 635 (Registered Trade Mark) computer and later on a Honeywell 645 (Registered Trade Mark) computer. The Multics philosophy utilizes 64 rings of protection numbered as rings 0—63 and is set forth generally in a paper entitled "Access Control to the Multics Virtual Memory" published by Honeywell Information Systems Inc. in the Multics Technical Papers, Order No. AG95, Rev. 0. A more detailed description of Multics ring protection is to be found on chapter 4 of a book entitled "The Multics System: An Examination of its Structure", by Elliott I. Organick, published by MIT Press, and also in the Multics System Programmers Manual 1969, MIT Project MAC. Briefly, the Multics system does not utilize a "pure ring protection strategy" but rather employs the "ring bracket protection strategy" wherein a user's access rights with respect to a given segment are encoded in an access-mode and a triple of ring number (r1, r2, r3) called the user's "ring brackets" for a given segment. A quotation from pages 137—139 from the Multics Technical Paper entitled, "Access Control to the Multics Virtual Memory" sets out the rules and conditions for using and changing rings.

This "ring protection concept" was first implemented with software techniques utilizing 64 separate rings. Subsequently an attempt was made to define a suitable hardware base for ring protection. The Honeywell 645 (Registered Trade Mark) computer represents a first such attempt. The Honeywell 645 (Registered Trade Mark) system differs from the "ringed hardware" concepts described supra in several respects which when taken together, add up to the fact that the Honeywell 645 (Registered Trade Mark) is a 2-ring rather than a 64-ring machine, and has in lieu of a "ring register", a master mode and a slave mode, which imparts greater power to the processor when in master mode than when in slave mode. "The access control field of the 645's SDW (segment descriptor word) contains no information about rings; in particular its does not contain ring brackets. It does, however, contain either:

a) access-mode information possibly including either of the two descriptors;
accessible in master mode only,
master mode procedure;
b) the specification of one of eight special 'directed' faults (traps) which is to occur whenever the segment descriptor word (SDW) is accessed.

"The procedure is only 'in master mode' when executing a procedure whose SDW indicates a 'master mode procedure'. The processor may enter master mode while executing a slave mode procedure by:
faulting,
taking an interrupt".

"The 645 processor's access control machinery interprets the SDW during the addressing cycle and causes the appropriate action to occur depending on the SDW and (usually) on the attempted access, as follows:

a. If the SDW implies a particular "directed fault", then that fault occurs.

b. Otherwise, if the SDW does not permit the attempted access, the appropriate access violation fault occurs.

c. Otherwise, the SDW permits the attempted access and the access is performed.

"When a fault occurs, the 645 enters master mode and transfers control to the

appropriate master mode fault handling procedure". (Access Control to the Multics Virtual Memory, supra pps. 157—158).

Another paper by Michael D. Schroeder and Jerome H. Saltzer entitled "A Hardware Architecture for Implementing Protection Rings" published in Communications of the ACM, March 1972 Vol. 15, No. 3, sets forth background and theory of ring protection and describes a hardware implementation of "ring protection".

Because the Multics and Honeywell 645 version of ring protection was implemented mainly in software, considerable operating system supervisor overhead was entailed particularly when calls to greater or lesser power were made by trapping to a supervisor procedure. What was required was an access control mechanism which had the functional capability to perform effectively its information protection function, was relatively simple in operation, was economic to build, operate and maintain, and did not restrict programming generality. The Honeywell 6000 (Registered Trade Mark) computer system met these requirements by implementing most of the ring protection mechanism in hardware. Hence special access checking logic, integrated with the segmented addressing hardware was provided to validate each virtual memory reference, and also some special instructions for changing the ring of execution. However certain portions of the ring system particularly outward calls and returns or calls to a lesser power and returns therefrom presented problems which required the ring protection function to be performed by transferring control to a supervisor. What is now needed are further improvements in hardware and techniques that will permit a full implementation of ring protection in hardware/firmware and will meet the criteria of functional capability, economy, simplicity and programming generality.

Accordingly the present invention has for an object to provide an improved computer ring protection mechanism.

Accordingly the present invention consists in an internally programmed data processing apparatus CPU having a virtual memory system, and being responsive to internally stored instruction words for processing information and having stored in said virtual memory system a plurality of different types of groups of information each information group-type associated with an address space bounded by a segment having adjustable bounds, and comprising means for protecting the information in said-virtual memory system from unauthorized users by restricting accessability to the information in accordance to levels of privilege, said means comprising in combination with an access checking mechanism:

(a) first means arranged in operation to store in said virtual memory system at least one segment table comprising a plurality of segment descriptors with each segment descriptor being associated with a predetermined one of said segments and each segment descriptor having a predetermined format containing an access information element and a base address element in predetermined positions of said format, said base address element being used for locating in said virtual memory system the starting location of a selected one of said segments, and said access information element for specifying the minimum level of privilege required for a predetermined type of access that is permitted in a selected one of said segments;

(b) a plurality of second means having a predetermined format, communicating with said first means, arranged to store in a predetermined portion of said second means, a segment number SEG for identifying a segment table and the location of a segment descriptor within said segment table, said second means also being arranged to store in a predetermined other portion of said second means, an offset address within the segment identified by said segment descriptor said offset address locating from said segment base the first byte of a word within said segment;

(c) third means responsive to an address syllable element of an instruction being executed for addressing one of said plurality of second means;

(d) fourth means arranged to store a displacement from said address syllable;

(e) fifth means, communicating with said first, second, third and fourth means, arranged to add the displacement D and said base address to said offset; and,

(f) sixth means responsive to said access information element in a selected one of said segment descriptors, restricting the accessability to the segment associated with said selected one of said segment descriptors in accordance to the level of privilege and the type of access specified in said access information element, wherein each group-type of information is associated with a predetermined ring number indicative of a level of privilege said legel of privilege decreasing as the associated ring number increases comprising means for determining the maximum effective address ring number EAR (i.e. minimum level of privilege) of a selected process to access a selected group of information, said means comprising;

(a) first means to store first information indicating the maximum ring number RD (i.e. minimum level of privilege) required to read information from said selected group;

(b) second means to store second information indicating the maximum ring number WR (i.e. minimum level of privilege) required to write information into said selected group;

(c) third means to store third information indicating the maximum ring number MAXR (i.e. minimum level of privilege) required to process information from said selected group; and,

(d) fourth means communicating with said first, second and third means, to determine the maximum of the contents of said first, second and third means whereby the effective address ring number EAR is generated.

The present invention, however, both as to organization and operation thereof may best be understood by reference to the following description which is given by way of example in conjunction with the accompanying drawings in which:

Figure 1 is a block diagram of a computer system utilizing the invention.

Figure 2 is a schematic diagram illustrating the levels of privilege of the invention.

Figure 3 is a flow diagram of the segmented address scheme utilized by the invention.

Figures 4A—4J are schematic diagrams of various novel hardware structures utilized in the invention.

Figure 5 is a schematic diagram of the computer ring protection hardware.

Figure 6 is a schematic diagram of the computer segmented addressing hardware.

Figures 7a—7h and Figures 8a—8c are detailed logic block diagrams of the ring protection hardware.

Figures 9a—9k is a legend of the symbols utilized in the diagrams of the invention.

Figure 10 is a schematic diagram of three stack segments, one each for ring 0, 1 and 3 respectively.

Figure 11A shows the format of the Enter Procedure instruction.

Figure 11B shows the format of a procedure descriptor.

Figure 11C shows the format of a gating procedure descriptor GPD the first word of the segment containing the procedure descriptors.

Figure 11D shows the format of the Exit Procedure instruction.

Figure 12 is a flow diagram of a portion of the Enter Instruction pertaining to ring crossing and ring checking.

Figure 13 schematically shows a segment descriptor and the segment containing procedure descriptors.

Figures 14—16 are flow diagrams showing various operations that are performed when the Enter Procedure instruction is executed.

Figure 17 is a flow chart of the Exit Instruction.

As previously discussed the ring concept of information protection was originated on MULTICS and implemented on various Honeywell (Registered Trade Mark) Computer Systems. The original MULTICS concept required 64 rings or level of privilege and later implementation had the equivalent of two rings on the Honeywell 645 and 8 rings on the Honeywell 6000 (Registered Trade Mark). The embodiment described herein groups data and procedure segments in the system into a hierarchy of 4 rings or classes. (Refer to Figure 2). The 4 rings or privilege levels are identified by integers 0—3; each ring represents a level of privilege in the system with level 0 having the most privilege and level 3 the least. Level 0 is known as the inner ring and level 3 as the outer ring. The basic notion as previously discussed is that a procedure belonging to an inner ring has free access to data in an outer ring. Conversely a procedure in an outer ring cannot access data in an inner ring without incurring a protection violation exception. Transfer of control among procedures is monitored by a protection mechanism such that a procedure execution in an outer ring cannot directly branch to a procedure in an inner ring. This type of control transfer is possible only by execution of a special "procedure-call" instruction. This instruction is protected against misuse in a number of ways. First, a gating mechanism is avilable to ensure that procedures are entered only at planned entry points called gates when crossing rings. The segment descriptor of such a procedure contains a gate bit indicating that procedures in this segment can be entered only via gates; information regarding these gates is contained at the beginning of the segment and is used by the hardware to cause entry at a legal entry-point. The procedure itself must then verify (in a way which, of necessity depends on the function of the procedure) that it is being legitimately called. A further hardward protection mechanism is available in the case that the calling procedure supplies an address as a parameter; it is then possible that the more privileged procedure would invalidly modify information at this address which the less privileged caller could not have done, since the ring mechanism would have denied him access; an address validation instruction is available to avoid this possibility.

An important convention is required

here in order to protect the procedure call mechanism. This states that it is not in general permissible to use this mechanism to call a procedure in a less privileged ring and return to the more privileged one. This restriction is necessary since there is no assurance that the procedure in the higher ring will, in fact, return; that it will not, accidentally or maliciously, destroy information that the more privileged procedure is relying upon; or that it will not, accidentally or maliciously, violate the security of the stack (see GLOSSARY for definition). Any of these could lead to unpredictable results and crash the system.

The level of privilege are quite independent of the process control mechanism and there is no notion here of privileged and non-privileged processes as in the IBM system 360 (Registered Trade Mark). Instead the same process can execute procedures at different levels of privilege (rings) subject to the restrictions imposed by the ring mechanism. In this sense the ring mechanism can be viewed as a method for subdividing the total address space assigned to a process according to level of privilege.

The ring mechanism defined herein permits the same segment to belong to up to 3 different rings at the same time i.e. there are 3 ring numbers in each segment descriptor, one for each type of possible access. Thus the same segment can be in ring one with respect to "write" access, ring two with respect to "execute" access and ring three with respect to "read" access. One obvious use for this is in the case of a procedure segment which can be written only by ring zero (perhaps the loader) but can be executed in ring three.

Of the four available rings, two are allocated to the operating system and two to users. Ring zero, the most privileged ring, is restricted to those operating system segments which are critical to the operation of the whole system. These segments form the hard core whose correctness at all times is vital to avoid disaster. Included would be the system information base, those procedures dealing with the organisation of physical memory or the initiation of physical data transfer operations, and the mechanisms which make the system function, like the "exception supervisor, the scheduler, and the resource management".

Ring one contains a much greater volume of operating system segments whose failure would not lead to catastrophe but would allow recovery. Included herein are the language translators, data and message management, and job and process management. Through the availability of two rings for the operating system, the problem of maintaining system integrity is made more tractable, since the smaller hard core which is critical is isolated and can be most carefully protected.

Rings two and three are available to the user to assign according to his requirement. Two important possibilities are debugging and proprietary packages. Programs being debugged may be assigned to ring two while checked out programs and data with which they work may be in ring two; in this way the effect of errors may be localized. Proprietary programs may be protected from their users by being placed in ring two while the latter occupy ring three. In these and other ways, these two rings may be flexibly used in applications.

The General Rules of the Ring System

1. A procedure in an inner ring such as ring 2 on Figure 2 has free access to data in an outer ring such as ring 3 and a legal access (arrow 201) results. Conversely a procedure in an outer ring such as ring 3 cannot access data in an inner ring such as ring 2 and an attempt to do so results in an illegal access (arrow 202).

2. A procedure in an outer ring such as ring 3 can branch to an inner ring such as ring 1 via gate 204 which results in a legal branch 203, but a procedure operating in an inner ring such as ring 2 may not branch to an outer ring such as ring 3.

3. Each segment containing data is assigned 2 ring values, one for read (RD) and one for write (WR). These ring values specify the maximum ring value in which a procedure may execute when accessing the data in either the read or write mode.

Each time a procedure instruction is executed, the procedure's ring number (effective address ring, EAR) is checked against the ring numbers assigned to the segment containing the referenced data. The EAR is the maximum number of process ring numbers in the processor instruction counter (see later description) and all ring numbers in base registers and data descriptors found in the addressing path. Access to the data is granted or denied based on a comparison of the ring numbers. For example, if a system table exists in a segment having a maximum read/ring value of 3 and a maximum write/ring value of 1, then a user procedure executing in ring 3 may read the table but may not update the table by writing therein.

Procedure Calls and the Stack Mechanism:

The procedure call and stack mechanism is an apparatus being described herein Procedure calls are used to pass from one procedure to another; to allow user procedures to employ operating system services: and to achieve a modular

structure within the operating system. A procedure call is effected by instructions and a hardware recognized entity called a stack.

A stack is a mechanism that accepts, stores and allows retrieval of data on a last-in-first-out basis. Stacks reside in special segments called stack segments. A stack segment consists of a number of contiguous parts called stack frames which are dynamically allocated to each procedure. The first stack frame is loaded into the low end of the segment and succeeding frames are loaded after it. The last frame loaded is considered the top of the stack. A T-register 114 (see Figure 1) locates the top of the stack for the currently active process. A virtual T-register exists in the process control block (PCB) of all other processes in the system.

A stack frame consists of three areas: a work area in which to store variables, a save area in which to save the contents of registers, and a communications area in which to pass parameters between procedures. Prior to a procedure call, the user must specify those registers he wishes saved and he must load into the communications area the parameters to be passed to the called procedure. When the call is made, the hardware saves the contents of the instruction counter and specified base registers to facilitate a return from the called procedure.

Each procedure call creates a stack frame within a stack segment and subsequent calls create additional frames. Each exit from one of these called procedures causes a stack frame to be deleted from the stack. Thus, a history of calls is maintained which facilitates orderly returns.

To ensure protection between procedures executing in different rings, different stack segments are used. There is one stack segment corresponding to each protection ring per process. A process control block (PCB) contains three stack base words (SBW) which point to the start of the stack segment for rings 0, 1 and 2 associated with the process. The ring 3 stack segment can never be entered by an inward call; therefore, its stack starting address is not required in the PCB.

The procedure call is used by users who have written their programs in a modular way to pass from one program module to another. It is used by user programs to avail themselves of operating system services. It is used by the operating system itself to achieve a responsive modular structure. The procedure call as is described in the above referenced patent application is effected by hardware instructions and the hardware recognizable stack mechanism.

The main requirements on a procedure call mechanism are:

1. Check the caller's right to call the caller;
2. Save the status of the caller which includes saving registers, instruction counter (for return), and other status bits;
3. Allow for the passing of parameters;
4. Determine valid entry point for the called procedure;
5. Make any necessary adjustments in the addressing mechanism;
6. Enter the new procedure.

When the called procedure terminates or exits, whatever was done in the call must be undone so that the status of the calling procedure is restored to what it was before the call.

As a preliminary to making a procedure call, the instruction PREPARE STACK is executed. This instruction causes those registers specified by the programmer in the instruction to be saved in the stack. It causes the status register (see Figure 1) to be saved, and provides the programmer with a pointer to parameter space which he may now load with information to be passed to the called procedure.

Another instruction ENTER PROCEDURE permits the procedure call via the following steps corresponding to the requirement specified above:

1. Ring checking—the caller's ring is checked to make sure that this ring may call the new procedure; the call must be to a smaller or equal ring number; and if ring crossing does occur the new procedure must be gated through a gate 204 of Figure
2. The new ring number will then be that of the called procedure.
2. The instruction counter is saved;
3. Base register 0 (see Figure 1) is made to point effectively to the parameters being passed;
4. The entry-point of the called procedure is obtained from a procedure descriptor whose address is contained in the ENTER PROCEDURE INSTRUCTION;
5. A point to linkage information is loaded in base register number 7.
6. The new procedure is entered by loading the new ring number and the address of the entry-point in the instruction counter.

The remainder of the current stack-frame is also available to the called procedure for storage of local variables.

When the called procedure wishes to return, it executes the instruction EXIT PROCEDURE. The registers and the instruction counter are then restored from their saving areas in the stack.

Referring to Figure 1 there is shown a block diagram and a computer hardware

system utilizing the invention. A main memory 101 is comprised of four modules of metal-oxide semi-conductor (MOS) memory. The four memory modules 1—4 are interfaced to the central processor unit 100 via the main store sequencer 102. The four main memory modules 1—4 are also interfaced to the peripheral subsystem such as magnetic tape units and disk drive units (not shown) via the main store sequencer 102 and the 10C (not shown). The main store sequencer gives the capability of providing access to and control of all four memory modules.

Operations of the CPU are controlled by a read only memory ROM, herein called the control store unit 110.

The control store interface adapter 109 communicates with the control store unit 110, the data mangagement unit 106, the address control unit 107 and the arithmetic logic unit 112 for directing the operation of the control store memory. The control store interface adapter 109 includes logic for control store address modification, testing, error checking, and hardware address generation. Hardware address generation is utilized generally for developing the starting address of error sequencers or for the initialization sequence.

The buffer store memory 104 is utilized to store the most frequently used or most recently used information that is being processed by the CPU.

The data management unit 106 provides the interface between the CPU 100 and main memory 101 and/or buffer store memory 104. During a memory read operation, information may be retrieved from main memory or buffer store memory. It is the responsibility of the data management unit to recognize which unit contains the information and strobe the information into the CPU registers at the proper time. The data management unit also performs the masking during partial write operations.

The instruction fetch unit 108 which interfaces with the data management unit 106, the address control unit 107, the arithmetic and logic unit 112 and the control store unit 110 is responsible for keeping the CPU 100 supplied with instructions.

The address control unit 107 communicates with the instruction fetch unit 108, the buffer store directory 105, the main store sequencer 102, the arithmetic logic unit 112, the data management unit 106, and the control store unit 110 via the control store interface adapter 109. The address control unit 107 is responsible for all address development in the CPU.

Interfacing with the address control unit 107, the instruction fetch unit 108 and the control store unit 110 is the arithmetic logic unit 112 which is the primary work area of the CPU 100. Its primary function is to perform the arithmetic operations and data manipulations required of the CPU.

Associated with the arithmetic logic unit 112 and the control store unit 110 is the local store unit 111 which typically is comprised of a 256-location (32 bits per location) solid state memory and the selection and read/write logic for the memory. The local store memory 111 is used to store CPU control information and maintain ability information. In addition, the local store memory 111 contains working locations which are primarily used for temporary storage of operands and partial results during data manipulation.

The central processing unit 100 typically contains 8 base registers (BR) 116 which are used in the process of address computation to define a segment number, an offset, and a ring number. The offset is a pointer within the segment and the ring number is used in the address validity calculation to determine access rights for a particular reference to a segment.

The instruction counter 118 communicates with the main memory local register (MLR) 103 and with the instruction fetch unit 108, and is a 32-bit register which contains the address of the next instruction, and the current ring number of the process (PRN). Also contained in the central processing unit is a T register 114 which also interfaces with the instruction fetch unit 108 and is typically a 32-bit register containing a segment number and a 16-bit or 22-bit positive integer defining the relative address of the top of the procedure stack. The status register 115 is an 8-bit register in the CPU which among other things contains the last ring number—i.e. the previous value of the process ring number (PRN).

The main memory 101 is addressed by the memory address register (MAR) 119, and the information addressed by (MAR) 119 is fetched and temporarily stored in the memory local register (MLR) 103.

Referring now to Figure 3 there is shown a flow diagram of the general rules for segmented address development shown in detail in the above mentioned copending patent application No. 21630/74, Serial No. 1,465,344. Figure 3 when read in conjunction with the above referenced patent application is self-explanatory. There is however one major difference between the address development as shown on Figure 3 to that of the above mentioned application and that is that in the address development of Figure 3 of the instant application as many as 16 levels of

indirection may be utilized in the address development whereas in the above referenced application the levels of indirection were limited to a maximum of two. This of course is a matter of choice with the designer and in no way alters the high level inventive concept.

Referring now to Figures 4A—4J, Figures 4A and 4B show the format of the instruction counter designated by reference numeral 118 on Figure 1. The instruction counter (IC) 118 is a 32-bit register which contains the address of the next instruction, and the current ring number of the process (PRN). Referring specifically to Figures 4A and 4B the TAG is a 2-bit field which corresponds to the TAG field of data descriptors shown and described in the above reference application entitled "Segmented Address Development". PRN is a 2-bit field which defines the current ring number of the process to be used in determination of access rights to main storage. SEG is typically either a 12-bit or a 6-bit field which defines the segment number where instructions are being executed. The OFFSET is typically either a 16-bit or a 22-bit field which defines the address of the instruction within the segment SEG.

Figures 4C—4F show the format of segment descriptors with Figures 4C and 4D showing the first and second word of a direct segment descriptor whereas Figures 4E and 4F show the first and second word of an indirect segment descriptor. Segment descriptors are two words long each word comprised of 32 bits. Referring to Figures 4C—4D which show the first and second word respectively of a direct segment descriptor, P is a presence bit. If P equals one, the segment defined by the segment descriptor is present in main storage. If P equals zero, the segment is not present and a reference to the segment descriptor causes a missing segment exception. All other fields in a segment descriptor have meaning only if P equals one. A is the availability bit. If A equals zero, the segment is unavailable (or locked) and a reference to the segment causes an unavailable segment exception. If A equals one, the segment is available (or unlocked, and can be accessed). I is the indirection bit. If I equals zero, the segment descriptor is direct. If I equals one, the segment descriptor is indirect. U is the used bit. If U equals zero, the segment has not been accessed. If U equals one, the segment has been accessed. U is set equal to one by any segment access. W is the written bit. If W equals zero, no write operation has been performed on the segment. If W equals one, a WRITE operation has been performed on the segment. W is set to one by any WRITE operation. GS is the gating-semaphore bits. When the procedure call mechanism referred to above requires that the segment be a gating segment or when the process communication mechanism (not shown) requires that the segment be a segment descriptor segment (SD) the GS bits are examined. To be a valid gating segment, the GS bits must have the value 10. To be a valid SD segment, the GS bits must have the value 01. If a gating or SD segment is not required, these bits are ignored. The BASE is a 24-bit field which defines the absolute address in quadruple words of the first byte of the segment. This field is multiplied by 16 to compute the byte address of the segment base. The SIZE is a field which is used to compute the segment size. If the segment table number, subsequently referred to as STN, is greater or equal to zero but less than or equal to six, the SIZE field is 18 bits long. The STN is a field indicating the segment table entry STE for selecting a segment descriptor. If the STN is greater than or equal to 8 but less than or equal to 15, the SIZE field is 12 bits long. The number of bytes in the segment is equal to 16 times (SIZE+1). If SIZE equals zero, the segment size is 16 bytes. RD is the read access field. This is a 2-bit field which specifies the maximum EAR (effective address ring number) for which a read operation is permitted on the segment. (A procedure is always permitted to read its own segment if EAR equals PRN). WR is the write access field. This is a 2-bit field which specifies the maximum EAR for which a write operation is permitted on the segment and the minimum PRN at which the segment may be executed. MAXR is the maximum ring number. This is a 2-bit field which specifies the maximum PRN at which the segment may be executed. WP is the write permission bit. This bit indicates whether a WRITE operation may be performed on the segment. If WP equals zero, no WRITE operation may be performed. If WP equals one, a WRITE operation may be performed if EAR is greater than or equal to zero but less than or equal to WR. EP is the execute permission bit. This bit specifies whether the segment may be executed. If EP equals zero, the segment may not be executed. If EP equals one, the segment may be executed at any PRN for which PRN is greater than or equal to WR but less than or equal to MAXR. MBZ is a special field which must be set to zero by software when the field is created, before its initial use by hardware.

Referring to Figures 4E—4F the definitions of the various fields are similar as above however word 0 includes a LOCATION field and word 1 includes a

RSU field. The LOCATION field is a 28-bit field which defines the absolute address of a direct segment descriptor. The value in the LOCATION field must be a muliple of 8. The RSU field is a special field which is reserved for software use.

Figures 4G—4H show the format of the base registers (BR) which are used in the process of address computation to define a segment table number, a segment table entry number, an offset, and a ring number. There are typically 8 base registers as shown by reference numeral 116 on Figure 1. A base register is specified or identified as base register 0 through 7. The size of a base register is 32 bits long. The base register format of Figure 4G is utilized for small segment i.e. where STN is greater or equal to 8 but less than or equal to 15, whereas the format of base register of Figure 4H is utilized for large segments i.e. STN is greater or equal to zero but less than or equal to six. Referring to Figures 4G—4H, TAG is a 2-bit field which corresponds to the TAG of a data descriptor referenced previously. RING is a 2-bit field which contains the ring number associated with the segmented address for protection purposes. SEG is a field previously referred to, which identifies a segment described in a segment table. STN is the segment table number, and STE is the segment table entry number. OFFSET is a 16-bit field or a 22-bit field depending on segment table number, which defines a positive integer. The OFFSET is used in the process of address development as a pointer within a segment.

Referring to Figures 4I—4J there is shown the format of the T-register. The T-register is a 32-bit register containing a segment number and a 16-bit or 22-bit positive integer defining the relative address of the top of the procedure stack previously mentioned. The T-register is shown by reference numeral 114 on Figure 1. The various fields of the T-register have the same definition as described above.

Referring now to Figures 3 and 4A—4J a more defined description of absolute address calculation and access checking is made. In general absolute address calculation consists of fetching a segment descriptor specified by STN and STE and using the segment descriptors in four ways: access checking, computation of the absolute address, bound checking, and updating (U and W flags). As described in copending patent application No. 21630/74, (Serial No. 1,465,344) the absolute address may be direct or indirect and is derived by first deriving an effective address from STN, STE, and SRA (segment relative address). STN is extracted from bits 4 through 8 of the base register BR specified in the address syllable of an instruction. If STN is 7, an out of segment table word array exception is generated. STE is extracted from the base register specified in the address syllable. If STN 4:4 (i.e., beginning at bit 4 and including the next 4 bits) is greater than or equal to zero or less than or equal to six, STE is in a base register bits 8 and 9. If STN 4:4 (i.e. 4 bits beginning at bit 4) is greater than or equal to 8 but less than or equal to 15, STE is in a base register BR bits 8 through 15. The segment relative address SRA for direct addressing is computed by adding the displacement in the address syllable; the offset of the base register BR; and the 32-bit contents of an index register, if specified in the address syllable. The sum of these three quantities is a 32-bit unsigned binary integer which must be less than the segment size appropriate to the segment STN, STE.

Indirect addressing is developed by fetching a data descriptor and developing an address from that descriptor. The effective address of the data descriptor is computed as in the direct addressing case with the exception that the index register contents are not used. In developing the address from the data descriptor the effective address may be computed by an indirection to segment ITS descriptor and an indirection to base ITBB descriptor. If the descriptor is ITS the STN and STE are extracted from the descriptor in the same manner as from a base register. SRA is computed by adding the displacement in the descriptor and the contents of an index register as specified in the syllable. If the descriptor is an ITBB descriptor then STN and STE are extracted from the base register specified in the BBR field (i.e. the base register implied by ITBB descriptor) of the descriptor as in direct addressing. SRA is computed by adding the displacement in the descriptor, the offset of the base register, and the contents of an index register is specified in the address syllable.

As shown on Figure 3 the indirection process may be extended up to 16 levels.

Every effective address contains protection information which is computed in address development and checks for access rights by the ring protection hardware of the absolute address calculation mechanism. The effective address contains protection information in the form of an effective address ring number EAR (see Figures 2J and 2K of above application No. 21630/74, (Serial No. 1,465,344). The EAR is computed from the base register ring number BRN and from the current process ring number PRN by taking the maximum ring number. In developing the EAR for indirect addressing

a somewhat more tedious but essentially similar procedure as indirect addressing is used. In indirect addressing the EAR for extraction of the first descriptor (EAR 1) is once again the maximum of the ring number from the base register specified in the address syllable and the current process ring number PRN in the instruction counter 115 of Figure 1 and stored in 00 register 512 of Figure 5. The EAR for extraction of the second descriptor (EAR 2), of multiple level indirection is the maximum of:

a. EAR 1;

b. The ring number in the first descriptor if indirection is indirection to segment;

c. The ring number from a base register 116 utilized as a data base register BBR if the first descriptor is an indirection to segment descriptor ITBB.

The EAR for extraction of the data of multiple level indirection is the maximum of:

a. EAR 2;

b. The ring number in the second descriptor if it is an indirection segment descriptor ITS;

c. The ring number in one of the base registers utilized as a data base register BBR if the second descriptor is an indirection to base descriptor ITBB.

Referring now to Figures 5 and 6, the transfers and manipulation of the various type ring numbers will be described at the system level. Detailed logic block diagrams for effecting the transfers and operations of Figure 5 will be later described. Referring first to Figure 6 an associative memory 600 is utilized in segmented address development. The associative memory 600 comprises essentially a UAS associator 609 which has circuitry which includes associative memory cells, bit sense amplifiers and drivers, and word sense amplifiers and drivers (not shown). A word or any part of a word contained in UAS associator 609 may be read, compared to another word with a match or no match signal generated thereby, or be written either in whole or in a selected part of the associator 609. For example, US register 607 may contain a segment number which may also be in the associative memory 600. A comparison is made with UAS associator 609 and if a match is found a "hit" results. The match or "hit" signal is provided to encoder 610. The function of encoder 610 is to transform the "hit" signal on one of the match lines to a 4 bit address. Encoder 610 provides this 4 bit address to UAB associator buffer 611 so that the information contained in that particular location of UAB associator buffer 611 is selected. Information in UAB associator buffer 611 may be transferred to UV register 613 for temporary storage or for transfer to QA or QB bus 614 and 615 respectively. By thus locating a prestored segment number of the associative memory 600 (which may have been placed there after a generation of an absolute address) regeneration of the same address is not necessary. In the drawing of Figure 6, UAB associator buffer 611 is shown as storing a first and second word of a segment descriptor; however other types of information may just as well be stored therein. This buffer 611 provides a function similar to that of buffer 104 in the more generalised diagram of Figure 1.

As mentioned supra the development of an absolute address of an operand from an effective address is disclosed in patent application No. 21630/74, (Serial No. 1,465,344). Briefly and with reference to Figure 6 any of 8 base registers 602 are addressed via UG and UH registers 603 and 604 respectively which contain base register addresses from an instruction address syllable or base register specified by the instruction formats. The base register 602 contain such information as TAG, base register ring number BRN, segment table number STN, segment table entry STE and OFFSET as shown or contained by base registers 1 and 2 of the group of base registers 602. Writing into the base registers is performed under micro-op control by UWB logic 601. For example it is shown that information from the UM register 502 of Figure 5 may be written into bit positions (2, 3) of a selected base register; also information from the QA bus may be written into the base registers and provisions are made to clear a selected base register i.e. write all zeroes. Reading out of any of the base registers is performed by UBR logic 605. In general the UBR logic 605 permits the appropriate base register to be strobed out onto bus QA or QB, or into UN register 608. Note that UN register 608 holds bits 8 through 31 of the base registers which is the OFFSET part of the segmented address. Moreover UBR logic 605 when addressed by an address contained in instruction buffer IB (not shown) reads out the segment number SEG (which is comprised of STN and STE) into US register 607 via UBS transfer logic 606. The comparison of the segment number SEG in US register 607 with the associative memory 600 may then be performed as previously described. It will be noted that bits (4—15) of QA bus 614 may also be read into or from US register 607. Similarly bits (8—31) from QA bus 614 may read into UN register 608. Also bits (9—11) of the US register 607 may be read into QA bus 614 as denoted by US (9—11) arrow (the arrows into various register and/or logic circuitry denote the source of data and that followed

by a number denote the bit numbers of that data).

Referring now to Figures 5 and 6, a 2-bit UP register 501 stores the current process ring number PRN. The current process ring numbers PRN is obtained from bits 2 and 3 of the instruction counter (118 or Figure 1) via bits IC (2—3) of the QA bus 614 of Figure 6. Bits IC (2—3) of QA bus 614 are transferred to 2-bit UV register 503 under control of a micro-operation UV9QA0. The micro-operations are obtained from micro-instructions in the control store unit 110. (On Figure 5 the dot surrounded by a circle indicates a micro-operation and the first two letters of the name of the micro-operation indicate the destination of the data to be transferred; the fourth and fifth letters indicate the source of the data transferred; the third character indicates whether a full or partial transfer is made with F indicating a full transfer while the sixth character indicates whether the signal doing the transferring is high or low with even numbers indicating a low signul and odd numbers indicating a high signal. As an example of the use of this convention bits 2 and 3 on QA bus indicating the tail of the arrow QA (2, 3) indicate PRN is the PRN process ring number that is being transferred under control of the micro-op UV9QA0 which says the transfer is made to register UV, is a partial transfer of the bus QA, and the source of the data is the bus QA and is an unconditional transfer as indicated by the sixth character being 0. Transfer to UV register from QA bus source is unconditional. This 0 will be the corresponding seventh character in the logic file name of the subcommand UV9QA1ϕ. Once the process ring number PRN is transferred from the QA bus 614 to the UV register 503 another transfer takes place under control of the micro-operation UM9UV0 from UV register 503 to UM register 502. Finally another transfer takes place from UM register 502 to UP register 501 under control of a micro-operation UP9UM0.

Two bit register UM 502 is utilized to generate the effective address ring number EAR during ITS and ITBB (i.e. indirection to segment and indirection to base), (EAR=MAX (BRN, PRN, DRN/BBR (BRN) etc.) address formation for address syllable 1 and address syllable 2 type instruction format. The EAR is generated according to the rules previously enunciated by utilizing one or more tests shown in block 510 and the maximum of the ring number is obtained and stored in UM register 502 which stores the effective address ring number EAR (detailed logic or making the comparisons of block 510 are later shown and described in detail). The UO register is used to save address syllable 1 effective address ring number EAR in the event the address syllable 2 is being utilized to extract EAR 2.

Two-bit UV register 503, and 2-bit UW register 504 is utilized mainly as storage for various ring numbers that are obtained from the outside of the ring checking hardware of Figure 5 and transferred or processed to other parts of the ring checking hardware. For example the base register ring number BRN is transferred from bit positions 2 and 3 of UBS transfer logic 606 to UV register 503 under control of the micro-operation UVFBS0; the maximum ring number MAXR of word 2 of the segment descriptor (also shown stored in bits 36 and 37 of UAB associator buffer 611) is transferred from UAB buffer 611 to UV register 503 under control of the micro-operation UVFAB1; also bits 34 and 35 of UAB buffer 611 which is the write ring number WR is transferred to UV register 503 under control of micro-operation UVFAB0. UW register 504 has similar transfers of other ring numbers from various parts of the system. For example bits 34 and 35 which are the write ring number WR of UAB buffer 611 may also be transferred to UW register 504 under control of micro-operation UWFAB1; bits 32 and 33, the read RD ring number of UAB buffer 611 may also be transferred to UW register 504 under control of micro-op UWFAB0; also bits 0 and 1 of QA bus 614 may be transferred to UW register 504 under control of micro-operation UW9QA0. Note also several transfer paths of UW register 504 into UV register 503 under control of the micro-operation UV9UW0; the transfer path of UV register 503 into UM register 502 under control of micro-operation UM9UV0; the transfer path of UM register 502 into UP register 501 under control of the micro-operation UP9UM0; the transfer path of UP register 501 into UM register 502 under control of micro-operation UM9UP0; the transfer path of UM register 502 into UO register 512 under control of micro-operation UO9UM0; and finally the transfer path of UO register 512 into UM register 502 under control of the micro-operation UM9UO0.

Briefly therefore UP register 501 holds the current process ring number PRN; UM register 502 and UO register 512 are utilized for transfer operations and also to generate the EAR; UV register 503 may shore for various purposes and at different times the current process ring number PRN, the base register ring number BRN, the maximum ring number MAXR, the write ring number WR, or the read ring number RD. UW register 504 may at various times hold the read ring number RD, the write ring

number WR, and bits 0 and 1 of bus QA. UMR 505 is logic, the details of which are shown on Figure 8d, which compares the contents of registers UM and UV and produces the greater of the two values in the registers and this value is stored in UM register 502 under micro-operation control UMFMR0. This is one way of generating the effective address ring number EAR. UMR logic 505 may also produce the greater value of the contents of register UP or of bits 2 and 3 of UBS logic 606. This is another method and/or additional step in generating the effective address ring number EAR. UMR logic 505 is also utilized to determine whether or not a write violation has occurred by transferring a write ring number WR into UV register 503 and then comparing the contents of the UM register 502 (holding EAR) with the contents of UV register 503 in order to determine which one has the greater contents. Since UM register 502 stores the effective address ring number EAR a comparison of the UM register and the UV register will indicate whether EAR is greater than WR or vice versa. If WP (i.e. write permission bit in the segment descriptor) is equal to 1 and if EAR lies in the range of $0 \leq EAR \leq WR$ then a write operation may be performed into the segment. Note that UMR logic 505 may have inputs directly or indirectly from all registers 501—504, from other logic 506, 507 and also from UBS logic 606.

UWV logic 506 corresponds to the detail logic of Figure 8a. UWV logic 506 has inputs directly or indirectly from registers 501—504 and from logic 505, 507 respectively and generates an execute violation signal when a comparison of UW, UM and UV registers 504, 502, and 503 respectively indicates that the statements that the maximum ring number MAXR is greater or equal to the effective address ring number EAR, and that EAR is greater or equal to the write ring number WR are not true i.e. in order for a procedure to be able to execute in a given segment indicated by the effective address the maximum ring number MAXR must be greater or equal to the effective address ring number and the effective address ring number EAR must be equal or greater than the write ring number WR. UWV logic 506 also performs tests shown in block 510. Indications may be given that the contents of UW register is less than or equal to the contents of the UV register; the contents of the UM register is greater than or equal to the contents of the UV register; the contents of the UV register is equal to the contents of the UM register; the contents of the UV register is greater or equal to the contents of the UM register; and the contents of the UM register is greater than the contents of the UW register. Of course when performing these tests different values of ring numbers may occupy the registers.

UEP logic 507 corresponds to the detail logic of Figure 8b. UEP logic 507 in combination with UWV logic 506 generates the read violation exception. However the read violation exception may be overridden if the effective address ring number EAR equals the current process ring number PRN, since a procedure is always permitted to read its own segment, and if the segment number of the procedure segment descriptor (not shown herein) and the segment number of the address syllable utilized in generation of the effective address are the same.

To illustrate the overriding of the read violation signal assume that the effective address read number EAR is greater than the read number RD which would generate a read violation high signal which would be applied as one input of AND gate 522. However the read violation exception signal may not be generated even though there is a read violation signal if the following two conditions exists:

1. The effective address ring number EAR is equal to the process ring number PRN; i.e. the contents of register UM is equal to the contents of the register UP; and,

2. The segment number contained in the address syllable of the segment in which a procedure desires to read is equal to the segment number of the procedure segment descriptor (not shown) of the current procedure in execution and this is indicated by setting a bit called a P bit and located as the thirteenth bit of UE register 650. (UE register 650 is a store for the contents of UAS associator 609 when a "hit" has resulted by a comparison of the contents of US register 607). Since this example assumes that EAR equals PRN, UEP logic 507 will apply a high signal to AND gate 520 as one input, and since it is also assumed that the segment number SEG of the address syllable of the segment being addressed is equal to the segment number SEG of the procedure segment descriptor (not shown) of the currently executing procedure, then the P bit of the procedure segment descriptor will be set and hence the other input applied to AND gate 520 will be high thus enabling AND gate 520; a high signal is therefore applied to the input of inverter 521 resulting in a low signal at the output of inverter 521 which low signal is then applied as another input of AND gate 522. Since there is a low signal to AND gate 522 no read violation exception signal can be generated by amplifier 523 even if

the third input signal applied to AND gate 522 is high.

To illustrate how a read violation signal is generated and not overridden, assume that the output of UEP logic 507 indicates that the contents of UM register is not equal to the contents of UP register. Then that input to AND gate 520 would be low and hence AND gate 520 would not be enabled and its output would be low and would be applied to the input of inverter 521. Since the input of inverter 521 is low its output would be high which would be applied as one input of AND gate 522. If also the effective address ring number EAR is greater than the read ring number RD (i.e. contents of UM register is greater than contents of UW register) that signal would be high and would be also applied to another input of AND gate 522. AND gate 522 has still a third input which must also be high in order to enable AND gate 522. This third input is high when AND gate 526 is enabled. Since AND gate 526 has one input terminal which is high when the 00 terminal of URV1F flop 524 is low, AND gate 526 is enabled by applying the micro-operation read violation interrogate signal AJERVA to one input terminal of AND gate 526 while the 00 terminal of URV1F flop 524 is low. Thus AND gate 522 will have all input terminals high, generating the read violation exception signal.

The execute violation exception is generated in two ways. It was seen earlier that an execute violation signal results when UWV logic 506 indicates that the inequalities WR is less than or equal to EAR, and EAR is less than or equal to MAXR are not true. This high execute violation signal is applied to a one-legged AND gate 550 which in turn is applied to the input terminal of two-legged AND gate 553 via amplifier 552. When an execute violation interrogate micro-operation signal AJEEVA is applied as another input of two-legged AND gate 553, this gate is enabled which in turn generates the execute violation exception via amplifier 554. The other method by which the execute violation exception is generated by the execute violation hardware 511 is when the execute permission bit EP is not set. When this condition is true it is indicated by the seventh bit of UY register 613 being high; this bit is then applied to the input terminal of one-legged AND gate 551 which is applied as a high signal to one input terminal of AND gate 553 via amplifier 552. When the execute violation interrogate micro-operation signal AJEEVA goes high, AND gate 553 is enabled and generates an execute violation exception via amplifier 554.

The write violation exception is also generated in two ways. It was seen previously how the UMR logic 505 generates a write violation signal when EAR is greater than WR. This write violation signal is applied to one input terminal of AND gate 545. AND gate 545 is enabled when its second input terminal goes high thus generating a write violation exception through amplifier 547. The second input terminal of AND gate 545 goes high when AND gate 542 is enabled. AND gate 542 is enabled when the input signals applied to its input terminals are high. One input signal is high when UWV1F flop 541 is low which in turn applies a low signal to the input terminal of inverter 543 which in turn applies a high signal to one input terminal of AND gate 542; the other input signal is high when the write violation interrogate micro-op signal AJEWVA is high and this happens when it is desired to interrogate a procedure for the write violation exception. (Flip-flops URV1F, URN1F, and UWV1F are set low when any interrupts or softward occurs). (UWV2F, URV2F, and URN2F flip-flops are utilized to store back-up excess checking information for ring checking). The other method for generating a write violation exception is when the write permission bit WP is not set. This condition is indicated by bit 6 of UV register 613 being high. When this condition exists and the high signal (i.e. the sixth bit of UV register) is applied as one input of AND gate 546 and the interrogate signal

AJEWVA is high and applied as another input of AND gate 546, then AND gate 546 is enabled and a write violation exception occurs via amplifier 547.

Logic circuitry 591 comprised of flip-flops 532 and 533 in conjunction with amplifier 530 and AND gate 531 and inverter 530A permit the formation in register UM 502 of the maximum value of ring number (i.e. EAR) under control of a splatter instruction subcommand (not described herein) from the instruction fetch unit IFU. Assuming URN1F flip-flop 532 is set to logical 0 whereas URN2F flip-flop 533 is set to logical 1, then during the execution of the splatter subcommand, input terminal 531A of AND gate 531 will be high; therefore if flip-flop 532 is low (logical 0) then the signal will be inverted by inverter 530A and AND gate 531 will be enabled. Hence the maximum value of the contents of UP register 501 or bits 2 and 3 of logic vector UBS 606 will be strobed into UM register 502. Conversely if flip-flop 532 is a logical 1, then the contents of UM register 502 is not changed via the above mentioned sources and the EAR derived in UM register 502 via the addressing process of indirection is the one utilized. Flip-flop

533 is the back-up store for the EAR of address-syllable 2 when utilized.

Referring now to Figures 7 and 8 and Figure 5 there is a correspondence wherein the detailed logic for hardware in Figure 5 is shown in Figures 7 and 8 as follows: Figure 7a and UW register 504; Figure 7b and UV register 503; Figure 7c and block 590; Figure 7d and block 591; Figure 7e and block 592; Figure 7f and UP register 501; Figure 7g and UO register 512; Figure 7h and UM register 502; Figure 8a and UWV logic 506; Figure 8b and UEP logic 507; and Figure 8d and UMR logic 505.

Referring to Figure 7a, the UW register 504 is comprised of two flip-flops 715a and 720a respectively, each flip-flop capable of holding one bit of information of the UW register. Coupled to flip-flop 715a are 4 AND gates 711a—714a which are OR'ed together, with each gate (except gate 713a) having two input terminals, and with at least one signal applied to each input terminal. AND gate 714a has one of its input terminals coupled to the set terminal OW00010 of the flip-flop 715a. Flip-flop 715a is also coupled to the terminal H27 for receiving from a clock a timing signal called a PDA signal. Flip-flop 720a coupled to AND gates 716a—719a which are OR'ed together. One input terminal of AND gate 716a is coupled to an input terminal of AND gate 711a; one input terminal of AND gate 717a is coupled to one input terminal of AND gate 712a and one input terminal of AND gate 719a is coupled to an input terminal of AND gate 714a, whereas the other input terminal of AND gate 719a is coupled to the set terminal UW00110 of the flip-flop 720a. Flip-flop 720a is also coupled to the H27 terminal for receiving PDA pulses.

AND gates 701a—704a are OR'ed together each having their output terminal coupled to the input terminal of inverter 705a. AND gate 706a is coupled to amplifier 708a; whereas AND gate 707a is coupled to amplifier 709a; one input terminal of AND gate 706a is coupled to one input terminal of AND gate 707a. The output terminal of inverter 705a is coupled to one input terminal of AND gate 714a and 719a; the output terminal of amplifier 708a is coupled to the input terminal of AND gate 713a and the output terminal of amplifier 709a is coupled to the input terminal of AND gate 718a.

The signals applied to the inputs of AND gates and the signals derived as outputs from amplifier, inverters, or flip-flops are designated by letters forming a special code. Since both data signals and control signals are either applied or derived there are two codes, one code for the control signals and one code for the data signals. The code for the control signals are previously described in detail and is summarized here. Briefly the first two characters of a control signal indicate the destination of data to be transferred; the third character indicates whether a full or partial transfer is to be effected with the letter F indicating full transfer and any other character indicating a partial transfer; the fourth and fifth character indicates the source of the data, and if the source is identified by more than two letters only the last two letters need be used; the sixth and seventh characters are usually numerals and indicate whether the signal is high or low i.e. an odd numeral in the sixth position indicates assertion and an even numeral in the sixth position indicates negation; the seventh position indicates whether this is the first, second, third, etc. level of occurrence of the signal. Data, on the other hand, is indicated differently. The first three characters of data indicates the source of the data, the fourth and fifth characters which may be numerals indicate the bit positions where the data is located in the source, and the sixth and seventh position are similar to the control signals in that they indicate whether the signal is high or low and the level of occurrence of the signal. Generally the format itself indicates whether the signal is a control signal or a data signal and by reference to Figures 5 and 6 the source and destination may be determined. There are exceptions to this general rule and they will be spelled out in the specification, and addenum.

As an example of this convention it will be noted on Figure 7a that the following signals are control signals: UWFAB11, UWFAB10, UW9QA10. The following signals are data signals UAB3410, UAB3210, UAB3510, UAB3310, QA00110, and QA00010. The following signals are exception PDARG10 is a timing signal whose source is the PDA clock; UWHOL10 is a hold signal for holding the information in the flip-flops 715a and 720a UW0BK10 and UW1BK10 are back-up logic whose main function is to extend the input capability of flip-flops 715a and 720a by connecting the UW register which is in fact formed by flip-flops 715a and 720a, to bit zero and bit 1 represented by flip-flops 715a and 720a respectively; and finally USCLR10 is the clear signal for clearing and setting the flip-flops to zero.

As an illustration of the above mentioned convention herein adopted the signal UWFAB11 applied to the input of one-legged AND gate 702a is a control signal which transfers data (bits 34 and 35) contained in UAB associator buffer 611 (the U in the signal has been omitted) to UW register 504 and is a full transfer to the

UW register 1; the odd number indicates the signal is assertion. Signal UWFAB10 applied to the input of one-legged AND gate 703a is a control signal with the same source and destination as the signal applied to AND gate 702a except that bits 32 and 33 of UAB are transferred to UW register. The signal UW9QA10 applied to one-legged AND gate 704a is also a control signal wherein data is transferred from QA bus 614 to the UW register and may be a partial transfer. The signal QA00010 applied to AND gate 706a is a data signal where data is on QA bus 614 (the third position is not herein utilized since the first two positions adequately describe where the data is) and this data signal represents the bit identified as 00 on QA bus 614. The signal QA00110 is similar to the previous signal except the data identified by this signal is the data on position 01 of the QA bus 614. Thus by utilizing this convention and Figures 5 through 9 the ring protection hardware is fully defined and may be easily built by a person of ordinary skill in the computer art.

Referring to Figure 7b there is shown the detailed logic block diagram for UV register 503. Signal UVH0L10 is a hold signal for UV register 503 which is generated via inverter 703b when none of the one-legged AND gates 701b—708b has a high signal applied to it. UVH0L10 signal is applied to AND gate 723b and causes information stored in the UV register 503 to be held therein. Signal UVH0L1E coupled to the input of AND gate 704b and to the outputs of AND gates 705b—708b extends the number of control signals that may generate the hold signal UVH0L10. Signal UV0BK10 coupled to the outputs of AND gates 710b—713b and to the input of AND gate 722b is also utilized to extend the number of inputs signals that may be applied to flip-flop 724b. Signal UV1BK10 coupled to the outputs of AND gates 716b—718b and to the input of AND gate 727b similarly extends the number of input signals that may be applied to flip-flop 729b.

Referring now to Figure 7g there is shown the detailed logic block diagram of UO register 512. AND gates 701g—704g are OR'ed together and their output is applied as an input to inverter 705g. AND gates 706g—709g are also OR'ed together and their outputs are coupled to flip-flop 710g. Also one input of AND gate 709g is coupled to the U000010 terminal of flip-flop 710g. AND gates 711g—714g are also OR'ed together and are similarly coupled to flip-flop 715g. It will be noted also that an input of AND gate 706g is coupled to an input of AND gate 711g; an input of AND gate 707g is coupled to an input of AND gate 712g and an input of AND gate 709g is coupled to an input of AND gate 714g. The UOH0L10 signal generated by inverter 705g is also coupled to an input of AND gate 709g and 714g and is utilized to hold information in the UO register 512. XOO represents a ground, whereas XNU means unused input.

Figure 7f is a detailed logic block diagram of UP register 501. It is similar to Figure 7g described supra except that different signals from different destinations and different sources are applied.

Referring now to Figure 7h there is shown the detailed logic block diagram of UM register 502. AND gate 701h—704h are OR'ed together to produce the UMH0L10 hold signal via inverter 705h. AND gates 706h—709h are OR'ed together and are coupled to the input of AND gate 704h in order to extend the range of signals that may be applied to produce the UMH0L10 hold signal. Similarly AND gates 711h—714h are OR'ed together and coupled to the input of AND gate 723h in order to extend the range of signals that may be applied to flip-flop 730h; and also AND gates 716h—719h are OR'ed together and are coupled to the input of AND gate 727h in order to extend the range of signals applied to flip-flop 731h. A line 740h for applying the PDA signals to flip-flop 730h and 731h is coupled at point 734h and 735h respectively. The input of AND gate 703h is also expanded to provide two further inputs URN1F00 and 1RNUM10 by coupling the output of amplifier 733h to the input of AND gate 703h.

Referring now to Figures 7c—7e there is shown detailed logic block diagrams of write exception control logic 590, IFU subcommand control logic 591, and read violation exception control logic 592 respectively. Referring first to Figure 7c there is shown flip-flops 705c and 710c which correspond to flip-flops 541 and 540 respectively. Under a micro-operation URW2F10 subcommand the information in flip-flop 710c is transferred to flip-flop 705c. The UWV1H10 hold signal is utilized to hold the information transferred to flip-flop 710c, whereas the UWV2H10 signal is utilized to hold the information transferred to flip-flop 705c. Similarly in Figure 7d information is transferred from flip-flop 710d to flip-flop 705d under micro-operation signal URNSW10, and in Figure 7e information from flip-flop 710e is transferred to flip-flop 709e under control of micro-operation signal URW2F10.

Referring now to Figures 8a, 8b and 8d there is shown detailed logic block diagrams of UWV logic 506, UWEP logic 507, and UMR logic 505 respectively. Referring first to Figure 8a there is shown logic for generating a high signal when one

of the test conditions 510 is true and also for generating the execute violation signal when the contents of UW register is less than or equal to the contents of UM register is less than or equal to the contents of UV register is not true. When the signal UWLEV10 is generated it indicates that the contents of UW register 504 is less than or equal to the contents of UV register 503. The logic for generating this signal was derived pursuant to the following Boolean expression:

$$X_1=\overline{(B\bar{C}\bar{D})+(AB\bar{D})\times(A\bar{C})}$$

Where X_1 represents the output of amplifier 805a and the various letters of the expression represent different input terminals of AND gates 801a—804a.

An indication that the contents of UV register 503 is greater than or equal to the contents of UM register 502 is had when UVGEM10 signal is generated. This signal is generated via inverter 820a in response to various inputs on AND gates 816a—819a which are OR'ed together and coupled to the input of inverter 820a. The logic for generating the UVGEM10 signal is made pursuant to the following Boolean expression:

$$X_2=\overline{(B\bar{C}\bar{D})+(AB\bar{D})+(A\bar{C})}$$

An indication that the contents of UM register 502 is greater than or equal to the contents of UV register 503 is indicated by generating signal UMGEV10 via inverter 810a in response to the various inputs of AND gates 806a—809a which are OR'ed together. The logic for generating this signal is derived from the following Boolean expression:

$$X_3=\overline{(B\bar{C}\bar{D})+(AB\bar{D})+(A\bar{C})}$$

(Wherein X_3 is the generated output signal).

Similarly the UVEQM10 signal is generated pursuant to the following Boolean expression:

$$X_4=\overline{(A\bar{C})+(\bar{A}C)+(B\bar{D})+(\bar{B}D)}$$

Generation of the UVEQUM10 signal indicates that the contents of the UV register 503 is equal to the contents of the UM register 502.

The generation of the UMGEW10 signal indicates that the contents of the UM register 502 is greater or equal to the contents of the UW register 504 and is generated pursuant to logic having the following Boolean expression:

$$X_5=\overline{(B\bar{C}\bar{D})+(AB\bar{D})+(A\bar{C})}$$

Generation of the UMGTW10 signal indicates that the contents of UM register 502 is greater than the contents of UW register 504 and this signal is generated by logic defined by the following Boolean expression:

$$X_6=(AB\bar{D})+\bar{C}(B\bar{D}+A)$$

The generation of the UWGMV00 signal indicates that the contents of UW register less than or equal to the contents of UM register less than or equal to the contents of UV register is not true. It is obtained when the UVGEM10 signal indicating that the contents of UV register is greater than or equal to the contents of the UM register, and the UMGEW10 signal indicating that the contents of the UM register is greater than or equal to the contents of the UW register are both high.

Referring now to Figure 8b a UMEQP10 signal is generated by logic derived from the following Boolean expression:

$$X_7=\overline{(A\bar{C})+(\bar{A}C)+(B\bar{D})+(\bar{B}D)}$$

When this signal is high it indicates that the contents of UM register 502 is greater than the contents of UP register 501.

Referring to Figure 8d there is shown the detailed logic block diagram for performing the operations of UMR logic 505 shown on Figure 5. One of the operations of this logic is to determine the maximum value of the contents of UP register 501 and of bits 2 and 3 of UBS logic 606. In order to do this there must be an indication whether contents of UP is less than the contents of UBS or the contents of UP is greater than the contents of UBS. The generation of UPBEB10 signal indicates that the contents of UP register 501 is less than or equal to bits 2 and 3 of UBS logic 606; whereas the generation signal UPGTB10 indicates that the contents of UP register 501 is greater than bits 2 and 3 of UBS logic 606. These signals are generated by logic which has been defined by the following Boolean expression:

$$X_8=\overline{(B\bar{C}\bar{D})+(AB\bar{D})+(A\bar{C})}$$

Where X_8 is the output of inverter 805d and the letters of the expression are various inputs of the AND gates 801d—803d.

To illustrate how the maximum value of the contents of UP register and UBS logic may be determined by the output signals UMPB010 and UMPB110 of amplifier 814d and 817d respectively, assume first that the contents of register UP are less than or equal to bits 2 and 3 of UBS logic because bit 2 is 1 and bit 3 is 1 whereas UB register

contains 01. This is indicated by the signal UPLEB10 being high and the signal UPGTB10 being low since it is the inverse of signals UPLEB10. This high UPLEB10 signal is applied to one input of AND gate 813d and also one input of AND gate 806d. If bit 2 of UBS logic is a 1 as indicated by signal UBS0210 then AND gate 813d is enabled and signal UMPB010 goes high and indicates that bit 2 on UBS logic is a 1. Moreover if bit 3 of UBS logic is a 1 indicated by input signal UBS0310 being applied as another input of AND gate 816d then AND gate 816d is enabled and signal UMPB110 is high or a 1. Therefore under the assumed conditions where bits (2, 3) UBS logic is greater or equal to the contents of UP register the maximum value of the two quantities is in UBS, and its number is binary 11 or decimal 4. Hence it is seen how a comparison is first made to determine which hardware contains the maximum, and then a determination is made as to the value of that maximum. By similar analysis one may see how the value of the UP register may be determined by signals UMPB010 and signals UMPB110 when the contents of UP register is greater than the second and third bit of UBS logic. Similarly the maximum value of UM register 502 or UV register 503 may be determined by signals UVGEM10 and UMGTV10 respectively, when UV register 503 is greater than or equal to UM register 502, and conversely when UM register 502 is greater than UV register 503.

Referring now to Figures 9a—9i a legend of symbols utilized in Figures 7 and 8 is shown. Figure 9a shows the symbol when there is a connection internally within the logic board. Figure 9b illustrates an output pin connection. Figure 9c indicates an input pin connection and is generally a source outside of the logic board illustrated. Figure 9d is the symbol utilized for an AND gate. Figure 9e is the symbol utilized for an amplifier; whereas Figure 9f is the symbol utilized for an inverter. Figure 9g illustrates three AND gates 901g—903g that are OR'ed together thus causing output 904g to go high when any one of AND gates 901g—903g is high. Figure 9h shows the symbol of a flip-flop having a 00 reset terminal and a 10 set terminal. A PDA line supplies the clock pulse for causing the flip-flop to switch states when other conditions are present on the flip-flop. Figure 9i represents a micro-operation control signal.

In order to enforce the ring protection scheme between procedures executing in different rings, the invention employs push-down stacks for its procedure linkage mechanism wherein a portion of each stack called a stack frame is dynamically allocated to each procedure. Different stack segments are used for each ring with one stack segment corresponding to one ring. Thus when a procedure is executed in ring RN its stack frame is located in the RN stack segment. Referring to Figure 10 there is shown three stack segments 1001—1003, with each stack segment having stack frames S1—S3 respectively. Ring 3 is assigned to stack segment 1001, ring 1 assigned to stack segment 1002 and ring 0 is assigned to stack segment 1003. Within each stack segment there is a procedure 11 associated with stack frame S1 of segment 1001, a procedure P2 associated with stack frame S2 of stack segment 1002 and a procedure P3 associated with stack frame S3 of stack segment 1003. The segmented addresses (i.e. segment number and segment relative address SEG, SRA) of the first bytes of the stack segments for rings 0, 1 and 2 respectively are located in stack base words SBW0—SBW2 respectively which are in turn located in process control block 104. Since the ring 3 stack segment can never be entered by an inward call (i.e. from a ring higher than ring 3) its stack starting address is not needed. Each stack frame S1, S2, S3 is divided into a working area 1005, 1006, 1007 respectively; an unused portion 1008, 1009, 1010, which is utilized for alignment purposes; a register saving area 1011, 1012, and 1013; and a communication area 1014, 1015, and 1016 respectively. The working area is utilized by its procedure as needed and may contain material required by the process such as local variables, etc. The saving area of the stack frame is utilized to save the contents of various registers such as the status register, the T-register and the instruction counter contents ICC. The communications area stores information which is needed to pass parameters between procedures. Prior to a call to a given procedure the user saves those registers he wishes saved and moreover loads into the communication area the parameters to be passed to the called procedure. When the call is made, the hardware saves the contents of the instruction counter and other specified registers to facilitate a return from the called procedure. Each procedure call creates a stack frame within a stack segment and subsequent procedure calls create additional frames. Hence a stack is created and consists of a number of contiguous parts called stack frames which are dynamically allocated to each procedure. These stacks reside in stack segments. Generally the first stack frame is loaded into the beginning of the segment and succeeding frames are loaded after it. The last frame loaded is considered the top

of the stack. A T-register 114 on Figure 1, locates the top of the stack for the currently active process. A procedure such as for example P1 which is executing in ring 3 may call a procedure P2 executing in ring 1 which in turn calls a procedure P3 which is now executing in ring 0. As each procedure is called it creates within its ring stack segment a stack frame (i.e. defining the environment for the procedure execution) and the T-register 114 is loaded which gives the address of the top of the stack for the current active process. The procedure P1 (as previously assumed) may call procedure P2 which in turn may call procedure P3 and since these calls are from a higher ring number to a lower ring number a ring crossing entailing an inward call is required and is accomplished in a manner to be described infra. During each change of procedure the necessary registers and parameters are saved in order to facilitate a return from the called procedure.

A procedure is always accessed through a procedure descriptor 1110 by means of the ENTER PROCEDURE INSTRUCTIONS. The format of the ENTER PROCEDURE INSTRUCTION 1100 is shown on Figure 11a. The operation code (OP) 1101 occupies bit positions 0 through 7. The complementary code 1102 is a one bit code and occupies bit position 8 to 9; if the complementary code is set to logical 1 the instruction is ENT, whereas if the complementary code is logical 0 the instruction is ENTSR and the base register must be base register 0 (BR0). The address syllable AS 1104 occupies bit positions 12 thru 31 and provides the address syllable AS of the procedure descriptor 1110. When an ENTER PROCEDURE INSTRUCTION requires a ring crossing a gating procedure descriptor 1120 is obligatorily accessed. This is indicated by the GS field 1302 of segment descriptor 1301 being set to logical 10. Generally the GS field is set to 10 when one of the ENTER PROCEDURE INSTRUCTIONS is utilized. As described in the application No. 21630/76, Serial No. 1,465,344, the segment descriptor is utilized to point to the base of the segment desired, in this instance the segment 1300 containing gate procedure descriptors GPD 1120. The first word of the segment 1300 containing the gating procedure descriptors (GPD's) is formatted as shown in Figure 11c. The TAG 1121 occupies bit positions 0 and 1 and must indicate a fault descriptor i.e. the TAG field must be set to logical 11. The Caller's Maximum Ring Number CMRN 1122 occupies bit positions 2 and 3, and indicates the maximum ring from which a calling procedure through the gated procedure descriptor GPD is legal. A call violation exception is generated if the caller's ring number is greater than CMRN 1122. The gated procedure descriptor address boundary GPDAB 1124 occupies bit positions 10 through 31 and it must be greater than the segment relative address SRA (i.e. the GPD's displacement in the segment of procedure descriptors 1300), otherwise an illegal GPD access exception occurs. Thus a gating procedure descriptor GPD is utilized as the first word of the segment containing procedure descriptors and is utilized to determine whether the caller has a right to access the segment via the caller's maximum ring number CMRN and whether or not the procedure descriptor called is within the gating procedure descriptor's address boundary. Once it is determined that there is a legal call to the segment and the caller has a right to enter the segment the address is obtained from the address syllable AS 1104 of enter instruction 1100 and the required procedure descriptor 1110 (see also Figure 13) is accessed. The format of procedure descriptor 1110 is shown on Figure 11b and is comprised of two 32 bit words—word 0 and 1 respectively. Word 0 contains the segmented address 1113 of the entry point EP of the procedure desired. The segmented address, as is the case with the segmented address of any operand, is comprised of the segment number SEG and the segment relative address SRA. Word 0 of the procedure descriptor includes an entry point ring number EPRN 1112 and a TAG field 1111. The value of the TAG is interpreted as follows:

a. if the TAG contains logical 00 the procedure descriptor is direct;

b. if the TAG is logical 01 the procedure descriptor is an extended descriptor and includes word 1 making a total of two words;

c. if the TAG is logical 10 the procedure descriptor is indirect and an illegal procedure descriptor exception occurs; and

d. if the TAG is logical 11 it is a fault procedure descriptor and an exception occurs.

Word 1 of the procedure descriptor is 32 bits long and is utilized when the TAG indicates an extended descriptor and contains the segmented address of a linkage section whose contents are loaded in base register BR 7 at procedure entry time.

Referring to Figure 12 a portion of the ENT instruction is shown and more specifically that portion which pertains to the ring crossing and ring checking requirements. The ENT instruction is called, 1201 and a comparison is made 1202 wherein the segmented part of the base register BRn is compared to the segmented part of the address of the T register, and if

they are not equal an illegal stack base register 1208 is indicated. If on the other hand they are equal another comparison 1203 is made wherein the 30th bit including the next two bits (i.e. bits 30 and 31) of base register, BRn is compared to 0 and if it is not equal to 0, then once again an illegal stack base register 1208 is indicated. If it is equal to 0 it indicates that the contents of BRn is aligned with respect to the word boundary and another comparison 1204 is performed to determine that the TAG of BRn (i.e. the two bits starting from bit 0) is equal to 0. A TAG having a logical 0 indicates information is accessed via a direct descriptor which is one of the requirements of the ENT instruction. If the TAG (i.e. bits 0 and 1 of BRn) is equal to 0 then the functions stated in flow charts of Figures 14 through 16 are performed (see flow chart Figure 12 block 1205). If these meet the necessary requirements a further check 1206 is made to determine whether the segment relative address of the entry point which was given (SRA_{EP}) is even, because instructions start on a half-word boundary. If it is not even then an illegal branch address exception is generated 1209 however if it is legal the ENT instruction is executed 1207 via further steps not shown.

Referring now to the flow charts of the access checking mechanism Figures 14—16, generally the following operations are performed each time the instruction ENTER PROCEDURE is issued:

a. the caller's right to call the callee is checked by first determining from the second word of the segment descriptor the call bracket in which the caller is executing. (The call bracket is determined by taking the minimum ring number from the write ring number field WR and the maximum ring number from the maximum ring number field MAXR).

b. a decision is made about the next process ring number by determining whether the caller is in the same call bracket as the callee, which implies don't do anything; whether the caller is in a call bracket requiring that he make an outward call in which case an exception condition is generated which is handled by a mechanism not described herein; or finally whether the caller is in a call bracket which requires an inward call (i.e. going to a call bracket which requires ring crossing from a larger ring number to a smaller ring number in which case the ring crossing must be at a valid entry point EP and the entry point must be validated).

c. a stack frame is created for the callee (i.e. space in the aforementioned format of the appropriate segment is allocated), and the stack frame registers are updated;

d. a branch to the entry point of the procedure pointed to by the procedure descriptor is performed.

Referring now to Figure 14 the access checking is started 1401 by obtaining the address syllable AS containing the effective address ring number EAR, the segment number of the procedure descriptor SEG_{PD}, and the segment relative address of the procedure descriptor SRA_{PD}. Having developed this information the procedure descriptor 1110 is fetched 1403 from (SEG_{PD}, SRA_{PD}) ignoring access rights to scratch pad memory. The procedure descriptor 1110 will yield the TAG which determines whether the descriptor is direct, extended, indirect, or a fault descriptor; the entry point ring number EPRN; the segment (SRA_{EP}) which contains the entry point and the segment relative address (SRA_{EP}) of the entry point. The TAG is tested 1404 to determine whether the descriptor 1110 is direct, extended, indirect or a fault descriptor by checking its field in accordance to the code hereinbefore described. Only a direct or extended procedure descriptor is legal. An indirect or fault descriptor is illegal and upon access invokes an exception mechanism not herein described. Once it is determined that a legal procedure descriptor has been accessed the actual call right checking begins at point A 1405.

Referring now to Figure 15 and continuing from point A 1405 the maximum ring number MAXR, the write ring number WR, and the execute permission bit EP of the segment containing the entry points SEG_{EP} are fetched; this information is contained in the segment descriptor for the segment containing the entry points (SEG_{EP}). The write ring number WR is compared to the maximum ring number MAXR 1503 and if the write ring number WR is greater than the maximum ring number MAXR the segment is nonexecutable and an execute violation exception 1513 occurs. If the write ring number WR is less than or equal to the maximum ring number MAXR then the execute permission bit EP is compared to logical 1 and if the EP bit is not logical 1 then once again an execute violation exception 1513 occurs; however if the EP bit is equal to one the effective address ring number EAR of the calling procedure is maximized with EPRN to give a new EAR_2,—[MAX (EAR_1 EPRN)] where EAR_1 is the maximum of PRN as found in the instruction counter IC, and all ring numbers in base registers and data descriptors, if any, found in the path which leads to the procedure descriptor. The

effective address ring number EAR_2 is then compared 1506 to the maximum ring number MAXR of the MAXR segment descriptor of SEG_{EP} which is the maximum ring number at which a procedure may execute. If EAR_2 is greater than MAXR the procedure call is an inward call which requires that the procedure be entered by a valid entry point and the access checking operation branch to point B 1507. The following checking operations are then performed:

a. the SEG_{EP} is checked to determine if it is a legal gate segment; and,

b. the caller's maximum ring number CMRN is checked to determine if it is greater than or equal to the effective address ring number EAR of the caller.

If these conditions are not true then an illegal gate segment exception 1603 or call violation exception 1615 occurs.

Referring now to branch point B 1507 of Figure 16 the first check 1602 that is made is to determine whether or not the segment which contains the procedure descriptors is a gate segment. This is done by examining the Gating/Semaphore field GS of the segment descriptor pointing to the segment of procedure descriptors, to determine if it is set to logical 10. If the GS field of the segment descriptor of the segment containing procedure descriptors is set to 10 it is then a gate segment and the first word of the segment containing procedure descriptors is a gated procedure descriptor GPD 1120 of Figure 11C and Figure 13. The first word 1120 of the segment containing procedure descriptors is then fetched from address SEG_{PD}, 0 ignoring access rights to scratch pad memory. It will be noted that the TAG field of the first word 1120 of the segment containing procedure descriptor SEG_{PD} 1300 must be a logical 11 (Figure 13) which indicates it is a fault descriptor. Moreover the MBZ field must be set to zero. These conditions are checked by hardware/firmware (arithmetic logic unit) stop 1605 and if these conditions do not hold an illegal gate segment exception 1603 results. However if these conditions do hold a check 1606 is further made to determine that the segment relative address of the procedure descriptor SRA_{PD} 1110 is a multiple of 8. If the condition of step 1606 does not hold an illegal system object address exception 1613 results otherwise the next step 1607 is performed. Step 1607 checks to determine whether or not the segment relative address of the procedure descriptor SRA_{PD} is within the address boundary GPDAB 1124 of the gated procedure descriptor 1120; if it is not within that address boundary it is an illegal procedure descriptor and an illegal GPD gated procedure descriptor access exception 1614 occurs. However if it is within the address boundary of the gated procedure descriptor (i.e. SRA_{PD} is less than GPDAB) then the caller's right to call the callee is checked 1608. This is performed by comparing the effective address ring number EAR_2 to the caller's maximum ring number CMRN 1122 as found in the first word 1120 of the segment of procedure descriptors 1300. If EAR_2 is greater than the caller's CMRN a call violation exception 1615 occurs which indicates that the caller in this particular instance has no right to legally call inward i.e. from a higher ring number to a lower ring number. On the other hand if EAR_2 is equal or less than CMRN, then the inward call is legal and a check is made 1609 to determine that the process ring number PRN which is the current process ring number found in the instruction counter IC just before the call was made is less than the maximum ring number MAXR of SEG_{EP}; and if it is the accessing mechanism branches to point C 1508, otherwise a new process ring number NPRN is calculated and set to a maximum ring number MAXR 1611. Generally the effective address ring number EAR_2 is the same as the process ring number PRN of the caller. Sometimes however, in cases where it is necessary to give maximum assurance that the caller will not be denied access to a given segment the EAR_2 is greater than the PRN. In those cases I RN is forced to take the value of EAR_2 in order to make sure that the call is returned to the maximum ring number upon an exit. To this point it will be noted that this checking mechanism was invoked because the EAR_2 was greater than the MAXR hence greater than the top of the call bracket of the procedure and hence an inward call was necessary which necessitated going through a valid gate, and the mechanism included these gating checks. By branching back to C 1508 (Figure 15) a further check 1509 is made to determine then that the process ring number PRN is greater than the write ring number WR of SEG_{EP} which in this context is the minimum ring number at which a procedure may execute. If the write ring number WR is greater than the process ring number PRN an outward call exception 1514 occurs. However if WR is less than or equal to PRN the call is legal and NPRN is set to PRN 1510.

Having made the above checks the inward call is made, and after performance of the desired operation a return back to the original point of the program in execution is made by the EXIT INSTRUCTION. During the ENTER INSTRUCTION the instruction counter IC

was saved in the saving area of the caller's stack frame before making the call. Moreover the caller's ring number was also saved during the ENTER INSTRUCTION and this was saved in base register 0 BRO.

The format of the EXIT INSTRUCTION 1130 is shown on Figure 11D. The operation code OP 1131 is found in bit positions 0—7 and the complementary code C 1133 is found in bit positions 12—15. The complementary code allows other instructions to use the same 8 bit op code. The MBZ field 1132 in bit positions 8—11 must be 0 otherwise an illegal format field exception occurs. (BRO is generally a pointer to the communications area of the caller's stack frame).

In performing the EXIT INSTRUCTION it is necessary to perform predetermined checks in order to ascertain that the caller didn't change his image which would permit him to operate a a different privilege than was intended. Referring to Figure 17 the first check performed 1701 is to determine if the TAG of the instruction counter content (ICC) indicates a direct descriptor. A logical 00 in the TAG field indicates that it is direct if it is not an illegal stack data exception 1702 occurs, whereas if it is equal to 0 the ring field in the instruction counter content ICC is set to the new process ring number NPRN 1703. This sets the new process ring number NPRN to what it used to be when the call was first made. However further checks are made in order to ascertain that there was no further cheating. Hence the base register 0 ring number located at bit position 2 and extending for 2 bit positions from and including bit position 2 must be equal to the new process ring number NPRN 1704. (It will be recalled that when the ENTER INSTRUCTION was called the ring number of the caller before the call was made was stored in bits 2 and 3 of base register 0 (BRO). If check 1704 indicates that the new process ring number NPRN is not equal to the ring number in bit positions 2 and 3 of the base register 0 (BRO) an illegal stack data exception 1702 occurs. The next check 1705 determines whether an inward or an outward return must be performed. Since an inward call was previously performed an outward return is implied in order to reach the original point from which the procedure was called. Moreover since the invention does not permit an outward call there is never a necessity to return inward. Hence the new process ring number NPRN is compared to the process ring number PRN 1705, and if NPRN is less than PRN an inward return is implied and an inward return exception 1706 is generated. However if check 1705 is passed successfully (i.e. NPRN is greater or equal to PRN) then a check is made to determine that a return is made to the segmented address SEGr that called the procedure and a return to the call bracket of the calling procedure is made and moreover that the execute bit EP is set. This is performed by fetching the segment descriptor SEGr of the calling procedure 1707 and making checks 1709, 1711, 1712. In performing checks 1709, 1711, 1712, check 1709 and 1711 determine that the new process ring number NPRN is greater than the minimum ring number WR but less than the maximum ring number MAXR (i.e. that the ring number is in the call bracket of the calling procedure where it should be). Finally check 1712 makes sure that the execute permission bit EP is set to 1. Thus a full cycle is concluded a call was performed via an ENTER INSTRUCTION; the required operation or processing was performed via the called procedure; then a return via an EXIT INSTRUCTION to the calling procedure was performed.

Having shown and described the preferred embodiment of the invention, those skilled in the art will realize that many variations of modifications can be made to produce the described invention and still be within the scope of the claimed invention.

Glossary of Terms

JOB—The job is the major unit of work for the batch user. It is the vehicle for describing, scheduling, and accounting for work he wants done.

JOB STEP—A smaller unit of batch work. It is generally one step in the execution of a job consisting of processing that logically belongs together.

TASK—The smallest unit of user-defined work. No user-visible concurrency of operation is permitted within a task.

PROGRAM—A set of algorithms written by a programmer to furnish the procedural information necessary to do a job or a part of a job.

PROCESS GROUP PLEX—The system's internal representation of a specific execution of a job.

PROCESS GROUP—A related set of processes, usually those necessary for performance of a single job step.

PROCESS—The controlled execution of instructions without concurrency. Its physical representation and control are determined by internal system design or convention.

Glossary of Terms (cont.)

PROCEDURE—A named software function or algorithm which is executable by a computational processor without concurrency. Its physical representation (code plus associated information, invocation, and use are determined by internal system or designed convention).

LOGICAL PROCESS—The collection of hardware resources and control information necessary for the execution of a process.

ADDRESS SPACE (SEGMENTATION)—The set of logical addresses that the CPU is permitted to transform into absolute addresses during a particular process. Although a processor has the technical ability of addressing every single cell of timing memory, it is desirable to restrict access only to those cells that are used during the process associated with the processor.

LOGICAL ADDRESS—An element of the process address space such as for example segment number SEG and Displacement D.

BASIC ADDRESS DEVELOPMENT—A hardware procedure which operates on a number of address elements to compute an absolute address which is used to refer to a byte location in core.

PROCESS CONTROL BLOCK—A process control block PCB, is associated with each process and contains pertinent information about its associated process, including the absolute address of tables defining the segment tables the process may access.

J. P. TABLES—A collection of logical addresses for locating a process control block associated with a process.

SEG_{PD}—The segment which contains the procedure descriptor.

SEG_{EP}—The segment which contains the entry point, as found in the procedure descriptor.

PRN—The process ring number, found in the instruction counter IC just before the call, or calculated by the ENTSR instruction.

EAR—The effective address ring number which is the maximum of:
(a) the process ring number PRN as found in the IC; or
(b) all ring numbers in the base register and data descriptors (if any) found in the path which leads to the procedure descriptor from the call instruction, including the entry point ring number EPRN located in the procedure descriptor itself.

MAXR—The maximum ring number at which a procedure may execute; MAXR is found in the segment descriptor of SEG_{EP}.

WR—The minimum ring number at which a procedure may execute; WR is found in the segment descriptor of SEG_{EP}.

EP—Execution permit bit found in the segment descriptor of SEG_{EP}.

CMRN—The caller's maximum ring number, as found in the first word of the segment SEG_{PD} if this segment is identified as a gate segment (i.e. with the code "gate" set).

NPRN—New process ring number.

EPRN—Entry point ring number (found in the process procedure descriptor).

Addendum

Signal Name	Type	Function
(1) WSCLR	Control	Clears register to which it is connected.
(2) PDARG	Control	Clock Signal PDA.
(3) PDURGIT	Connecting	Pin connected to PDA at one end and resistor at the other.
(4) UWOBK	Connecting	Expands inputs to UW register.
(5) UWHOL	Control	Holds information in register to which it is connected.
(6) UW1BK	Control	Same as UWOBK but is connected to different input terminal of UW register.
(7) UW00000		Reset terminal of one flip-flop of register UW.
(8) UW00010		Set terminal of flip-flop of register UW.
(9) UW00100 UW00110		Same as 7+8 but different flip-flop.
(10) UVSPS	Control	Spare Control Input.

Addendum (cont.)

Signal Name	Type	Function
(11) UVSPD	Data	Spare Data Input.
(12) UVOBK	Expander	Same as UW0BK and UW1BK, but it connects different registers and gates.
(13) UV00000 UV00010 UV00100 UV00110		Same as UW00000, UW00010, UW00100, UW00110, but applies to flip-flop UV.
(14) UWV1S	Control	Control input for UWV1F.
(15) UWV1D	Data	Data input for UWV1F.
(16) UWV2F	F/F	Write control flip-flop.
(17) UWV1S UWV2S	Control	Control unit for UWV1F, UWV2F.
(18) UWV1D	Data	Data input for UWV1F.
(19) UWV1H	Control	Hold UWV1F flip-flop.
(20) UWV1C	Control	Clear UWV1F.
(21) UWV2C	Control	Clear UWV2F.
(22) URN1S URN2S	Control	Control inputs for URN1F, URN2F.
(23) URN1D	Data	Data Input for URN1F.
(24) URNSW	Control	Transfer URN1F to URN2F and URN2F to URN1F.
(25) URN2F	F/F	Control loading max (UP, UBS2, 3 to UM).
(26) URN1H	Control	Hold URN1F flip-flop.
(27) URN2C	Control	Clear URN2F.
(28) URW1S URW2S	Control	Control inputs for URV1F, URV2F.
(29) URW1D	Data	Data Input for URV1F.
(30) URV2F	F/F	Read control flop.
(31) XNU		Indicates terminal not used herein.
(32) XOO		Grounded Input.

WHAT WE CLAIM IS:—

1. An internally programmed data processing apparatus CPU having a virtual memory system, and being responsive to internally stored instruction words for processing information and having stored in said virtual memory system a plurality of different types of groups of information each information group-type associated with an address space bounded by a segment having adjustable bounds, and comprising means for protecting the information in said-virtual memory system from unauthorized users by restricting accessability to the information in accordance to levels of privilege, said means comprising in combination with an access checking mechanism;

(a) first means arranged in operation to store in said virtual memory system at least one segment table comprising a plurality of segment descriptors with each segment descriptor being associated with a predetermined one of said segments and each segment descriptor having a predetermined format containing an access information element and a base address element in predetermined positions of said format, said base address element being used for locating in said virtual memory system the starting location of a selected one of said segments, and said access information element for specifying the minimum level of privilege required for a predetermined type of access that is permitted in a selected one of said segments;

(b) a plurality of second means having a predetermined format, communicating with said first means, arranged to store in a predetermined portion of said second means, a segment number SEG for identifying a segment table and the location of a segment descriptor within said segment table, said second means also being arranged to store in a predetermined other portion of said second means, an offset address within the segment identified by said segment descriptor said offset address locating from said segment base the first byte of a word within said segment;

(c) third means responsive to an address syllable element of an instruction being executed for addressing one of said plurality of second means;

(d) fourth means arranged to store a displacement from said address syllable,

(e) fifth means, communicating with said first, second, third and fourth means, arranged to add the displacement D and said base address to said offset; and,

(f) sixth means responsive to said access

information element in a selected one of said segment descriptors, restricting the accessability to the segment associated with said selected one of said segment descriptors in accordance to the level of privilege and the type of access specified in said access information element, wherein each group-type of information is associated with a predetermined ring number indicative of a level of privilege said level of privilege decreasing as the associated ring number increases comprising means for determining the maximum effective address ring number EAR (i.e. minimum level of privilege) of a selected process to access a selected group of information, said means comprising:

(a) first means to store first information indicating the maximum ring number RD (i.e. minimum level of privilege) required to read information from said selected group;

(b) second means to store second information indicating the maximum ring number WR (i.e. minimum level of privilege) required to write information into said selected group;

(c) third means to store third information indicating the maximum ring number MAXR (i.e. minimum level of privilege) required to process information from said selected group; and,

(d) fourth means communicating with said first, second and third means, to determine the maximum of the contents of said first, second and third means whereby the effective address ring number EAR is generated.

2. Apparatus according to claim 1, wherein said second means for storing the maximum ring number WR additionally indicates the minimum ring number WR (i.e. maximum level of privilege) required to process information from said selected group.

3. Apparatus according to claim 1 or claim 2, wherein said fourth means to generate the effective address ring number comprises a comparator for comparing binary numbers.

4. Apparatus according to any one of claims 1 to 3 wherein the sixth means restricting the accessibility to the segment includes comparator means, communicating with said second means, to compare the effective address ring number EAR with the write ring number WR, and further including means communicating with said comparator means to generate a write-violation-exception signal when EAR is greater than WR.

5. Apparatus according to claim 4, wherein the sixth means restricting the accessibility to the segment includes seventh means, communicating with said second and third means thereof to compare the maximum ring number MAXR and the write ring number WR with the effective address ring number EAR, and further including eighth means, communicating with said seventh means for generating an execute-violation-exception signal when the MAXR is not equal or greater than EAR which in turn is not equal or greater than WR.

6. Apparatus according to claim 5, wherein in that the sixth means restricting the accessibility to the segment includes ninth means, communicating with said first means, for comparing the effective address ring number EAR with the read ring number RD, and further including tenth means, communicating with said ninth means, to generate a read-violation-exeption signal when EAR is greater than RD.

7. Apparatus according to claim 6, wherein in that the sixth means restricting the accessibility to the segment includes eleventh means to store a process ring number PRN of a currently executing process, and also including twelfth means to communicate with said eleventh means, and further including thirteenth means communicating said said twelfth means for overriding said read-violation-exception signal when the effective address ring number EAR is equal to the process ring number PRN of the currently executing process.

8. Apparatus according to any one of the preceding claims wherein the access checking mechanism supervises transfer of control of said CPU from a first selected procedure (i.e. caller) having a first ring number indicative of a minimum level of privilege associated with said caller, to a second selected procedure (i.e. the callee) having a second ring number associated with said callee indicative of a minimum level of privilege associated with said callee, said access checking mechanism comprising

(a) first means for checking the caller's right to call the callee;

(b) second means, communicating with said first means, to compare the caller's ring number to the callee's ring number;

(c) third means responsive to said second means to permit a transfer of control of said CPU from said caller to said callee when the ring number of the caller is greater than the ring number of callee (i.e. inward call); and,

(d) fourth means also responsive to said second means to deny a transfer of control of said CPU from said caller to said callee when the ring number of said caller is less than the ring number of the callee (i.e. outward call).

9. Apparatus according to claim 8, wherein the access checking mechanism includes a plurality of ring stack-segment means each of said ring stack-segment means having associated with it a ring stack-segment number, indicative of the minimum level of privilege required by a selected one of said procedures to access a selected one of said ring stack segments.

10. Apparatus according to claim 9 wherein there are four ring stack segment means having ring numbers 0 to 3 respectively.

11. Apparatus according to claim 9 or claim 10 wherein the access checking mechanism includes stack-frame-element means associated with selected ones of said procedures, said stack-frame-element means being grouped within said ring stack-segment means in accordance with the ring number of the associated procedure of said stack-frame-element means, said stack frame element means to save said register of said caller prior to passing control to said callee.

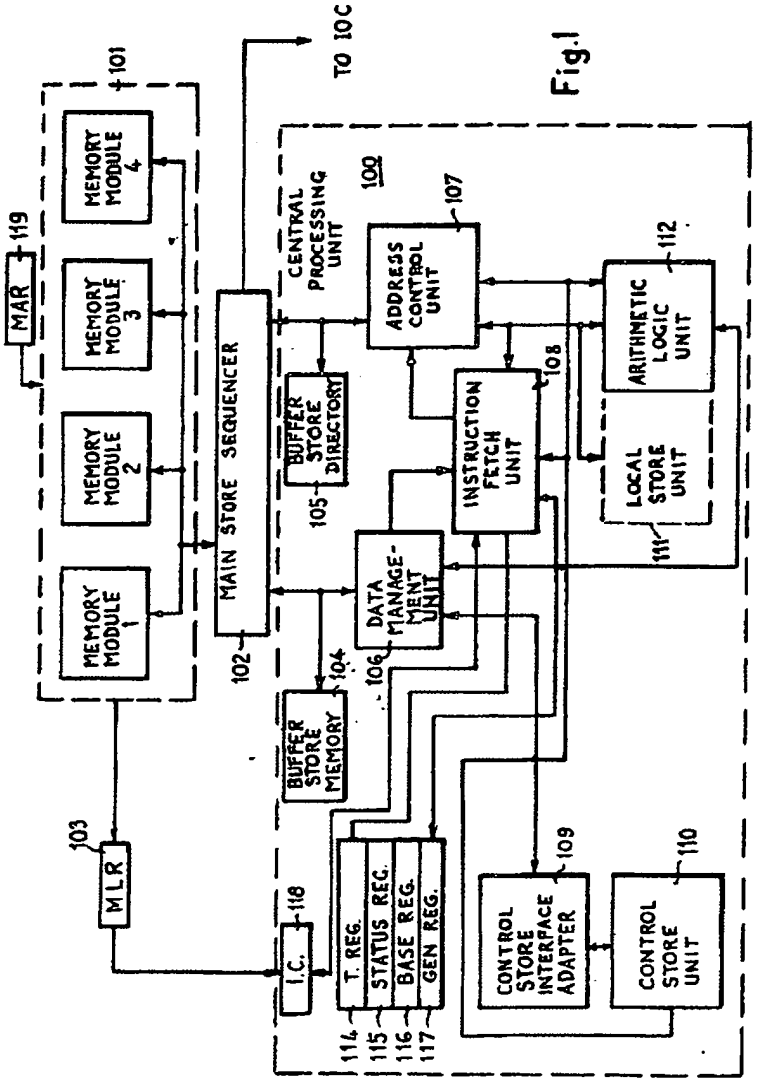
12. Apparatus according to claim 11, wherein the access checking mechanism includes first sub-element means, responsive to said first, second, third and fourth means, for communicating between a selected one of said stack-frame-means in a first ring stack-segment being associated with one ring number, and a selected other of said stack-frame-means in a second ring stack-segment associated with another ring number.

BARON & WARREN,
16, Kensington Square,
London, W8 5HL.
Chartered Patent Agents.

Printed for Her Majesty's Stationery Office, by the Courier Press, Leamington Spa, 1977
Published by The Patent Office, 25 Southampton Buildings, London, WC2A 1AY, from which copies may be obtained.

1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 1

Fig.1



1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 2

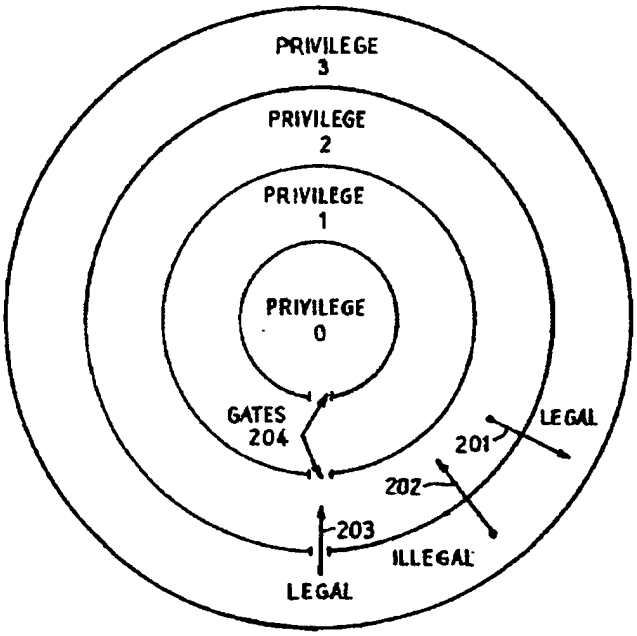
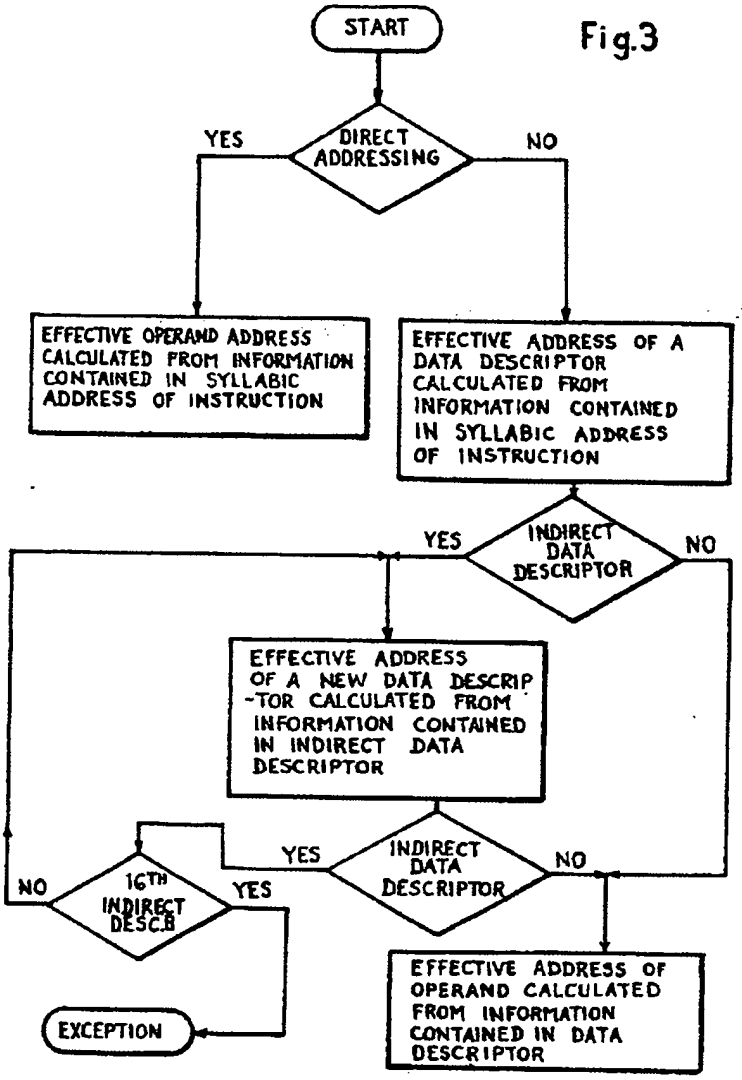


Fig.2

Fig.3


START
DIRECT ADDRESSING
YES
NO
EFFECTIVE OPERAND ADDRESS CALCULATED FROM INFORMATION CONTAINED IN SYLLABIC ADDRESS OF INSTRUCTION
EFFECTIVE ADDRESS OF A DATA DESCRIPTOR CALCULATED FROM INFORMATION CONTAINED IN SYLLABIC ADDRESS OF INSTRUCTION
INDIRECT DATA DESCRIPTOR
YES
NO
EFFECTIVE ADDRESS OF A NEW DATA DESCRIP-TOR CALCULATED FROM INFORMATION CONTAINED IN INDIRECT DATA DESCRIPTOR
INDIRECT DATA DESCRIPTOR
YES
NO
16TH INDIRECT DESC.?
NO
YES
EXCEPTION
EFFECTIVE ADDRESS OF OPERAND CALCULATED FROM INFORMATION CONTAINED IN DATA DESCRIPTOR

1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 4

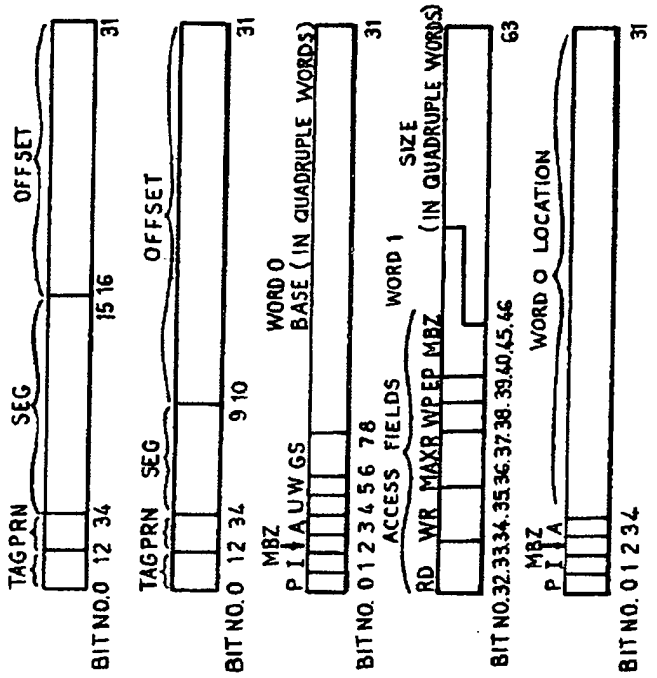


Fig.4 A Fig.4 B Fig.4C Fig.4 D Fig.4 E

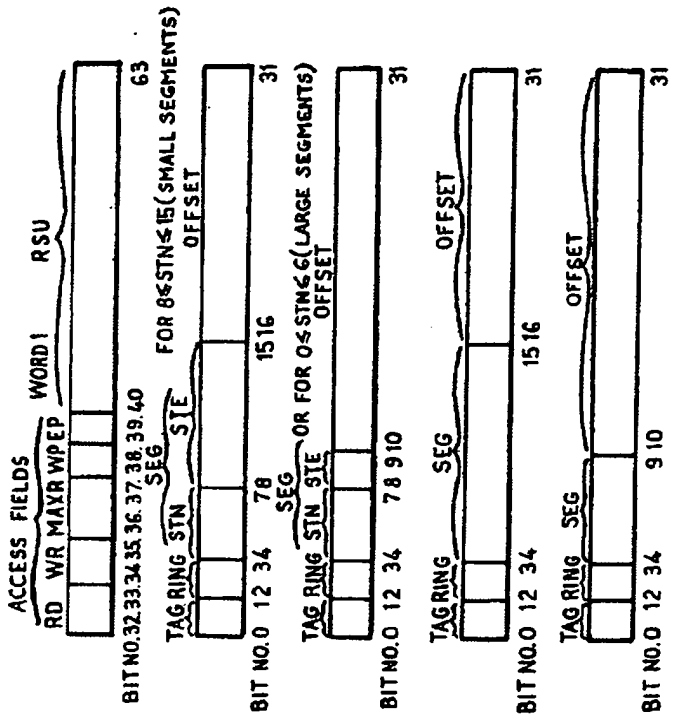
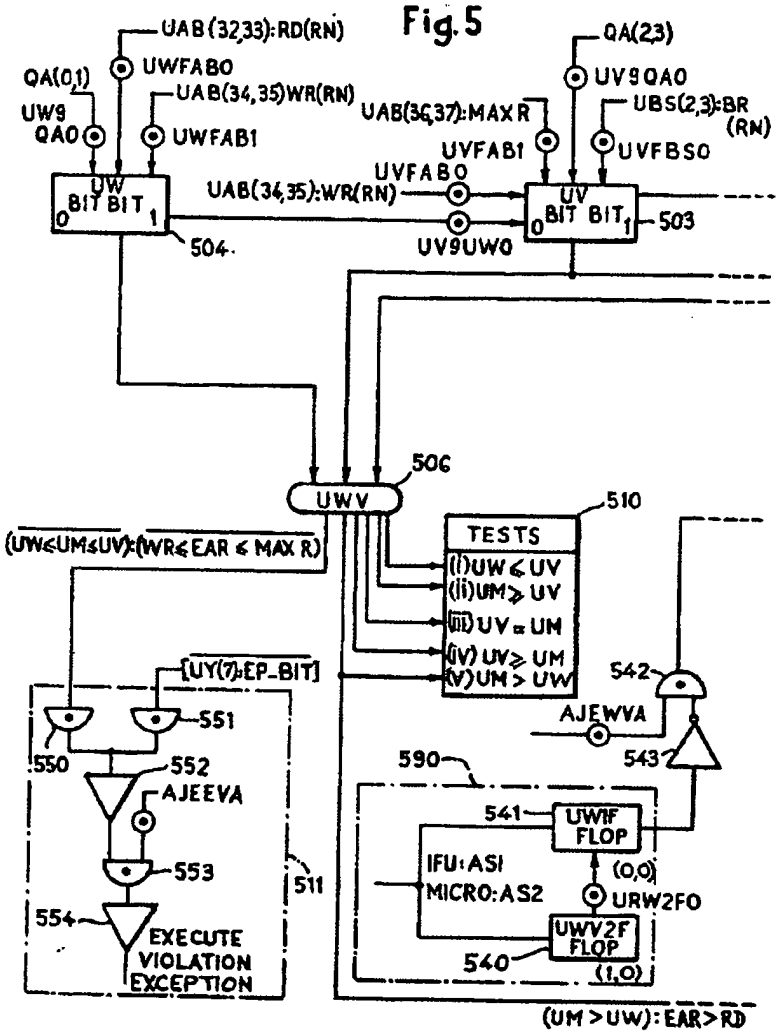


Fig.4F Fig.4G Fig.4H Fig.4I Fig.4J

Fig. 5


UAB(32,33):RD(RN)
QA(0,1)
UWFABO
UAB(34,35)WR(RN)
UW9 QA0
UWFAB1
UW BIT 0 BIT 1
504
QA(23)
UV9QA0
UAB(36,37):MAXR
UBS(2,3):BR (RN)
UVFAB1
UVFBSO
UVFABO
UAB(34,35):WR(RN)
UV BIT 0 BIT 1
503
UV9UW0
506
UWV
510
TESTS
(i)UW ≤ UV
(ii)UM ≥ UV
(iii)UV = UM
(iv) UV ≥ UM
(v)UM > UW
(UW≤UM≤UV):(WR≤EAR≤MAX R)
[UY(7):EP_BIT]
551
550
552
AJEEVA
553
554
EXECUTE VIOLATION EXCEPTION
511
542
AJEWVA
543
590
541
UWIF FLOP
IFU:ASI
MICRO:AS2
(0,0)
URW2FO
UWV2F FLOP
540
(1,0)
(UM > UW):EAR>RD

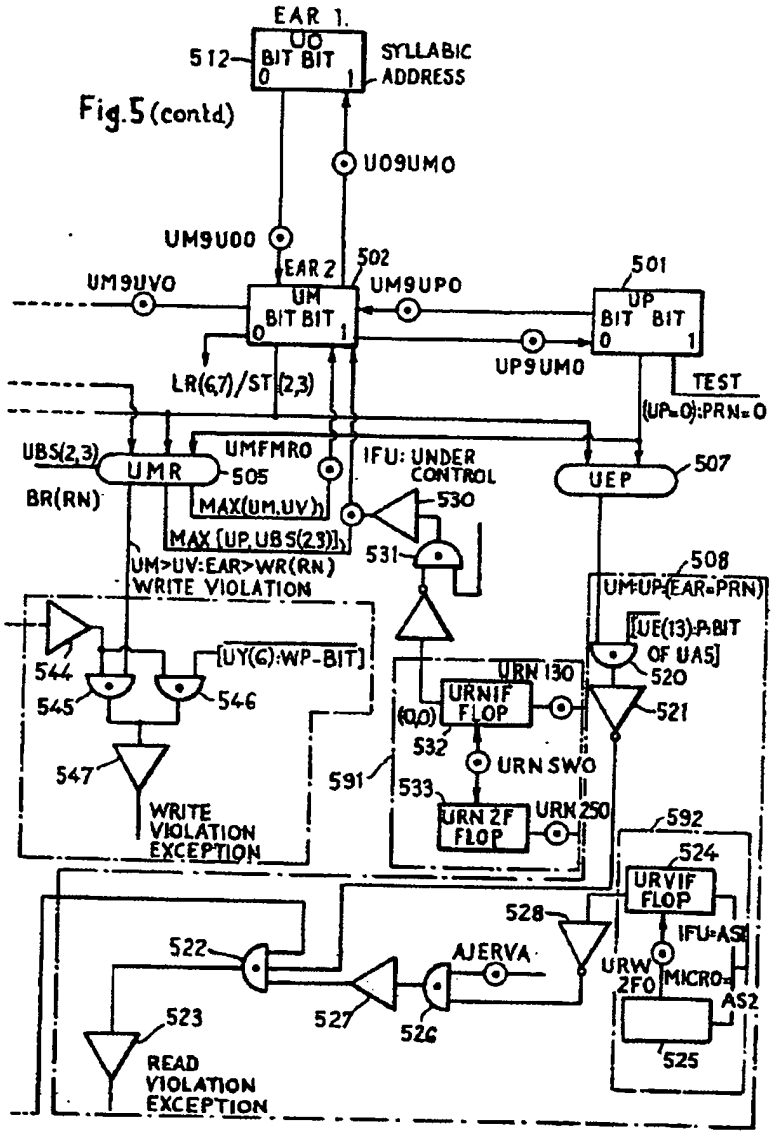

EAR 1.
U0
512
BIT BIT
0 1
SYLLABIC ADDRESS
U09UM0
UM9U00
EAR 2
502
UM9UV0
UM9UP0
UM
BIT BIT
0 1
501
UP
BIT BIT
0 1
UP9UM0
TEST
(UP=0):PRN=0
LR(6,7)/ST(2,3)
UBS(2,3)
UMR
505
UMFMR0
IFU: UNDER CONTROL
UEP
507
BR(RN)
MAX(UM.UV)
530
MAX(UP.UBS(23))
531
UM>UV:EAR>WR(RN) WRITE VIOLATION
508
UM=UP=(EAR=PRN)
[UE(13)=A BIT OF UA5]
520
521
[UY(6):WP-BIT]
544
545
546
547
WRITE VIOLATION EXCEPTION
URN 130
URNIF FLOP
(0,0)
532
URN SW0
591
533
URN 2F FLOP
URN 250
592
524
URVIF FLOP
528
IFU=AS1
AJERVA
URW 2F0
MICRO=
AS2
522
523
527
526
525
READ VIOLATION EXCEPTION

Fig.5 (contd)

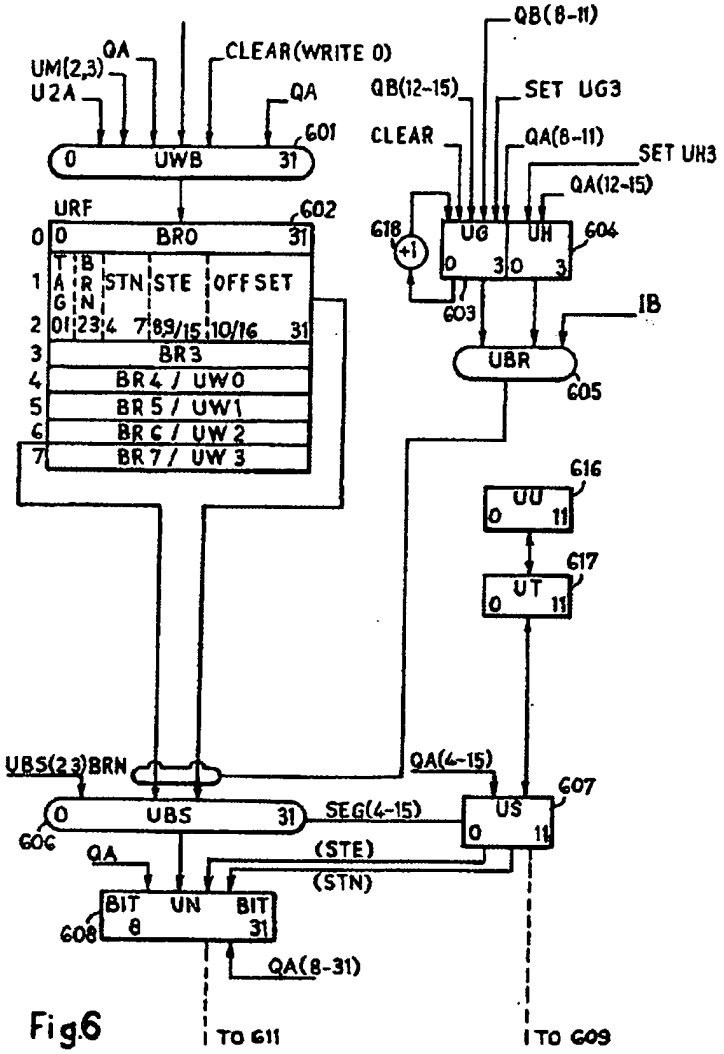


Fig.6

1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 9

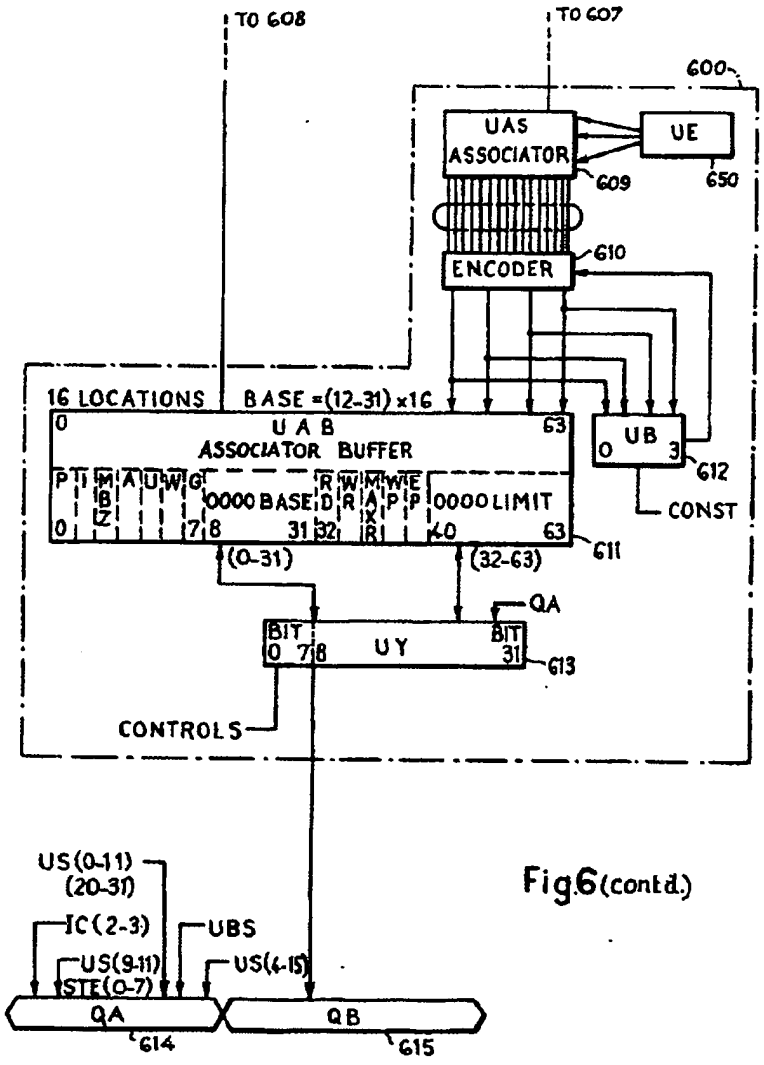


Fig.6(contd.)

1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 10

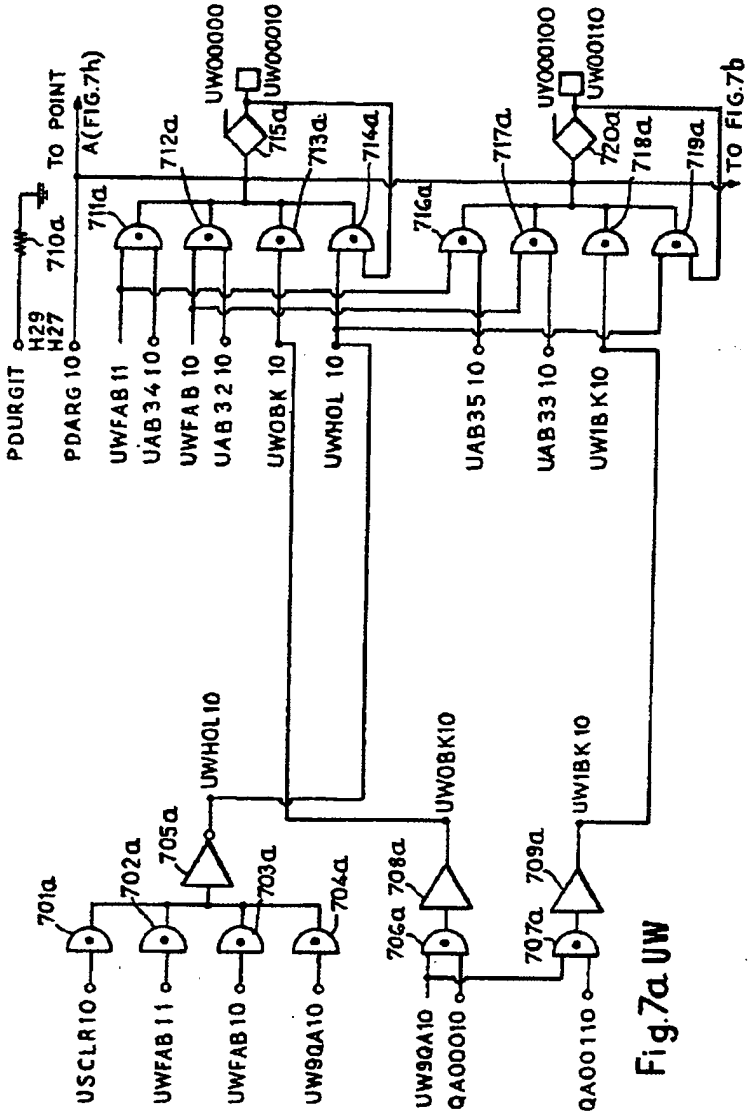
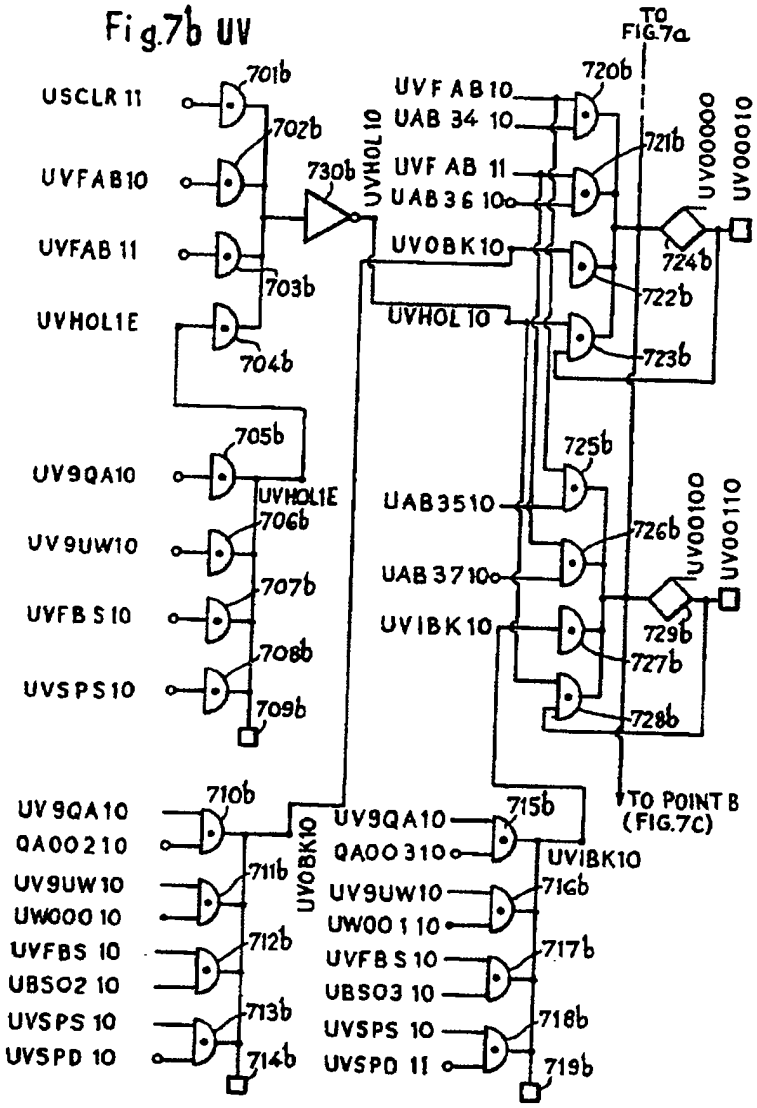


Fig.7a UW

1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 11

Fig.7b UV



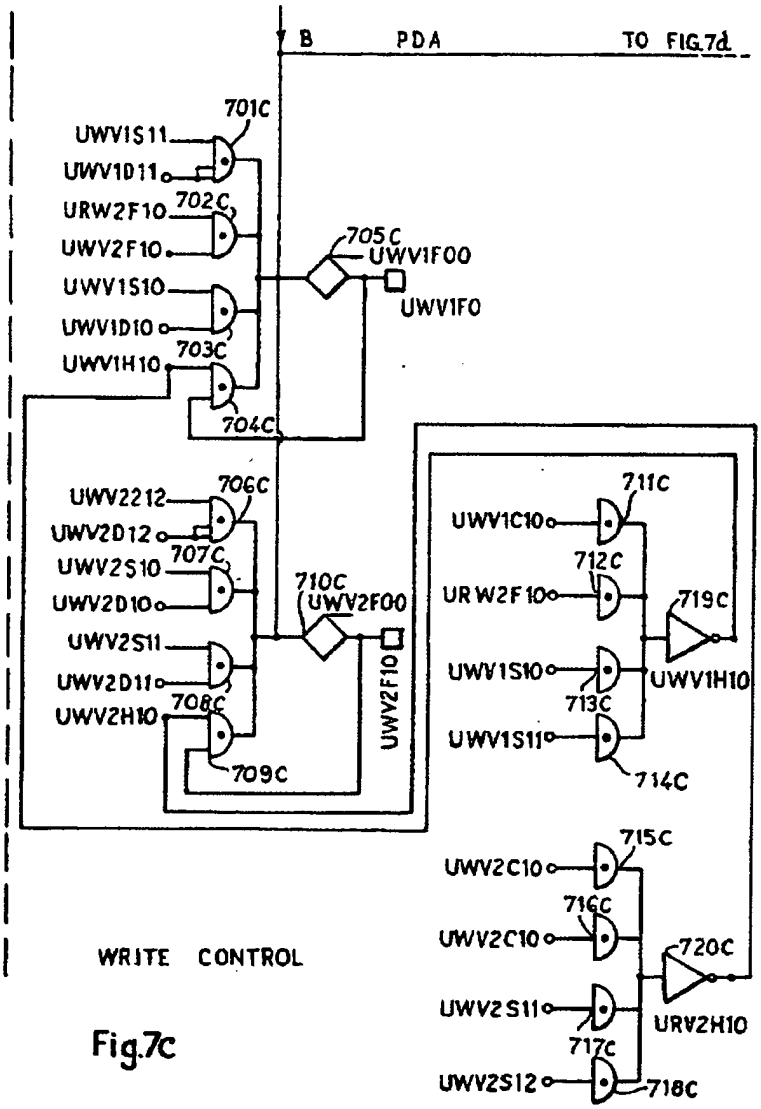


Fig.7c

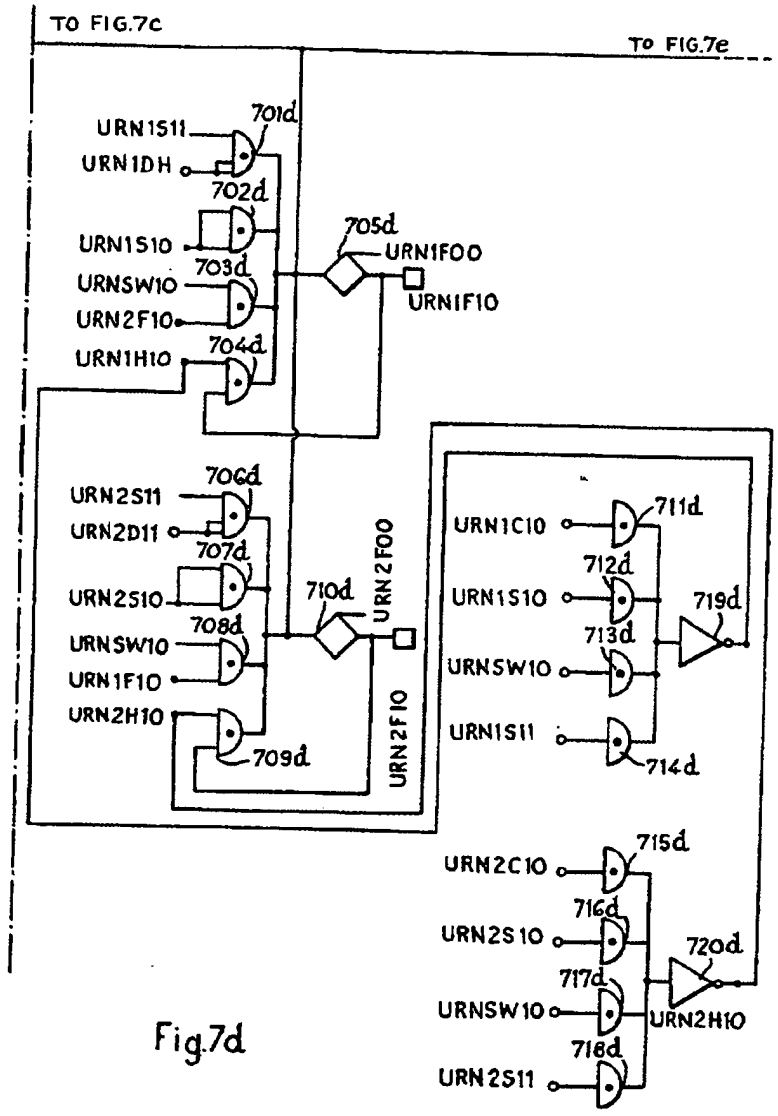

TO FIG.7c
TO FIG.7e
701d
URN1S11
URN1DH
702d
URN1S10
705d
URN1F00
703d
URNSW10
URN1F10
URN2F10
704d
URN1H10
706d
URN2S11
URN2D11
707d
URN2S10
710d
URN2F00
708d
URNSW10
URN1F10
URN2H10
709d
URN2F10
711d
URN1C10
712d
URN1S10
719d
713d
URNSW10
URN1S11
714d
715d
URN2C10
716d
URN2S10
720d
717d
URNSW10
URN2H10
718d
URN2S11

Fig.7d

1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 14

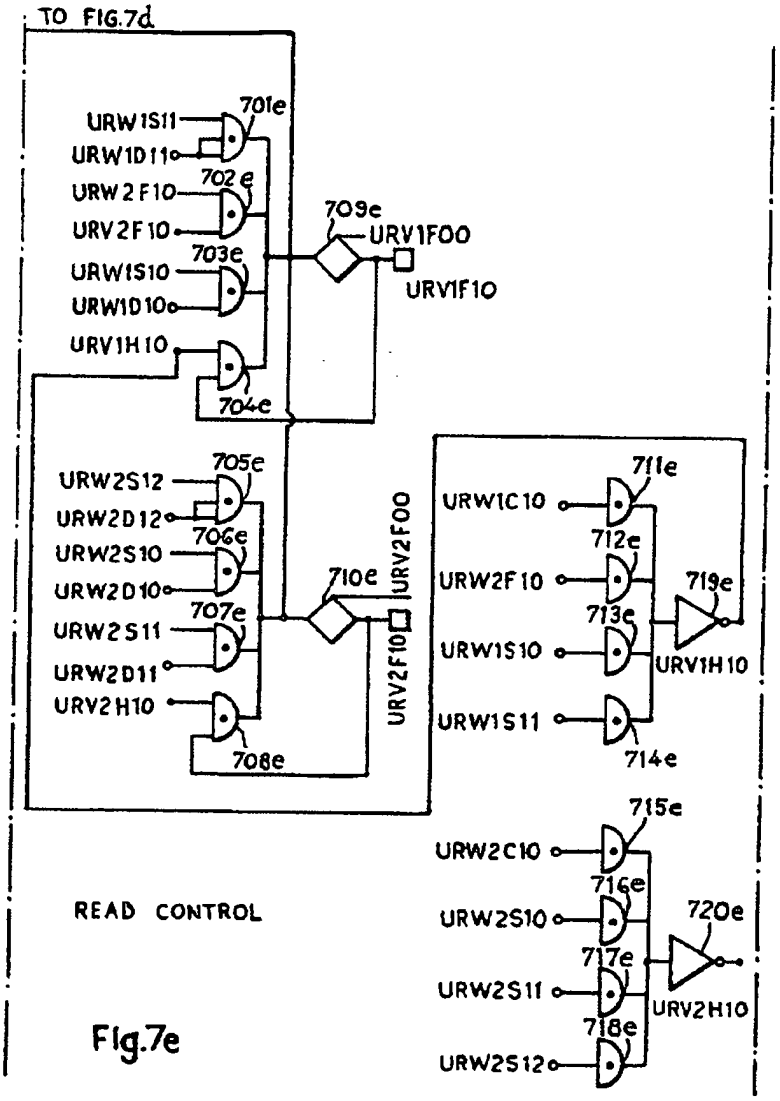
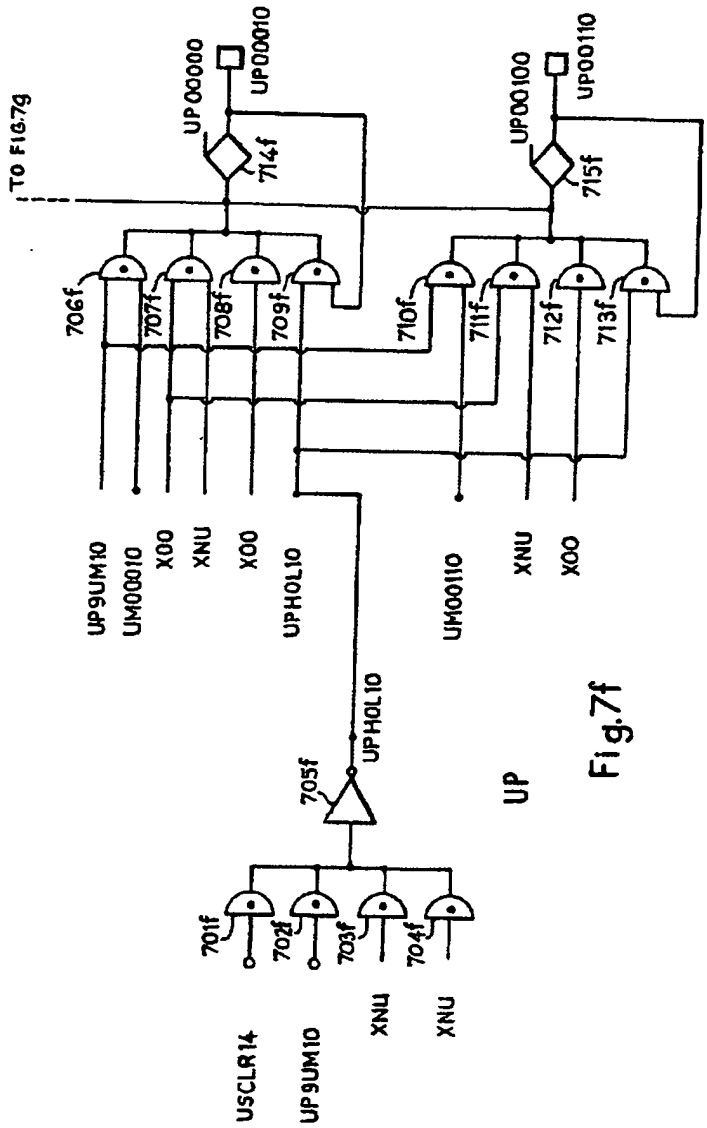


Fig.7e

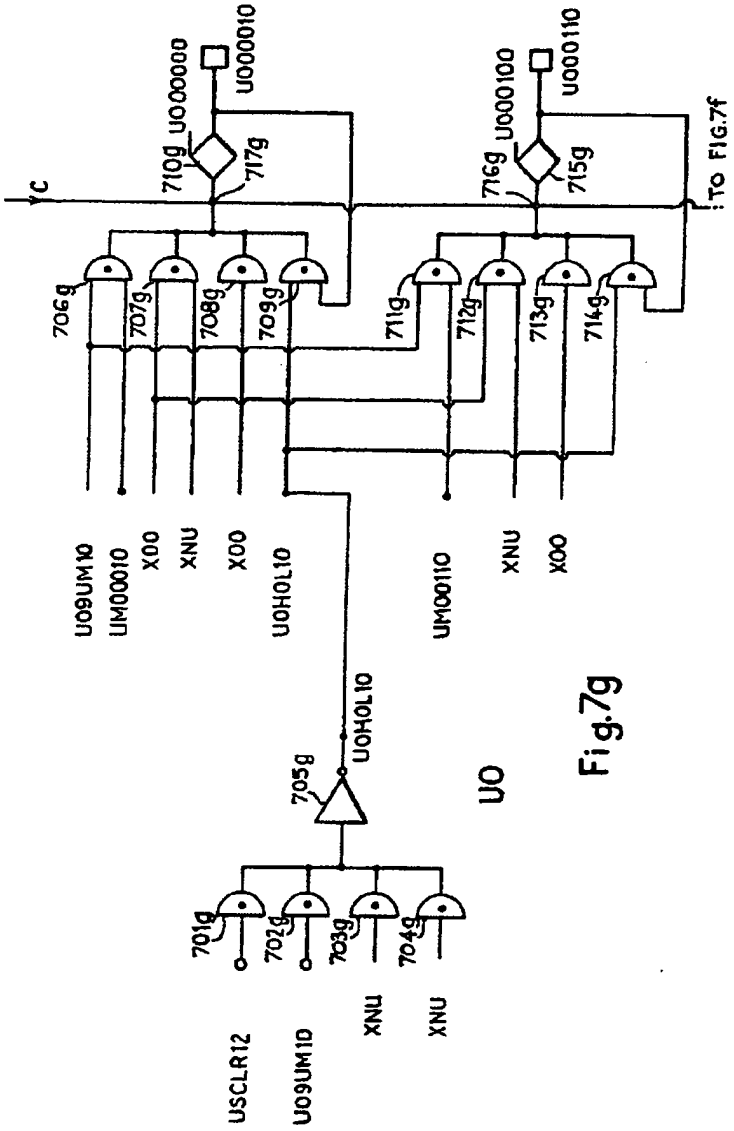
1483282 COMPLETE SPECIFICATION
29 SHEETS This drawing is a reproduction of the Original on a reduced scale
Sheet 15



UP

Fig.7f

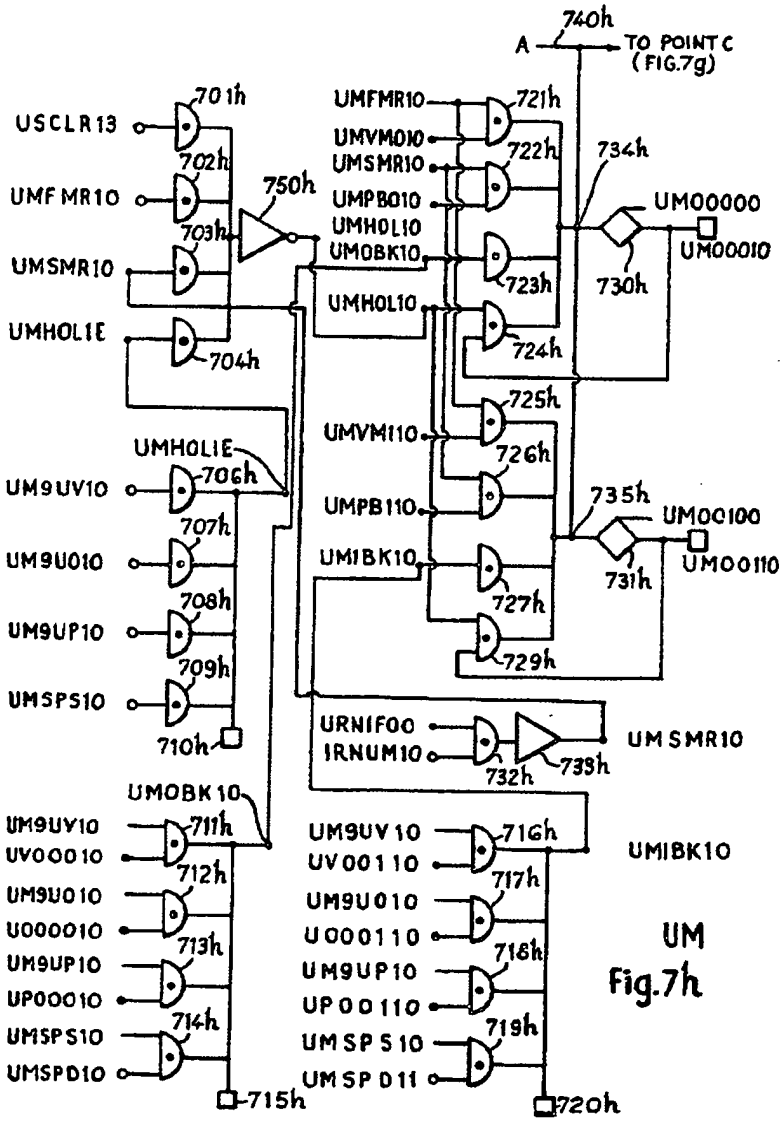
1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 16



U0

Fig.7g

1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 17



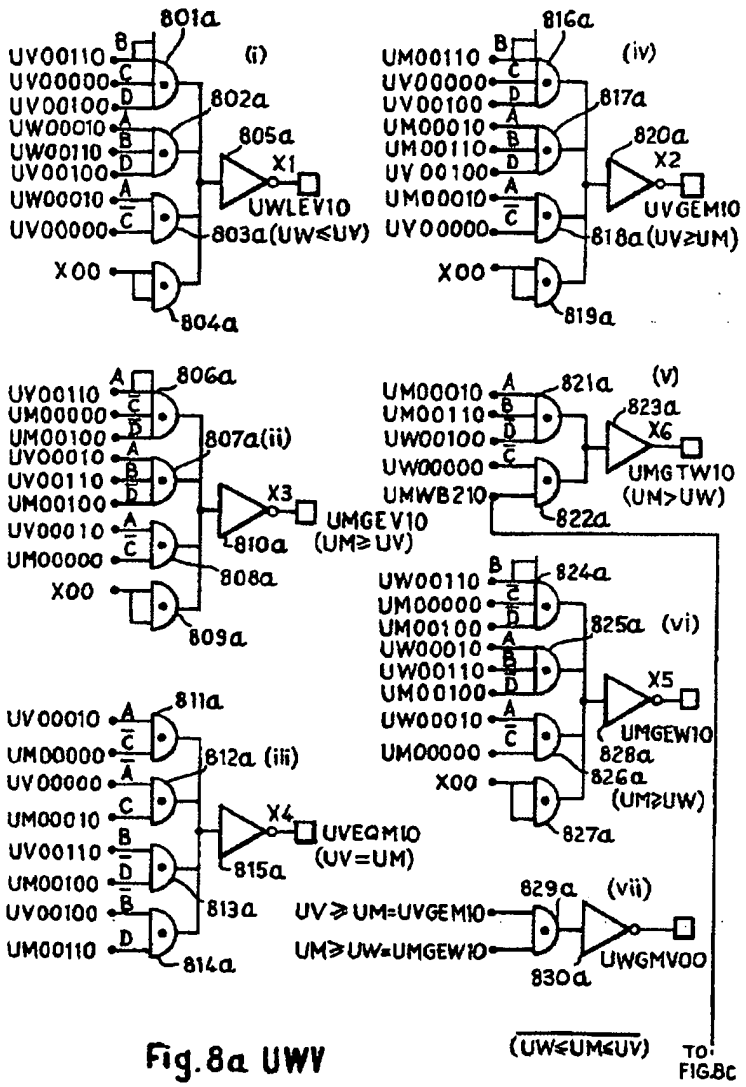

801a
(i)
UV00110
UV00000
UV00100
UW00010
UW00110
UV00100
UW00010
UV00000
X00
802a
805a
X1
UWLEV10
803a(UW≤UV)
804a
816a
(iv)
UM00110
UV00000
UV00100
UM00010
UM00110
UV00100
UM00010
UV00000
X00
817a
820a
X2
UVGEM10
818a(UV≥UM)
819a
806a
UV00110
UM00000
UM00100
UV00010
UV00110
UM00100
UV00010
UM00000
X00
807a(ii)
X3
UMGEV10
(UM≥UV)
810a
808a
809a
821a
(v)
UM00010
UM00110
UW00100
UW00000
UMWB210
823a
X6
UMGTW10
(UM>UW)
822a
811a
UV00010
UM00000
UV00000
UM00010
UV00110
UM00100
UV00100
UM00110
812a (iii)
X4
UVEQM10
(UV=UM)
815a
813a
814a
824a
UW00110
UM00000
UM00100
UW00010
UW00110
UM00100
UW00010
UM00000
X00
825a
(vi)
X5
UMGEW10
828a
826a
(UM≥UW)
827a
829a
(vii)
UV≥UM=UVGEM10
UM≥UW=UMGEW10
UWGMV00
830a
(UW≤UM≤UV)
TO FIG.8c

Fig.8a UWV

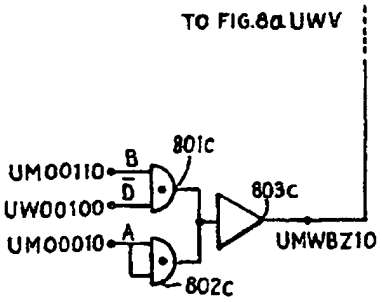


Fig.8c

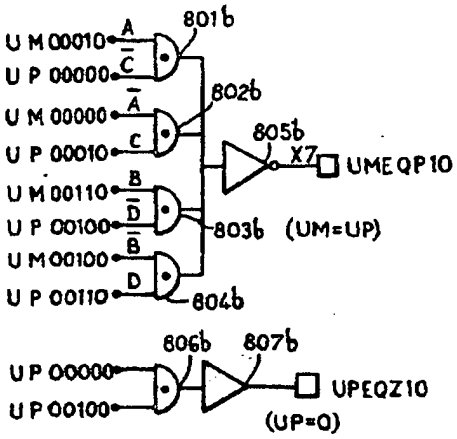


Fig.8b UEP

1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 20

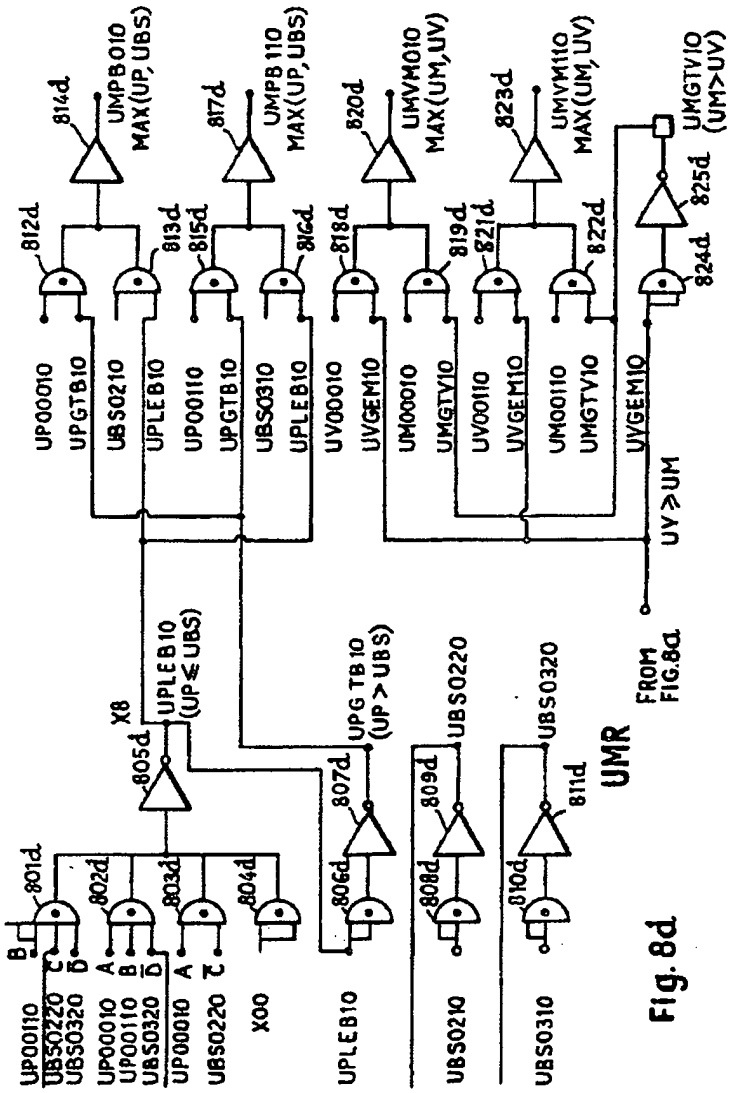


Fig. 8d

KEY TO SYMBOLS

Fig. 9a ——— INTERNAL SIGNAL SOURCE

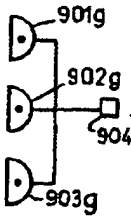
Fig. 9b ——— OUTPUT PIN

Fig. 9c ——— INPUT PIN

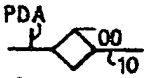
Fig. 9d  AND GATE

Fig. 9e  AMPLIFIER

Fig. 9f  INVERTER

Fig. 9g 

"AND-OR" GATES

Fig. 9h 

FLIP-FLOP

Fig. 9i ⊙——— MICRO-OPERATION

Fig. 9j X::Y——— X::Y

Fig. 9k a::β——— START OF BIT α WHERE THERE ARE β BIT POSITIONS INCLUDING BIT α

1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 22

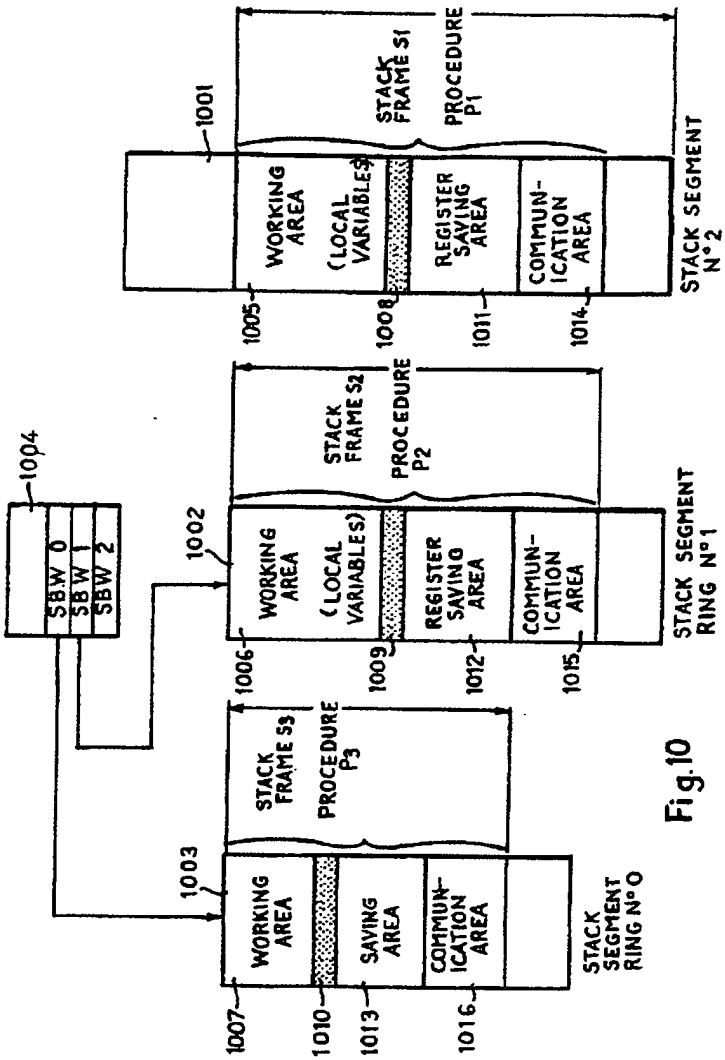


Fig.10

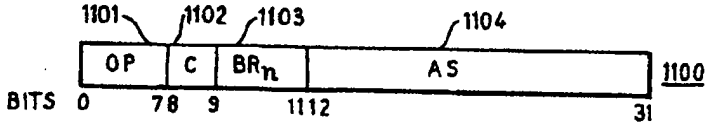


Fig. 11A

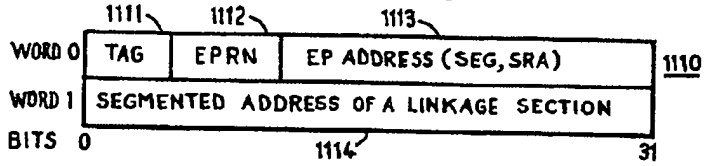


Fig. 11B

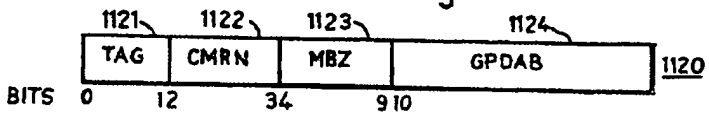


Fig. 11C

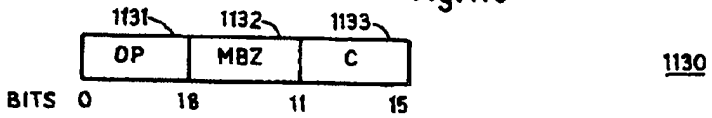


Fig. 11D

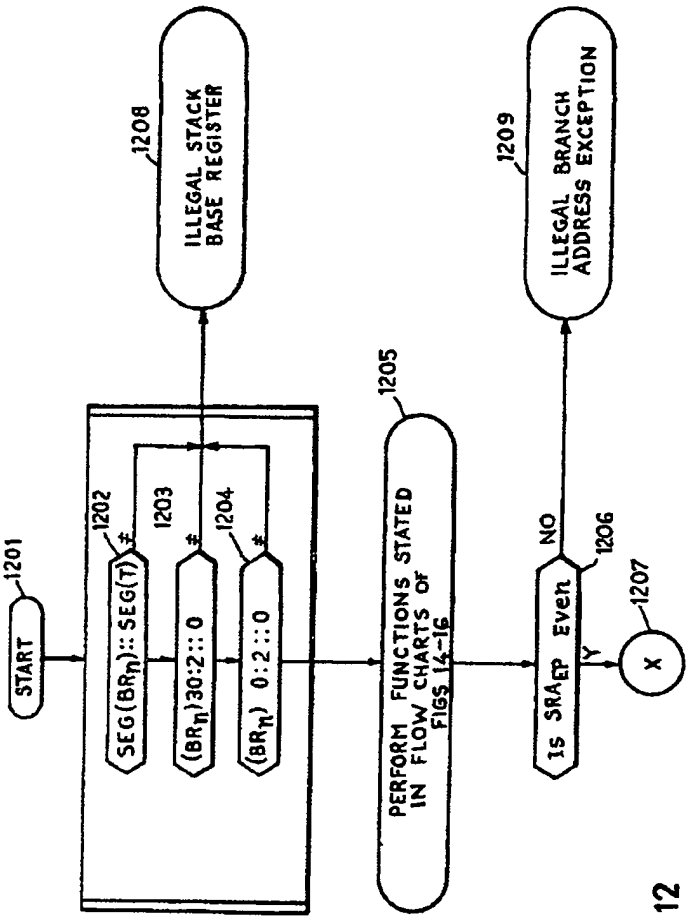

START
1201
SEG(BRn):: SEG(T)
1202
(BRn)30:2:: 0
1203
(BRn) 0:2:: 0
1204
ILLEGAL STACK BASE REGISTER
1208
PERFORM FUNCTIONS STATED IN FLOW CHARTS OF FIGS 14-16
1205
IS SRAEP Even
NO
Y
1206
ILLEGAL BRANCH ADDRESS EXCEPTION
1209
X
1207

Fig. 12

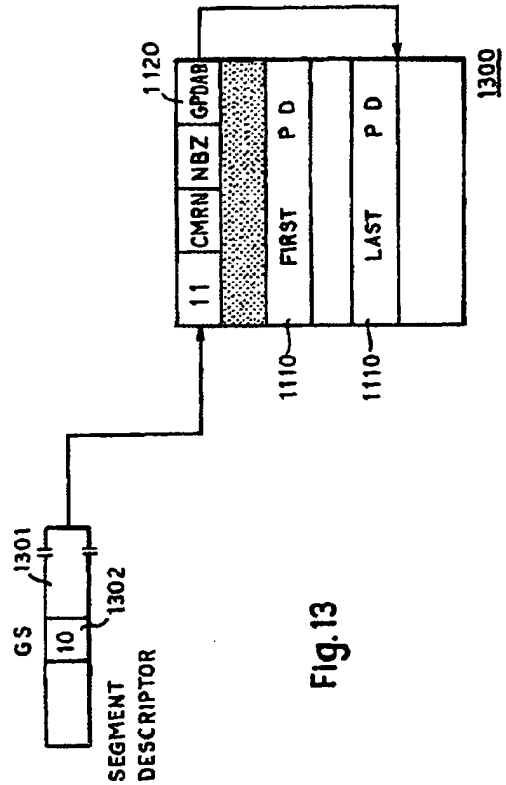

1120
GPDAB
NBZ
CMRN
11
FIRST
P D
LAST
P D
1300
1110
1110
1301
1302
GS
10
SEGMENT DESCRIPTOR

Fig.13

1483282 COMPLETE SPECIFICATION
29 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 26

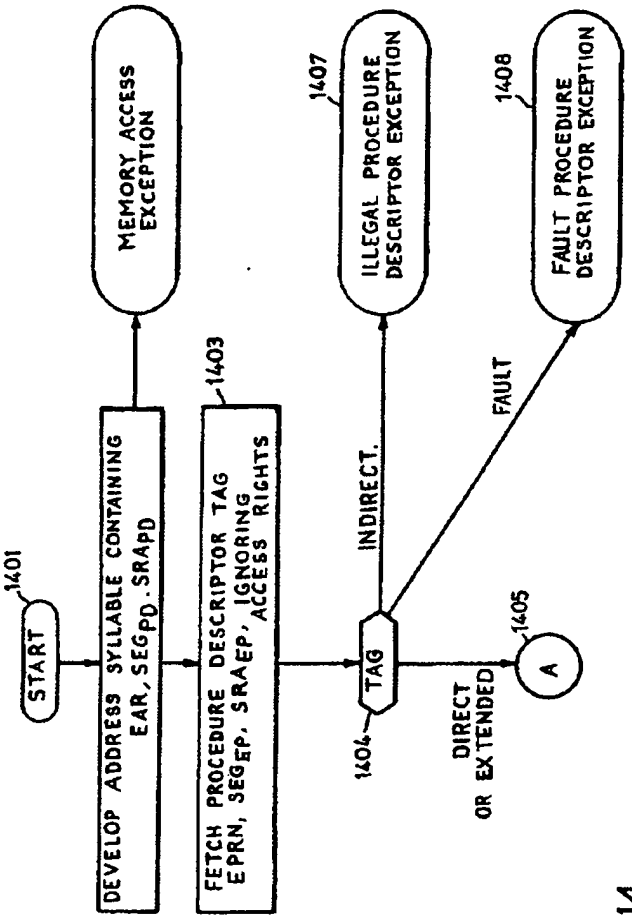


Fig.14

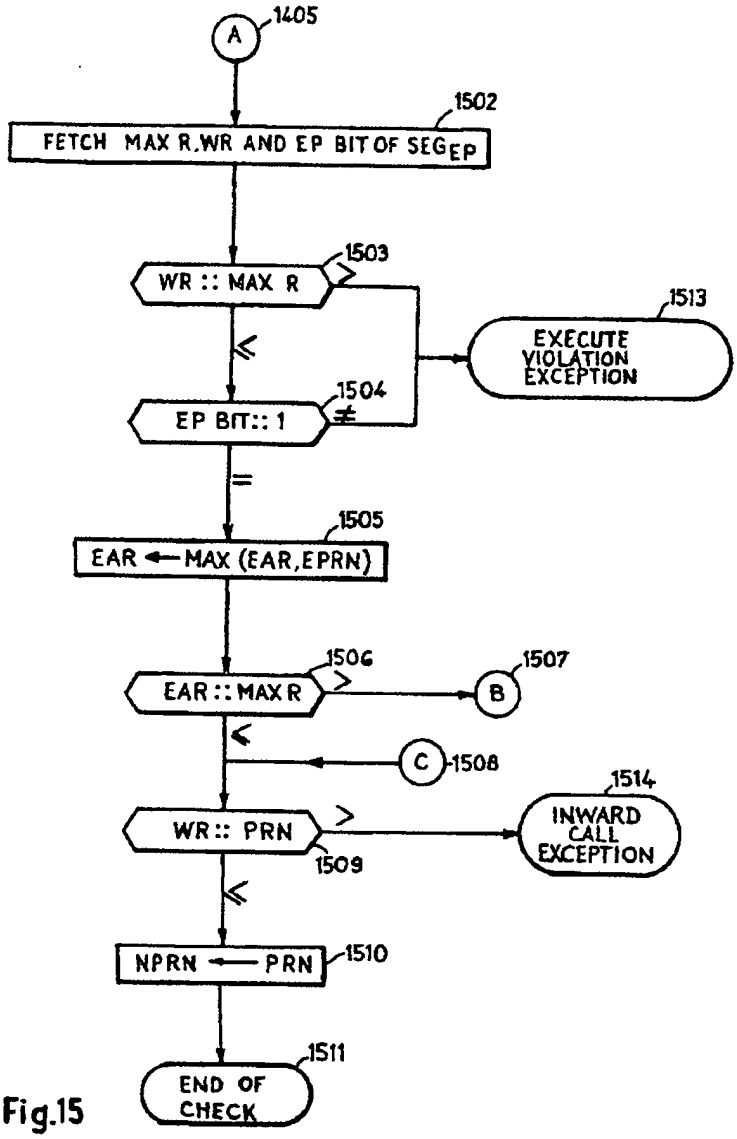

A
1405
1502
FETCH MAX R,WR AND EP BIT OF SEG EP
1503
WR :: MAX R
1513
EXECUTE VIOLATION EXCEPTION
1504
EP BIT:: 1
1505
EAR ← MAX (EAR,EPRN)
1506
EAR :: MAX R
1507
B
C
1508
1514
WR :: PRN
INWARD CALL EXCEPTION
1509
NPRN ← PRN
1510
1511
END OF CHECK

Fig.15

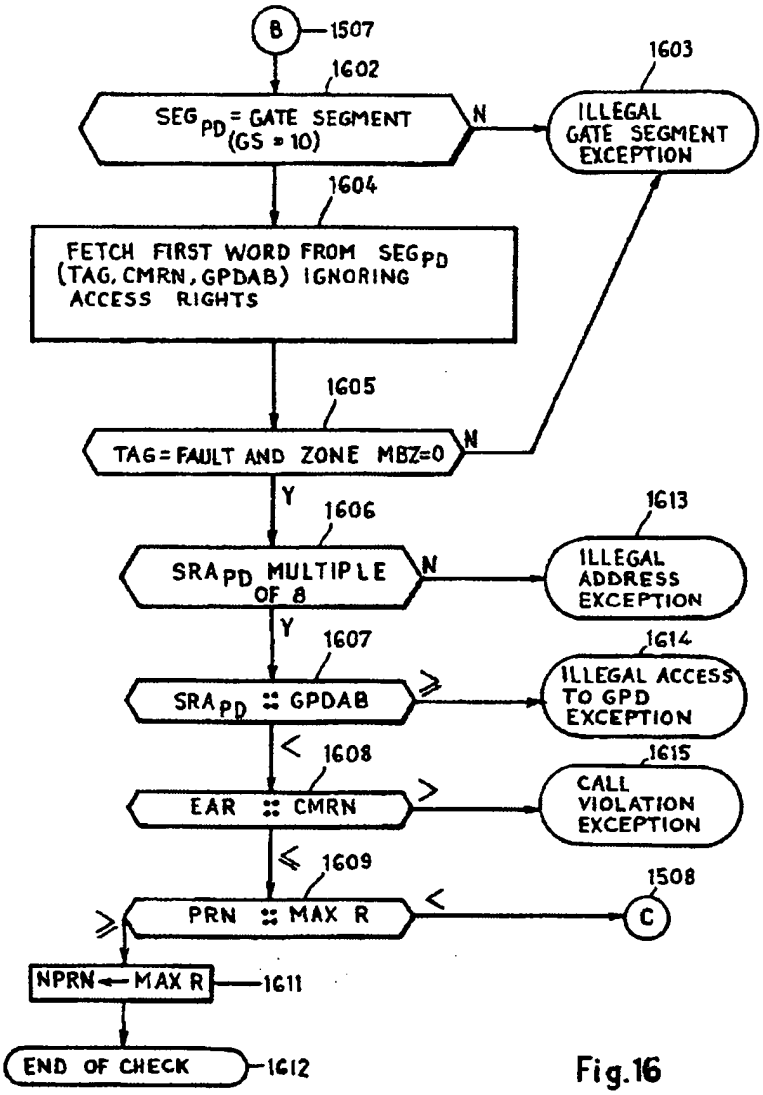

B
1507
1602
SEG PD = GATE SEGMENT (GS = 10)
N
1603
ILLEGAL GATE SEGMENT EXCEPTION
1604
FETCH FIRST WORD FROM SEG PD (TAG, CMRN, GPDAB) IGNORING ACCESS RIGHTS
1605
TAG = FAULT AND ZONE MBZ=0
N
Y
1606
SRA PD MULTIPLE OF 8
N
1613
ILLEGAL ADDRESS EXCEPTION
Y
1607
SRA PD :: GPDAB
≥
1614
ILLEGAL ACCESS TO GPD EXCEPTION
<
1608
EAR :: CMRN
>
1615
CALL VIOLATION EXCEPTION
≤
1609
PRN :: MAX R
<
1508
C
≥
NPRN ← MAX R
1611
END OF CHECK
1612

Fig.16

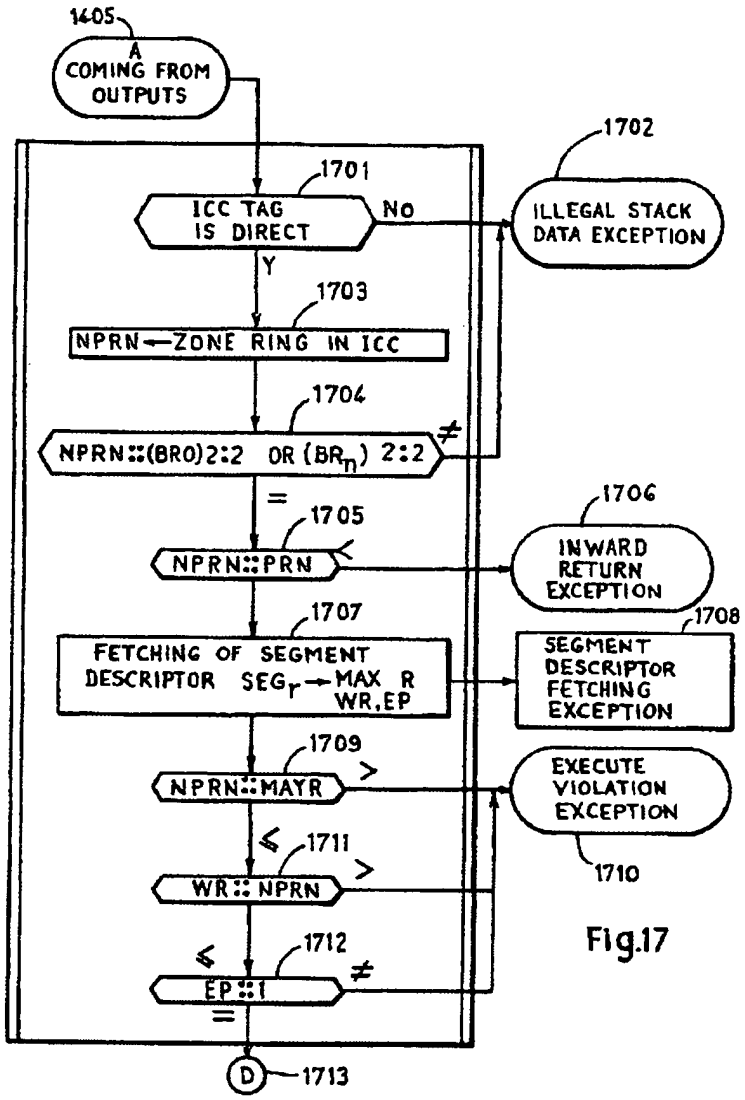

1405
A
COMING FROM OUTPUTS
1701
ICC TAG IS DIRECT
No
Y
1702
ILLEGAL STACK DATA EXCEPTION
1703
NPRN←ZONE RING IN ICC
1704
NPRN::(BRO)2:2 OR (BRn) 2:2
≠
=
1705
NPRN::PRN
1706
INWARD RETURN EXCEPTION
1707
FETCHING OF SEGMENT DESCRIPTOR SEGr→MAX R WR,EP
1708
SEGMENT DESCRIPTOR FETCHING EXCEPTION
1709
NPRN::MAYR
>
EXECUTE VIOLATION EXCEPTION
1710
≤
1711
WR::NPRN
>
≤
1712
EP::1
≠
=
D
1713

Fig.17

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 892 521 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
20.01.1999 Bulletin 1999/03

(51) Int Cl.[6]: **H04L 9/32**

(21) Application number: **98305646.6**

(22) Date of filing: **15.07.1998**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **15.07.1997 US 892792**

(71) Applicant: **Hewlett-Packard Company**
Palo Alto, California 94304 (US)

(72) Inventor: **Zamek, Steven**
Palo Alto, California 94303 (US)

(74) Representative: **Jehan, Robert et al**
Williams, Powell & Associates,
4 St Paul's Churchyard
London EC4M 8AY (GB)

(54) Method and apparatus for long term verification of digital signatures

(57) The time over which a digital signature can be verified is extended well beyond the expiration of any or all of the certificates upon which that signature depends. In a "save state" approach, an archive facility is used to store public key infrastructure (PKI) state, e.g. cryptographic information, such as certificates and certificate revocation lists (CRLs), in addition to non-cryptographic information, such as trust policy statements or the document itself. This information comprises all that is necessary to re-create the signature verification process at a later time. When a user wants to verify the signature on a document, possibly years later, a long term signature verification (LTSV) server re-creates the precise state of the PKI at the time the document was originally submitted. The LTSV server restores the state, and the signature verification process executes the exact process it performed (or would have performed) years earlier. Another embodiment of the invention combines the strength of cryptography with the proven resilience of (non-public key) technology and procedures currently associated with secure data stores by saving the PKI state for future verification; and protecting the PKI state information from intrusion by maintaining it in a secure storage facility which is protected by services, such as firewalls, access control mechanisms, audit facilities, intrusion detection facilities, physical isolation, and network isolation.

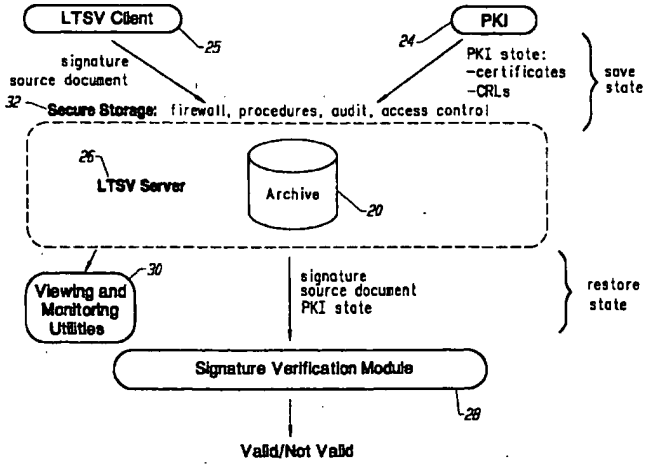


FIG. 3

EP 0 892 521 A2

Printed by Jouve, 75001 PARIS (FR)

Description

This invention relates to a method and apparatus for the long term verification of digital signatures.

The technology of digital signatures opens up the likelihood of increased use of digital networks (including the Internet) for electronic commerce. It is now feasible to send and receive digitally signed documents that represent transactions of some value to one or more parties.

Currently, a digital signature is verifiable only as long as the digital certificates upon which it depends have not expired. Given the expectation that a certificate's life span is in the area of one to two years duration, current technology does not support the emerging needs of the electronic commerce market, where the durability of digital signatures over time is a requirement.

For certain applications, the recipient of digitally signed documents should be able to verify the authenticity of a document years after the document was signed, just as the document's authenticity can be verified at the time of signing. Unfortunately, the current state of the technology does not provide for the verification of these digital signatures after certificate expiration because it is the nature of keys and certificates used for signing and encrypting documents to expire after a specific period of time (typically after a year or two). This is due, at least in part, to the fact that the strength of keys is expected to degrade over time because of such factors as improvements in computing speed and breakthroughs in cryptoanalysis. Moreover, the longer the key is in use, the longer that an adversary has to attempt to crack the key. Therefore, it is standard practice to replace keys periodically. This is why certificates have specific expiration dates.

An examination of the current state of the technology reveals that a digital signature verification module would fail if presented with a request to verify a signed document in which any of the associated certificates had expired. Fig. 1 is a block schematic diagram illustrating certification expiration. This simple example demonstrates that, given a certificate 10 having a two-year life span *(e.g.* from 4/1/96 to 4/1/98), a signature could be successfully verified six months *(e.g.* on 10/1/96) after certificate issuance (100); but this same signature would not be successfully verified three years later *(e.g.* on 4/1/99) (102). This behavior is clearly unacceptable if the duration of a document, for example contract, must extend beyond the duration of the certificates' life.

Further, some current systems use certificate revocation lists (CRLs) to revoke certificates and remove them therefrom, once those certificates expire. This means that a record of those CRLs generally disappears, making long term signature verification impossible using known techniques.

It is known to reconstruct past trust (see A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography, CRC Press, pp. 583 (1996)). In this approach, both signature reverification relative to a past point in time and resolution of disputes may require reconstruction of chains of trust from a past point in time. This requires archival of keying material and related information for reconstruction of past chains of trust. Direct reconstruction of such past chains is taught to be unnecessary if a notarizing agent is used. A notarizing agent is defined as a general service capable not only of ascertaining the existence of a document at a certain time, but of vouching for the truth of more general statements at certain points in time. The original verification of the notary is taught to establish the existence of a trust chain at that point in time, and subsequently its record thereof is taught to serve as proof of prior validity. It is taught that details of the original trust chain may be recorded for audit purposes. It is not taught that a document can be verified based upon the existence of expired certificates. Rather, reliance is placed upon the use of the notarizing agent. It is further taught that the archived keying material can be used as evidence at a future time to allow resolution of disputed signatures by non-automated procedures.

It would be advantageous to provide a technique for extending the time over which the authenticity and integrity of digital signatures can be accurately verified beyond the time that any relevant certificates expire.

The present invention seeks to provide improved signature verification.

According to an aspect of the present invention there is provided a method of enabling long term verification of digital signatures as specified in claim 1.

According to another aspect of the present invention there is provided apparatus as specified in claim 11.

The prefered embodiment provides a method and apparatus which effectively extends the time over which a digital signature can be verified, *i.e.* well beyond the expiration of any or all of the certificates upon which that signature depends. The invention can be used for any application domain where users want digital signatures to be applied to long lasting documents *(e.g.* contracts), and be independently verifiable years or decades after signing the document. The preferred embodiment provides two alternative approaches to constructing a solution which delivers long term signature verification (LTSV).

One embodiment of the invention provides an approach for solving the LTSV problem that is referred to herein as the "save state" approach. This embodiment of the invention largely entails the use of cryptographic information and techniques. Thus, an archive facility is used to store the public key infrastructure (PKI) state, *e.g.* cryptographic information, such as certificates and CRLS, in addition to the document itself. This information comprises all that is necessary to re-create the signature verification process at a later time. It may also be desirable to store the source document separately from the cryptographic information (such as the signature, certificates, and CRLs) for reasons of privacy. For ex-

ample, a user may want to have control over the source document. The PKI state information may contain either or both of cryptographically protected information, such as certificates and CRLs, and information that is not cryptographically protected, such as the public key of a root certification authority or policy information.

When a user wants to reverify the signature on a document, possibly years later, an LTSV server re-creates the precise state of the PKI at the time the document was originally submitted. The LTSV server restores the state, and the signature verification process executes the exact process it performed (or would have performed) years earlier. The time used as the basis for re-creation of the signature verification process does not have to be the time of submittal. Rather, the time could be some other relevant time, such as when a document was signed by the originator or when it was verified by a recipient.

Another embodiment of the invention combines the strength of cryptography with the proven resilience of (non-public key) technology and procedures currently associated with secure data stores. An example of this embodiment provides a mechanism that:

- Saves the PKI state for future reverification; and
- Protects the PKI state information from intrusion by either maintaining it in a secure storage facility which is protected by services, such as firewalls, access control mechanisms, audit facilities, intrusion detection facilities, physical isolation, and network isolation; and/or employing a cryptographic protection mechanism, for example using the LTSV server to sign the PKI state information or using a keyed hash algorithm.

In addition, other non-cryptographic features may be added to such approaches to deliver a highly secure and trusted LTSV solution, including, for example utilities for viewing the PKI state information (cryptographic as well as non-cryptographic) and visually monitoring the security of the system. These utilities can be used to provide visual evidence for purposes of dispute resolution.

One enhancement to the secure storage approach herein disclosed maintains certain evidence, such as certificate chains, in an archive. This information need not be used for actual reverification, but merely as supporting evidence in case of a dispute.

An embodiment of the present invention is described below, by way of example only, with reference to the accompanying drawings, in which:

Fig. 1 is a block schematic diagram illustrating certification expiration;

Fig. 2 is a block schematic diagram illustrating a "save state" embodiment of the invention;

Fig. 3 is a block schematic diagram illustrating a "save state" "secure storage" embodiment of the invention;

Fig. 4 is a flow diagram that provides two alternative scenarios that illustrate the applicability of time stamps to the preferred embodiments;

Figs. 5a-5c provide block schematic diagrams that illustrate three long term signature verification usage scenarios;

Fig. 6 is a block schematic diagram that illustrates trust between two entities ; and

Fig. 7 is a block schematic diagram that illustrates a long term signature verification trust model.

The meanings of some of the terms used herein may differ somewhat from common usage. The following definitions are meant to clarify the meaning of each in the context of its usage herein.

Archive: Any facility for the storage and retrieval of electronic information.

Certificate: An artifact upon which digital signatures are based. A certificate securely binds an entity with that entity's public key.

Cryptographic Refresh: A means of solving the key degradation problem when storing cryptographic information for long periods of time. The process involves re-encoding the old cryptographic artifacts *(e.g.* encrypted data, digital signatures, and message digests) with stronger algorithms and/or longer keys.

Document: A document can be any information which can be represented electronically or optically *(e. g.* an arbitrary bit stream).

Key Degradation/Algorithm Degradation: The process whereby the protection afforded a document by encryption under a key loses effectiveness over time. For example, due to factors such as improvements in computing speed and breakthroughs in cryptoanalysis, it is expected that a document securely encrypted today would be crackable years later. This property could affect any cryptographic information, including digital signatures. This problem can be generalized to keyed and non-keyed cryptographic processes and artifacts, such as one-way hash algorithms. The security provided by these are also expected to diminish over time.

LTSV: Long Term Signature Verification. The herein described method and apparatus for verifying a digital signature after the certificates used for such verification have expired.

LTSV client: The entity which requests/utilizes the services of the LTSV server.

LTSV server: The entity which delivers the LTSV services. This does not imply, however, that this entity must be stand-alone component.

LTSV submission: A request from an LTSV client to

an LTSV server to perform the necessary functions required to enable reverification of a digital signature some time in the future *(e.g.* save PKI state).

PKI: Public Key Infrastructure. Refers to all components, protocols, algorithms, and interfaces required to deliver the capabilities to digitally sign and verify documents. For purposes of clarity herein, a PKI does not include a service module for long term signature verification (LTSV server), although in practice a PKI might be designed to encompass such a module.

Signature Reverification: The re-creation of the digital signature verification process after the original verification. This specifically refers to the process associated with the verification process, based upon the restoration of the previously saved PKI state.

Signature Verification: The process by which a digital signature, for a given document, is determined to be authentic or not.

Signature Verification Module: The module which is responsible for performing the verification of digital signatures.

Time stamp: A digital time stamp is an electronic indicator which associates the current date and time with a particular document. Time stamps are useful for proving that a document existed at a particular time. It is desirable that time stamps be secure, durable over time, and trusted by those using them.

The discussion herein assumes an understanding of public key, digital signatures, and PKI infrastructure using X.509 certificates. Practical information concerning application of such techniques is considered to be well known to those skilled in the art. Background information may be found, for example, in B. Schnier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, John Wiley & Sons, Inc. (1996); W. Ford, M. Baum, Secure Electronic Commerce, Prentice Hall PTR (1997); and in the X.509 v.3 specification ([X.509-AM] ISO/IEC JTC1/SC 21, Draft Amendments DAM 4 to ISO/IEC 9594-2, DAM 2 to ISO/IEC 9594-6, DAM 1 to ISO/IEC 9594-7, and DAM 1 to ISO/IEC 9594-8 on Certificate Extensions, 1 December 1996). The system described herein may be built upon the X.509 infrastructure.

The following discussion provides some background on cryptographic techniques. Cryptographic algorithms can generally be divided into two categories: public key *(e.g.* RSA) and secret key *(e.g.* DES). Both types of algorithms transform plain text into cypher text using a key(s) for the encryption and decryption processes.

Both public key and secret key algorithms are considered to be secure. One is not better than another in terms of security. The strength of each algorithm, in terms of it being cracked, is largely a function of the length of the key used. The primary distinguishing characteristic of public key, however, is that it uses two keys (one to encrypt and another to decrypt), while secret key algorithms use only one key (the same key is used for encryption and decryption). For this reason, secret key algorithms are sometime referred to as symmetric algorithms and public key algorithms are called asymmetric.

One problem with secret key algorithms is that a key must be distributed between all participants. This means that some secure channel must be available for the distribution of the keys.

In practice, each entity in a public key-based system has a key pair, *i.e.* one private key and one public key. The private key is known only to its owner, the public key is known to all correspondents. It is computationally infeasible to determine a private key from the public key.

The two primary services provided by public key cryptography are secure exchange of symmetric keys (by using public key techniques to encrypt a symmetric session key), and non-repudiation via digital signatures.

Public key cryptography can be used to solve the key exchange problem associated with secret key algorithms by using this technology to encrypt the secret key under the public key of the recipient. It can then be decrypted by the recipient using his/her private key.

Digital signatures are possible by encrypting data with the private key of the signing entity. Any entity can decrypt it with the signer's publicly available public key and know that no one else could have encrypted it because that private key is only known by that one individual. This particular use of public key provides the non-repudiation service, which is a primary use of public key cryptography. A digital signature is very powerful notion, it generally exhibits the following characteristics:

- Cannot be forged;
- Is independently verifiable;
- Is not reusable or transferable to a different piece of data; and
- Includes data integrity checks, allowing tamper-detection.

The new services provided by public key cryptography do not come for free, however, because these services require the existence of a supporting public key infrastructure. The strength of a public key system depends upon the assurance that all participants know the public key of any entity with whom they wish to correspond. If a secure correspondence between a user and his/her public key cannot be maintained, then it may be possible to impersonate another entity or read encrypted data intended for another.

The standard solution to this problem is the issuance of a digital certificate (X.509 certificate) to each participant. This certificate securely binds its owner's name with his/her public key. It is issued by a trusted third party, called a certification authority (CA), and is signed by that CA, thereby making it tamper proof. Certificates are issued for a limited period of time (start and

stop dates), during which the certificate is considered valid. A certificate is considered expired after the ending validity date.

The public keys of entities (which are embedded in the X.509 certificates) must be publicly available. The distribution or access mechanisms available are numerous.

The secure operation of a public key infrastructure rests upon certain points of trust. Certainly each entity must trust its own CA. However, when a given PKI domain is expanded to encompass relationships with multiple CAs, the number of points of trust are also expanded. The trust placed in a particular end entity *(i.e.* that entity's certificate or signature) is directly related to the trust relationships among the CAs which certify those entities.

CAs can create trust relationships with other CAs by certifying each other. This can be a unidirectional trust relationship, whereby one CA can merely issues a certificate to another CA, just as a CA issues a certificate to an end user. Two CAs can also mutually agree to trust each other (bidirectional trust relationship) by issuing a cross-certificate -- a special form of certificate which contain two individual certificates, one for each direction.

If two entities are in the same CA domain, then there is no concern with respect to CA trust because they both trust the same CA. This is not the case, however, when dealing with the scenario where entities which have been certified by different CAs attempt to conduct a secure transaction. The security of this transaction depends upon the trust between the CAs. More generally, the security provided by the PKI depends upon the trust models embodied in the trust relationships among the various CAs which choose to trust one another. In concrete terms, any change in these trust relationships can cause a signature verification to either succeed or fail.

The preferred method and apparatus effectively extend the time over which a digital signature can be verified, *i.e.* well beyond the expiration of any or all of the certificates upon which that signature depends. They can be used for any application domain where users want digital signatures to be used on long lasting documents *(e.g.* contracts), and be independently verifiable years or decades after signing the document. The preferred embodiment of the invention provides two alternative approaches to constructing a solution which delivers long term signature verification (LTSV).

Fig. 2 is a block schematic diagram illustrating a "save state" embodiment of the invention. This embodiment, largely entails the use of cryptographic information and techniques. Thus, an archive facility 20 is used to store a public key infrastructure (PKI) state 24, *e.g.* cryptographic information, such as certificates and CRLS, in addition to the source document itself. For example, the LTSV client 25 requests the services of an LTSV server 26 to accomplish storage of such information. This step is shown as the "save state" step in Fig. 2. The PKI state information may contain either or both of cryptographically protected information, such as certificates and CRLs, and information that is not cryptographically protected, such as the public key of a root certification authority or policy information.

This information comprises all that is necessary to re-create the signature verification process at a later time, *i.e.* during the "restore state" step, for example, as requested by the LTSV client. It may also be desirable to store the source document separately from the cryptographic information (such as the signature, certificates, and CRLs) for reasons of privacy. For example, a user may want to have control over the source document.

When a user wants to reverify the signature on a document, possibly years later, the LTSV server 26 recreates the precise state of the PKI at the time the document was originally submitted. The LTSV server restores the state, and the signature verification process 28 executes the exact process it performed (or would have performed) years earlier. The time used as the basis for re-creation of the signature verification process does not have to be the time of submittal. Rather, the time could be some other relevant time, such as when a document was signed by the originator or when it was verified by a recipient.

Fig. 3 is a block schematic diagram illustrating a "save state" "secure storage" embodiment of the invention. This embodiment of the invention combines the strength of cryptography with the proven resilience of (non-public key) technology and procedures currently associated with secure data stores. An example of this embodiment:

- Saves the PKI state for future reverification (as described above in connection with Fig. 2); and
- Protects the PKI state information from intrusion by maintaining it in a secure storage facility which is protected by services, such as firewalls, access control mechanisms, audit facilities, intrusion detection facilities, physical isolation, and network isolation; and/or employing a cryptographic protection mechanism, for example using the LTSV server to sign the PKI state information or using a keyed hash algorithm.

In addition, other non-cryptographic features may be added to such approach to deliver a highly secure and trusted LTSV solution, including, for example utilities 30 for viewing the PKI state information (cryptographic as well as non-cryptographic) and visually monitoring the security of the system. These utilities can be used to provide visual evidence for purposes of dispute resolution.

One enhancement to the secure storage approach herein disclosed maintains certain evidence, such as certificate chains, in an archive. This information need

not be used for actual reverification, but merely as supporting evidence in case of a dispute. See A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography, CRC Press, pp. 583 (1996), for one manner in which this enhancement may be implemented in the context of a notary service (discussed above).

There are other embodiments of the invention in which a hybrid LTSV solution could be constructed by combining cryptographic and non-cryptographic techniques. The best combination for a particular application domain depends upon the security requirements of the application(s), in combination with cost constraints.

It is presently preferred to employ the second embodiment of the invention (discussed above) due to the cryptographic strength associated with its ability to recreate the complete digital signature verification process, combined with the trust instilled by more conventional techniques used for providing secure storage, and in conjunction with audit and viewing facilities with which to view evidence and monitor the secure storage controls. In practice, the most useful embodiment of the invention for a particular application may be that which is the least expensive and which still meets the user or application requirements.

Several issues related to the design of a system which implements LTSV are described below. Alternatives for the resolution of the issues are presented, as well as a discussion of the advantages and disadvantages associated with each alternative. The best approach to any given solution depends upon the security requirements of the application(s) using the LTSV services, as well as the cost constraints. There is no best solution for all applications.

When to Save the PKI State

Signature reverification is preferably associated with a particular time because the outcome of this process could change, depending upon the state of the PKI *(e.g.* because of certificate revocations or the creation/removal of cross certificates). There are numerous possibilities with regard to when the PKI state should be saved, including:

- At signature creation time. This approach is used when an individual wants to document the validity of his/her signature at the time it was created. This is the most accurate time to store the PKI state because it reflects the state at the time of signing, which is presumably the critical time in evaluating the authenticity of that signature. Changes to the PKI state occur after that time, some of which could impact the outcome of a signature reverification. Therefore, saving of the PKI state at any time after signing introduces the possibility of inconsistencies between the signer's and recipient's perspectives on a signature's validity.
- At signature verification time. This approach is useful when a recipient wants to document the validity of a signed document received from another individual.
- At archival time. When a user decides that a document should be archived for long term storage is also an appropriate time to save the PKI state.
- When explicitly requested. There may occur certain application specific document life cycle milestones, at which time the user may desire the PKI state to be saved for future reverification.
- Just before changes in PKI state *(e.g.* certificate revocation). This approach requires a tight integration with the PKI because changes in the PKI must be monitored.

The correct time at which to save the PKI state is preferably determined by the constraints and needs of the application using the LTSV services. A robust LTSV solution is able to accommodate the needs of all (or most) applications in this respect.

Contents of the PKI State.

The exact composition of the PKI state varies somewhat from one PKI vendor's product to another's, depending upon the implementation chosen by each vendor. Moreover, certain information is stored in a different format from one vendor to another. In addition, the contents of a PKI state may change over time as well, as new capabilities (and supporting data) are added to the system. Finally, the required contents of the PKI state may change from one application to another, depending upon the needs *(e.g.* level of security and legal requirements) of each application.

Notwithstanding these uncertainties, there are classes of PKI state information which are candidates for saving. These classes include:

- Certificate chain (list of certificates from one entity to another, including certification authorities (CAs) and the end entities).
- CRLs (one for each CA in certificate chain).

  CA policy statements or identifiers.
- Attribute certificates.
- Date and time.
- Trust information (e.g., public key(s) or certificate(s) of trusted root CA(s), policy constraints).

  Policy constraints are, for example, non-crypto-

graphic information stored within the LTSV archive. The public key of the trusted root CA may or may not be cryptographically protected. If it is embedded in a certificate, then it is signed by the CA. However, it could just as well be an isolated public key, in which case it is unprotected by cryptography.

It is possible that the items in the above list may not be supported or available from certain PKI implementations. Further, the PKI state from another implementation might include some additional data. Therefore, the list above is only an example of what might be considered important pieces of PKI state information, given the current state of the technology. An implementation of an LTSV service is preferably tied to the implementation of a specific PKI until such time as the technology evolves and comprehensive standards emerge.

How to Store the PKI State.

Storage of the PKI state is preferably accomplished in either of two general ways:

- Store all of the PKI state relevant to each document separately; and
- Store the PKI state centrally, and only store references to the PKI state information with each document. This approach enables storage efficiencies by eliminating the redundant storage of PKI state information over multiple documents. For example, given two documents submitted to the LTSV server at about the same time, it is possible that the CRLs contained in the PKI state are exactly the same for both submissions. Central storage of this information allows the LTSV server to store this information only once.

The storage requirements for the save state solution for LTSV may be quite large, depending upon the size of the certificates, the length of the certificate chains and -- more importantly -- the size of the CRLs. The choice of storage technique may have a great impact on the total data storage requirements. It is clearly undesirable to store massive CRLs with every document that is stored for long term archival and possible future reverification. For this reason, the second alternative listed above is presently considered to be the preferred approach.

However, this second approach may present certain difficulties in applications where the LTSV server is an entirely separate component from the PKI, and where support of multiple PKIs is a primary design goal of the LTSV server. In this case, it would be advantageous for the PKI state to remain opaque to the LTSV server, thereby providing ease of support of multiple PKI vendors. Given that what constitutes the PKI state for one vendor may be different for another vendor, it is desirable to maintain an opaque interface between the LTSV server and the PKI. On the other hand, storage efficiencies can be derived only if the LTSV server is informed about the contents and format of the PKI state information. These conflicting requirements -- acceptable storage size and opaqueness -- pose a challenge for the design of an LTSV service.

Some of the possible alternatives are listed below:

- Keep the interface opaque and store the PKI state as it currently exists (full certificate chains and CRLs). This option focuses entirely on the opaqueness requirement, and sacrifices the data size requirement. The primary advantage of this solution is simplicity and quick deployment.
- Remove the opaqueness requirement by making the PKI state visible to the LTSV server. This allows the LTSV server to manage the certificates and CRLs manually -- thereby avoiding duplication of these objects in the data store. This solution potentially sacrifices the ease of multi-vendor support at the expense of achieving efficient storage.
- Compromise by making the CRLs visible to the LTSV server, where other PKI state information is opaque. This solution is interesting because it is probable that the CRLs are the largest piece of data comprising the PKI state. Because CRLs are standard across nearly all PKIs, the visibility should not pose a problem in terms of multi-vendor support. This solution address both of the requirements, but does put the burden of management of the CRLs onto the LTSV server.
- An alternative embodiment of the invention provides a variation on the solution above that breaks up the PKI state into multiple pieces, each of which is opaque. The PKI indicates which of these objects is common across multiple signed documents *(e.g.* CRLs and certificates). The PKI labels these objects with unique handles (identifiers), thereby allowing the LTSV server to store these objects and retrieve them efficiently when needed for signature reverification -- all the while maintaining the opaqueness of these objects.
- Encourage PKI vendors to make concise cryptographically protected assertions about the state of revocation, as an alternative to using CRLs. (For example, CRLs indicate who has been revoked. It would be more efficient if the PKI could make a statement that a certificate has not been revoked at a given point in time. This could eliminate the need for storing CRLs.) This approach is non-standard, but acceptable because these PKI-generated assertions are not seen by any application outside the PKI. A major benefit of this approach is that the opaqueness of the state is preserved while some of

the storage inefficiencies of the state information are removed.

For cases where the LTSV server is dedicated to a particular PKI, it is preferred to create a close integration between the two components, thereby allowing the LTSV server to know about the content and format of the PKI state information, and store it in the most efficient manner possible. For cases where the LTSV server must be insulated from the PKI *(e.g.* for portability across multiple PKIs), one of the options listed above (with the possible exception of the first two) may be used.

Location of Source Data.

The source data associated with an LTSV submission can be stored either by the client or by the LTSV server itself. Some LTSV clients do not choose to submit clear text to the LTSV server for storage because of concerns over privacy. (Privacy of the channel between the LTSV client and the LTSV server can be achieved by having the client encrypt the submission under the public key of the LTSV server.) A submission to the LTSV may be encrypted, such that the LTSV is not able to decrypt it. That is acceptable with the LTSV server. However, the client must determine how to decrypt the submission.

Given that the LTSV server views the source data as a bit stream, it is possible that the message could be encrypted by the LTSV client before submission. (The fact that a general purpose LTSV server treats the source document as a bit stream does not preclude the possibility of implementing an application specific LTSV server that is aware of the contents of the submitted data.) The LTSV server treats the encrypted data as the source. Such prior encoding may be sufficient for some applications' needs for privacy. In this case, however, either the client must maintain the decryption key, or the key must be divulged and stored by the LTSV server (which is probably not acceptable).

Alternatively, the LTSV client may submit a message digest (resulting from a one-way hash function) as the source document. The client, in this case, is responsible for maintaining the real source document. If the source document is stored by the client, then only the PKI state information is stored in the LTSV server's archive (along with some reference to the source document or the submitter).

Whether the source data is stored by the client or the LTSV server, it must be produced if and when a reverification of that document is required. It is a required component of any signature verification process.

Key and Algorithm Degradation.

If cryptographically encoded information *(e.g.* digital signatures or encrypted data) is stored for a significant period of time, the issue of key and algorithm degradation must be addressed, *i.e.* the probable loss in effectiveness of a cryptographic key or algorithm over time. Although signed documents are expected to be sealed securely with strong cryptographic algorithms, the strength of an algorithm and associated key length decreases over time with the advent of faster computers and new developments in cryptoanalysis. It is expected that cryptographic algorithms and key lengths have limited life spans. It is generally acknowledged that they should be examined, modified, and/or replaced at periodic intervals. This legitimate security concern increases with the length of time for which a document is valid, and it becomes a very serious threat as the time span approaches multiple decades.

For example, a digital signature performed today, using RSA and a 512-bit key, is considered very strong *(i.e.* it would take years to forge it). But, it is also expected that this same signature may be easily forgeable within ten years or so. This is because of the increased ability to search the key space faster (and thereby find the key used to sign the message) with newer computers or computing techniques. Similarly, there may continue to be developments in techniques for factoring large prime numbers (the difficulty of which is the basis for the strength of the RSA algorithm). It is reasonable for both of these abilities to improve over time (although the pace of these changes is less certain).

It is, therefore, prudent to protect cryptographically encoded documents from this threat when those documents must live beyond a few years. This is the case with the documents expected to be submitted to the LTSV server, and especially so when using the save state approach herein disclosed. Hence, the LTSV facility should address this problem. Not only must the signed documents stored in the archive be protected from this threat, but all other cryptographic data or metadata stored in the archive should be protected. (The cryptographic data primarily include keyed information. That is, any information that is signed or encrypted with a private key. Such information may also include non-keyed cryptographic data, such as the output from a hash algorithm, such as MD5.) This data could also include such items as certificates and CRLs, which are, themselves, digitally signed by the issuing CA.

There are any number of ways that the LTSV facility addresses this problem. For example:

- Periodically countersign all data in need of cryptographic refresh through the use of nested signatures. Under this approach, the LTSV server effectively refreshes the cryptographic strength of the data by signing it with successively longer keys (or stronger algorithms) every few years. Each counter signature has the effect of locking in the cryptographic strength of the enclosed signature(s), thereby extending the cryptographic life of the enclosed document. This countersignature is prefera-

bly performed by the LTSV server using a key owned by that server. Performance shortcuts may be required to avoid the costly unraveling of signatures at reverification time, or the potentially time consuming task of countersigning every document in the archive. Such shortcuts include, for example, removing a previous countersignature before applying a new one, or countersigning the entire archive or portions thereof instead of each individual document.

- A modification of the cryptographic approach suggested above provides for countersigning the information in the archive once, but with an extremely long key, *i.e.* a key which is expected to be unbreakable for decades or more. This eliminates all need for countersigning. This may be merely a theoretical solution because finding an algorithm and key length which is secure for that long is impossible to predict. Therefore, there is still a need to provide some backup mechanism, just in case the original algorithm were cracked, for example.

- Protect the cryptographic information in the archive by insulating the archive itself, rather than the individual documents contained in the archive, thereby eliminating the need for a cryptographic solution. In this approach, the archive is protected via access controls and other procedural controls. If the archive can be effectively insulated from intrusion and modification, then key degradation is not an issue and cryptographic refresh is not necessary.

- Use a time stamp facility to seal the cryptographic information in time. Such a facility is expected to provide all of the necessary characteristics required for solving the key degradation problem. This time stamp facility could use one of the techniques listed above, or it could even be an independent service (see below for a discussion of time stamping).

Relationship to Time stamping.

A secure and comprehensive LTSV solution preferably includes an association with a time stamping mechanism. For long term verification of digital signatures, it is often necessary to know the time at which particular events occurred *(e.g.* time of signing or verifying a signature) to determine if a document was valid at that specific time. If there were uncertainty concerning when a document was signed, then the later reverification of that document could be compromised because of the uncertainty of when it was signed.

Fig. 4 is a flow diagram that provides two alternative scenarios that illustrate the applicability of time stamps.

In scenario 1:

- Alice signs a document at time T1, and sends it to Bob (140).

- Alice's certificate is revoked at time T2 (142).

- Bob verifies Alice's signature at time T3 (144).

In scenario 2:

- Alice's certificate is revoked at time T1 (150).

- Alice signs a document at time T2, and sends it to Bob (152).

- Bob verifies Alice's signature at time T3 (154).

When Bob performs the verification (at time T3), he does not know when Alice signed the document. This is critical, because if Alice's key (certificate) were revoked before signing the message, then the signature verification by Bob should fail, and Bob should not trust the contents of the message. If, on the other hand, the revocation occurred after the act of signing, then the signature can be presumed to be valid and trustworthy. For simplicity, this example does not consider the complicating issue of CRL latency, *i.e.* the time between the initiation of certificate revocation and the time when this information becomes available on a CRL.

This example demonstrates the need for a secure and trusted time stamp mechanism in the domain of digital signatures. The mere recording of the current date and time when creating a digital signature is not sufficient for most application because the source of that time may not be trusted by the recipient. The impact, however, also applies not only to the short term signature verification process, but also to the long term verification of digital signatures. Given the example above, the LTSV server could save the PKI state (at time T1) associated with scenario 1 or scenario 2 (or both). The outcome of a signature verification on this message years later is greatly affected by the PKI state used for this verification process, as well as the target time for the verification.

The problem highlighted above demonstrates the preference that the LTSV service to be cognizant of time. It should:

- Be able to determine in a secure fashion the time at which a document was originally signed;

- Be able to re-create accurately the PKI state which was active at a target time in the past;

- Be able to determine the current date and time accurately; and

- At a minimum, save the PKI state associated with a particular target time.

These requirements establish the preference for the integration of a time stamp facility with the signing and verification (and reverification) process. When a document is signed, it is also preferably time stamped to document in a secure fashion the precise moment at which that event occurred. The LTSV service should know the time for which the PKI state is to be saved, be sure to save the appropriate state (the state active at the target time), and execute its signature reverification process in the context of the correct time.

Usage Scenarios.

Figs. 5a-5c provide block schematic diagrams that illustrate three long term signature verification usage scenarios.

In scenario 1, a client (EntityA) 50 submits a document to a LTSV facility 52 for long term signature verification. This is a simple case where EntityA is interested in documenting that it possessed some piece of information.

In scenario 2, EntityB 56 receives a document from EntityA 54 and submits that document to the LTSV facility 58. In this case, EntityB wants to document that it received some information from EntityB.

In scenario 3, EntityA 60 sends the same document to EntityB 64 and to the LTSV facility 62. This case represents a carbon copy feature, whereby EntityA can document the information it sent to EntityB by additionally filing it with the LTSV facility.

Each of the scenarios described above raises issues with respect to encryption, private key access, and trust models.

Encryption and Private Key Access.

A document can be encrypted and/or signed. Ideally, the LTSV facility accepts any such document. This raises a problem, however, with respect to how the LTSV module works with respect to the encryption. When encrypting under a public key system, the document is effectively encrypted under the public key of the recipient, thereby guaranteeing that the recipient (the possessor of the corresponding private key) is the only entity which can decrypt the information. (For purposes of this discussion, interaction with symmetric keys and algorithms is ignored.)

When applying this principle to scenario 1, it is clear that if the signed message is also encrypted, then it could be encrypted under the public key of the LTSV module. This allows the LTSV component to unwrap the signed document and preserve it for long term verification. This is a useful feature because it provides confidentiality between EntityA and the LTSV service. This scenario does not preclude the possibility that the source document sent signed and encrypted to the LTSV module could itself be encrypted under a key known only to EntityA. That is, it is not necessary that the LTSV have access to the plain text version of the source document. The LTSV module treats that encrypted document as the source. If EntityA does decide to encrypt the document first under a secret key before submitting the document to the LTSV service, then it is the responsibility of EntityA to maintain possession of that key if and when decryption of that document is required.

In Scenario 2, if the message from EntityA to EntityB is encrypted (under the public key of EntityB) and then forwarded -- unchanged -- to the LTSV service by EntityB, then it is unreadable by the LTSV component because it does not possess the private key required to decipher and unwrap the enclosed signed document. This unwrapping (decipherment) is essential for the LTSV module to do its job.

There exist several alternatives for addressing this problem:

- Allow the LTSV facility to have access to EntityB's private key;
- Do not allow EntityA to send encrypted documents to EntityB; or
- Have EntityB strip off the privacy aspect of the signed and encrypted document received from EntityA. Because EntityB wants to preserve EntityA's signature on the document, and be able to verify it at a later time, this stripping process can not alter the validity of the signature. EntityA can then either send the stripped *(i.e.* plain text) document to the LTSV service, or it can re-encrypt it (still preserving the original signature by EntityA) under the public key of the LTSV module.

The latter approach above is presently the preferred approach. The first approach above raises significant security concerns because it requires distribution of an entity's private key. The second approach above is unacceptably restrictive on the usage of the system.

Trust.

Digital signature verification is always performed between two (and only two) entities. The verification process is based upon (among other things) the trust relationship(s) in place between those two entities -- the originator (signer) and the recipient (verifier).

Fig. 6 is a block schematic diagram that illustrates trust between two entities according to the invention. In this situation, EntityA 70 has been issued a certificate by CA1 72, EntityB 74 has been issues a certificate by CA2 76, and CA's 1 and 2 have been cross certified. (A cross-certificate is a special type of certificate which indicates mutual trust between two CAs.) The resulting trust model sets up a path of trust between EntityA and EntityB, enabling them to verify digitally signed docu-

ments from one another successfully. (A trust model is comprised of the trust relationships among PKI entities (CAs and end users), embodied by the certificates and cross-certificates issues among these entities, as well as the underlying policies which enable this trust.) Note that if any of the three paths in this model were not in place, then sufficient trust would be lacking for the successful exchange of digitally signed messages between the two end parties. Signature verification would fail if any entity in this path is not trusted.

This trust path is commonly referred to as the certificate chain because it is composed of the certificates between the two entities. When considering the save state approach to long term signature verification, it is this entire trust path (among other things) which must be archived as part of the PKI state for later signature reverification. Moreover, the trust path stored by the LTSV facility must contain the relevant trust information existing at the time of the request, not at some other time (before or after) where the trust relationships may be different between the entities. For example, a cross certificate between to CAs could either be created or removed at some point in time. This could effect the trust between two entities and affect the outcome of a signature verification.

As discussed above, the time associated with the existing trust model between two entities is extremely important to the LTSV facility, but there are also ramifications with respect to how the LTSV module works -- specifically, what trust information is needed and stored by the LTSV component for later signature verification. This gets complicated when the LTSV component is included, which may or may not be trusted (via some trust path) by some entities.

Consider the three scenarios illustrated in Figs. 5a-5c:

Scenario 1 is fairly straightforward. There are only two entities involved. The trust path stored by the LTSV facility is the path between those two parties (EntityA and LTSV). It is assumed that trust exists between these entities, otherwise EntityA would not submit a request to that service.

Scenario 2, however, raises certain issues. When EntityB sends a request to the LTSV service, what signature does EntityB want to later verify? Most likely, EntityB wants to reverify EntityA's signature at a later time -- it wants the LTSV service to document that the signed document received from EntityA was valid (contained a valid signature) at the time it was received. This raises two general questions:

- Whether the LTSV service is trusted by EntityA. It can be assumed that the communicating parties (EntityA with EntityB, and EntityB with the LTSV) have developed some trust between themselves. But in this case, it is possible that there exists no trust path between EntityA and the LTSV component.
- The trust path that is to be stored by the LTSV facility. There exist three possible trust paths which can be stored by the LTSV, *i.e.* the path between Entities A and B; the path between EntityB and the LTSV component itself; and the path between EntityA and the LTSV component, if it exists.

Fig. 7 is a block schematic diagram that illustrates a long term signature verification trust model. Given scenario 2, where EntityB 84 submits a signed document, received from EntityA 80, to the LTSV component 88, the LTSV can save the trust model embodied in the original signed document (EntityA 80 → CA1 82 → CA2 86 → EntityB 84). Later verification of this signature recreates the verification process originally performed by EntityB when it received this document from EntityA. If, however, the PKI state stored by the LTSV service were to contain only the trust path between the submitter and the service (EntityB 84 -> CA2 86 -> CA3 90 -> LTSV 88), then reverification of the original document, as originally performed, is impossible. In fact, this is exactly the paradigm described in scenario 1, where the trust path between the submitter and the LTSV are of interest.

The above discussion reveals that there are good reasons for the LTSV component to be able to store either trust path, depending upon the requirements of the client.

In scenario 2, the LTSV would most likely store the trust path corresponding to the message from EntityA to EntityB (to reverify the signed document from EntityA to EntityB). In scenario 1, the LTSV would store the trust path corresponding to the submission itself -- from EntityA to the LTSV.

Similarly, scenario 3 represents a case where flexibility in which trust path(s) to store is required. In this case, EntityA's submission to the LTSV facility may be with the intent to either reverify its correspondence with EntityB, or to reverify the submission itself (between EntityA and the LTSV). In fact, both trust paths may be of use to the client. The requirements on the LTSV are determined by the business of the particular application being deployed. For this reason, the interface to the LTSV preferably supports the ability of the client to indicate the needs in terms of trust paths as it impacts the requirements for later reverification.

The disclosures in United States patent application no 08/892,792, from which this application claims priority, and in the abstract accompanying this application are incorporated herein by reference.

Claims

1. A method of enabling long term verification of digital signatures, comprising the steps of:

   submitting a source document or digest thereof to a signature verification entity; and

using an archive facility to store a public key infrastructure (PKI) state relative to said document at a selected archival time.

2. A method as in claim 1, comprising the steps of:

using said archived PKI state to re-create said PKI state relative to said document at a selected time after a certificate associated with said signature has expired;
wherein the time over which a digital signature associated with said document can be verified is extended beyond expiration of any or all of any certificates upon which that signature depends.

3. A method as in claim 1 or 2 comprising the step of:
storing said source document separately from any associated cryptographic information.

4. A method as in claim 1, 2 or 3 wherein the selected archival time used as the basis for subsequent recreation of a signature verification process is the time of said source document submittal;
is the time when said source document was signed by its originator; or in the time when said source document was verified by a recipient.

5. A method as in any preceding claim, comprising the step of;
protecting said PKI state information from intrusion by maintaining it in a secure storage facility preferably comprising of at least one of a firewall, access control mechanism, audit facility, intrusion detection facility, physical isolation and network isolation; or protecting non-cryptographic PKI state information from intrusion by protecting it in an archive via any of a signature and keyed hash algorithm.

6. A method as in any preceding claim comprising the step of:
providing utilities for viewing said PKI state information and for visually monitoring system security.

7. A method as in any preceding claim, wherein classes of PKI state information may include one or more of certificate chain from one entity to another, including certification authorities (CAs) and the end entities; certificate revocation lists (CRLs), one for each CA in certificate chain; certificate practice statements; attribute certificates; policy constraints; trust information; and date and time.

8. A method as in any preceding claim, comprising the step of:
periodically countersigning all data in need of cryptographic refresh through the use of nested signatures and/or countersigning information in said archive facility once with an extremely long key.

9. A method as in any preceding claim, comprising at least one of the steps of:

protecting said archive facility itself, rather than individual documents contained in said archive; and
employing a cryptographic protection mechanism at said signature verification entity.

10. A method as in any preceding claim, comprising the step of:
using a time stamp facility to seal cryptographic information in time.

11. Apparatus for long term verification of digital signature, comprising:

a source document; and
an archive facility for storing a public key infrastructure (PKI) state relative to said document at a selected archival time.

12. Apparatus as in claim 11, comprising:
either of a signature and a keyed hash system for protecting non-cryptographic PKI state information from undetected modification, wherein said noncryptographic PKI state information is maintained in an archive.

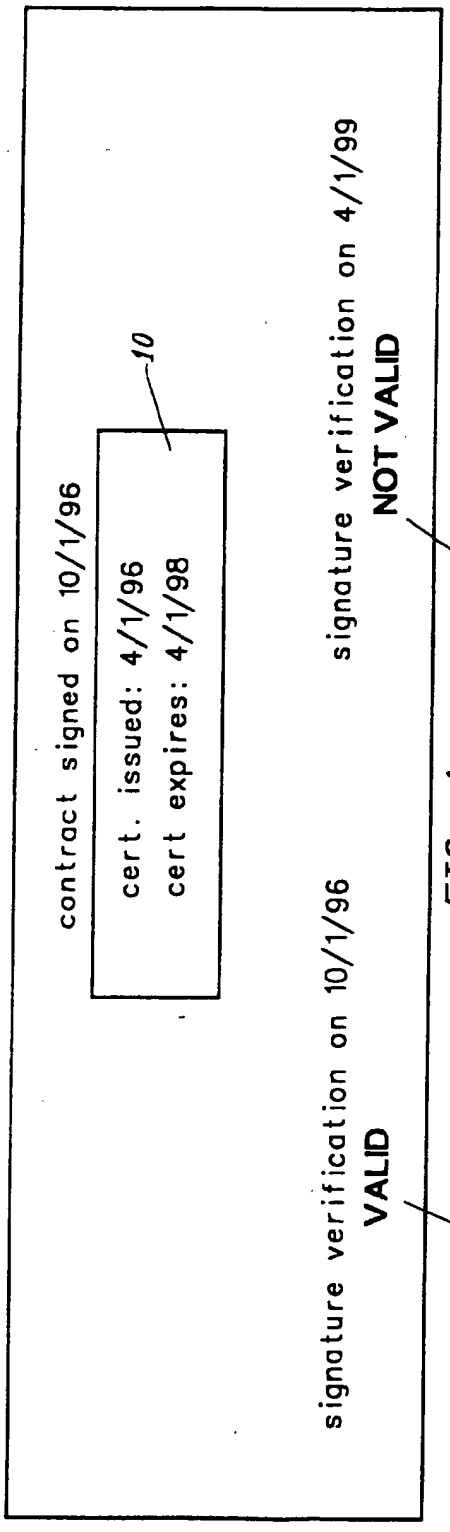


FIG. 1
(PRIOR ART)

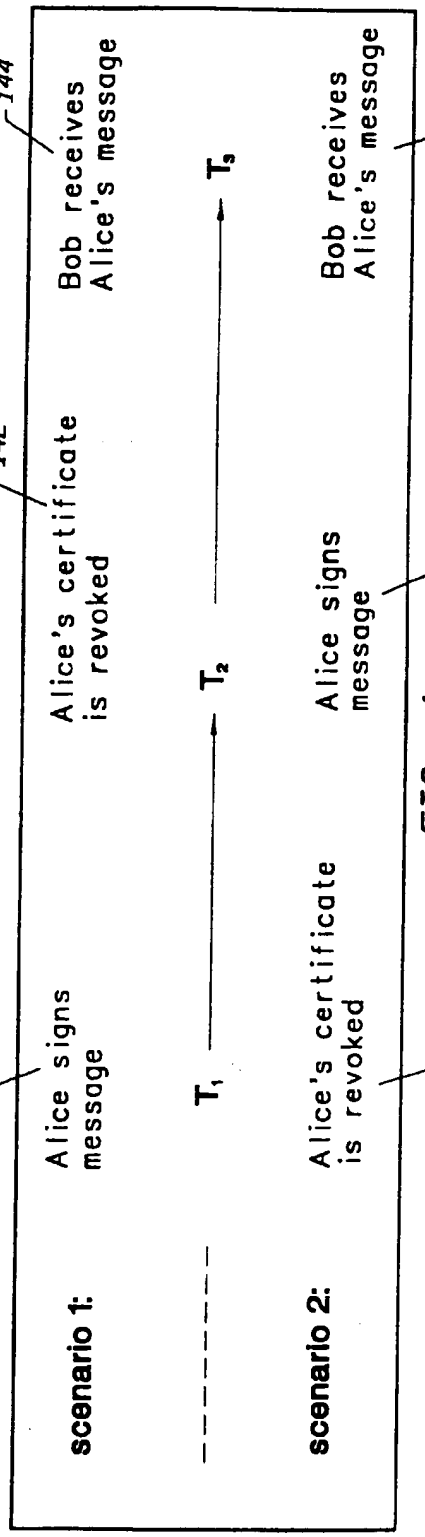


FIG. 4

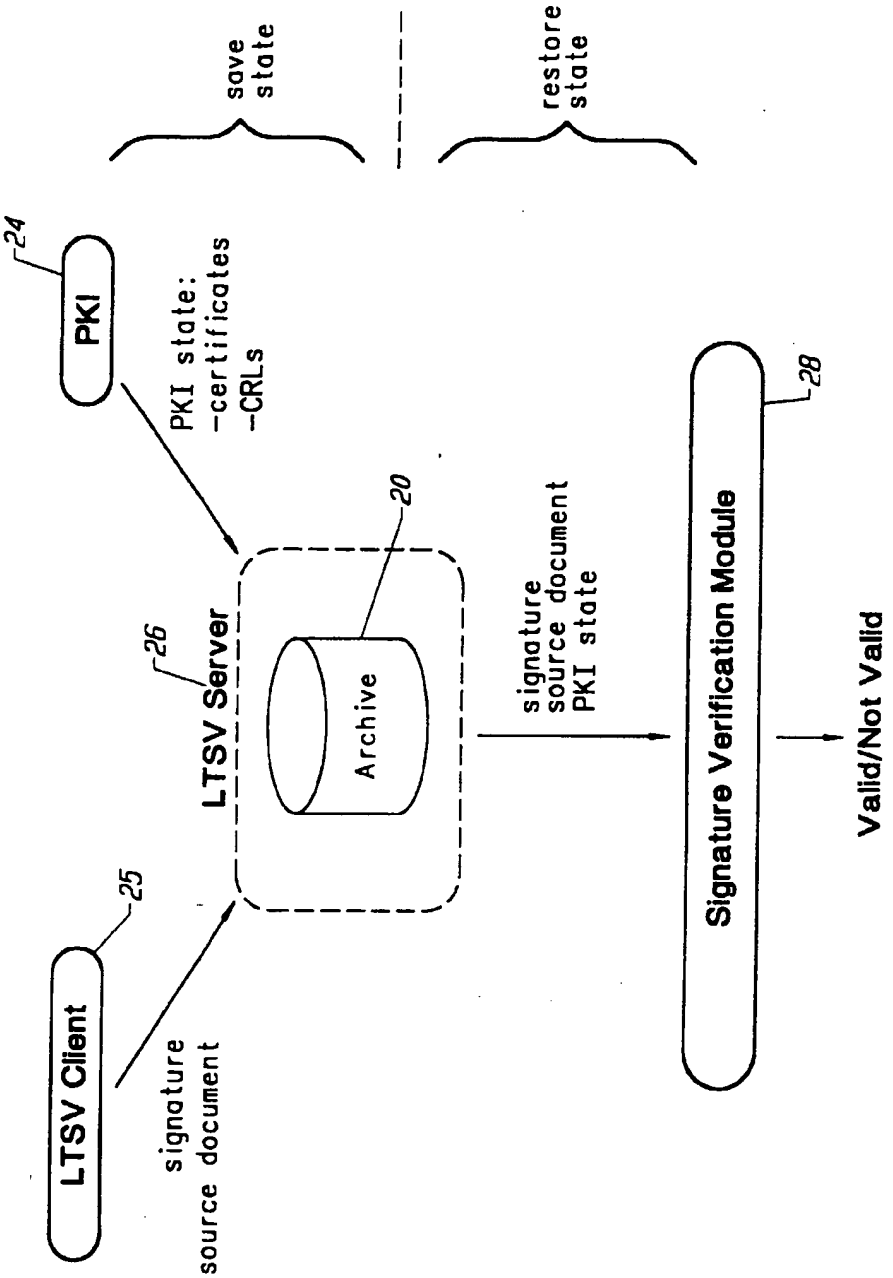


FIG. 2

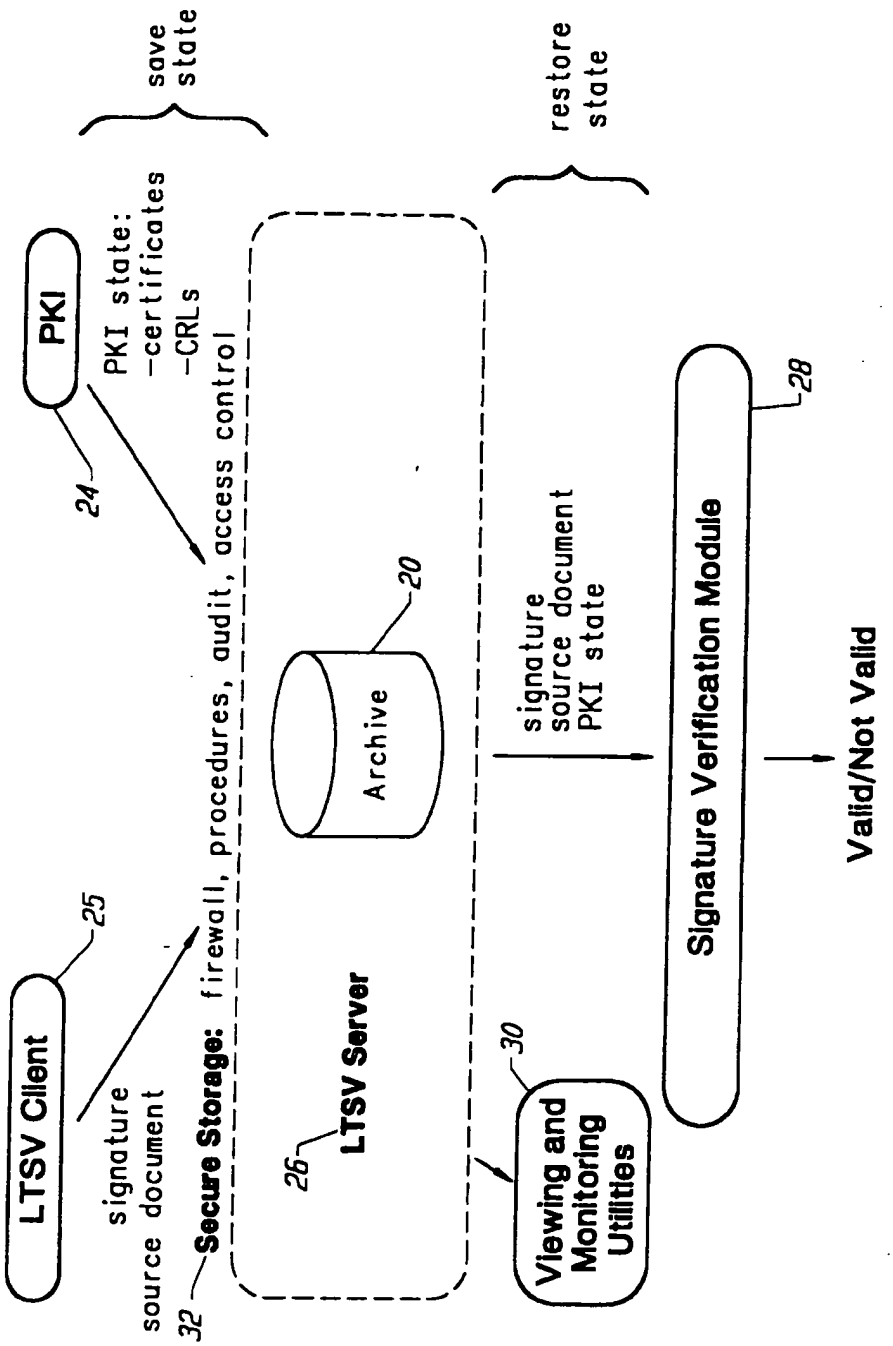


FIG. 3

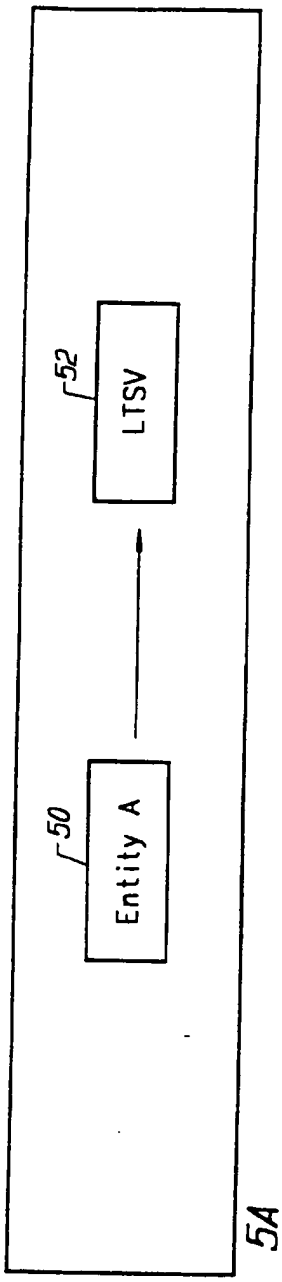


FIG. 5A

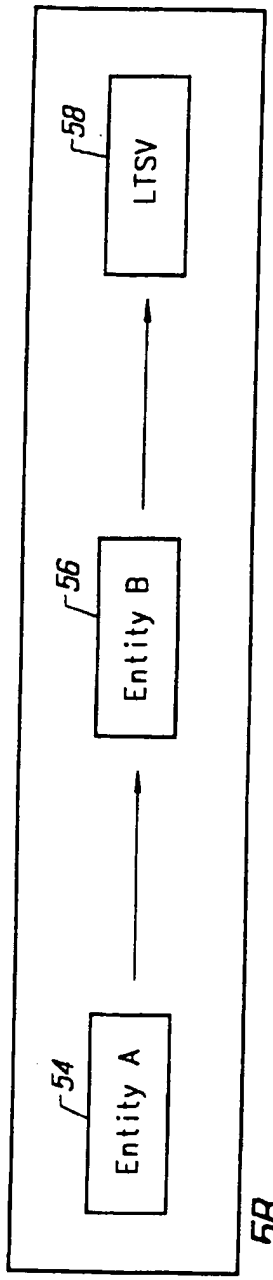


FIG. 5B

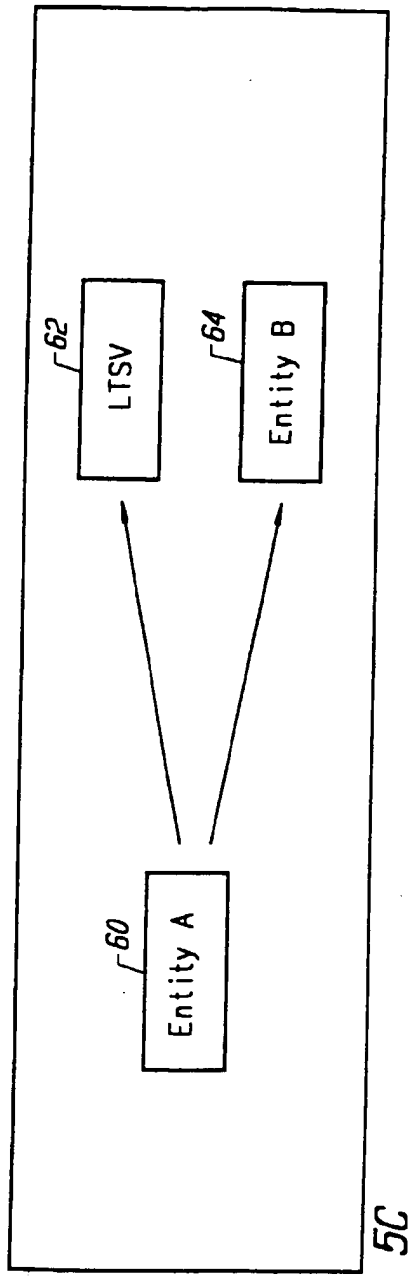
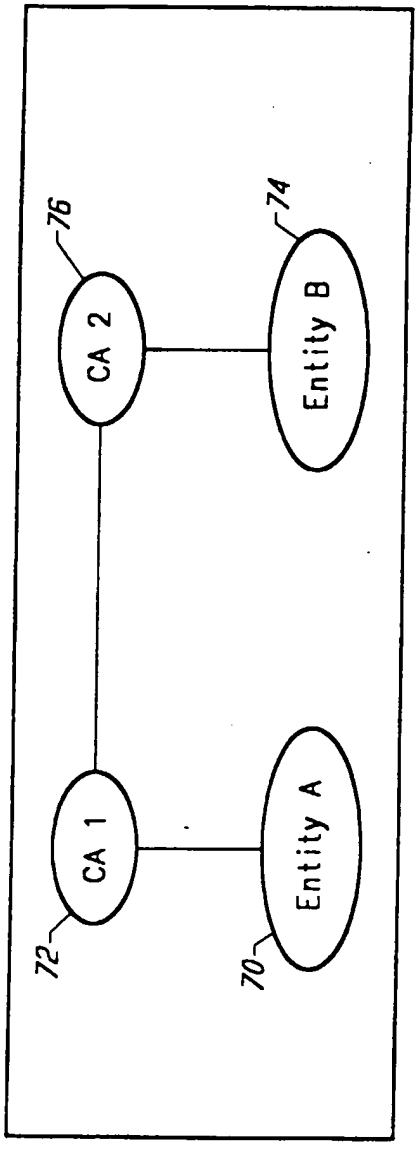


FIG. 5C

FIG. 6



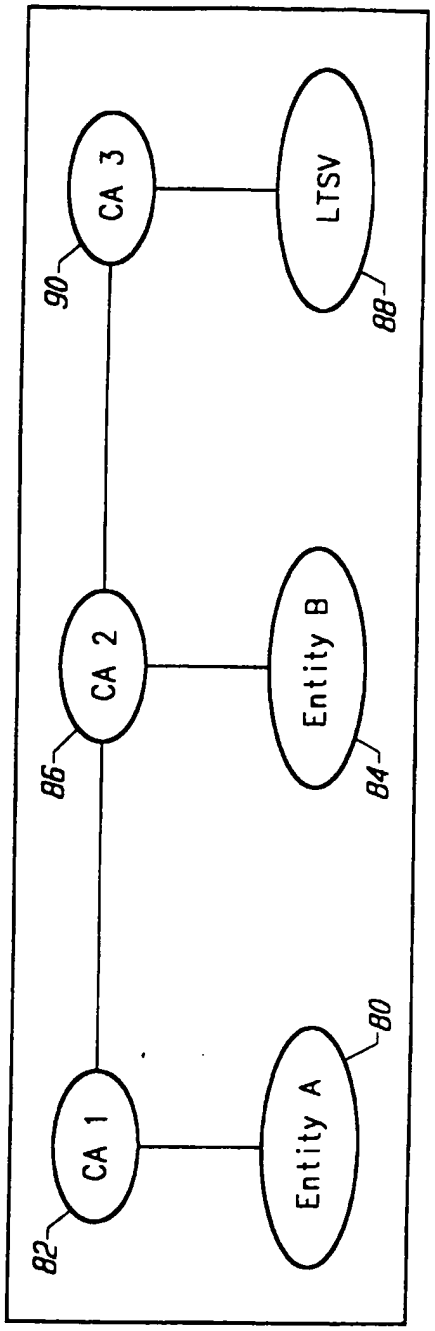


FIG. 7

09

(12) UK Patent Application (19) GB (11) 2 316 503 (13) A

(43) Date of A Publication 25.02.1998

(21) Application No 9617596.3

(22) Date of Filing 22.08.1996

(71) Applicant(s)
ICL Personal Systems Oy

(Incorporated in Finland)

PO Box 458, SF-00101 Helsinki, Finland

(72) Inventor(s)
Tapani Lindgren

(74) Agent and/or Address for Service
S M Dupuy
International Computers Limited, Cavendish Road,
STEVENAGE, Hertfordshire, SG1 2DY,
United Kingdom

(51) INT CL^6
G06F 1/00

(52) UK CL (Edition P)
G4A AAP

(56) Documents Cited
GB 2238604 A EP 0332304 A2 WO 93/11480 A1
US 5375206 A US 4924378 A

(58) Field of Search
UK CL (Edition O) G4A AAP
INT CL^6 G06F

(54) **Software licence management**

(57) A software licence management method and system is for a computer system including at least one server (1,5) and particularly for a plurality of computers connected via a network. Before a service (2) can offer functionality to a user it has to check that the user has a licence for that service. A licensing subsystem (3) has associated with it a ticket database (4) that hold tickets corresponding to existing licences. Tickets, if available, are issued to a service on request, thereby verifying the existence of a licence. The receipt of a ticket allows a service to offer functionality.

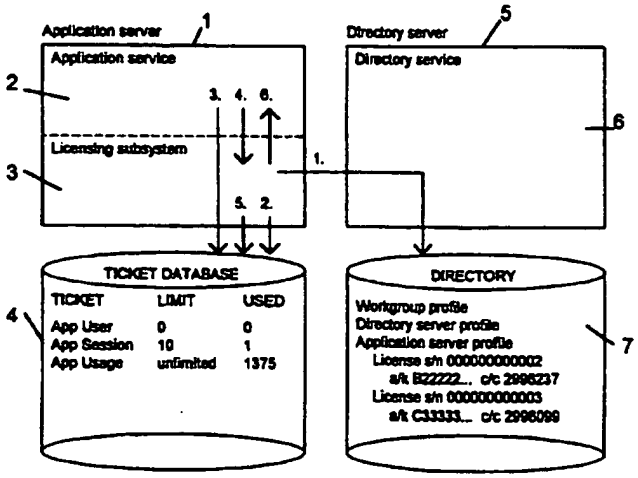


FIG 1

GB 2 316 503 A

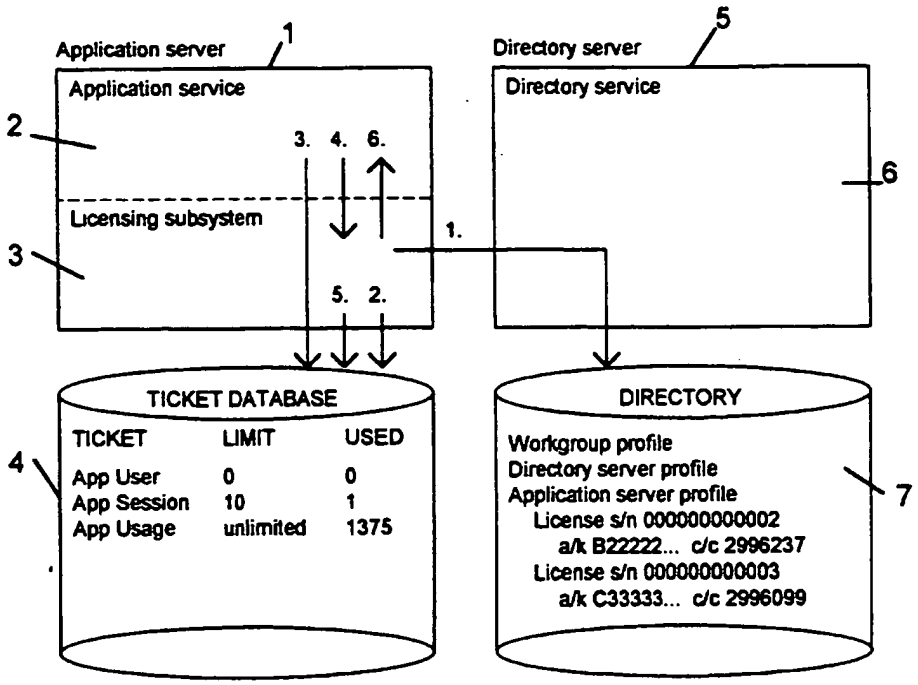

Application server
1
Application service
2
3. 4. 6.
Licensing subsystem
3
1.
5. 2.
Directory server
5
Directory service
6
TICKET DATABASE
TICKET LIMIT USED
4
App User 0 0
App Session 10 1
App Usage unlimited 1375
DIRECTORY
Workgroup profile
Directory server profile
Application server profile
License s/n 000000000002
a/k B22222... c/c 2996237
License s/n 000000000003
a/k C33333... c/c 2996099
7

FIG 1

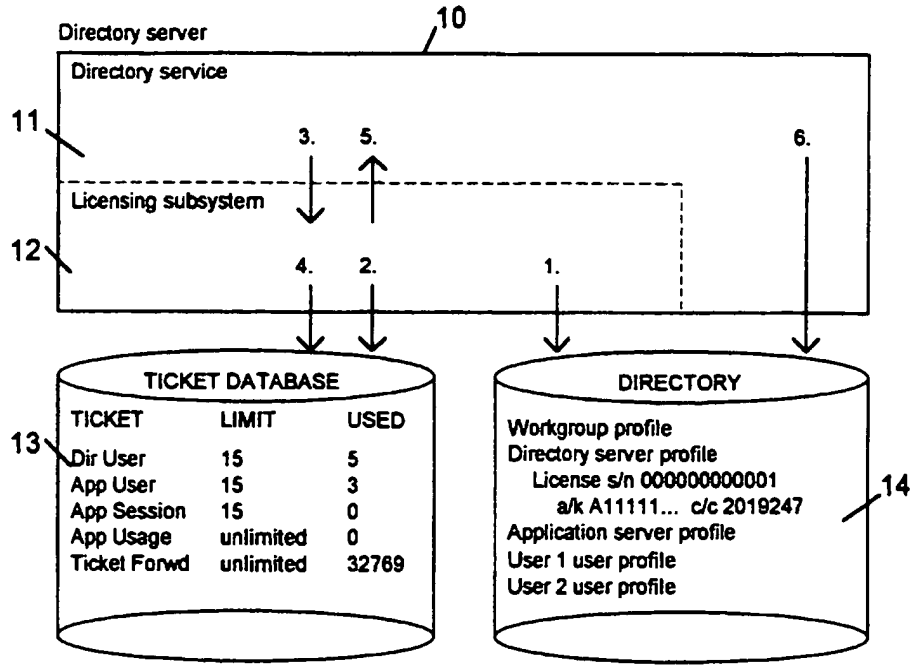

Directory server
10
Directory service
11
3.
5.
6.
Licensing subsystem
12
4.
2.
1.
TICKET DATABASE
TICKET
LIMIT
USED
13
Dir User
15
5
App User
15
3
App Session
15
0
App Usage
unlimited
0
Ticket Forwd
unlimited
32769
DIRECTORY
Workgroup profile
Directory server profile
License s/n 000000000001
a/k A11111... c/c 2019247
14
Application server profile
User 1 user profile
User 2 user profile

FIG 2

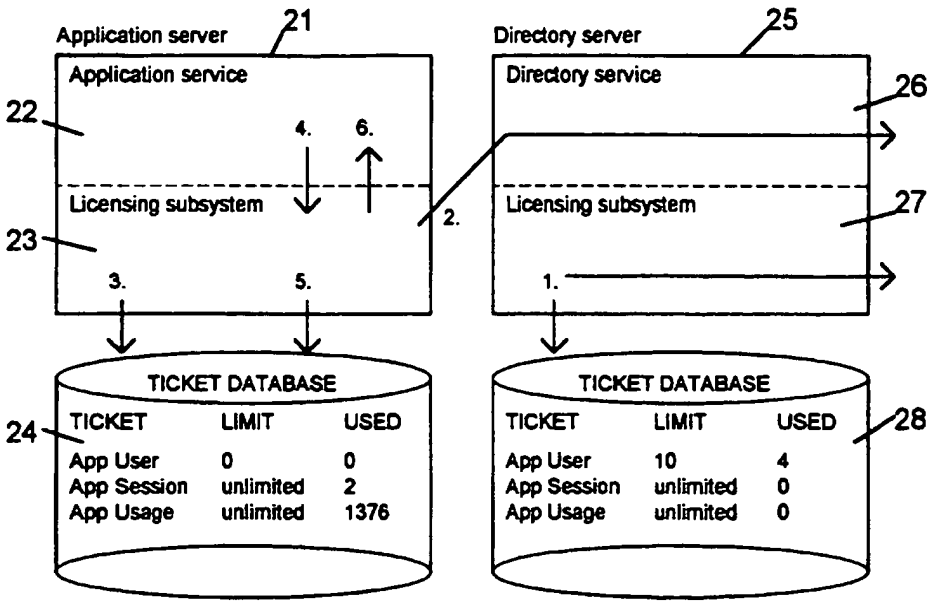

Application server
21
Application service
22
4.
6.
Licensing subsystem
23
2.
3.
5.
TICKET DATABASE
TICKET
LIMIT
USED
24
App User
0
0
App Session
unlimited
2
App Usage
unlimited
1376
Directory server
25
Directory service
26
Licensing subsystem
27
1.
TICKET DATABASE
TICKET
LIMIT
USED
28
App User
10
4
App Session
unlimited
0
App Usage
unlimited
0

FIG 3

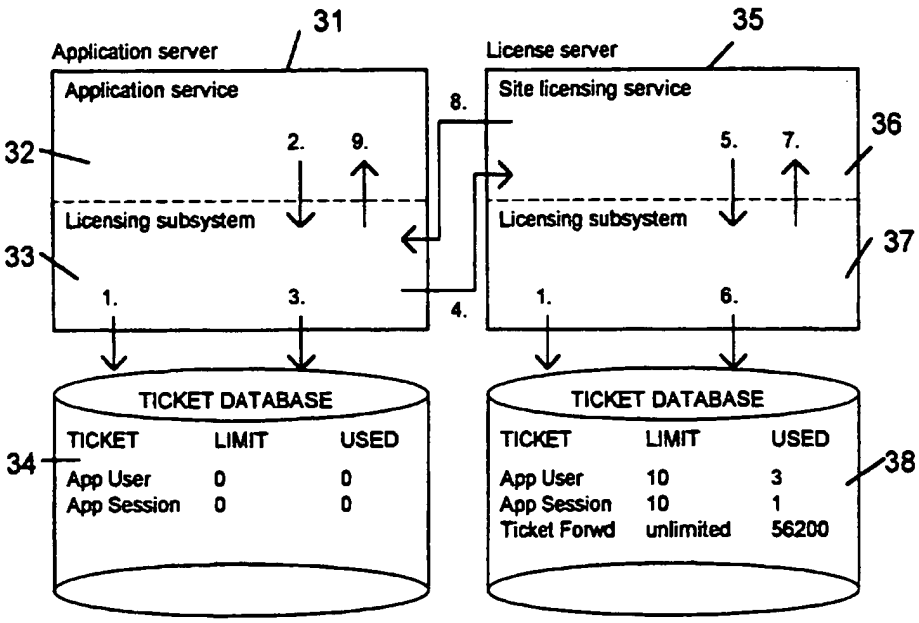

31
Application server
Application service
32
2.
9.
Licensing subsystem
33
1.
3.
8.
4.
35
License server
Site licensing service
36
5.
7.
Licensing subsystem
37
1.
6.
TICKET DATABASE
TICKET
LIMIT
USED
34
App User
0
0
App Session
0
0
TICKET DATABASE
TICKET
LIMIT
USED
38
App User
10
3
App Session
10
1
Ticket Forwd
unlimited
56200

FIG 4

2316503

SOFTWARE LICENCE MANAGEMENT

This invention relates to software licence management and in particular to licence management for software running on a plurality of computers connected via a network.

Conventionally, licences have been provided by software vendors as separate licences for individual workstations or as a single licence for a number of workstations. Various schemes have been proposed in order to try and make unlicensed software unusable, in particular pirated (illegal) copies of software. Other schemes have been proposed such as in order to achieve low initial software costs but licensing royalties consistent with the extent of use, in order not to deter low-usage users from purchasing particular forms of software, and thus to reduce piracy, whilst still enabling a vendor to collect higher dues from high-usage users.

The present invention is, particularly, concerned with a distributed system consisting of various server and client programs running on various computers which are connected via a local or wide area network, and an object is to provide server software licensing which ensures that all software running in the network has been purchased legally.

According to one aspect of the present invention there is provided a software licence management method for use with a computer system including at least one server, the method being such that before a service can offer functionality to a user, the said service shall verify that the user has a licence for said service, and wherein the computer system further includes a licensing subsystem with which are associated service tickets corresponding to existing licences, the method including the steps of the said service requesting a respective service ticket from the licensing

subsystem prior to offering functionality to the user, and the licensing subsystem issuing a said service ticket to the said service, if one is available, thereby verifying the licence exists and allowing the said service to offer functionality.

According to another aspect of the present invention there is provided a computer system including at least one server and a software licence management system, the management system being such that before a service can offer functionality to a user, the service shall verify that the user has a licence for said service, the management system including a licencing subsystem with which are associated service tickets corresponding to existing licences, and the management system being such that a said service ticket is issued to a service, if one is available, upon request by the service, thereby verifying existence of a licence and allowing the said service to offer functionality.

Embodiments of the invention will now be described with reference to the accompanying drawings, in which:

Figure 1 Illustrates obtaining a session or usage ticket for an application,

Figure 2 Illustrates obtaining a user ticket for a directory server,

Figure 3 Illustrates independent licence sharing, and

Figure 4 Illustrates licence sharing with a site licencing service,

Various terms used in the following will first be defined. For the purposes of the description the software is considered to relate to a Groupware Office system which provides various facilities including mail, for example.

Definitions

"Server"	An instance of server software running on a server computer. Usually only one such instance runs on any one computer. Each "server" implements one or more collections of related functions called "function sets", examples of which are directory, mail, library etc. The "directory function set" includes functions to access a database that contains information about the Groupware Office system.
"Client"	Any piece of software that connects to the "server" using a "client-server protocol" to access the functions offered by the "server". A "client" may be a program run by a user on a workstation, or a part of any other program.
"Session"	An instance of client-server dialogue between one "client and one "server". Each "session" allows the "client" to use the functions of one or more "function sets".
"Directory Server"	A server that implements the directory function set.
"Mail Server"	A server that implements the mail function set. [A server may be a directory server and a mail server simultaneously.]
"User"	A person (actual or virtual) listed in the database of a directory server.

"User profile" The information pertaining to one user stored in a directory database entry, such as the name of the user, user authentication information, the list of servers and function sets the user is permitted to access, etc.

"Server profile" The name and network address of a server and the list of services offered by it, as stored in a directory database entry.

"Service profile" Information stored in a directory database entry about one service in one server. If the same type of service is offered by more than one server, each instance has its own profile.

"Site profile" Information stored in a directory database entry about one site.

"Site" A set of servers connected to a single directory server. Each server belongs to exactly one site, and each site has exactly one directory server. Other servers in the site are optional, usually unlimited in number, and sometimes called member servers or application servers.

"Enterprise" A set of sites that share their directory databases. The directory servers in each site replicate directory information to other directory servers in the enterprise. Each directory server contains both "local" and "external" information. One of the directory servers, the "enterprise directory server" controls the others, which are

"site directory servers".

"Subsystems" — Collections of programs and/or subroutines that perform a set of interrelated functions. Some subsystems implement a function set within a server program, while others run independently as stand-alone applications. Many subsystems are collections of common subroutines called by other subsystems. Client programs are also subsystems.

"Subsystem id" — A respective unique number identifying every type of subsystem. Some systems use two ids, a "real subsystem id" when dealing with licensing issues and an "alias subsystem id" when performing a task on behalf of a virtual entity, such as "generic gateway no 9".

Not every subsystem software needs to be purchased individually. Most collections of subroutines can be used freely by other subsystems.

"Services" — Those subsystems that need to be explicitly purchased.

Service types may include directory service, mail service, fax gateway, X.400 gateway, enterprise option, library service, power library option, etc. Each service is located in one server, either as a function set of the server program or a standalone application running in the same computer. Many services of the same type can exist in different servers.

The software licence management system of the present invention proceeds from the premise that before offering any usable functionality to the users, services shall verify from a "licensing subsystem" that a licence for the service exists. To achieve that, it is proposed that the licensing subsystem holds "service tickets" and a service requests a permission to offer its functions to the user by requesting a corresponding "service ticket" be provided from the licensing subsystem. Each service knows that kind of tickets are needed to fulfil the service's functionality. The licensing subsystem has to keep track of the available licences and of the service tickets it has issued. A service ticket may be considered as partially the equivalent of a password in that one must be provided before a service can operate.

A "licence" is a permission to use one or more services within certain limits. Typically these limits are "license duration", which specifies the maximum length of the period when the licence may be used (the "active" period) and the "licence size", which specifies the maximum number of users of the licence. The interpretation of "number of users" varies from service to service. It may, for example, mean the number of users in the local directory that are allowed to use the service, or the number of concurrent sessions that are connected to the service. Licence duration and/or size may also be unlimited.

When a customer purchases a Groupware, for example, software product which employs the software licence management method and system of the invention from a supplier, as well as the media containing the software itself and associated documentation, there is obtained a single licence to one or more services. Each said product has a unique serial number. The license is supplied in the form of a licence agreement document on which the licence information is printed. This licence information consists of the serial number of the product and an "activation key" for the licence. The licence

size and duration and the included services are encoded into the activation key.

The software license management method and system of the invention is such that the Groupware software may be copied and installed by the customer without any technical restrictions, but before any of the services can be used, a corresponding licence must be installed and activated. Licence installation consists of entering the license information (serial number and activation key) into the server profile of a server in the directory server's database, ie in the site directory, in the server profile of the server in question. Licence activation consists of setting the active period of the installed licence so that service tickets can be issued. Typically licence installation and licence activation are performed simultaneously by the server setup program. The license information is stored in the site directory, in the server profile of the server in question.

As will be appreciated, there also exists "evaluation licences" which allow a prospective customer to use a service for a short trial period before actually purchasing it. These licences typically have a very short duration and a relatively small size. The product serial numbers associated with such evaluation licences are not necessarily unique, since the licence information may be distributed on CD-ROMs or via public networks.

As mentioned above, each software product contains just one licence, although that licence may include a large number of services, for example, enough to build a complete Groupware Office site with all of the basic services. Alternatively, the licence may include just one service. Product with that kind of licence could be used to expand the capacity or functionality of an existing Groupware Office System.

The core of the process of designing a product is, therefore, determining what services will be included and the size and duration of the licence. This information is encoded into a number, the covert code, which may be a 7-digit number, for example. The building of the covert code is discussed in more detail hereinafter.

The amount of information that can be encoded into the covert code is limited by the size of the code. Therefore, there are some necessary restrictions on what kinds of licences are possible. The most obvious limitation is that the size and duration can only take certain discrete values. Also, the same size and duration will apply to all services covered by the licence. Another restriction is that only the most common groups of services can be combined freely into a multi-service licence. Other services will have to be licensed individually.

The covert code, which specifies the properties of the software licence, is thus a part of the product description in the logistics database. When the product is manufactured it has the unique serial number, referred to above, assigned to it. The actuation key for the license is calculated as a function of the serial number and the covert code using a secret algorithm. The serial number and activation key may be printed on a label, which is attached to the licence agreement document.

When creating a site, a customer must have a licence that includes a site creation ticket. This licence is installed for the directory server. The customer may also install additional licences for the directory server and for other servers. Each licence may apply to one or more services. Some licences are valid only in that server for which they are installed, whilst other licences may be shared with other servers at the same site (see later). Shared licences would usually be installed in the server profile of the directory

server, although optionally, and with some restrictions, another server may be designated as the licence server. The serial number of the first licence installed in a directory server's profile can be used to identify the site uniquely. Thus the directory server is computer number 1. Other servers in the site will use the same site id but differing computer numbers for identification.

When a service program is about to execute an action which requires that a customer possesses a licence for that service, the service program must first obtain a corresponding service ticket from the licensing subsystem. The actions concerned are ones which are potentially profitable for the customer and may, for example, include namely:

setting up a new Groupware site;

creating a new user account;

setting up an instance of the mail service;

enabling mail usage for a user and creating a user mailbox;

starting a session between a mail UI client and the mail server;

sending a mail message;

relaying a mail message to an X.400 mail network.

Each kind of action requires a specific kind of service ticket. To obtain a ticket the service needs to specify the ticket type and the number of tickets. The service tickets are only identified by ticket type. There is a licensing subsystem in each server and it counts the number of

different tickets in all available licences and keeps track of how many licences of each type are being used in the server.

The steps involved in obtaining various licences will now be described in greater detail. With respect to Figure 1 there will, firstly, be described the case of an application service running in a separate server from the directory server obtaining a session or usage ticket.

Figure 1 illustrates schematically an application server 1, providing an application service 2 and having a licensing subsystem 3, with an associated ticket database 4, a directory server 5 providing a directory service 6 and having an associated directory 7. The ticket database 4 has stored therein details of ticket types, the limit, if any, of the number of such tickets which are available and the number of used tickets for each type. The ticket types as illustrated are "App User" (Application User), "App Session", "App Usage". The directory 7 has stored therein, the "Workgroup profile", the "Directory server profile", the Application server profile. In the example illustrated, the application server has two associated licenses whose serial numbers (s/n) are 000000000002 and 000000000003, respectively, whose activation keys (a/n) are of the form B22222... and C33333..., respectively, and whose covert codes (c/c) are 2996237 and 2996099, for example, respectively.

When the licensing subsystem 3 on the application server starts, it fetches the application server's server profile from the directory service 6, 7 using the directory API (Application Programming Interface) (Step 1 in Figure 1). The licensing subsystem 3 analyses the licences and updates the limits of each ticket type in the local ticket database 4. The numbers of used tickets are not modified at this time, the old accumulated values being maintained (step 2).

When the application service 2 starts, and before any user logs in, it tells the licensing subsystem 3 to set the number of used session tickets to zero. This frees any session tickets that may have been left unreturned at the end of a session because of a system crash etc. (Step 3). The application service 2 then requests a service ticket (session or usage) from the licensing subsystem 3, since without a ticket it cannot proceed. (Step 4). The licensing subsystem checks the ticket availability in the local ticket database 4 and updates the used ticket count (step 5), to take into account the requested ticket, before issuing the ticket to the application service (step 6), which then proceeds since it has determined that there exists the appropriate licence.

In the embodiment of Figure 2, the procedure whereby a directory service obtains a ticket for adding a user to a directory is illustrated.

A directory server 10 provides a directory service 11 and includes a licensing subsystem 12 with an associated ticket database 13, the directory service 11 having an associated directory 14. The ticket database 13 has stored therein details of ticket types, the limit, if any, of the number of such tickets which are available, and the number of used tickets for each type. The ticket types are illustrated as "Dir User" (Directory User), "App User", "App Session", "App Usage" and "Ticket Forwd" (Ticket Forwarding). The directory 14 has stored therein the "Workgroup profile", the "Directory server profile", the "Application Server profile" and the user profile of two users, User 1 and User 2. The Directory server has a license serial number (s/n) 000000000001, with an actuation key (a/) of the form A11111..., and a corresponding covert code (c/c) 2019247, for example.

When the licensing subsystem 12 on the directory server 10 starts, it fetches the directory server's server profile directly from the directory 14 (step 1). The licensing

subsystem 12 analyses the licenses and updates the limits of each ticket type in the ticket database 13. The numbers of used tickets are not modified at this time, the old accumulated values being maintained (step 2).

When it is desired to add a new user to the directory, the directory service 11 requests a user ticket from the licensing subsystem 12 (step 3). The licensing subsystem 12 checks ticket availability in the local ticket database 13 and updates the used ticket count (step 4) to take into account the requested ticket. The licensing subsystem 12 issues the requested user ticket to the directory service 11 (step 5). The directory service then adds the new user to the directory 14, that is it adds its user profile.

To ensure consistency, the directory service 11 may periodically count the number of users in the directory 14 and tell the licensing subsystem 12 to set the used ticket count accordingly.

When a licence is installed, the start time of its active period will be fixed. By default this is the same as the installation time, but any time in the past or in the future may be specified. If the licence has a limited period, the end time will also be set. The licence will be active whenever the current time is after the start time and before the end time.

A customer may wish to deactivate a licence so that it cannot be used. Thus can be done at any time by altering the end time of the licence with the server setup program. The end time may be altered freely, as long as the active period does not exceed the licence duration.

Once installed, limited-duration licences are fixed, ie they cannot be removed, except by remaining the entire site directory, or moved to another server, and their start time

may not be altered. These restrictions, however, do not apply to unlimited-duration licences. They may be removed, reinstalled, moved or altered freely. The only restriction that remains is that a licence may only be installed for one server at a time.

A further restriction applies to the licence that has been used to create a site. This licence cannot be removed or deactivated, except by removing the entire site.

The licensing method described with reference to Figures 1 and 2 applies only to local licences, ie the tickets included in a license can only be issued in one server, the server whose server profile contains the licence. Often there is a need to share a single licence between two or more servers, so that tickets can be issued in all of them. Most commonly, the user tickets for an application are needed in the directory server, and session and usage tickets in the application servers.

If a licence includes an unlimited number of a certain kind of service ticket, sharing the licence is not very complicated. Any server can read the licences in any other server's profile. If the licensing subsystem in a server can verify that another server's licence contains an unlimited supply of freely shareable tickets, it will deduce that these tickets may be issued without limit in any server, independently of other servers. This is independent license sharing.

Not all licences are necessarily shareable, even if they contain an unlimited number of tickets. Whether each licence is shareable or not is a licence-specific property, which is coded in the covert code together with other licence properties.

The first implementation of the licensing subsystem capable

of independent licence sharing will not scan every server profile for available licences. It will only scan its own server profile and the directory server's profile. Therefore, all licenses that are meant to be shared, should be installed for the directory server.

If the tickets to be shared are limited in number, the situation is more complicated. For each "pool" of shareable tickets, there must be a single process that is responsible for keeping track of their usage. It has to co-ordinate the activities of the licensing subsystems in various servers and make sure that no ticket is issued more than once. To achieve this a site licensing service can be implemented. This is an extension to the licensing subsystem that allows the licensing subsystems of various servers to communicate using a client-server protocol. The site licensing service, together with the licensing subsystem in the same server, control the usage of tickets installed for that server. Another server's licensing subsystem may connect to the site licensing service and ask the latter to obtain a service ticket on its behalf.

Licenses that are shareable by independent sharing would also be shareable by the site licensing service, with the addition that also limited-number tickets could be shared. Some types of licences will still be unshareable, since shareability is a licence-specific property. The licensing service could itself require a licence. A site licensing service could be expanded to support also client licensing and enterprise-wide licence sharing.

An example of independent licence sharing will now be described with reference to Figure 3 in which an application server 21 provides an application service 22 and includes a licensing subsystem 23 with an associated ticket database 24. A directory server 25 provides a directory service 26 and includes a licensing subsystem 27, with a associated ticket

database 28, and a directory (not shown but containing information of the type illustrated in Figures 1 and 2). The ticket databases 24 and 28 have details of ticket type, limit and usage as indicated.

The licensing system 27 on the directory server 25 fetches the server profile from the directory (not shown), analyses the licences therein, and updates the ticket limits (step 1). The licensing system 23 on the application server 21 fetches the application server's server profile from the directory (not shown) using the directory API. It also fetches the directory server's server profile (step 2).

The Application server's licensing subsystem 23 analyses the licences in the server's own profile. In this case there are none, since the example is concerned with licence sharing. The licensing subsystem 23 then analyses the directory server's licences. Because there are unlimited session and usage tickets in a shareable licence, the local limit is also set to unlimited. The user ticket limit is set to 0, because they are limited (10 according to ticket database 28) and limited tickets cannot be shared with this method (step 3).

The application service 22 then requests an application session ticket from its licensing subsystem 23 (step 4). The ticket is granted because there are an unlimited supply of them. The used ticket count is updated in the local ticket database 24 (step 5), although it is only needed for statistics as the number is unlimited. The session ticket is then issued to the application service 22, which then proceeds since it has determined that there exists an appropriate licence.

License sharing in the case of a site licensing service will now be described with reference to Figure 4, in which an application server 31 provides an application service 32 and includes a licensing subsystem 33 with an associated ticket

database 34. A license server 35 provides a site licensing service 36 and includes a licensing subsystem 37 with an associated ticket database 38.

The licensing subsystems 33 and 27 of the servers 31 and 35, fetch their corresponding server profiles from a directory (not shown), analyse installed licences and store the ticket limits in the local databases 34 and 38 (step 1). The application server 31 need not have any licences.

The application service 32 requests a service ticket, for example an application session ticket, from the local licensing subsystem 33 (step 2). The local licensing subsystem 33 in the application server 31 will first attempt to issue the ticket locally, but this will fail as there are no licences installed for the application server 31, as indicated by the lack of available tickets in the ticket database 34 (step 3). The licensing subsystem 33 in the application server 31 will then connect to the site licensing service 36 using the client-server protocol and request the ticket remotely (step 4). The site licensing service 36 requests the ticket from the local licensing subsystem 37 and it also request a ticket-forwarding ticket (step 5). The licensing subsystem 37 of the license server 35 checks ticket availability and updates the used ticket counts in the ticket database 38 (step 6). The tickets are issued to the site licensing service 36 (step 7) which forwards the application ticket to the client ie licensing subsystem 33 (step 8), which as a result issues the application ticket to the application service 32, allowing that to proceed (step 9).

Whenever a licensing subsystem issues a service ticket, or a ticket is returned such as because it is an unused ticket (any number can be requested) or because it is a session ticket, which are required to be returned at the end of a session, the transaction can, optionally, be logged to a log file which is separate from other log files in the system.

The information in this separate log file may be used to implement a pay-by-usage licensing scheme (delayed billing). Logging can be enabled or disabled by an administrator. Each server has its own log file and all kinds of tickets issued in the server will be logged the same way. Logging parameters for each kind of ticket could be specified for certain types of licences, although such a licence could not be shared by the independent sharing method.

The proposed licensing method allows for introducing new services while retaining compatibility with old licences. The licensing subsystems will initially support some types of licences and service tickets that are not yet connected to any particular service. New services can be assigned to these items without making any modifications to existing administration programs and the licensing subsystem. The method could be extended further by adding new license/ticket combinations to the licensing subsystem, although all existing combinations would need to be kept unchanged. This would involve updating the licensing subsystem in all servers where the new services would be used. Older subsystems would not accept the new kind of licenses not issue tickets for the new services. The licenses and tickets could be defined statically, as they are now, although there could be other possibilities.

As discussed above, the covert code specifies the licence duration, licence size and included services. An example of a covert code comprises a 7-digit decimal number, with the digits numbered from right to left, starting from zero eg in number 6543210, digit no 0 is "0", digit no 1 is "1" etc.

Licence duration may be encoded in the last digit ie digit 0, as follows:

Digit No 0	Licence Duration
"0"	10 days
"1"	1 month (31 days)
"2"	3 months (92 days)
"3"	6 months (184 days)
"4"	1 year (366 days)
"5"	2 years
"6"	3 years
"7"	Unlimited (small size)
"8"	Unlimited (medium size)
"9"	Unlimited (large size)

Licence size may be coded in the next-to-last digit, digit no 1. However, its interpretation may depend on the licence duration. Limited duration licences may be one of, for example, 30 different sizes; duration digits "7", "8" or "9" select small, medium or large licence sizes respectively.

Digit No 1	Licence size for each duration type			
	Limited	Unlimited Small	Unlimited Medium	Unlimited Large
"0"	1	1	60	400
"1"	2	2	80	500
"2"	5	5	100	600
"3"	10	10	125	800
"4"	15	15	150	1000
"5"	20	20	175	1200
"6"	30	25	200	1500
"7"	50	30	225	2000
"8"	100	40	250	3000
"9"	Unlimited	50	300	Unlimited

The services that are included in a licence may be encoded into four digits, digits no 2 to no 5, of the covert

code. These digits are called the service code. The licence may apply to one kind of service tickets only, to a group of related service tickets that are used by one service, or to a group of selected services. The service code can be chosen to represent a particular name of service, such as "basic directory service", "basic mail service", "basic calendar service", in any desired manner but it will indicate what types of tickets are included and how many licence service tickets are included for each type of service.

The digit no 6, the most significant digit, may be used to specify a particular product line. In the examples shown in the drawings the covert codes all commence with the number 2, indicating they relate to the same product line.

Any number of licences may be installed in the server profile of any server. The activation key is verified, and the covert code calculated from the serial number and the activation key at license installation time. The mapping of covert code to service ticket is, preferably, not stored in the directory, rather it is recalculated by a licensing subsystem every time it starts up. All tickets of the same type are indistinguishable. The licensing subsystems do not keep track of individual tickets issued.

Any number of identical tickets may be obtained at once by a service from the corresponding licensing subsystem, providing of course that they are available. Tickets can be returned if they are not used.

The licensing subsystem does not force services to obtain tickets rather it is the service's responsibility to offer services only to legal users and without obtaining a respective ticket, a service which requires a licence will not function.

Session tickets are associated with client-server sessions.

Unless a service wants to allow unlimited usage, it should obtain a session ticket whenever a session starts. Determining when each session starts and ends is the responsibility of the service. Session tickets may not be applicable to all services. It is important that session tickets are returned when the sessions end, otherwise they will be unusable, at least until the licencing subsystem is resynchronised. This is achieved at server start up, when there are no sessions in existence, by setting the used session ticket count to zero.

When a user is given the right to use a service, the associated user ticket should be obtained first. Because in a currently preferred embodiment, users are created and user rights given by the directory service, the licenses that include user tickets should be installed into the directory server. The directory service is the only service that requests user tickets and it is responsible for maintaining consistency of the used ticket counts. It periodically counts all users in the directory and their user rights and sets the number of tickets in use as appropriate.

Some kinds of tickets are "consumable" e.g. for sending mail messages, and these will not be returned unless, for example, the message is cancelled.

Clearly if an originally purchased licence becomes inadequate, due for example to an increased number of users, then supplemental licences can be purchased which when installed will increase the number of available tickets for a service. Additional functionality can of course also be purchased subsequently, in order to add new features to a system, and the appropriate software and licence installed in an appropriate server.

It is considered that with the above description of the licence management system and method proposed by the

invention, a software developer will have difficulty producing the corresponding code for licence management for a particular software product written in a particular language, and hence no further description is considered necessary in this respect.

CLAIMS

1. A software licence management method for use with a computer system including at least one server, the method being such that before a service can offer functionality to a user, the said service shall verify that the user has a licence for said service, and wherein the computer system further includes a licensing subsystem with which are associated service tickets corresponding to existing licences, the method including the steps of the said service requesting a respective service ticket from the licensing subsystem prior to offering functionality to the user, and the licensing subsystem issuing a said service ticket to the said service, if one is available, thereby verifying the licence exists and allowing the said service to offer functionality.

2. A method as claimed in Claim 1, including the step of installing licence information comprising a licence serial number and a licence activation key into the computer system, the activation key containing encoded details of the licensed services, and wherein the computer system calculates, from the serial number and the activation key, information including the types of service tickets associated with a particular licence, the numbers of service tickets, and the duration of the licence.

3. A method as claimed in Claim 2 wherein the licensing subsystem maintains a log of the numbers of the maximum available and issued service tickets.

4. A method as claimed in Claim 2 or Claim 3, wherein a covert code is calculated by the computer system from the serial number and activation key and wherein mapping of the covert code to service tickets is calculated by

the licencing subsystem each time it is started.

5. A method as claimed in any one of the preceding claims wherein the computer system comprises a plurality of computers connected in a network and wherein a said server comprises a directory server, providing a directory service and including a respective licencing subsystem, together with a directory database and a ticket database, wherein stored in the directory database are directory server profile details, licence details and user profile details, and wherein the ticket database includes details of service tickets available in accordance with the respective licence details and issued, and wherein adding a user to the computer system includes the steps of starting the directory server licensing subsystem, the directory server licensing subsystem fetching the directory server profile with licence details from the directory database and updating the ticket database, the requesting of a user service ticket by the directory service from the licensing subsystem, the checking of ticket availability in the ticket database by the licensing subsystem, the issuing of a ticket by the licensing subsystem to the directory service, and the adding to the directory database of the new user's profile by the directory service.

6. A method as claimed in any one of Claims 1 to 4 wherein the computer system comprises a plurality of computers connected in a network, wherein a said server comprises an application server providing an application service and including a respective licensing subsystem with a respective ticket database, and another said server comprises a directory server providing a directory service and with a respective directory database, wherein stored in the directory database are directory server profile details, application server profile details and licence details, and wherein the ticket

database includes details of service tickets available in accordance with the licence details and issued, and wherein obtaining a use ticket for the application service includes the steps of starting the application server licensing subsystem, the subsystem fetching the application server profile and licence details from the directory database and updating the ticket database accordingly, starting the application service without providing funtionality, the requesting by the application service of a user service ticket from the licensing subsystem, the checking of ticket availability in the ticket database by the licensing subsystem, and the issuing of a service ticket to the application service by the licensing subsystem, the application service then providing its functionality to a user.

7. A method as claimed in any one of Claims 1 to 4 and for independent licence sharing, wherein the computer system comprises a plurality of computers connected in a network, wherein a said server comprises an application server providing an application service and including a respective licensing subsystem with a respective ticket database, and another said server comprises a directory server providing a directory service and including a respective licensing subsystem with a respective ticket base and with a respective directory database, wherein stored in the directory database are directory server profile details, application server profile details and shareable licence details, the number of service tickets being unlimited, wherein the directory server ticket database includes details of service tickets available in accordance with the shareable licence details and issued, and wherein the application server ticket database includes details of service tickets issued, and wherein obtaining a service ticket for the application service includes the steps of the directory server licensing system fetching the server profile from the

directory database, analysing the shareable licence details and updating the corresponding ticket types and ticket limits in the directory server ticket database, the application server licensing subsystem fetching the application server and the directory server profiles and shareable licence details from the directory database and analysing them and updating the corresponding ticket types in the application server ticket database, starting the application service without providing functionality, the requesting by the application service of a service ticket from the application server licensing system, the granting of a service ticket, and the issuing of the service ticket to the application service by the application server licensing system, the application service then providing its functionality to a user.

8. A method as claimed in any one of Claims 1 to 4 and for licence sharing with site licensing, wherein the computer system comprises a plurality of computers connected in a network, wherein a said server comprises an application server providing an application service and including a respective licensing subsystem with a respective ticket database, another said server comprises a site licensing server providing a site licensing service and including a respective licensing subsystem with a respective ticket database, and a further said server comprises a directory server providing a directory service and having a directory database, wherein stored in the directory database are directory server profile details, site licensing server profile details, application server profile details and licence details, wherein the site licensing subsystem ticket database includes details of service tickets available in accordance with the licence details and issued, and wherein obtaining a service ticket for the application service when the application server has no

respective licence includes the steps of the licensing subsystems fetching their corresponding server profiles from the directory database, analysing the installed licence details and the site licensing server updating the respective ticket database, starting the application service without providing functionality, the requesting by the application service of a service ticket from the site licensing service, the requesting of a service ticket and a ticket-forwarding ticket by the site licensing service from its licensing subsystem, the checking of ticket availability and the issuing of the service and ticket-forwarding tickets to the site licensing service, the forwarding of the service ticket to the application server licensing subsystem, and the issuing of the service ticket to the application service, the application service then providing its functionality to a user.

9. A computer system including at least one server and a software licence management system, the management system being such that before a service can offer functionality to a user, the service shall verify that the user has a licence for said service, the management system including a licencing subsystem with which are associated service tickets corresponding to existing licences, and the management system being such that a said service ticket is issued to a service, if one is available, upon request by the service, thereby verifying existence of a licence and allowing the said service to offer functionality.

10. A computer system as claimed in Claim 9, wherein the management system includes means for calculating from an input licence serial number and input licence activation key, information including the types of service tickets associated with a particular licence, the numbers of service tickets and the duration of the licence, said

information being encoded in the activation key.

11. A computer system as claimed in Claim 10, and including a log in which are stored the numbers of the maximum available and issued service tickets.

12. A computer system as claimed in Claim 9 or Claim 10, and wherein the calculating means include means for calculating a covert code from the serial number and activation key, and the licensing subsystem including means for mapping the covert code into service tickets each time the licensing subsystem is started.

13. A computer system as claimed in any one of Claims 9 to 12 and comprising a plurality of computers connected in a network, wherein a said server comprises a directory server, providing a directory service and including a respective licensing subsystem, together with a directory database and a ticket database, wherein stored in the directory database are directory server profile details, licence details and user profile details, and wherein the ticket database includes details of service tickets available in accordance with respective licence details and issued.

14. A computer system as claimed in any one of Claims 9 to 12 and comprising a plurality of computers connected in a network, wherein a said server comprises an application server providing an application service and including a respective licensing subsystem with a respective ticket database, and another server comprises a directory server providing a directory service with a respective directory database, wherein stored in the directory database are directory server profile details, application server profile details and licence details, and wherein the ticket database includes details of service tickets available in accordance with the licence

details and issued.

15. A computer system as claimed in Claim 14 and for independent licence sharing, wherein the directory server includes a respective directory licensing subsystem and a respective ticket database, shareable licence details, for which the number of service tickets available is unlimited, being stored in the directory database, the directory ticket database including details of service tickets available in accordance with the shareable licence details and issued, and the application server ticket database including details of service tickets issued.

16. A computer system as claimed in Claim 14 and for licence sharing with site licensing, and including another said server comprising a site licensing server providing a site licensing service and including a respective licensing subsystem with a respective ticket database, the directory database also including site licensing server profile details, and wherein the site licensing ticket database includes details of service tickets available in accordance with the licence detailed and issued.

17. A software licence management method substantially as herein described with reference to an as illustrated in Figure 1, Figure 2, Figure 3, or Figure 4, of the accompanying drawings.

18. A computer system including at least one server and a software licence management system substantially as herein described with reference to and as illustrated in Figure 1, or Figure 2, or Figure 3, or Figure 4 of the accompanying drawings.

The Patent Office

29

Application No: GB 9617596.3 **Examiner:** Mike Davis
Claims searched: 1-18 **Date of search:** 26 September 1996

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.O): G4A (AAP)

Int Cl (Ed.6): G06F

Other:

Documents considered to be relevant:

Category	Identity of document and relevant passage		Relevant to claims
X	GB 2236604 A	(SUN MICROSYSTEMS)	1,9 at least
X	EP 0332304 A2	(DIGITAL EQUIPMENT)	"
X	WO 93/11480 A1	(INTERGRAPH)	"
X	US 5375206	(HUNTER ET AL)	"
X	US 4924378	(HERSHEY ET AL)	"

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

(12) UK Patent Application (19) GB (11) 2 022 969 A

(21) Application No 7924218
(22) Date of filing 11 Jul 1979
(23) Claims filed 11 Jul 1979
(30) Priority data
(31) 14400/78
(32) 12 Apr 1978
(33) United Kingdom (GB)
(43) Application published 19 Dec 1979
(51) INT CL³ G06K 15/20
(52) Domestic classification H4T 4A2 4B1
(56) Documents cited None
(58) Field of search H4T
(71) Applicants Data Recall Limited, Sondes Place, Dorking, Surrey RH4 3EF
(72) Inventor Mark-Eric Jones
(74) Agents Reddie & Grose

(54) **Video display control apparatus**

(57) Video display control apparatus for a visual display device (11, Fig. 1, not shown) employing a television-type raster in a word processor has a display memory (13), a column counter 25 and a row counter 28 adapted to address the display memory. Each location of the display memory has an address comprising a column number and a row number. A clock oscillator (18) and a timing chain (19) produce raster timing signals and column and row timing signals. The count in the column counter 25 tracks the line being scanned, and the count in the row counter tracks successive groups of lines in the raster. The display data output of the display memory controls a character matrix memory (15) acting through a parallel-to-serial converter (16) to cause alphanumeric characters to be displayed in rows by the display device. So that the information display by the display device can be varied in a convenient manner, the row counter 28 is coupled to the display memory 13 through a random access memory 30 which stores information from a central processor unit (14). This stored information determines which set of sequential row addresses shall be supplied to the display memory as the row counter 28 carries out its counting sequence, and includes an instruction associated with a selected row address which causes a reset signal 31 to be supplied to the column counter 25 so that for this row the characters displayed start at the character stored in the first column of locations in the display memory, the column addresses generated by the column counter 25 being otherwise selectable as any set formed by a predetermined number of consecutive column addresses for alphanumeric character locations in the display memory.

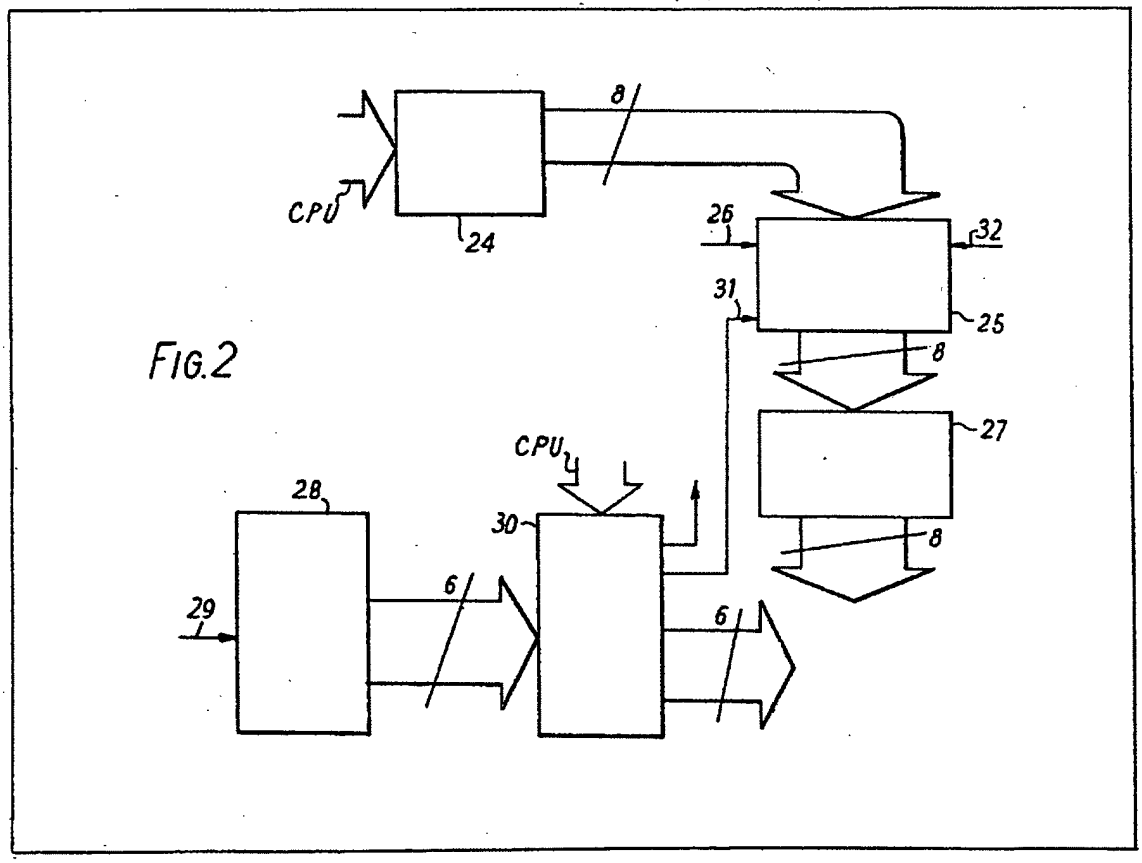
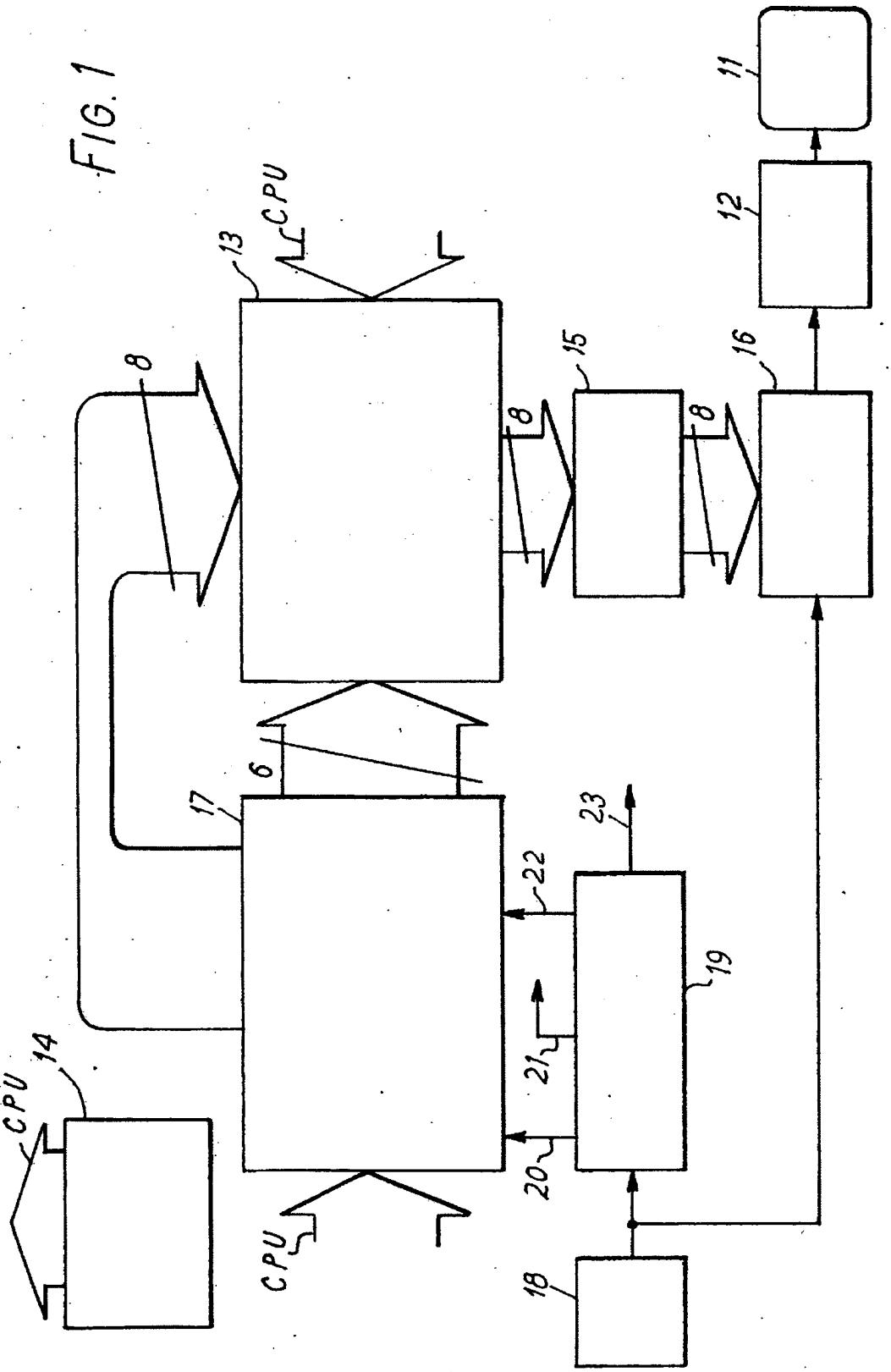


FIG.2

GB 2 022 969 A

2022969

FIG. 1



2022969

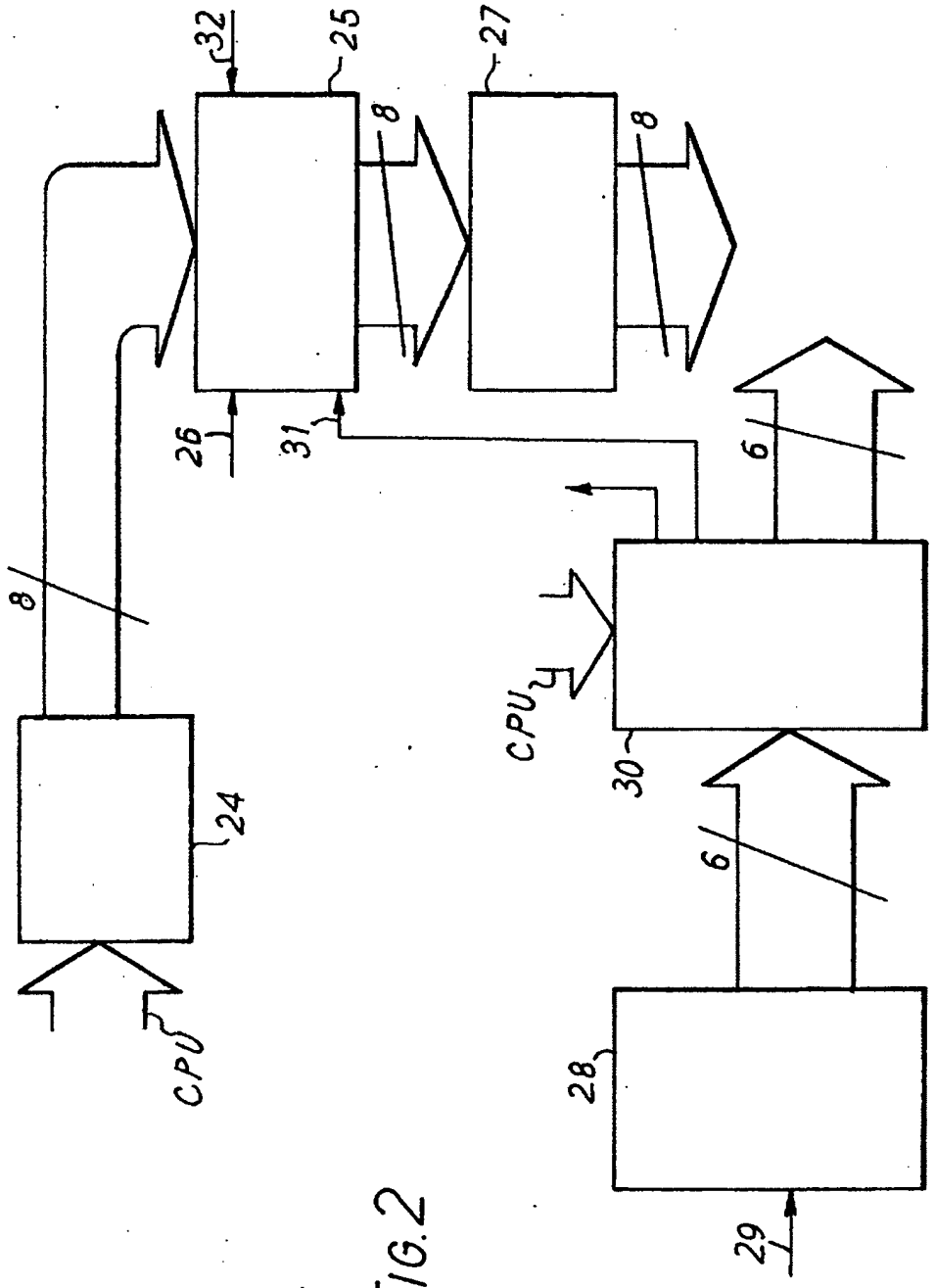


FIG.2

SPECIFICATION

Video display control apparatus

This invention relates to video display control apparatus for use with a visual display device employing a television-type scanning raster.

Visual display devices are now employed in monitoring or simply displaying information constituting the output of, for example, a computing system, a commercial information disseminating network, or a word processor. At present, such display devices are usually in the form of a cathode ray tube operated with a television-type scanning raster. It is frequently the case that the quantity of data stored in the system supplying the visual display device is greater than the amount that can be displayed simultaneously.

An object of the present invention is to provide control apparatus enabling a visual display device to vary the information display thereby in a convenient manner.

According to the present invention, therefore, there is provided video display control apparatus for use with a visual display device employing a television-type scanning raster, the control apparatus including a display memory, a column counter and a row counter adapted to address the display memory, each of a plurality of locations of the display memory having an address comprising a column number and a row number, timing means for producing raster timing signals and column and row timing signals, the timing means being so coupled to the column counter and the row counter that, in operation, the count in the column counter changes in a manner representative of the scanning of a line of the raster and the count in the row counter changes in a manner representative of the succession of lines in the raster, and means coupled to data output terminals of the display memory for producing display signals representative of display data held in addressed locations of the said plurality of locations, characterised in that the row counter is coupled to the display memory through a random access memory adapted to store a row holding instruction relating to a selected row address and to supply to the column counter a row holding signal such that the column counter in response thereto carries out its column counting or countings for the selected row address through a predetermined series of column numbers, the column counter being adapted to count a predetermined number of column numbers starting from a column number which is selectable except in the presence of the row holding instruction.

Since the count in the column counter changes in a manner representative of the scanning of a line of the raster and the count in the row counter changes in a manner representative of the succession of lines in the raster, and the column and row timing signals are such that the count in the column counter changes faster than the count in the row counter. Although the terms column and row are thus associated with the scanning of a line of the raster and the succession of lines in the raster respectively, the lines of the raster in the display in operation may be so orientated as to run from top to bottom of the display as viewed by a user. Normally, however, the lines will be orientated so as to run from left to right in the display.

Preferred features of the apparatus are defined in the sub-claims appended hereafter.

The invention will now be described in more detail, solely by way of example, with reference to the accompanying drawings, in which:—

Fig. 1 is a block diagram of a word processor embodying the invention; and

Fig. 2 is a block diagram showing in more detail part of the embodiment of Fig. 1.

In the word processor of Fig. 1, a cathode ray display tube 11 receives a video signal from a video output stage 12. Scanning circuitry for the cathode ray display tube 11 is not shown and produces a scanning raster on the screen of the tube 11, the scanning raster being formed by a large number of horizontal lines. The scanning of the raster is similar to that of a television raster except that there is no interlacing of the lines. The lines in the scan making up each frame of the raster are produced in sequence starting at the top of the frame. A display memory 13 stores alphanumeric character codes in a plurality of locations arranged to represent, for example, an array of 128 columns by 64 rows. The character codes are supplied to the display memory 13 by a central processor unit 14 which receives this information from a flexible disc, not shown, or a keyboard, not shown.

Whenever one of the locations containing a character code in the display memory is addressed, the character code is supplied to a character matrix memory 15 which stores a character scan dot code for each possible alphanumeric character. In the present example, each alphanumeric character is formed by a selection of dots from a matrix of 10 by 13 dot positions, each matrix being 13 dots high and 10 dots wide. Consequently, 13 line scans are required to scan each complete character. Thus one row consists of 13 horizontal successive lines of dots, in coded form, supplied by the matrix memory 15 to a parallel-toserial converter 16 in the form of a 10 bit shift register. The serial output of this converter is supplied to the video output stage 12 which correspondingly supplies video dot signals to the cathode ray display tube 11.

The display memory 13 is addressed by an addressing unit 17 which provides the address for each of the alphanumeric character locations of the display memory in the form of a 6 bit row address combined with an 8 bit column address. In effect, a selected succession of 80 column addresses is supplied 13 times to the display memory 13 during the supplying of each row address to the display memory 13. Consequently, each of the 13 horizontal lines of dots in coded form supplied to the converter 16 consists of 80 groups of dots, each group lying in a respective

column and being a selection of the dots forming the character at the location defined by the respective column and the current row.

Timing signals; in the form of pulses, are generated as follows.

A clock oscillator 18 generates clock pulses at, for example, 50 megahertz. The clock pulses are supplied directly to the shift register constituting the converter 16 and thus the dot rate is set at the frequency of the clock oscillator 18. The clock pulses are also supplied directly to a timing chain 19 which consists of a chain of frequency dividers (not shown). Four outputs 20, 21, 22 and 23 from the timing chain 19 are shown. Streams of pulses at successively lower rates are supplied at these outputs 20 to 23. The highest pulse rate, which is at the output 20, is supplied to the addressing unit 17 to determine the rate at which column addresses are generated. This rate is accordingly the character clock rate and may be, for example, 5 megahertz. The pulses supplied at the output 21 are generated at a rate which is used as the line frequency for the raster of the cathode ray display tube 11. Each pulse at the output 21 is very short and corresponds substantially to a line sync pulse. The rate of the pulses at the output 22 is 1/13th that of the pulses at the output 21. The pulses at the output 22 are supplied to the addressing unit 17 where they serve to determine the row address rate. The rate of the pulses at the output 23 is 1/68th of the rate of the pulses at the output 22. The pulses at the output 23 are accordingly used as frame sync pulses, i.e. the pulses which determine the instants at which rasters on the cathode ray display tube 11 are completed.

The central processor unit 14 supplies to the addressing unit 17 information which determines which succession of 80 of the 128 columns is to be addressed by the addressing unit, and which one of the 64 rows is to serve as the starting row during addressing by the addressing unit. This facility enables the cathode ray display tube 11 to display the information contained in any array of 80 columns by 64 rows selected from the array of 128 columns by 64 rows representing the stored alphanumeric characters in the display memory 13. For example, if the array represented by the locations in the display memory 13 is considered to consist of columns 1 to 128 numbered from the left and rows 1 to 64 numbered from the top, the addressed array may consist of columns 21 to 100 by rows 10 to 64 followed by rows 1 to 9. Furthermore, the information supplied to the addressing unit 17 by the central processor unit 14 can include an instruction for a selected row of the addressed array to consist of the locations in columns 1 to 80 of that row while the other rows consist of the locations in another succession of 80 columns, for example, columns 21 to 100. The means whereby this latter operation is carried out will now be described with reference to Fig. 2.

In Fig. 2, the addressing unit 17 is shown to consist of a roll left right offset latch 24 which holds the current value of the left hand column to be displayed, this value being supplied to the latch by the central processor unit, a column counter 25 coupled to the latch 24 to receive therefrom an 8 bit output representing the left hand column value held by the latch 24, and receiving at an input 26 the character rate pulses supplied by the output 20 of the timing chain 19, a buffer 27 coupled to the 8 bit output of the counter 25 and having an 8 bit output at which the column addresses supplied to the display memory 13 appear in operation, a row counter 28 which receives at an input 29 the row rate pulses provided at the output 22 of the timing chain 19, and a random access memory 30 coupled to the row counter 28 to receive therefrom a 6 bit output, and having an 8 bit output of which 6 bits are supplied to the display memory 13 as the row addresses, the 7th bit of the output being supplied to a reset input 31 of the column counter 25 and the 8th bit of the output being supplied to the display memory as a blanking signal to force the main memory to provide no alphanumeric character as output during the active time of the signal on the 8th bit of the output of the random access memory 30. The random access memory 30 also receives an input from the central processor unit which determines the prevailing relationship between the 6 bit output of the row counter 28 and the first 6 bits of the output of the random access memory 30 which are supplied as row addresses to the display memory 13. The input to the random access memory 30 from the central processor unit also determines for each row address generated by the random access memory 30 the accompanying values of the 7th and 8th bits of the output of the random access memory. In particular, the value of the 7th bit for each row address is either high or low, and in response to one of these values, the column counter 25 is reset to zero. The column counter 25 is arranged to count a succession of 112 column numbers starting from the number of the left hand column supplied to it by the latch 25 unless the counter 25 is reset to zero in which case the count of 112 successive column numbers is started at zero. Consequently, in the display on the cathode ray display tube 11, rows of alphanumeric characters are presented which start at the left hand end with the character in the left hand column determined by the value supplied to the counter 25 by the latch 24 when for the row address supplied to the display memory 13 by the random access memory 30 the 7th bit of the output of the random access memory 30 is not such as to reset the column counter 25. However, when the 7th bit of the output of the random access memory 30 accompanying the row address supplied to the display memory 13 is such as to reset the column counter 25, the corresponding row of alphanumeric characters displayed by the cathode ray display tube 11 starts at its left hand end with the character occurring in the first column of locations in the display memory 13 for that row. Line fly-back blanking pulses are supplied to another input 32 of the column counter 25 to set the counter 25 to the start of each cycle of

counting each blanking pulse occurring during the last 32 counts. In the present example, the column counter 25 is capable of counting from 0 to 255. It will be realized that the selection of the left hand column by means of the left hand column number supplied by the latch 24 to the counter 25 enables that area of the array of locations containing alphanumeric characters in the display memory 13 which is to be displayed by the cathode ray display tube 11 to be shifted to the left and to the right. Such shifting is referred to as rolling. The fixing of a particular row to the first 80 columns by the 7th bit of an output from the random access memory 30 enables rows thus selected to be held in the display on the cathode ray display tube 11 while the other rows are rolled to the left or to the right. This facility is particularly useful in the case of rows constituting headings for information appearing in the display.

The row counter 28 is such as to count from 0 to 67 and supplies its count in coded form as the 6 bit output to the random access memory 30. In a manner determined by the instructions received by the random access memory 30 from the central processor unit, the random access memory 30 translates the count of the row counter 28 into an 8 bit output signal in which the first 6 bits constitutes a row address, the 7th bit constitutes the signal to be supplied to the reset input 31 of the column counter, and the 8th bit constitutes a signal to the display memory 13 instructing that memory 13 to either provide the contents of the addressed locations or to provide a blank output signal.

The counting operation carried out by the row counter 28 is synchronised with the raster of the cathode ray display tube 11 so that the counts 64, 65, 66, and 67 occur during the frame fly-back blanking time. This locking of the counting cycle of the counter 28 to the raster timing ensures that rows of characters are automatically placed in the desired positions in the displayed array.

The random access memory 30 may be a Motorola MCM 6810AL which has a capacity of a 128 times 8 bits. The display memory 13 may be formed of 32 Texas Instruments TMS4044—15, each being a 4K by 1 bit static random access memory. The character matrix memory 15 may be formed of 8 Texas Instruments TMS4044—15. Where the random access memory 30 is a Motorola MCM 6810AL, the 6 bit input from the row counter 28 is multiplexed with the input which the random access unit 30 receives from the central processor unit.

CLAIMS

1. Video display control apparatus for use with a visual display device employing a television-type scanning raster, the control apparatus including a display memory, a column counter and a row counter adapted to address the display memory, each of a plurality of locations of the display memory having an address comprising a column number and a row number, timing means for producing raster timing signals and column and row timing signals, the timing means being so coupled to the column counter and the row counter that, in operation, the count in the column counter changes in a manner representative of the scanning of a line of the raster and the count in the row counter changes in a manner representative of the succession of lines in the raster, and means coupled to data output terminals of the display memory for producing display signal representative of display data held in addressed locations of the said plurality of locations, characterised in that the row counter is coupled to the display memory through a random access memory adapted to store a row holding instruction relating to a selected row address and to supply to the column counter a row holding signal such that the column counter in response thereto carries out its column counting or countings for the selected row address through a predetermined series of column numbers, the column counter being adapted to count a predetermined number of column numbers starting from a column number which is selectable except in the presence of the row holding instruction.

2. Apparatus according to claim 1, wherein a latch for storing a selected column number is coupled to the column counter, and the column counter is adapted to effect counting of a predetermined number of column numbers starting from the column number stored in the latch except in the presence of the row holding instruction.

3. Apparatus according to claim 1 or 2, characterised in that the column counter has a reset input terminal, the random access memory is so coupled to the column counter as to supply row holding instructions to the reset input terminal, and the column counter is such as to reset to the count zero whenever a row holding instruction is present at the reset input terminal.

4. Apparatus according to claim 3, characterised in that the random access memory is adapted to encode the count in the row counter as a different count related thereto by a constant which is selectable.

5. Apparatus according to claim 4, wherein the said locations of the display memory are filled by a central processor unit which is arranged to supply the column number to be stored to the said latch, and to supply the instructions to the random access memory which determine the said constant and determine the said selected row address.

6. Video display control apparatus substantially as described herein before with reference to the accompanying drawings.

Printed for Her Majesty's Stationery Office by the Courier Press, Leamington Spa, 1979. Published by the Patent Office, 25 Southampton Buildings, London, WC2A 1AY, from which copies may be obtained.

Booking by means of a virtual access ticket

Publication number: EP1103922
Publication date: 2001-05-30
Inventor: LAUTENSCHLAGER WOLFGANG (DE); STUERZ HEINZ (DE)
Applicant: CIT ALCATEL (FR)
Classification:
- international: ***G06Q10/00; G07B15/00;*** **G06Q10/00; G07B15/00;** (IPC1-7): G07F7/08; G06F17/60; G07B15/00; G07F17/42
- European:
Application number: EP20000124578 20001110
Priority number(s): DE19991056359 19991124

Also published as:

EP1103922 (A3)
DE19956359 (A1)

Cited documents:

EP0950968
US5598477
NL9301902
EP0713198
GB2317258
more >>

Report a data error here

Abstract of **EP1103922**

The booking method has a reservation request received from a customer by a reservation agent, with the customer charge logged by the agent and an electrical signal containing coded data corresponding to an access authorisation transmitted back to the customer, for storage on an electronic data carrier, acting as a virtual entry ticket. Also included are Independent claims for the following: (a) a central server for a reservation booking method; (b) a computer program for a reservation booking method

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 964 572 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
15.12.1999 Bulletin 1999/50

(51) Int. Cl.[6]: **H04N 5/00**

(21) Application number: **98401374.8**

(22) Date of filing: **08.06.1998**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(71) Applicant:
CANAL+ Société Anonyme
75711 Paris Cedex 15 (FR)

(72) Inventor: **Declerck, Christophe**
28210 Senantes (FR)

(74) Representative:
Cozens, Paul Dennis et al
Mathys & Squire
100 Grays Inn Road
London WC1X 8AL (GB)

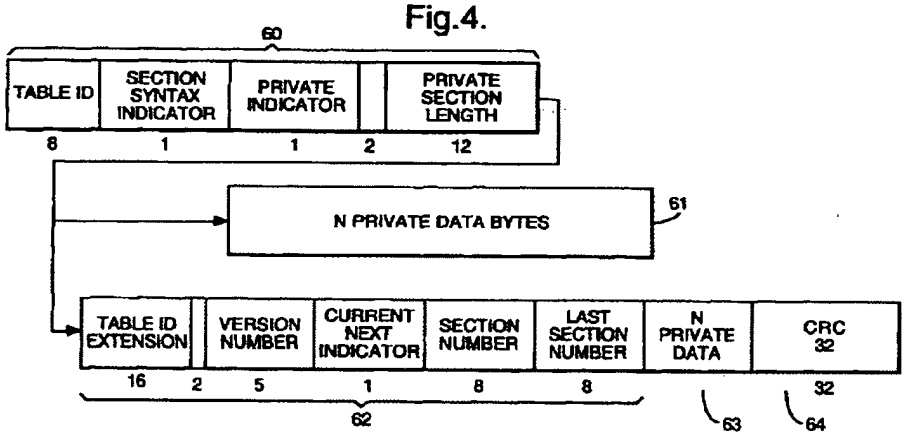
(54) Decoder and security module for a digital transmission system

(57) A decoder 12 in particular for a digital television system and adapted to receive a transport packet stream containing table or section data encapsulated within the packet payloads. The decoder is characterised in comprising a means 80 for filtering table or section data configurable in response to filter data received from a portable security module 30 such as a smart card.

The invention equally extends to a portable security module 30 including a memory holding such data as is necessary to configure the table or section filter 80, and a method for processing a transport packet stream including encapsulated table and section data using such a decoder 12 and security module 30.

In a preferred embodiment, the filter 80 is adapted to filter out conditional access messages in response to the table or section filter data received from the portable security module 30, these messages being thereafter forwarded to the security module for processing.

Fig.4.



EP 0 964 572 A1

Printed by Xerox (UK) Business Services
2.16.7/3.6

Description

[0001] The present invention relates to a decoder and security module for a digital transmission system and method of operating a decoder and security module, in particular for use in a digital television system.

[0002] Conventional digital television broadcast systems transmit data in the form of discrete transport stream packets or transport packets, each packet being of a predetermined length and containing a header and a payload. The MPEG standard is the currently favoured standard in this domain and sets out, amongst other things, a predetermined format for such packets.

[0003] The packet header comprises general descriptive data regarding the packet, whilst the payload comprises the data to be processed at the receiver. The packet header includes at least a packet ID or PID identifying the packet. The payload of the packet may contain audio, video or other data such as application data or, in particular, conditional access system data.

[0004] Conventionally, the incoming data stream is filtered by a receiver/decoder according to the PID of each packet. Data requiring immediate processing such as audio or visual data is communicated to an appropriate processor in the form of what is conventionally known as a packetised elementary stream or PES. This continuous flux of data, which is formed by assembling the payloads of the transport packets, itself comprises a sequence of packets, each PES packet comprising a packet header and payload.

[0005] Other data not requiring immediate processing may also be encapsulated within the payloads of the transport packets. Unlike PES data, which is treated immediately by a processor to generate a real time output, this sort of data is typically processed in an asynchronous manner by the decoder processor. In this case, data is formatted in a single table or a series of sections or tables, each including a header and a payload, the header of the section or table including a table ID or TID.

[0006] In the case where the access to a transmission is to be restricted, for example, in a pay TV system, conditional access data may be included in a table or section broadcast in the transport stream with the transmission. This conditional access data is filtered by the receiver/decoder and passed to a portable security module, such as smart card, inserted in the decoder. The data is then processed by the smart card in order to generate, for example, a control word subsequently used by the decoder to descramble a transmission.

[0007] One problem with known systems lies in the volume of data that will be received and processed by the receiver/decoder and notably the volume of conditional access messages eventually forwarded to the smart card or security module. In particular, the processing capabilities of a smart card processor and the capacity of the communication channel between the decoder and smart card may be insufficient to handle a given volume of messages. This problem is exacerbated by the increasing tendency for programmes to be transmitted with multiple conditional access messages enabling access by different operators to the same programme (e.g. a football match or a thematic television channel).

[0008] According to the present invention, there is provided a decoder for a digital transmission system adapted to receive a transport packet stream containing table, section or other packetised data encapsulated within the packet payloads and characterised in that the decoder comprises a means for filtering the encapsulated data configurable in response to filter data received from a portable security module.

[0009] Filtering data at the table or section level in response to information from the security module enables a more precise identification and selection of data to be carried out, for example, to extract relevant conditional access messages addressed to the module. In practice, and as will be described below, this filtering at the table or section level may be carried out after and in addition to a filtering carried out at the transport packet level.

[0010] Preferably, the means for filtering encapsulated data is configurable in response to filter data comprising at least a table ID or section ID value transmitted by the portable security module. The means for filtering encapsulated data may equally be configurable in accordance with other data received from the portable security module.

[0011] In a preferred embodiment, the means for filtering encapsulated data is further adapted to forward to the security module conditional access data obtained in accordance with the filter data received from the security module.

[0012] Whilst the present invention is particularly adapted to enable a reduction of the volume of conditional access messages communicated between the decoder and the module, it will be nevertheless appreciated that the encapsulated data may be configured by the security module to extract data other than conditional access data and having a destination other than the security module.

[0013] Conditional access data filtered and forwarded to the security module may comprise entitlement control messages (ECMs) and/or entitlement management messages (EMMs).

[0014] Even within a group of messages associated with a single conditional access system there may be a large number of messages irrelevant to a particular user within that system. For example, within a single conditional access system a number of different groups of users may be defined leading to the generation of a number of EMMs, not all of which may be relevant to a given user.

[0015] Preferably therefore, filter data provided by the security module comprises data used by the filter means to

extract group and/or individual entitlement management messages addressed to the security module.

[0016] In one embodiment, the decoder is adapted to receive a control word generated by the security module in response to the conditional access data forwarded thereto, the control word being used by the decoder to descramble a scrambled transmission.

[0017] In addition to a filtering at the table or section level, the decoder may further carry out a transport level filtering in order, for example, to extract only these packets comprising data associated with the particular conditional access system used by the security module. Preferably, therefore the decoder further comprises a means for filtering transport packet data configurable in response to data received from the security module.

[0018] Advantageously, the means for filtering transport packet data may be configurable in response to data representing the identity of the conditional access system received from the security module.

[0019] In one embodiment, the transport packet filtering means is adapted to extract transport packets containing a program map table and a conditional access table, the decoder further comprising selection means adapted to receive the program map table and conditional access table from the transport packet filtering means and conditional access identity data from the security module and thereafter configure the transport packet filtering means to extract transport packet data associated with the conditional access system in question.

[0020] In order to preserve security in the system, some or all communications between the security module and the decoder may be encrypted. In particular, the descrambling control word generated by the security module and eventually transmitted to the decoder may be encrypted.

[0021] The present invention has been described above in relation to a decoder. Other aspects of the invention relate to a method of filtering encapsulated data in a transport packet stream and a security module for use with a decoder or method of the present invention. In one embodiment, the security module may conveniently comprise a smart card.

[0022] Whilst the present invention may apply to any packet transmission system comprising a transport stream layer and a table or section layer, the present invention is particularly applicable to a decoder adapted to receive an MPEG compatible data stream.

[0023] In this regard, the term "table, section or other packetised data" refers in its broadest sense to any data table, alone or in a sequence, and comprising a header and payload and that is itself encapsulated within a transport packet stream. As will be described in the preferred embodiment, the present invention is particularly applicable to filtering of data contained within an MPEG table, notably a single MPEG short form table. Other embodiments are nevertheless conceivable, for example, in which filtering is carried out on PES packets encapsulated within the transport packet payloads.

[0024] In the context of this application, the term MPEG refers to the data transmission standards developed by the International Standards Organisation working group "Motion Pictures Expert Group" and in particular but not exclusively the MPEG-2 standard developed for digital television applications and set out in the documents ISO 13818-1, ISO 13818-2, ISO 13818-3 and ISO 13818-4. In the context of the present patent application, the term MPEG includes all variants, modifications or developments of MPEG formats applicable to the field of digital data transmission.

[0025] As used herein, the term "smart card" includes, but not exclusively so, any chip-based card device, or object of similar function and performance, possessing, for example, microprocessor and/or memory storage. Included in this term are devices having alternative physical forms to a card, for example key-shaped devices such as are often used in TV decoder systems.

[0026] The term "decoder" or "receiver/decoder" used herein may connote a receiver for receiving either encoded or non-encoded signals, for example, television and/or radio signals, which may be broadcast or transmitted by some other means. Embodiments of such receiver/decoders may include a decoder integral with the receiver for decoding the received signals, for example, in a "set-top box", a decoder functioning in combination with a physically separate receiver, as well as a decoder including additional functions, such as a web browser or integrated with a video recorder or a television.

[0027] As used herein, the term "digital transmission system" includes any transmission system for transmitting or broadcasting digital data, for example primarily audiovisual or multimedia digital data. Whilst the present invention is particularly applicable to a broadcast digital television system, the invention may also be applicable to a fixed telecommunications network for multimedia internet applications, to a closed circuit television, and so on.

[0028] As used herein, the term "digital television system" includes for example any satellite, terrestrial, cable and other system.

[0029] There will now be described, by way of example only, a preferred embodiment of the invention, with reference to the following figures, in which:

Figure 1 shows the overall architecture of a digital TV system according to this embodiment;

Figure 2 shows the architecture of the conditional access system of Figure 1;

Figure 3 shows the hierarchy of MPEG-2 packets, in particular those associated with conditional access messages;

Figure 4 shows the structure of long form and short form MPEG-2 private sections;

Figure 5 shows the elements of a receiver/decoder for use in this embodiment;

Figure 6 shows the elements of the receiver/decoder used to process the transport stream, in particular in relation to conditional access messages; and

Figure 7 shows the structure of the PID and section filters of the filter unit of Fig. 6.

[0030] An overview of a digital television broadcast and reception system 1 is shown in Figure 1. The invention includes a mostly conventional digital television system 2 which uses the MPEG-2 compression system to transmit compressed digital signals. In more detail, MPEG-2 compressor 3 in a broadcast centre receives a digital signal stream (for example a stream of audio or video signals). The compressor 3 is connected to a multiplexer and scrambler 4 by linkage 5. The multiplexer 4 receives a plurality of further input signals, assembles one or more transport streams and transmits compressed digital signals to a transmitter 6 of the broadcast centre via linkage 7, which can of course take a wide variety of forms including telecom links.

[0031] The transmitter 6 transmits electromagnetic signals via uplink 8 towards a satellite transponder 9, where they are electronically processed and broadcast via a national downlink 10 to earth receiver 11, conventionally in the form of a dish owned or rented by the end user. The signals received by receiver 11 are transmitted to an integrated receiver/decoder 12 owned or rented by the end user and connected to the end user's television set 13. The receiver/decoder 12 decodes the compressed MPEG-2 signal into a television signal for the television set 13.

[0032] A conditional access system 20 is connected to the multiplexer 4 and the receiver/decoder 12, and is located partly in the broadcast centre and partly in the decoder. It enables the end user to access digital television broadcasts from one or more broadcast suppliers. A smartcard, capable of decrypting messages relating to commercial offers (that is, one or several television programmes sold by the broadcast supplier), can be inserted into the receiver/decoder 12. Using the decoder 12 and smartcard, the end user may purchase events in either a subscription mode or a pay-per-view mode.

[0033] An interactive system 17, also connected to the multiplexer 4 and the receiver/decoder 12 and again located partly in the broadcast centre and partly in the decoder, may be provided to enable the end user to interact with various applications via a modemmed back channel 16.

[0034] The conditional access system 20 will now be described in more detail.

[0035] With reference to Figure 2, in overview the conditional access system 20 includes a Subscriber Authorization System (SAS) 21. The SAS 21 is connected to one or more Subscriber Management Systems (SMS) 22, one SMS for each broadcast supplier, by a respective TCP-IP linkage 23 (although other types of linkage could alternatively be used). Alternatively, one SMS could be shared between two broadcast suppliers, or one supplier could use two SMSs, and so on.

[0036] First encrypting units in the form of ciphering units 24 utilising "mother" smartcards 25 are connected to the SAS by linkage 26. Second encrypting units again in the form of ciphering units 27 utilising mother smartcards 28 are connected to the multiplexer 4 by linkage 29. The receiver/decoder 12 receives a "daughter" smartcard 30. It is connected directly to the SAS 21 by Communications Servers 31 via the modemmed back channel 16. The SAS sends, amongst other things, subscription rights to the daughter smartcard on request.

[0037] The smartcards contain the secrets of one or more commercial operators. The "mother" smartcard encrypts different kinds of messages and the "daughter" smartcards decrypt the messages, if they have the rights to do so.

[0038] The first and second ciphering units 24 and 27 comprise a rack, an electronic VME card with software stored on an EEPROM, up to 20 electronic cards and one smartcard 25 and 28 respectively, for each electronic card, one card 28 for encrypting the ECMs and one card 25 for encrypting the EMMs.

[0039] The operation of the conditional access system 20 of the digital television system will now be described in more detail with reference to the various components of the television system 2 and the conditional access system 20.

Multiplexer and Scrambler

[0040] With reference to Figures 1 and 2, in the broadcast centre, the digital audio or video signal is first compressed (or bit rate reduced), using the MPEG-2 compressor 3. This compressed signal is then transmitted to the multiplexer and scrambler 4 via the linkage 5 in order to be multiplexed with other data, such as other compressed data.

[0041] The scrambler generates a control word used in the scrambling process and included in the MPEG-2 stream in the multiplexer. The control word is generated internally and enables the end user's integrated receiver/decoder 12

to descramble the programme.

[0042] Access criteria, indicating how the programme is commercialised, are also added to the MPEG-2 stream. The programme may be commercialised in either one of a number of "subscription" modes and/or one of a number of "Pay Per View" (PPV) modes or events. In the subscription mode, the end user subscribes to one or more commercial offers, or "bouquets", thus getting the rights to watch every channel inside those bouquets. In the preferred embodiment, up to 960 commercial offers may be selected from a bouquet of channels.

[0043] In the Pay Per View mode, the end user is provided with the capability to purchase events as he wishes. This can be achieved by either pre-booking the event in advance ("pre-book mode"), or by purchasing the event as soon as it is broadcast ("impulse mode"). In the preferred embodiment, all users are subscribers, whether or not they watch in subscription or PPV mode, but of course PPV viewers need not necessarily be subscribers.

Entitlement Control Messages

[0044] Both the control word and the access criteria are used to build an Entitlement Control Message (ECM). This is a message sent in relation with a scrambled program; the message contains a control word (which allows for the descrambling of the program) and the access criteria of the broadcast program. The access criteria and control word are transmitted to the second encrypting unit 27 via the linkage 29. In this unit, an ECM is generated, encrypted and transmitted on to the multiplexer and scrambler 4. During a broadcast transmission, the control word typically changes every few seconds, and so ECMs are also periodically transmitted to enable the changing control word to be descrambled. For redundancy purposes, each ECM typically includes two control words; the present control word and the next control word.

[0045] Each service broadcast by a broadcast supplier in a data stream comprises a number of distinct components; for example a television programme includes a video component an audio component, a sub-title component and so on. Each of these components of a service is individually scrambled and encrypted for subsequent broadcast to the transponder 9. In respect of each scrambled component of the service, a separate ECM is required. Alternatively, a single ECM may be required for all of the scrambled components of a service. Multiple ECMs are also generated in the case where multiple conditional access systems control access to the same transmitted program.

Programme Transmission

[0046] The multiplexer 4 receives electrical signals comprising encrypted EMMs from the SAS 21, encrypted ECMs from the second encrypting unit 27 and compressed programmes from the compressor 3. The multiplexer 4 scrambles the programmes and sends the scrambled programmes, the encrypted EMMs and the encrypted ECMs to a transmitter 6 of the broadcast centre via the linkage 7. The transmitter 6 transmits electromagnetic signals towards the satellite transponder 9 via uplink 8.

Programme Reception

[0047] The satellite transponder 9 receives and processes the electromagnetic signals transmitted by the transmitter 6 and transmits the signals on to the earth receiver 11, conventionally in the form of a dish owned or rented by the end user, via downlink 10. The signals received by receiver 11 are transmitted to the integrated receiver/decoder 12 owned or rented by the end user and connected to the end user's television set 13. The receiver/decoder 12 demultiplexes the signals to obtain scrambled programmes with encrypted EMMs and encrypted ECMs.

[0048] If the programme is not scrambled, that is, no ECM has been transmitted with the MPEG-2 stream, the receiver/decoder 12 decompresses the data and transforms the signal into a video signal for transmission to television set 13.

[0049] If the programme is scrambled, the receiver/decoder 12 extracts the corresponding ECM from the MPEG-2 stream and passes the ECM to the "daughter" smartcard 30 of the end user. This slots into a housing in the receiver/decoder 12. The daughter smartcard 30 controls whether the end user has the right to decrypt the ECM and to access the programme. If not, a negative status is passed to the receiver/decoder 12 to indicate that the programme cannot be descrambled. If the end user does have the rights, the ECM is decrypted and the control word extracted. The decoder 12 can then descramble the programme using this control word. The MPEG-2 stream is decompressed and translated into a video signal for onward transmission to television set 13.

Entitlement Management Messages (EMMs)

[0050] The EMM is a message dedicated to an individual end user (subscriber), or a group of end users. Each group may contain a given number of end users. This organisation as a group aims at optimising the bandwidth; that is,

access to one group can permit the reaching of a great number of end users.

[0051] Various specific types of EMM can be used. Individual EMMs are dedicated to individual subscribers, and are typically used in the provision of Pay Per View services; these contain the group identifier and the position of the subscriber in that group.

[0052] Group subscription EMMs are dedicated to groups of, say, 256 individual users, and are typically used in the administration of some subscription services. This EMM has a group identifier and a subscribers' group bitmap.

[0053] Audience EMMs are dedicated to entire audiences, and might for example be used by a particular operator to provide certain free services. An "audience" is the totality of subscribers having smartcards which bear the same conditional access system identifier (CA ID). Finally, a "unique" EMM is addressed to the unique identifier of the smartcard.

Subscriber Management System (SMS)

[0054] A Subscriber Management System (SMS) 22 includes a database 32 which manages, amongst others, all of the end user files, commercial offers, subscriptions, PPV details, and data regarding end user consumption and authorization. The SMS may be physically remote from the SAS.

[0055] Each SMS 22 transmits messages to the SAS 21 via respective linkage 23 which imply modifications to or creations of Entitlement Management Messages (EMMs) to be transmitted to end users.

[0056] The SMS 22 also transmits messages to the SAS 21 which imply no modifications or creations of EMMS but imply only a change in an end users state (relating to the authorization granted to the end user when ordering products or to the amount that the end user will be charged).

[0057] The SAS 21 sends messages (typically requesting information such as call-back information or billing information) to the SMS 22, so that it will be apparent that communication between the two is two-way.

Subscriber Authorization System (SAS)

[0058] The messages generated by the SMS 22 are passed via linkage 23 to the Subscriber Authorization System (SAS) 21, which in turn generates messages acknowledging receipt of the messages generated by the SMS 21 and passes these acknowledgements to the SMS 22.

[0059] In overview the SAS comprises a Subscription Chain area to give rights for subscription mode and to renew the rights automatically each month, a Pay Per View Chain area to give rights for PPV events, and an EMM Injector for passing EMMs created by the Subscription and PPV chain areas to the multiplexer and scrambler 4, and hence to feed the MPEG stream with EMMs. If other rights are to be granted, such as Pay Per File (PPF) rights in the case of downloading computer software to a user's Personal Computer, other similar areas are also provided.

[0060] One function of the SAS 21 is to manage the access rights to television programmes, available as commercial offers in subscription mode or sold as PPV events according to different modes of commercialisation (pre-book mode, impulse mode). The SAS 21, according to those rights and to information received from the SMS 22, generates EMMs for the subscriber.

[0061] The EMMs are passed to the Ciphering Unit (CU) 24 for ciphering with respect to the management and exploitation keys. The CU completes the signature on the EMM and passes the EMM back to a Message Generator (MG) in the SAS 21, where a header is added. The EMMs are passed to a Message Emitter (ME) as complete EMMs. The Message Generator determines the broadcast start and stop time and the rate of emission of the EMMs, and passes these as appropriate directions along with the EMMs to the Message Emitter. The MG only generates a given EMM once; it is the ME which performs cyclic transmission of the EMMs.

[0062] On generation of an EMM, the MG assigns a unique identifier to the EMM. When the MG passes the EMM to the ME, it also passes the EMM ID. This enables identification of a particular EMM at both the MG and the ME.

[0063] In systems such as simulcrypt which are adapted to handle multiple conditional access systems e.g. associated with multiple operators, EMM streams associated with each conditional access system are generated separately and multiplexed together by the multiplexer 4 prior to transmission.

Conditional Access Messages in the Transport Stream

[0064] The different nature of ECM and EMM messages leads to differences vis à vis the mode of transmission of the messages in the MPEG transport stream. ECM messages, which carry the control words needed to descramble a programme are necessarily linked to the video and audio streams of the programme being transmitted, in contrast EMM messages are general messages broadcast asynchronously to transmit rights information to individual or groups of customers. This difference is reflected in the placing of ECM and EMM messages within the MPEG transport stream.

[0065] As is known, MPEG transport packets are of a fixed length of 188 bytes including a header. In a standard packet, the three bytes of the header following the synchronisation data comprise:

TABLE I

Transport error indicator	1 bit
Payload unit indicator	1 bit
Transport priority	1 bit
PID	13 bits
Transport scrambling control	2 bits
Adaptation field control	2 bits
Continuity counter	4 bits

[0066] The characteristics of these fields are largely determined by the MPEG standard.

[0067] Referring to Figure 3, the organisation of data within a transport stream will be described. As shown, the transport stream contains a programme association table 40 ("PAT"), the PID in the header of the packet being fixed by the MPEG-2 standard at a value of 0x00. The programme access table 40 provides the entry point for access to programme data and contains a table referring to the PID values of the programme map tables ("PMT") 41, 42 associated with a number of programmes. Each programme map table 41, 42 contains in turn a reference to the PID values of the packet streams of the audio tables 43 and video tables 44 of that programme.

[0068] As shown, the programme map table 42 also contains references to the PID values of other packets 45, 46 containing additional data relating to the programme in question. In the present case ECM data generated by a number of conditional access systems and associated with the programme in question is contained within the referred packets 45, 46.

[0069] In addition to the programme access table PAT 40, the MPEG transport stream further comprises a conditional access table 47 ("CAT"), the PID value of which is fixed at 0x01. Any packet headers containing this PID value are thus automatically identified as containing access control information. The CAT table 47 refers to the PID values of MPEG packets 48, 49, 50 associated with EMM data associated with one or more conditional access systems. As with the PMT packets, the PID values of the EMM packets referred to in the CAT table are not fixed and may be determined at the choice of the system operator.

Private Section Data

[0070] In conformity with the MPEG-2 standard, information contained with a packet payload is subject to a further level of structure according to the type of data being transported. In the case of audio, visual, teletext, subtitle or other such rapidly evolving and synchronised data, the information is assembled in the form of what is known as a packetised elementary stream or PES. This data stream, which is formed by assembling the payloads of the transmitted packets, itself comprises a sequence of packets, each packet comprising a packet header and payload. Unlike the transmitted packets in the transport stream, the length of PES packets is variable.

[0071] In the case of other data, such as application data or, in this example, ECM and EMM data, a different format from PES packeting is proscribed. In particular, data contained in the transport packet payload is divided into a series of sections or tables, the table or section header including a table ID or TID identifying the table in question. Depending on the size of the data, a section may be contained entirely within a packet payload or may be extended in a series of tables over a number of transport packets. In the MPEG-2 context, the term "table" is often used to refer to a single table of data, whilst "section" refers to one of a plurality of tables with the same TID value.

[0072] As with transport packet data and PES packet data, the data structure of a table or section is additionally defined by the MPEG-2 standard. In particular, two possible syntax forms for private table or section data are proposed; a long form or a short form, as illustrated in Figure 4.

[0073] In both the short and long form, the header includes at least the data 60 comprising:

TABLE II

Table id	8 bits
Section syntax indicator	1 bit

TABLE II (continued)

Private indicator/reserved	1 bit
ISO reserved	2 bits
Section length	12 bits

[0074] The private indicator and private section lengths are comprised of data not fixed by the MPEG-2 standard and which may be used by the system operator for his own purposes.

[0075] In the case of short form, the header 60 is immediately followed by the payload data 61. In the case of the long form, a further header section 62 is provided before the payload 63 and the message equally includes a CRC check value 64. The long form, which is typically used when a message is so long that it must be divided into a number of sections, contains the information necessary to assemble the sections, such as the section number, the number of the last section in the sequence of sections etc.

[0076] For further information regarding the long and short form table data, the reader is directed to the MPEG-2 standard.

[0077] In the case of conditional access ECM and EMM messages, the data may usually be accomodated in a single table and the short form will be the appropriate format. A specific syntax for such short form conditional access messages is proposed in the context of the present invention, namely:

TABLE III

Table id (filter data)	8 bits (1 byte)
Section syntax indicator	1 bit
Private indicator/reserved	1 bit
ISO reserved	2 bits
Section length	12 bits
CA specific header field (filter data)	56 bits (7 bytes)

[0078] For such CA messages, the table id value may be set by the system operator at, for example, 0x80 and 0x81 for ECM messages (for example, odd and even messages) and 0x82 to 0x8F for EMM messages. These values are not MPEG-2 proscribed and may be chosen at the discretion of the system operator.

[0079] Equally, in the case of the CA specific header field, hereby designated as the first 7 bytes of the payload following the header, the parameters may be set by the system operator to reflect, for example, the fact that the CA message is an EMM message carrying individual, group or audience subscription information. In this manner the "header" of such a table or section is extended.

[0080] The advantages of such message syntax will become clear later, with regard to the processing and filtering of messages by the receiver/decoder, notably by using the Table id and CA specific field data.

Receiver/decoder

[0081] Referring to Figure 5, the elements of a receiver/decoder 12 or set-top box for use in a digital broadcast system and adapted to be used in the present invention will now be described. As will be understood, the basic elements of this decoder are largely conventional and their implementation will be within the capabilities of one skilled in the art.

[0082] As shown, the decoder 12 is equipped with several interfaces for receiving and transmitting data, in particular a tuner 70 for receiving broadcast MPEG transmissions, a serial interface 71, a parallel interface 72, and a modem 73 for sending and receiving data via the telephone network. The decoder also includes a first and second smart card reader 74 and 75, the first reader 74 for accepting the subscription smart card and the second reader 75 for accepting bank and/or other smart cards.

[0083] The decoder also includes a receiver 76 for receiving infra-red control signals from a handset remote control 77 and a Peritel ouput for sending audiovisual signals to a television 13 connected to the decoder.

[0084] Processing of digital signals received via the interfaces and generation of output signals is handled by an ensemble of hardware and software elements here grouped together as a central control unit 78. The software architecture of the control unit within the decoder may correspond to that used in a known decoder and will not be described here in any detail. It may be based, for example, on a virtual machine interacting via an interface layer with a lower level

operating system implemented in the hardware components of the decoder. In terms of hardware architecture, the control unit 78 will be equipped with a processor, memory elements such as ROM, RAM, FLASH memory etc. as in known decoders.

[0085] Applications processed by the control unit 78 may be resident applications stored in the ROM or FLASH of the decoder or applications broadcast and downloaded via the MPEG interface 2 of the decoder. Applications can include program guide applications, games, interactive services, teleshopping applications, as well as initiating applications to enable the decoder to be immediately operational upon start-up and applications for configuring aspects of the decoder. Applications are stored in memory locations in the decoder and represented as resource files comprising graphic object descriptions files, unit files, variables block files, instruction sequence files, applications files, data files etc.

Filtering of Conditional Access Data

[0086] Figure 6 shows in schematic form the elements necessary for processing packet and table data in accordance with this embodiment of the invention. As will be understood, the elements shown in this figure may be implemented in hardware, software or in combination of the two.

[0087] The broadcast transmission received from the satellite receiver are passed via the conventional tuner 70 and an associated demodulator unit 79. The tuner 70 typically scans a range of frequencies, stopping when a chosen carrier frequency is detected within that range. The signals are then treated by the demodulator unit 79 which extracts and forwards the transport packet stream to a demux and filter unit 80. The filter structure of the demux and filter unit 80 will be described in detail below in relation to Figure 7. As will be understood, the actual choice of components needed to implement such a unit is at the discretion of the manufacturer and the most important aspect of such a unit is the chosen filter configuration.

[0088] In the case of data encrypted in accordance with a conditional access system as per the present embodiment, the filter unit interacts with a smart card 30 (or any other secure device) inserted in the decoder 12 and a channel parameter application 81, typically implemented as a software application in the decoder.

[0089] The filter unit 80 extracts from the transport packet stream the PMT and CAT tables present in the stream. Referring back to Figure 3, this filtering operation is carried out at a PID level, the CAT table being identified by the PID value 0x01 and the appropriate PMT table corresponding to the chosen broadcast channel being extracted via the PAT table (PID value: 0x00) and the PID value of the chosen channel identified in the PAT table.

[0090] The channel parameter application 81 additionally receives from the smart card 30 an identification of the conditional access system associated with that smart card. Again, referring back to Figure 3, a first conditional access system is associated with ECM and EMM data in the packets 45 and 48, respectively. Using the conditional access system ID received from the smart card 30 and the PMT and CAT tables received from the filter unit 80, the application 81 determines the PID values of the conditional access packets associated with the conditional access system in question and returns these values to the filter unit 80.

[0091] In the case of a simplified system, where a relatively small number of ECM and EMMs are emitted, no other filtering may be necessary and these PID values may be used by the filter unit 80 to extract all relevant ECM and EMM private sections from the identified packets and to thereafter forward the data contained within these sections to the smart card 30.

[0092] This conditional access data is then processed by the microprocessor within the smart card 30 and the control word associated with the transmission passed to a descrambling unit 83. The descrambling unit 83 receives scrambled audiovisual or other data information extracted from the transport packet stream by the demux and filter unit 80, descrambles the information using the control word and thereafter passes the data to a convention MPEG-2 chip which prepares the data for subsequent display on the associated television display.

[0093] However, whilst a PID level filter enables an extraction of those ECM and EMM messages associated exclusively with the conditional access system in question, there may nevertheless be a large proportion of messages irrelevant to the user. These messages may include group EMM messages for other user groups, individual EMM messages for other users etc. The throughput of conditional access messages passed to the smart card may therefore be very high. Given the limitations of the processor power and memory of smart cards, this throughput may be in practice more than the card can handle.

[0094] In order to overcome this problem, the smartcard 30 is adapted to pass further filter data to the unit 80 for use in a section or table level filter process.

[0095] Referring to the Table III above, tables containing conditional access data include Table id and CA specific header fields which are chosen to identify, for example, the presence of an EMM or ECM (table id values 0x80 or 0x81 and 0x82 to 0x8F, respectively) and the type of message (CA specific data identifying the group concerned by a group EMM message, the presence of an audience EMM message etc.). Depending on the data that it requires, the smart card 30 will send the necessary table id and CA specific data to configure the filter unit to extract and return only those conditional access messages of interest to the smart card. In this way, the flow of data sent to the smart card may be

reduced to conform with the processing capabilities of the smart card microprocessor.

[0096] Referring to Figure 7, the details of the filtering unit 80 will be described. Typically, the unit may be implemented as a hardware resource, driven by a firmware managing application with the receiver/decoder. As shown, a first set of filters 85 carries out a PID filtering process using the CA PID information received from the channel parameter application. The PID filters 85 may equally be configured to extract other relevant packets such as the PMT, CAT tables sent to the channel parameter application. Other PID filters (not shown) may be used to extract the audiovisual PES packet information eventually sent to the descrambler etc.

[0097] Once stripped of the packet header, the private section or table data is then routed to a set of prefilters 86 adapted to filter the 8 bytes in the extended header of a table. As shown in Table III, 1 byte of the extended header is associated with the table id, 7 bytes with the CA specific information. The filtering operation is carried out by comparison of the 8 byte pattern in a table with the filter data received from the smart card. Some bits within the 8 byte, 64 bit pattern may be masked or ignored in the evaluation. In this embodiment, 32 different patterns are proposed, a subset of these patterns being applied by the prefilters in dependence of the information received from the smart card. If one pattern matches, the section is sent to the FIFO buffer element 87. If no pattern matches, the section is ignored. The filters 86 equally act to extract from the appropriate sections the PMT and CAT table information, which is passed to a FIFO buffer 88.

[0098] Due to the characteristics of the transport layer, the arrival of sections is bursty. The buffer capacity of the buffers 87, 88 must be sufficient to handle an average rate of 5Mbits/s, with the insertion of packets being based on a regular allocation with a possible deviation of ± 25%.

[0099] In order to better understand the invention, a proposed example of operating instructions handled by the section filters 86 will now be outlined.

Filter_all_sections (Filter_id, Target, Mask, Trigger_conditions, p/n)
This command retrieves every section matching the target except masked bits after trigger_conditions occured.
Filter_next_section (Filter_id, Target, Mask, Trigger_conditions, p/n)
This command retrieves the next section matching the target except masked bits after trigger_conditions occured.
Trigger_conditions are related to other filters previously identified as matching.
Filter_id is an index between 0 and 31, pointing to a filter and an output queue. In addition, it gives the queueing priority, 0 being the highest priority.
Target is an 8 bytes pattern.
Mask is an 8 bytes pattern showing the bits to be masked in the target, value 0 means masked.
Trigger_conditions is a 32 bib bitmap, ORing filter_id triggering that filter. Bit set at 0 means no trigger condition. Self trigger condition is ignored.
p/n is a value, normally set to 1, positive for normal operation as described above. When set to 0 it means negative filtering, i.e., retrieve sections not matching target.

Examples of use:

Example 1:

[0100]

Filter_all_sections(5, 0x8C7C453AA8BBFF00, 0XFF557FFFEEFFFF00, 0, 1) will capture all EMMs corresponding To matching criteria.

Example 2:

[0101]

Filter_next_section(0, 0x8000000000000000, 0xFF00000000000000, 0, 1)
Filter_next_section(1, 0x8100000000000000, 0xFF00000000000000, 5, 1)
Filter_next_section(2, 0x8000000000000000, 0xFF00000000000000, 3, 1)

will start an ECM capture process with odd/even toggle.

Example 3:

[0102]

Filter_next_section(8, 0xPMT_TID0000Version_number00000000, 0xFF00001F00000000, 0, 0)
Filter_next_section(1, 0x8100000000000000, 0xFF00000000000000, 0x14, 1)
Filter_next_section(2, 0x8000000000000000, 0xFF00000000000000, 0x12, 1)

will start an ECM capture process with odd/even toggle, starting when there is a change in the PMT.

[0103] In terms of communication of CA messages and filter data to and from the smart card 82 and filter unit 80, a standard protocol such as ISO7816 may be used. Since not all of the data in the filtered private section is required by the smart card 82, the section may be modified and a message of the following format sent to the smart card:

Table id	8 bits
Zero	11 bits
Filter id	5 bits
CA specific header field	56 bits
CA message	N*8 bits

[0104] The meaning of each of these terms will be clear from the above description. In terms of the filter data sent from the smart card 82 to the filter 80, the following format may be used:

Number of filters	8 bits
Filtering instruction	5 bits
Filter id	5 bits
Target	64 bits
Mask	64 bits
Trigger conditions	5 bits
p/n	1 bit

Number_of_filters describes the number of filters to be set in this instruction.
Filtering_instruction is describing the type of instruction (filter next section, filter all sections).
Filter_id is an index pointing to a filter and an output queue. In addition, it gives the queueing priority, 0 being the highest priority.
Target is the target pattern.
Mask is a pattern showing the bits to be masked in the target, value 0 means masked.
Trigger_conditions is a bitmap. ORing filter_id triggering that filter. Bit set at 0 means no trigger condition. Self trigger condition is ignored.
p/n is a value, normally set to 1, positive for normal operation as described above. When set to 0 it means negative filtering, i.e., retrieve sections not matching target.

[0105] In practice, communications between the smart card and the receiver/decoder may be subject to a level of encryption or scrambling for security reasons. In particular, communications between the smart card 82 and filter unit 80, as well as the control word stream sent to the descrambler unit 83 may be encoded in this way. Encryption algorithms suitable for this purpose are widely known (RSA, DES etc.).

Claims

1. A decoder adapted to receive a transport packet stream containing table, section or other packetised data encapsulated within the packet payloads and characterised in that the decoder comprises a means for filtering the encapsulated data configurable in response to filter data received from a portable security module.

2. A decoder as claimed in claim 1 in which the means for filtering encapsulated data is configurable in response to filter data comprising at least a table ID or section ID value transmitted by the portable security module.

3. A decoder as claimed in claim 1 or 2 in which the means for filtering encapsulated data is further adapted to forward to the security module conditional access data obtained in accordance with the filter data received from the security module.

4. A decoder as claimed in claim 3 in which conditional access data forwarded to the security module comprises entitlement control messages (ECMs) and/or entitlement management messages (EMMs).

5. A decoder as claimed in claim 3 or 4 in which filter data provided by the security module comprises data used by the filter means to extract group and/or individual entitlement management messages addressed to the security module.

6. A decoder as claimed in any of claims 3 to 5 in which the decoder is adapted to receive a control word generated by the security module in response to the conditional access data forwarded thereto, the control word being used by the decoder to descramble a scrambled transmission.

7. A decoder as claimed in any preceding claim further comprising a means for filtering transport packet data configurable in response to data received from the security module.

8. A decoder as claimed in claim 7, in which the means for filtering transport packet data is configurable in response to data representing the identity of the conditional access system received from the security module.

9. A decoder as claimed in claim 8 in which the transport packet filtering means is adapted to extract transport packets containing a program map table and a conditional access table, the decoder further comprising selection means adapted to receive the program map table and conditional access table from the transport packet filtering means and conditional access identity data from the security module and thereafter configure the transport packet filtering means to extract transport packet data associated with the conditional access system in question.

10. A decoder as claimed in any preceding claim adapted to process encrypt and/or decrypt communications to and from the portable security module.

11. A security module for use with a decoder as claimed in any preceding claim and characterised in comprising a memory means for storing filter data subsequently communicated to the decoder to configure the means for filtering encapsulated data.

12. A security module as claimed in claim 13 comprising a smart card.

13. A method of processing a transport packet stream containing table, section or other packetised data encapsulated within the packet payloads characterised by receiving the transport stream in a decoder and filtering the encapsulated data in response to filter data received from a portable security module.

14. A method of processing a transport packet stream as claimed in claim 13 further comprising generating encapsulated data including conditional access data and filtering at the decoder using the encapsulated data and in response to filter data supplied by the portable security module.

Fig.1.

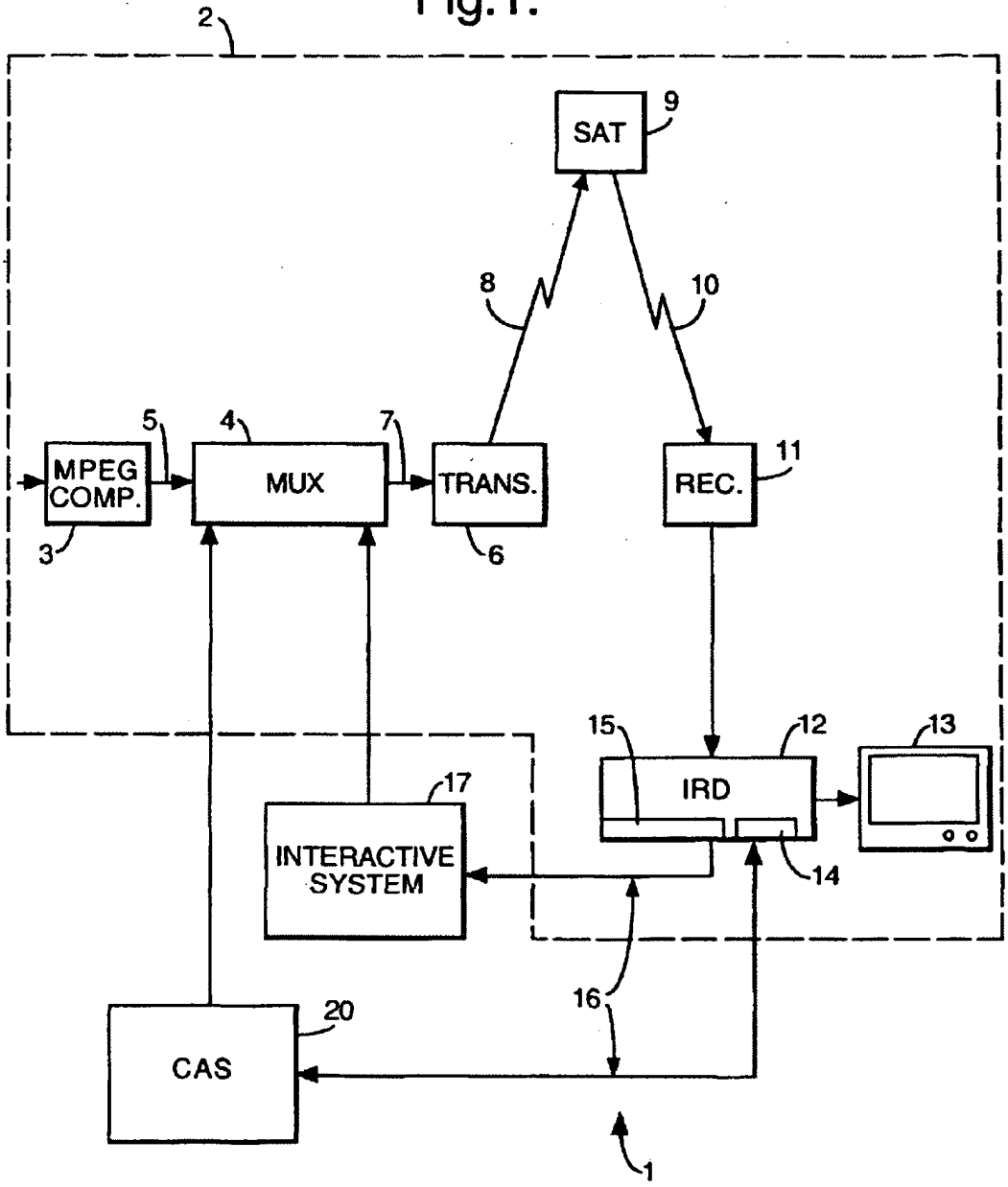


Fig.2.

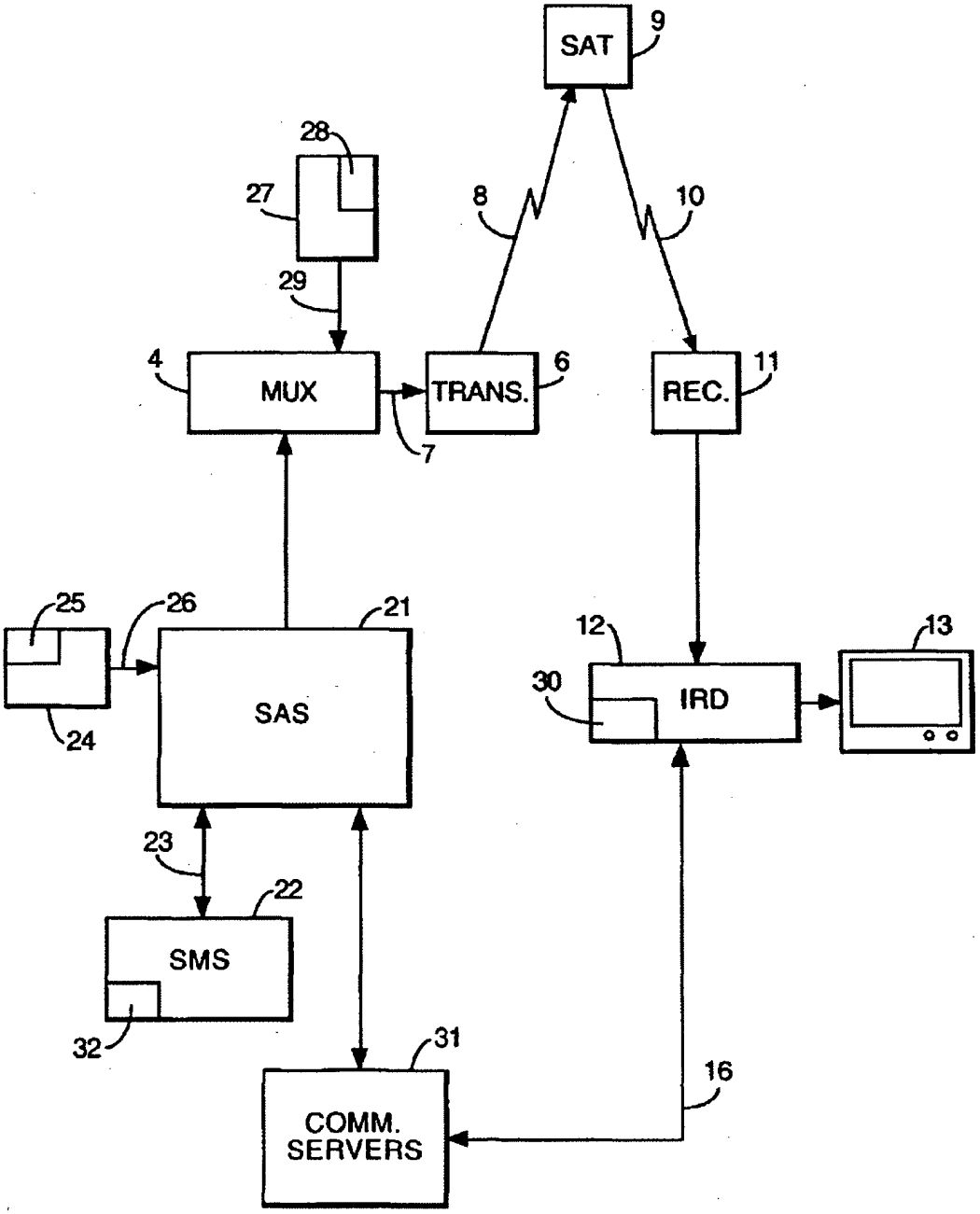


Fig.3.

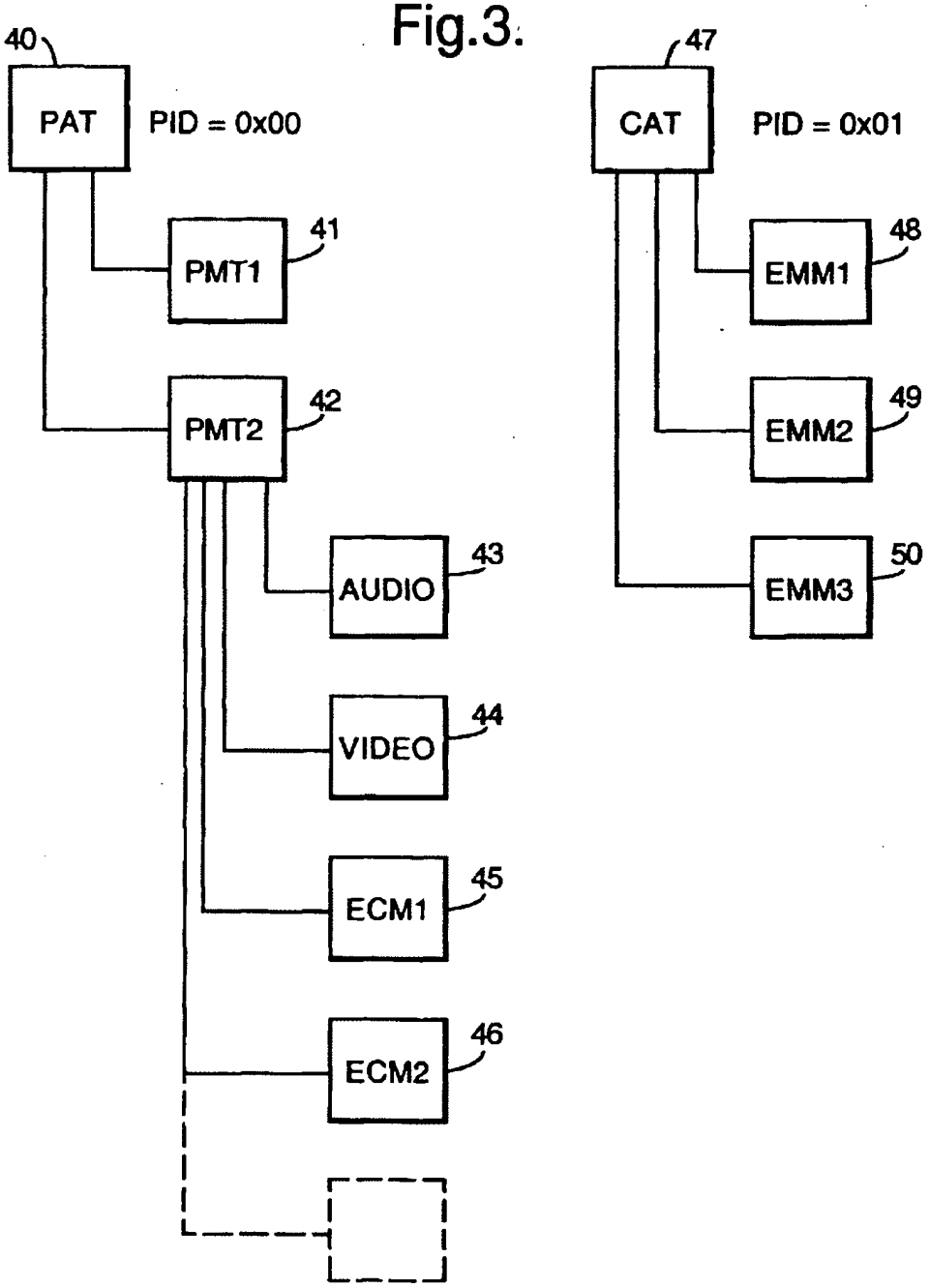


Fig.4.

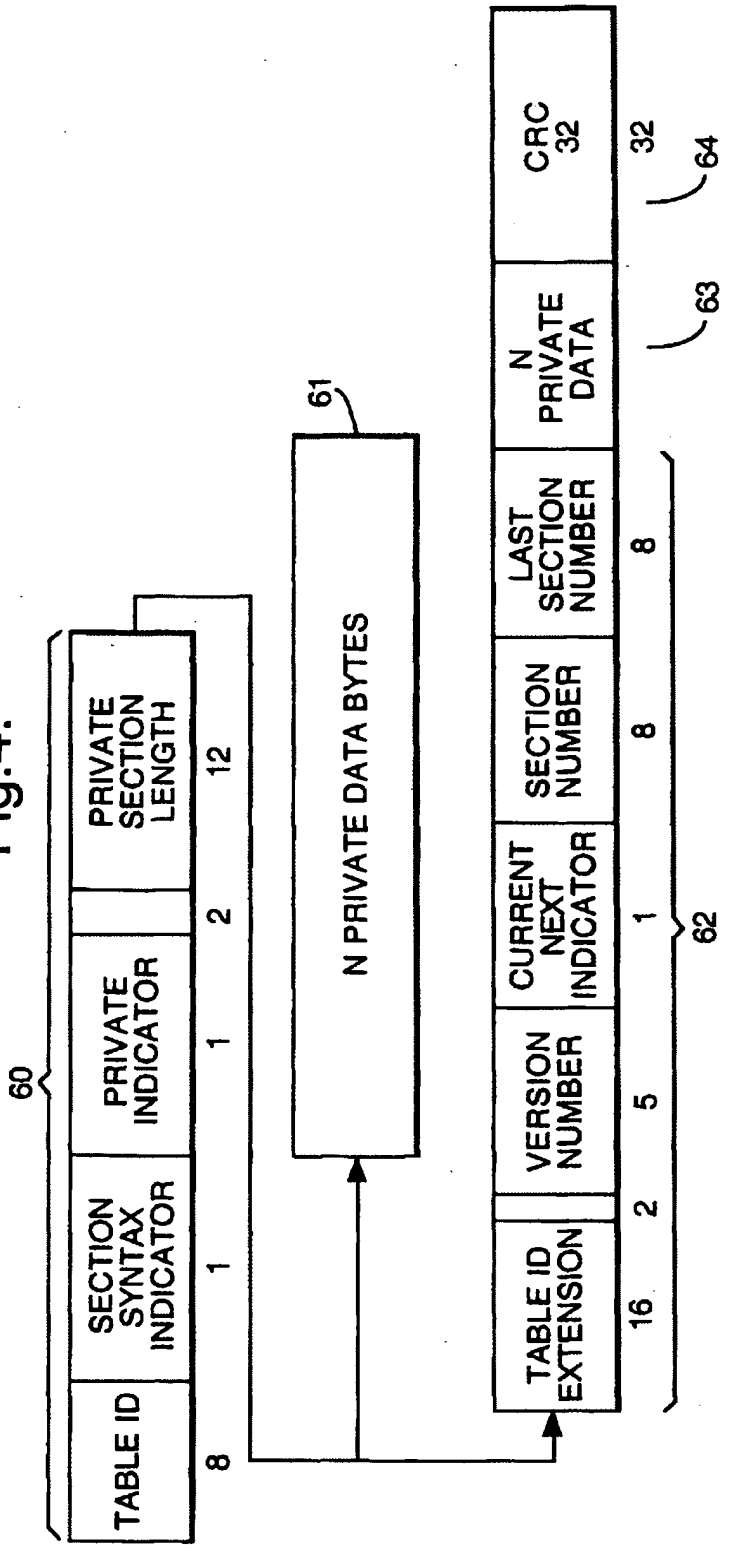


Fig.5.

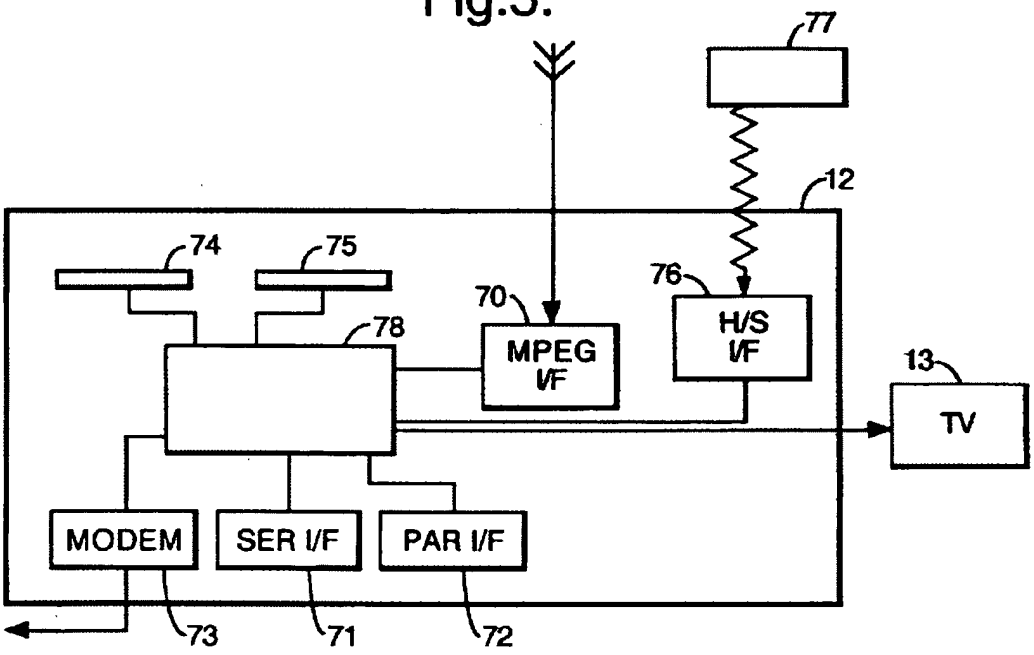


Fig.6.

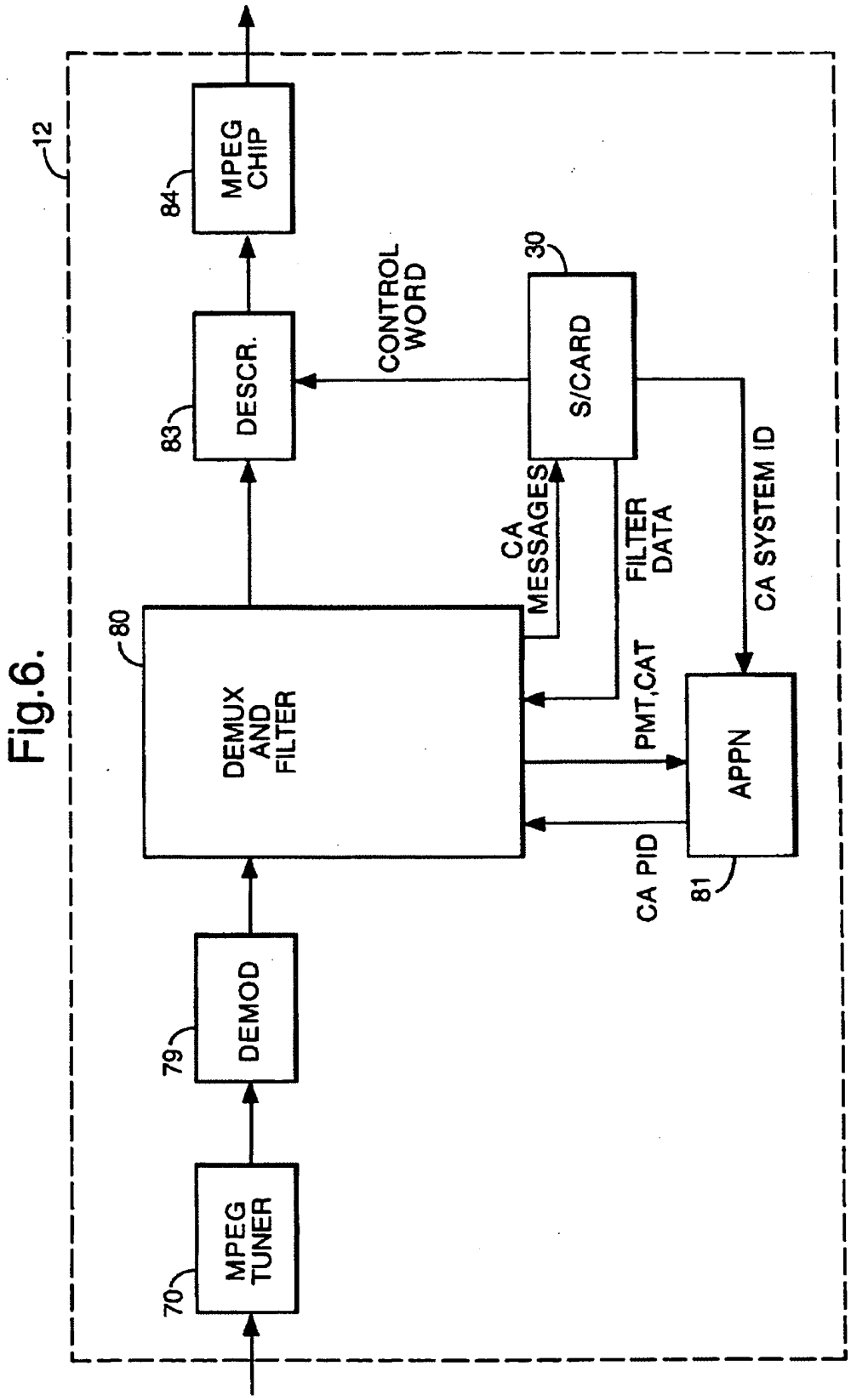
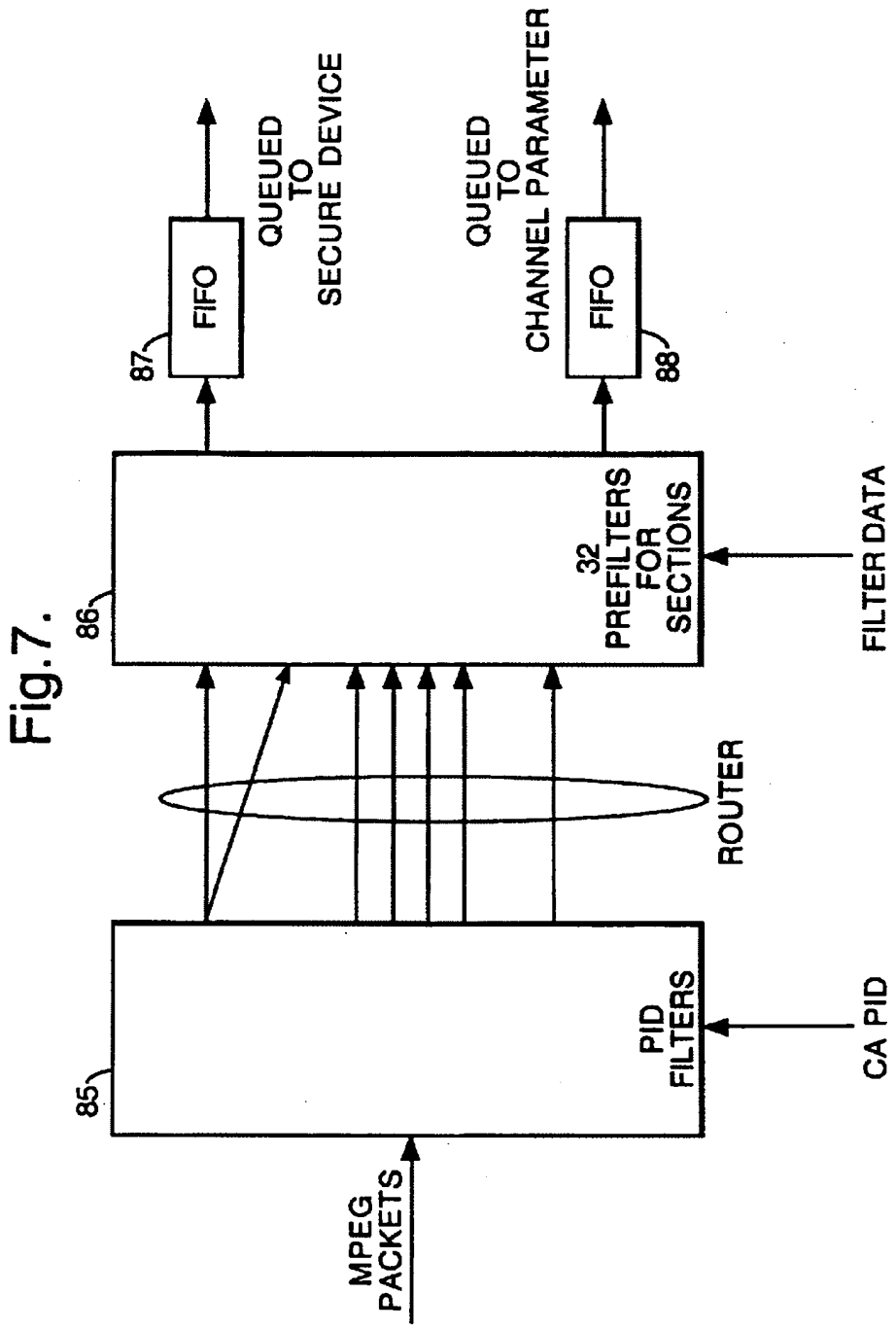


Fig.7.



European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 98 40 1374

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
X	WO 95 29560 A (THOMSON CONSUMER ELECTRONICS) 2 November 1995 * page 1, line 35 - page 2, line 25 * * page 4, line 23 - page 8, line 35 * * figure 3 *	1,3-5,8, 10-14	H04N5/00
A		2,6,7,9	

X	WO 97 46008 A (THOMSON CONSUMER ELECTRONICS) 4 December 1997 * page 3, line 17 - page 10, line 9 *	1-3,6-14	
A		4,5	

A	"FUNCTIONAL MODEL OF A CONDITIONAL ACCESS SYSTEM" 21 December 1995 , EBU REVIEW- TECHNICAL, NR. 266, PAGE(S) 64 - 77 XP000559450 * the whole document *	1-14	

A	SCHOONEVELD VAN D: "STANDARDIZATION OF CONDITIONAL ACCESS SYSTEMS FOR DIGITAL PAY TELEVISION" PHILIPS JOURNAL OF RESEARCH, vol. 50, no. 1/02, July 1996, pages 217-225, XP000627672 * page 218, line 12 - page 220, line 9 *	1-14	

TECHNICAL FIELDS SEARCHED (Int.Cl.6)
H04N

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	3 November 1998	Fassnacht, C

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

```
E26         1  PN=BR 9810991
E27         1  PN=BR 9810992
E28         1  PN=BR 9810993
E29         1  PN=BR 9810994
E30         1  PN=BR 9810995
E31         1  PN=BR 9810996
E32         1  PN=BR 9810997
E33         1  PN=BR 9810998
E34         1  PN=BR 9810999
E35         1  PN=BR 9811000
E36         1  PN=BR 9811001
E37         1  PN=BR 9811002
E38         1  PN=BR 9811004
E39         1  PN=BR 9811005
E40         1  PN=BR 9811006
E41         1  PN=BR 9811007
E42         1  PN=BR 9811008
E43         1  PN=BR 9811009
E44         1  PN=BR 9811010
E45         1  PN=BR 9811011
E46         1  PN=BR 9811012
E47         1  PN=BR 9811013
E48         1  PN=BR 9811014
E49         1  PN=BR 9811015
E50         1  PN=BR 9811016

Enter P or PAGE for more
```

? s e3

```
      S1        1  PN='BR 9810967'
```

? t 1/7/1

1/7/1
DIALOG(R)File 351: Derwent WPI

0009253575 *Drawing available*
WPI Acc no: 1999-181268/199915
Related WPI Acc No: 1996-465320; 1997-363998; 1998-363180; 1999-154174; 1999-154175; 1999-154176; 1999-154177; 1999-154178; 1999-154179; 1999-243551; 2002-060946; 2002-499082; 2002-705909; 2002-722051; 2002-722052; 2003-677663; 2003-898213; 2004-155029; 2004-478232; 2004-579235; 2004-623798; 2005-809338; 2007-015228
XRPX Acc No: N1999-133079
method for decrypting an instance of service that has been decrypted with short-term key
Patent Assignee: SCIENTIFIC-ATLANTA INC (SCAT)
Inventor: AKINS G L; PALGON M S; PINDER H G; WASILEWSKI A J; AKINS G

Patent Family (8 patents, 79 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
WO 1999009743	A2	19990225	WO 1998US16079	A	19980731	199915	B

AU 199915816	A	19990308	AU 199915816	A	19980731	199929	E
EP 1000511	A2	20000517	EP 1998960147	A	19980731	200028	E
			WO 1998US16079	A	19980731		
BR 199810967	A	20011030	BR 199810967	A	19980731	200173	E
			WO 1998US16079	A	19980731		
EP 1000511	B1	20011114	EP 1998960147	A	19980731	200175	E
			WO 1998US16079	A	19980731		
DE 69802540	E	20011220	DE 69802540	A	19980731	200207	E
			EP 1998960147	A	19980731		
			WO 1998US16079	A	19980731		
JP 2003521820	W	20030715	WO 1998US16079	A	19980731	200347	E
			JP 2000510276	A	19980731		
JP 2005253109	A	20050915	JP 2000510276	A	19980731	200560	E
			JP 2005120425	A	20050418		

Priority Applications (no., kind, date): US 199754575 P 19970801; US 1998126921 A 19980731

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
WO 1999009743	A2	EN	113	29		
National Designated States,Original	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					
Regional Designated States,Original	AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW NL OA PT SD SE SZ UG ZW					
AU 199915816	A	EN			Based on OPI patent	WO 1999009743
EP 1000511	A2	EN			PCT Application	WO 1998US16079
					Based on OPI patent	WO 1999009743
Regional Designated States,Original	DE FR GB IT NL					
BR 199810967	A	PT			PCT Application	WO 1998US16079
					Based on OPI patent	WO 1999009743
EP 1000511	B1	EN			PCT Application	WO 1998US16079
					Based on OPI patent	WO 1999009743
Regional Designated States,Original	DE FR GB IT NL					
DE 69802540	E	DE			Application	EP 1998960147
					PCT Application	WO 1998US16079
					Based on OPI patent	EP 1000511

				Based on OPI patent	WO 1999009743
JP 2003521820	W	JA	136	PCT Application	WO 1998US16079
				Based on OPI patent	WO 1999009743
JP 2005253109	A	JA	59	Division of application	JP 2000510276

Alerting Abstract WO A2

NOVELTY - The method involves receiving a second message in a receiver together with the instance of the service. The second message includes a key derivation value that is used with a long-term key to obtain the short-term key to decrypt the instance of the service.

DESCRIPTION - A control word is combined into an encrypted coded message (ECM) (107) with other service-related information. The ECM (107) is authenticated by Control Word Encrypt & Message Authenticate function (204) which produces a message authentication code using a keyed-hash value derived from the message content combined with a secret which can be shared with the receiving set-top box (113). This secret is preferably part or all of a multisession key (MSS) (208). The message authentication code is appended to the rest of the ECM (107). The CAW (202) is always encrypted before being sent along with the other parts of the ECM to MX (200). This encryption is preferably a symmetric cipher such as the Triple-DES algorithm using two distinct 56-bit keys (which taken together comprise MSS (208).

USE - The invention concerns systems for protecting information and more particularly concerns systems for protecting information that is transmitted by a wired or wireless medium against unauthorized access.

ADVANTAGE - The service distribution organizations require access restrictions which are both more secure and more flexible than those in conventional systems

DESCRIPTION OF DRAWINGS - The drawing is a block diagram of service instance encryption techniques.

107 encrypted coded message

204 Control Word Encrypt & Message Authenticate function

200 MX

Title Terms /Index Terms/Additional Words: METHOD; INSTANCE; SERVICE; SHORT; TERM; KEY

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
H04L-009/08			Main		"Version 7"
H04H-001/00; H04N-007/167; H04N-007/173			Secondary		"Version 7"
H04H-0001/00	A	I	L	R	20060101
H04L-0009/08	A	I	L	R	20060101
H04N-0005/00	A	I		R	20060101
H04N-0007/16	A	I		R	20060101
H04N-0007/167	A	I		R	20060101
H04N-0007/173	A	I	F	R	20060101

H04H-0001/00	C	I	L	R	20060101
H04L-0009/08	C	I	F	R	20060101
H04N-0005/00	C	I		R	20060101
H04N-0007/16	C	I		R	20060101
H04N-0007/167	C	I		R	20060101
H04N-0007/173	C	I	L	R	20060101

File Segment: EPI;
DWPI Class: W02; W03
Manual Codes (EPI/S-X): W02-F05A1B; W03-A16C3A

Original Publication Data by Authority

Australia

Publication No. AU 199915816 A (Update 199929 E)
Publication Date: 19990308
Assignee: SCIENTIFIC-ATLANTA INC; US (SCAT)
Language: EN
Application: AU 199915816 A 19980731 (Local application)
Priority: US 199754575 P 19970801
US 1998126921 A 19980731
Related Publication: WO 1999009743 A (Based on OPI patent)
Current IPC: H04H-1/00(R,I,M,JP,20060101,20051220,A,L) H04H-1/00 (R,I,M,JP,20060101,20051220,C,L) H04L-9/08(R,I,M,JP,20060101,20051220,A,L) H04L-9/08 (R,I,M,JP,20060101,20060310,C,F) H04N-5/00(R,I,M,EP,20060101,20051008,A) H04N-5/00 (R,I,M,EP,20060101,20051008,C) H04N-7/16(R,I,M,EP,20060101,20051008,A) H04N-7/16 (R,I,M,EP,20060101,20051008,C) H04N-7/167(R,I,M,EP,20060101,20051008,A) H04N-7/167 (R,I,M,EP,20060101,20051008,C) H04N-7/173(R,I,M,JP,20060101,20051220,A,F) H04N-7/173 (R,I,M,JP,20060101,20051220,C,L)

Brazil

Publication No. BR 199810967 A (Update 200173 E)
Publication Date: 20011030
Assignee: SCIENTIFIC-ATLANTA INC (SCAT)
Inventor: WASILEWSKI A J
AKINS G L
PALGON M S
PINDER H G
Language: PT
Application: BR 199810967 A 19980731 (Local application)
WO 1998US16079 A 19980731 (PCT Application)
Priority: US 199754575 P 19970801
US 1998126921 A 19980731
Related Publication: WO 1999009743 A (Based on OPI patent)

Current IPC: H04H-1/00(R,I,M,JP,20060101,20051220,A,L) H04H-1/00
(R,I,M,JP,20060101,20051220,C,L) H04L-9/08(R,I,M,JP,20060101,20051220,A,L) H04L-9/08
(R,I,M,JP,20060101,20060310,C,F) H04N-5/00(R,I,M,EP,20060101,20051008,A) H04N-5/00
(R,I,M,EP,20060101,20051008,C) H04N-7/16(R,I,M,EP,20060101,20051008,A) H04N-7/16
(R,I,M,EP,20060101,20051008,C) H04N-7/167(R,I,M,EP,20060101,20051008,A) H04N-7/167
(R,I,M,EP,20060101,20051008,C) H04N-7/173(R,I,M,JP,20060101,20051220,A,F) H04N-7/173
(R,I,M,JP,20060101,20051220,C,L)

Germany

Publication No. DE 69802540 E (Update 200207 E)
Publication Date: 20011220
Assignee: SCIENTIFIC-ATLANTA INC; US (SCAT)
Language: DE
Application: DE 69802540 A 19980731 (Local application)
EP 1998960147 A 19980731 (Application)
WO 1998US16079 A 19980731 (PCT Application)
Priority: US 199754575 P 19970801
US 1998126921 A 19980731
Related Publication: EP 1000511 A (Based on OPI patent)
WO 1999009743 A (Based on OPI patent)

EPO

Publication No. EP 1000511 A2 (Update 200028 E)
Publication Date: 20000517
Assignee: SCIENTIFIC-ATLANTA, INC., One Technology Parkway South, Norcross, Georgia 30092, US
Inventor: AKINS, Glendon, L., III, 2510 Windward Lane N.E., Gainesville, GA 30501, US
PALGON, Michael, S., 1196 Poplar Grove Drive, Atlanta, GA 30306, US
PINDER, Howard, G., 4317 Stilson Circle, Norcross, GA 30092, US
WASILEWSKI, Anthony, J., 10680 Wren Ridge Road, Alpharetta, GA 30022, US
Agent: Kugele, Bernhard, NOVAPAT INTERNATIONAL SA, 9, Rue du Valais, 1202 Geneve, CH
Language: EN
Application: EP 1998960147 A 19980731 (Local application)
WO 1998US16079 A 19980731 (PCT Application)
Priority: US 199754575 P 19970801
US 1998126921 A 19980731
Related Publication: WO 1999009743 A (Based on OPI patent)
Designated States: (Regional Original) DE FR GB IT NL
Original IPC: H04N-7/167(A)
Current IPC: H04H-1/00(R,I,M,JP,20060101,20051220,A,L) H04H-1/00
(R,I,M,JP,20060101,20051220,C,L) H04L-9/08(R,I,M,JP,20060101,20051220,A,L) H04L-9/08
(R,I,M,JP,20060101,20060310,C,F) H04N-5/00(R,I,M,EP,20060101,20051008,A) H04N-5/00
(R,I,M,EP,20060101,20051008,C) H04N-7/16(R,I,M,EP,20060101,20051008,A) H04N-7/16
(R,I,M,EP,20060101,20051008,C) H04N-7/167(R,I,M,EP,20060101,20051008,A) H04N-7/167
(R,I,M,EP,20060101,20051008,C) H04N-7/173(R,I,M,JP,20060101,20051220,A,F) H04N-7/173
(R,I,M,JP,20060101,20051220,C,L)
Original Abstract:

A cable television system provides conditional access to services. The cable television system includes a headend from which service "instances", or programs, are broadcast and a plurality of set top units for receiving the instances and selectively decrypting the instances for display to system subscribers. The service instances are encrypted using public and/or private keys provided by service providers or central authorization agents. Keys used by the set tops for selective decryption may also be public or private in nature, and such keys may be reassigned at different times to provide a cable television system in which piracy concerns are minimized.

Publication No. EP 1000511 B1 (Update 200175 E)
Publication Date: 20011114
Assignee: Scientific-Atlanta, Inc., 5030 Sugarloaf Parkway, Lawrenceville, GA 30044, US
Inventor: AKINS, Glendon, L., III, 2510 Windward Lane N.E., Gainesville, GA 30501, US
PALGON, Michael, S., 1196 Poplar Grove Drive, Atlanta, GA 30306, US
PINDER, Howard, G., 4317 Stilson Circle, Norcross, GA 30092, US
WASILEWSKI, Anthony, J., 10680 Wren Ridge Road, Alpharetta, GA 30022, US
Agent: Kugele, Bernhard, NOVAPAT INTERNATIONAL SA, 9, Rue du Valais, 1202 Geneve, CH
Language: EN
Application: EP 1998960147 A 19980731 (Local application)
WO 1998US16079 A 19980731 (PCT Application)
Priority: US 199754575 P 19970801
US 1998126921 A 19980731
Related Publication: WO 1999009743 A (Based on OPI patent)
Designated States: (Regional Original) DE FR GB IT NL
Original IPC: H04N-7/167(A)
Current IPC: H04H-1/00(R,I,M,JP,20060101,20051220,A,L) H04H-1/00 (R,I,M,JP,20060101,20051220,C,L) H04L-9/08(R,I,M,JP,20060101,20051220,A,L) H04L-9/08 (R,I,M,JP,20060101,20060310,C,F) H04N-5/00(R,I,M,EP,20060101,20051008,A) H04N-5/00 (R,I,M,EP,20060101,20051008,C) H04N-7/16(R,I,M,EP,20060101,20051008,A) H04N-7/16 (R,I,M,EP,20060101,20051008,C) H04N-7/167(R,I,M,EP,20060101,20051008,A) H04N-7/167 (R,I,M,EP,20060101,20051008,C) H04N-7/ 173(R,I,M,JP,20060101,20051220,A,F) H04N-7/173 (R,I,M,JP,20060101,20051220,C,L)
Claim:

1. Verfahren der Entschlusselung einer Diensteeinheit (325), die mit einem gegebenen Kurzzeitschlussel (319) verschlusselt wurde, wobei das Verfahren in einem Empfanger (333) ausgefuhrt wird, der ein Offentlich/Privat-Schlusselpaar besitzt, und das Verfahren durch die folgenden Schritte **gekennzeichnet** ist:
   - o im Empfanger eine erste Nachricht (315) zu empfangen, deren Inhalt einen ersten Langzeitschlussel (309) einschliesst und unter Verwendung des offentlichen Schlussels (312) fiir den Empfanger (333) verschlusselt wurde;
   - o den privaten Schlussel (337) zur Entschlusselung des Inhalts zu verwenden;
   - o den ersten Schlussel (309) zu speichern;
   - o im Empfanger (333) zusammen mit der verschlusselten Diensteeinheit (329) eine zweite Nachricht (323) zu empfangen, wobei die zweite Nachricht (323) einen Indikator fur einen zweiten Kurzzeitschlussel (319) einschliesst;
   - o den Indikator und den ersten Schlussel (309) zu benutzen, um den zweiten Schlussel zu erhalten; worin der zweite Schlussel dem gegebenen Schlussel (319), mit dem der Dienst verschlusselt wurde, gleichwertig ist, und
   - o den zweiten Schlussel zur Entschlusselung der empfangenen Diensteeinheit zu

verwenden.

1. A method of decrypting an instance of a service (325) that has been encrypted with a given short-term key (319), the method being carried out in a receiver (333) that has a public key-private key pair and the method being **characterised by** the following steps:
   - o receiving a first message (315) in the receiver whose contents include a first long-term key (309), the contents having been encrypted using the public key (312) for the receiver (333);
   - o using the private key (337) to decrypt the contents;
   - o storing the first key (309);
   - o receiving a second message (323) in the receiver (333) together with the encrypted instance of the service (329), the second message (323) including an indicator for a second short-term key (319);
   - o using the indicator an the first key (309) to obtain the second key; wherein the second key is equivalent to the given key (319) that encrypted the service, and
   - o using the second key to decrypt the received instance of the service.

1. Procede de decryptage d'une instance d'un service (326) qui etait cryptee avec une cle a court terme donnee (319), le procede etant execute dans un recepteur (333) qui comporte une paire de cle publique-cle privee et le procede etant **caracterise par** les etapes suivantes:
   - o recevoir un premier message (315) dans le recepteur dont le contenu comprend une premiere cle a long terme (309), le contenu ayant ete crypte en utilisant la cle publique (312) pour le recepteur (333),
   - o utiliser la cle privee (337) pour decrypter le contenu,
   - o memoriser la premiere cle (309),
   - o recevoir un second message (323) dans le recepteur (333) en meme temps que l'instance cryptee du service (329), le second message (323) comprenant un indicateur pour une seconde cle a court terme (319),
   - o utiliser l'indicateur et la premiere cle (309) pour obtenir la seconde cle, dans lequel
   - o la seconde cle est equivalente a la cle donnee (319) qui a crypte le service, et
   - o utiliser la seconde cle pour decrypter l'instance recue du service.

Japan

Publication No. JP 2003521820 W (Update 200347 E)
Publication Date: 20030715
Language: JA (136 pages)
Application: WO 1998US16079 A 19980731 (PCT Application)
JP 2000510276 A 19980731 (Local application)
Priority: US 199754575 P 19970801
US 1998126921 A 19980731
Related Publication: WO 1999009743 A (Based on OPI patent)

Original IPC: H04L-9/08(A) H04H-1/00(B) H04N-7/167(B) H04N-7/173(B)
Current IPC: H04L-9/08(A) H04H-1/00(B) H04N-7/167(B) H04N-7/173(B)

Publication No. JP 2005253109 A (Update 200560 E)
Publication Date: 20050915
CONDITIONAL ACCESS SYSTEM
Assignee: SCIENTIFIC-ATLANTA INC (SCAT)
Inventor: AKINS GLENDON L III
PALGON MICHAEL S
PINDER HOWARD G
WASILEWSKI ANTHONY J
Language: JA (59 pages)
Application: JP 2000510276 A 19980731 (Division of application)
JP 2005120425 A 20050418 (Local application)
Priority: US 199754575 P 19970801
US 1998126921 A 19980731
Original IPC: H04L-9/08(A)
Current IPC: H04H-1/00(R,I,M,JP,20060101,20051220,A,L) H04H-1/00 (R,I,M,JP,20060101,20051220,C,L) H04L-9/08(R,I,M,JP,20060101,20051220,A,L) H04L-9/08 (R,I,M,JP,20060101,20060310,C,F) H04N-5/00(R,I,M,EP,20060101,20051008,A) H04N-5/00 (R,I,M,EP,20060101,20051008,C) H04N-7/16(R,I,M,EP,20060101,20051008,A) H04N-7/16 (R,I,M,EP,20060101,20051008,C) H04N-7/167(R,I,M,EP,20060101,20051008,A) H04N-7/167 (R,I,M,EP,20060101,20051008,C) H04N-7/173(R,I,M,JP,20060101,20051220,A,F) H04N-7/173 (R,I,M,JP,20060101,20051220,C,L)

WIPO

Publication No. WO 1999009743 A2 (Update 199915 B)
Publication Date: 19990225
CONDITIONAL ACCESS SYSTEM
RESEAU D'ACCES CONDITIONNEL
Assignee: SCIENTIFIC-ATLANTA, INC., Intellectual Property Dept., One Technology Parkway South, Norcross, GA 30092, US Residence: US Nationality: US (SCAT)
Inventor: AKINS, Glendon, L., III, 2510 Windward Lane N.E., Gainesville, GA 30501, US
PALGON, Michael, S., 1196 Poplar Grove Drive, Atlanta, GA 30306, US
PINDER, Howard, G., 4317 Stilson Circle, Norcross, GA 30092, US
WASILEWSKI, Anthony, J., 10680 Wren Ridge Road, Alpharetta, GA 30022, US
Agent: GARDNER, Kelly, A., Scientific-Atlantic, Inc., Intellectual Property Dept., One Technology Parkway South, Norcross, GA 30092, US
Language: EN (113 pages, 29 drawings)
Application: WO 1998US16079 A 19980731 (Local application)
Priority: US 199754575 P 19970801
US 1998126921 A 19980731
Designated States: (National Original) 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
(Regional Original) AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW NL OA PT SD SE SZ UG ZW
Original IPC: H04N-7/167(A)
Current IPC: H04H-1/00(R,I,M,JP,20060101,20051220,A,L) H04H-1/00

(R,I,M,JP,20060101,20051220,C,L) H04L-9/08(R,I,M,JP,20060101,20051220,A,L) H04L-9/08
(R,I,M,JP,20060101,20060310,C,F) H04N-5/00(R,I,M,EP,20060101,20051008,A) H04N-5/00
(R,I,M,EP,20060101,20051008,C) H04N-7/16(R,I,M,EP,20060101,20051008,A) H04N-7/16
(R,I,M,EP,20060101,20051008,C) H04N-7/167(R,I,M,EP,20060101,20051008,A) H04N-7/167
(R,I,M,EP,20060101,20051008,C) H04N-7/173(R,I,M,JP,20060101,20051220, A,F) H04N-7/173
(R,I,M,JP,20060101,20051220,C,L)
Original Abstract:
A cable television system provides conditional access to services. The cable television system includes a headend from which service "instances", or programs, are broadcast and a plurality of set top units for receiving the instances and selectively decrypting the instances for display to system subscribers. The service instances are encrypted using public and/or private keys provided by service providers or central authorization agents. Keys used by the set tops for selective decryption may also be public or private in nature, and such keys may be reassigned at different times to provide a cable television system in which piracy concerns are minimized.

Un reseau de television par cable assure un acces conditionnel a des services. Le reseau de television par cable comprend une tete de reseau a partir de laquelle on diffuse les "instances" de service ou programmes. Ce reseau comprend aussi une pluralite d'unites decodeurs concues pour recevoir les instances et dechiffrer selectivement les instances qui vont s'afficher pour les abonnes du reseau. Les instances de service sont chiffrees par des cles publiques et/ou privees fournies par des fournisseurs de service ou des agents d'autorisation centraux.Les cles utilisees par les decodeurs permettant un dechiffrement selectif peuvent aussi etre publiques ou privees et de telles cles peuvent etre reaffectees a differents moments pour assurer un reseau de television par cable dans lequel les risques de piratage sont minimises.

?

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 934 765 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
11.08.1999 Bulletin 1999/32

(51) Int. Cl.[6]: **A63F 9/22**

(21) Application number: **98400285.7**

(22) Date of filing: **09.02.1998**

(84) Designated Contracting States:
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(60) Divisional application:
98202314.5

(71) Applicant:
CANAL+ Société Anonyme
75711 Paris Cedex 15 (FR)

(72) Inventors:
- **Agasse, Bernard**
  95610 Eragny/Oise (FR)
- **Bayassi, Mulham**
  75015 Paris (FR)

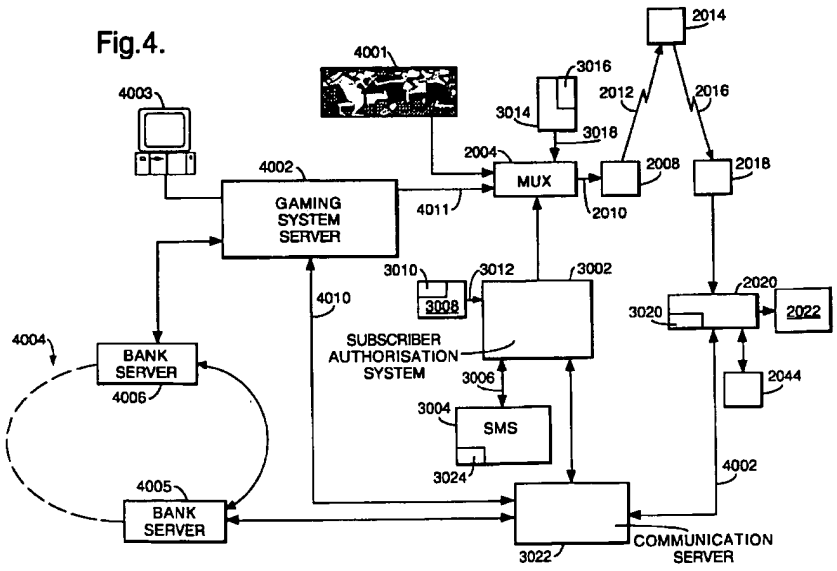
(74) Representative:
Cozens, Paul Dennis et al
Mathys & Squire
100 Grays Inn Road
London WC1X 8AL (GB)

Remarks:
The application is published incomplete as filed (Article 93 (2) EPC).

(54) Interactive gaming system

(57) An interactive gaming and audiovisual transmission system comprising a central gaming computer 4002 for processing gaming data, a decoder 2020 adapted to receive gaming data from the central gaming computer 4002 together with transmitted audiovisual data, the decoder further including a card reading device for interacting with a user's bank card in order to credit a gaming account held by the central gaming computer means in response to a transfer of credit from the user's bank account.

Fig.4.



EP 0 934 765 A1

Printed by Xerox (UK) Business Services
2.16.7/3.6

Description

[0001] The present invention relates to an interactive gaming and digital audiovisual transmission system, in particular a gaming and digital television transmission system.

[0002] Broadcast transmission of digital data is well-known in the field of pay TV systems, where scrambled audiovisual information is sent, usually by a satellite or satellite/cable link, to a number of subscribers, each possessing a decoder capable of descrambling the transmitted program for subsequent viewing. Terrestrial digital broadcast systems are also known. Recent systems have also used the broadcast link to transmit other data, in addition to or as well as audiovisual data, such as computer programs or interactive applications to the decoder or to a connected PC.

[0003] The increasing sophistication of such technology, in particular in relation to the receiver/decoder devices used in the systems, has led to an increase in the possible services that may be provided thereby. In particular, a number of systems have been proposed using interactive technology to enable a viewer to, for example, participate in a quiz show, or to select further information regarding a product currently being displayed on a shopping channel.

[0004] In the case of gaming applications, a number of largely theoretical systems have been proposed to enable a viewer to gamble a sum of money on the outcome of a sporting event or casino-type game broadcast over a television network. In most of these systems, a viewer is usually obliged to open an initial account with the controlling gaming authority by phoning or mailing a money transfer to the gaming authority before any gambling can be carried out. The disadvantages of this sort of procedure will be apparent.

[0005] Alternative systems are also known, in which the viewer buys credits to be gambled in the form of an electronic purse, i.e. a smart card or the like, the credits in the purse being available for subsequent gaming operations. The card is inserted in the decoder and the credits used thereafter in the subsequent gaming operations. When the contents of the purse are exhausted, the viewer buys a new card or re-charges the card at a suitable sales point. This system again implies a certain infra-structure to be put in place to enable a user to obtain the necessary credits to be gambled.

[0006] The present invention seeks to overcome some or all of the disadvantages of these prior art systems.

[0007] According to the present invention, there is provided an interactive gaming and audiovisual transmission system comprising a central gaming computer means for processing gaming data, a decoder adapted to receive gaming data from the central gaming computer together with transmitted audiovisual data, the decoder further including a card reading device for interacting with a user's bank card in order to credit a gaming account held by the central gaming computer means in response to a transfer of credit from the user's bank account.

[0008] In this way, the present invention enables a user to simply and quickly open and credit a gaming account from the comfort of his home, avoiding the more elaborate payment methods of the known systems.

[0009] The type of bank card used in this transaction may be of the debit or credit type. The card reading device may in particular comprise a smart card reader adapted to interact with a bank card in the form of a smart card.

[0010] Advantageously, the decoder is further equipped with a second card reading device. For example, in the case where the decoder forms part of a television subscription service, the subscriber may be provided with a subscription card in the form of a smart card or the like. The provision of two card reader devices in the decoder permits the decoder to carry out credit transactions on a bank card inserted in one reader whilst the subscription card is held in the second reader.

[0011] In one realisation, the decoder may be adapted to obtain transfer of credit information in the form of an electronic certificate generated by the bank card in response to transaction data submitted by the decoder. This transaction information may include, for example, the details of the bank account of the gaming authority to be credited in the operation, the sum of money to be transferred etc.

[0012] Typically, data is entered by the user into the decoder using a handheld remote control. In the case where a credit transaction is to be carried out, it may be necessary to enter the bank card PIN number using the remote control. In one embodiment, the decoder is provided with a handheld remote control, some or all of the data sent to the decoder being encrypted by the handheld remote control and subsequently decrypted by the decoder. In this way, interception by third parties of sensitive data emitted by the remote control may be avoided.

[0013] Preferably, the decoder is adapted to transmit transfer of credit information from the decoder to a bank server via a network communication link, for example, using an modem integrated in the decoder.

[0014] The decoder may be adapted to directly communicate transfer of credit information to a bank computer. However, preferably, the system further comprises an intermediate communications server, adapted to receive transfer of credit information communicated from the decoder and to forward this information on to a bank server.

[0015] The intermediate communications server may further be adapted to communicate with the central gaming computer means, for example, to inform the central communication means of a transfer of credit instruction being forwarded from the intermediate communication means to a bank computer, so as to permit

the gaming computer means to set up an account without having to verify the transaction carried out at an associated bank server.

[0016] The central gaming computer means may equally be adapted to receive and transmit credit information to or from a bank server via a network communication link. This may be necessary, for example, in the case of a win or in order to verify the transfer of funds from the bank account of a user to the gaming authorities bank account before opening a gaming account.

[0017] Preferably, the decoder is adapted to communicate gaming information to the central gaming computer during gaming operation via a network communication link. This may be the same link as used to communicate transfer of credit information to a bank computer, for example, using a modem device integrated in the decoder.

[0018] Some or all of the gaming information communicated from the decoder to the central gaming computer during gaming operation may be encrypted by the decoder. For example, the decoder may be adapted to transmit in encrypted form a code word entered by the user associated with the gaming account of the user held by the central gaming computer.

[0019] The decoder may be adapted to directly communicate information to the central gaming computer during gaming operation. However, preferably, the system further comprises an intermediate communications server, adapted to receive information communicated from the decoder during gaming operation and to forward this information on to the central gaming computer. This may be the same intermediate server as used for the transfer of credit information between the decoder and a bank.

[0020] In the case where gaming information is encrypted by the decoder, the intermediate communications server may be adapted to simply pass this information "as is" to the central gaming computer. However, in one embodiment, the intermediate communications server is adapted to decrypt information received from the decoder and to re-crypt this information for subsequent communication to the central gaming computer. This may be required, for example, in the case where different encryption algorithms are used by the decoder and central gaming computer.

[0021] The intermediate communications server may further be adapted to communicate information to and from other computer devices, for example, computer databases holding TV subscriber information. In this way, the intermediate communications server may obtain directly information regarding the user of the system (name, address etc) to be used in setting up a gaming account, without the user having to re-enter the same information.

[0022] The communication means used to transmit gaming data from the central gaming computer to the decoder may be defined in a number of different ways and by a number of different communication elements. For example, some or all of the gaming data sent from the gaming computer to the decoder may be transmitted via a transmitter means used to transmit audiovisual data to the decoder.

[0023] In addition, or alternatively, some or all of the gaming data sent from the central gaming computer to the decoder may be sent via a network communication link, for example, the same network used to communicate information from the decoder to the central gaming computer during gaming operation.

[0024] In practice, a mixture of these two communication paths may prove optimal, the network path being used for rapid dialogue between the decoder and the gaming computer during real-time operation and the transmission path being used for relatively fixed data, such as screen format display data or the like.

[0025] The present invention also extends to a gaming system for processing gaming data, comprising:

means for transmitting gaming data to a user's decoder;
means for receiving data from the user's decoder; and
means for connection to a bank server holding the user's bank account in order to transfer credit to or from the account.

[0026] The gaming system may include a gaming account held by the gaming system which can be credited in response to the transfer of credit.

[0027] The gaming system may be adapted to communicate with the decoder and the bank server via a communications server. If so, the gaming system may be adapted to receive encrypted information from the communications server.

[0028] The present invention also provides a interactive gaming and audiovisual transmission system comprising a gaming system as aforementioned, said user's decoder, and said bank server.

[0029] As mentioned above the system may be used to permit gaming in relation to various events. For example, the central gaming computer may be adapted to generate a computer game (computer blackjack or the like), the computer generated images being transmitted via the audiovisual link to the decoder.

[0030] However, as will be appreciated, the combination of gaming and audiovisual systems makes the present invention particularly adapted to permit gaming in relation to televised sports, such as horse racing or the like. In one embodiment, the present invention comprises a central gaming computer adapted to provide gaming data related to a real-time sporting event, the decoder being adapted to receive both gaming data and associated audiovisual data of the event.

[0031] In the context of the present application the term ((audiovisual transmission system)) refers to all transmission systems for transmitting or broadcasting primarily audiovisual or multimedia digital data. The

present invention is particularly, but not exclusively, applicable to a broadcast digital television system.

[0032] In this application the term ((smart card)) is used to mean any conventional chip-based card device possessing, for example, microprocessor and/or memory storage. Also included in this term are chip devices having alternative physical forms, for example key-shaped devices such as are often used in TV decoder systems.

[0033] In the present application, the term "decoder" is used to apply to an integrated receiver/decoder for receiving and decrypting an encrypted transmission, the receiver and decoder elements of such a system as considered separately, as well as to a receiver capable of receiving non-encrypted broadcasts. The term equally covers decoders including additional functions, such as web browsers, together with decoder systems integrated with other devices, for example, integrated VHS/decoder devices or the like.

Figure 1 shows the overall architecture of a digital television system, as may be incorporated in the gaming system of the present invention;

Figure 2 shows the conditional access system of the television system of Figure 1;

Figure 3 shows the structure of the decoder of Figures 1 and 2;

Figure 4 shows a gaming system incorporating the television system of Figures 1 and 2; and

Figure 5 shows a flow diagram of the logical steps involved in a gaming transaction

Digital Television System

[0034] An overview of a digital television broadcast and reception system 1000 adaptable to the present invention is shown in Figure 1. The system includes a mostly conventional digital television system '2000, which uses the known MPEG-2 compression system to transmit compressed digital signals. In more detail, the MPEG-2 compressor 2002 in a broadcast centre receives a digital signal stream (typically a stream of video signals). The compressor 2002 is connected to a multiplexer and scrambler 2004 by linkage 2006. The multiplexer 2004 receives a plurality of further input signals, assembles one or more transport streams and transmits compressed digital signals to a transmitter 2008 of the broadcast centre via linkage 2010, which can of course take a wide variety of forms including telecom links.

[0035] The transmitter 2008 transmits electromagnetic signals via uplink 2012 towards a satellite transponder 2014, where they are electronically processed and broadcast via notional downlink 2016 to earth receiver 2018, conventionally in the form of a dish owned or rented by the end user. The signals received by receiver 2018 are transmitted to an integrated receiver/decoder 2020 owned or rented by the end user and connected to the end user's television 2022. The receiver/decoder 2020 decodes the compressed MPEG-2 signal into a television signal for the television set 2022.

[0036] A conditional access system 3000 is connected to the multiplexer 2004 and the receiver/decoder 2020, and is located partly in the broadcast centre and partly in the decoder. It enables the end user to access digital television broadcasts from one or more broadcast suppliers. A smart card, capable of decrypting messages relating to commercial offers (that is, on or several television programmes sold by the broadcast supplier), can be inserted into the receiver/decoder 2020. Using the decoder 2020 and smart card, the end user may purchase events in either a subscription mode or a pay-per-view-mode.

[0037] An interactive system 4000, also connected to the multiplexer 2004 and the receiver/decoder 2020 and again located partly in the broadcast and partly in the decoder, enables the end user to interact with various applications via a modemmed back channel 4002. Such interactive applications may include an interactive shopping service, a quiz application, an interactive programme guide etc.

[0038] In point of fact, whilst the interactive system 4000 has been represented as a discrete logical block, the physical elements of this system, such as the server or servers used to handle communications between the receiver/decoder and central servers, may be elements shared with the conditional access system 3000. This will become clear in the description of the gaming system of Figure 4.

Conditional Access System

[0039] With reference to Figure 2, the conditional access system 3000 includes a Subscriber Authorization System (SAS) 3002. The SAS 3002 is connected to one or more Subscriber Management Systems (SMS) 3004, one SMS for each broadcast supplier, by a respective TCP-IP link 3006 (although other types of linkage could alternatively be used). Alternatively, one SMS could be shared between two broadcast suppliers, or one supplier could use two SMSs, and so on.

[0040] First encrypting units in the form of ciphering units 3008 utilising ((mother)) smart cards 3010 are connected to the SAS by linkage 3012. Second encrypting units again in the form of ciphering units 3014 utilising mother smart cards 3016 are connected to the multiplexer 2004 by linkage 3018. The receiver/decoder 2020 receives a ((daughter)) smart card 3020. It is connected directly to the SAS 3002 by Communications Servers 3022 via the modemmed back channel 4002. The SAS sends amongst other things subscription

rights to the daughter smart card on request.

[0041] The smart cards contain the secrets of one or more commercial operators. The ((mother)) smart card encrypts different kinds of messages and the ((daughter)) smart cards decrypt the messages, if they have the rights to do so.

[0042] The first and second ciphering units 3008 and 3014 comprise a rack, an electronic VME card with software stored on an EEPROM, up to 20 electronic cards and one smart card 3010 and 3016 respectively, for each electronic card, one (card 3016) for encrypting the ECMs and one (card 3010) for encrypting the EMMS.

[0043] Also shown in Figure 2 is a handheld remote control used by the viewer to control and program functions of the receiver/decoder 2020.

Multiplexer and Scrambler

[0044] With reference to Figures 1 and 2, in the broadcast centre, the digital video signal is first compressed (or bit rate reduced), using the MPEG-2 compressor 2002. This compressed signal is then transmitted to the multiplexer and scrambler 2004 via the linkage 2006 in order to be multiplexed with other data, such as other compressed data.

[0045] The scrambler generates a control word CW used in the scrambling process and included in the MPEG-2 stream in the multiplexer 2004. The control word CW is generated internally and enables the end user's integrated receiver/decoder 2020 to descramble the programme. Access criteria, indicating how the programme is commercialised, are also added to the MPEG-2 stream. The programme may be commercialised in either one of a number of ((subscription)) modes and/or one of a number of ((Pay Per View)) (PPV) modes or events.

[0046] In the subscription mode, the end user subscribes to one or more commercial offers, of ((bouquets)), thus getting the rights to watch every channel inside those bouquets. In the preferred embodiment, up to 960 commercial offers may be selected from a bouquet of channels. In the Pay Per View mode, the end user is provided with the capability to purchase events as he wishes. This can be achieved by either pre-booking the event in advance (((pre-book mode))), or by purchasing the event as soon as it is broadcast (((impulse mode))).

[0047] Both the control word CW and the access criteria are used to build an Entitlement Control Message (ECM); this is a message sent in relation with a scrambled program. The message contains a control word (which allows for the descrambling of the program) and the access criteria of the broadcast program. The access criteria and control word are transmitted to the second encrypting unit 3014 via the linkage 3018. In this unit an ECM is generated, encrypted with an exploitation key Cex and transmitted on to the multiplexer and scrambler 2004.

Programme Transmission

[0048] The multiplexer 2004 receives encrypted EMMs from the SAS 3002, encrypted ECMs from the second encrypting unit 3014 and compressed programmes from the compressor 2002. The multiplexer 2004 scrambles the programmes and communicates the scrambled programmes, the encrypted EMM (if present) and the encrypted ECMs to a transmitter 2008 of the broadcast centre via linkage 2010. The transmitter 2008 transmits electromagnetic signals towards the satellite transponder 2014 via uplink 2012.

Programme Reception

[0049] The satellite transponder 2014 receives and processes the electromagnetic signals transmitted by the transmitter 2008 and transmits the signals on to the earth receiver 2018, conventionally in the form of a dish owned or rented by the end user, via downlink 2016. The signals received by receiver 2018 are transmitted to the integrated receiver/decoder 2020 owned or rented by the end user and connected to the end user's television set 2022. The receiver/decoder 2020 demultiplexes the signals to obtain scrambled programmes with encrypted EMMs and encrypted ECMs.

[0050] If the programme is not scrambled the receiver/decoder 2020 decompresses the data and transforms the signal into a video signal for transmission to television set 2022.

[0051] If the programme is scrambled, the receiver/decoder 2020 extracts the corresponding ECM from the MPEG-2 stream and passes the ECM to the ((daughter)) smart card 3020 of the end user. This slots into a housing in the receiver/decoder 2020. The daughter smart card 3020 controls whether the end user has the right to decrypt the ECM and to access the programme. If not, a negative status is passed to the receiver/decoder 2020 to indicate that the programme cannot be descrambled. If the end user does have the rights, the ECM is decrypted and the control word extracted. The decoder 2020 can then descramble the programme using this control word. The MPEG-2 stream is decompressed and translated into a video signal onward transmission to television set 2022.

Subscriber Management System (SMS)

[0052] A Subscriber Management System (SMS) 3004 includes a database 3024 which manages, amongst others, all of the end user files, commercial offers (such as tariffs and promotions), subscriptions, PPV details, and data regarding end user consumption and authorization. The SMS may be physically remote from the SAS

[0053] Each SMS 3004 transmits messages to the SAS 3002 via respective linkage 3006 to enable modifications to or creations of Entitlement Management Mes-

sages (EMMs) to be transmitted to end users.

[0054] The SMS 3004 also transmits messages to the SAS 3002 which imply no modifications or creations of EMMs but imply only a change in an end user's state (relating to the authorization granted to the end user when ordering products or to the amount that the end user will be charged).

Entitlement Management Messages and Entitlement Control Messages

[0055] ECMs or Entitlement Control Messages are encrypted messages embedded in the data stream of a transmitted program and which contain the control word necessary for descrambling of part or all of a program. Authorisation of a given receiver/decoder is controlled by EMMs or Entitlement Management Messages, transmitted on a less frequent basis and which supply an authorised receiver/decoder with the exploitation key necessary to decode the ECM.

[0056] An EMM is a message dedicated to an individual end user (subscriber), or a group of end users. A group may contain a given number of end users. This organisation as a group aims at optimising the bandwidth; that is, access to one group can permit the reaching of a great number of end users.

[0057] Various specific types of EMM may be used. Individual EMMs are dedicated to individual subscribers, and are typically used in the provision of Pay Per View services. So-called ((Group)) subscription EMMs are dedicated to groups, of say, 256 individual users, and are typically used in the administration of some subscription services. This EMM has a group identifier and a subscribers' group bitmap

[0058] For security reasons, the control word CW embedded in an encrypted ECM changes on average every 10 seconds or so. In contrast, the exploitation key Cex used by the receiver to decode the ECM is changed every month or so by means of an EMM. The exploitation key Cex is encrypted using a personalised key corresponding to the identity of the subscriber or group of subscribers recorded on the smart card. If the subscriber is one of those chosen to receive an updated exploitation key Cex, the card will decrypt the message using its personalised key to obtain that month's exploitation key Cex.

[0059] The operation of EMMs and ECMs will be well-known to one skilled in the art and will not be described here in any more detail.

Receiver/Decoder Structure

[0060] Referring to Figure 3, the elements of a receiver/decoder 2020 or set-top box for use in a digital broadcast system and adapted to be used in the present invention will now be described. As will be understood, the elements of this decoder are largely conventional and their implementation will be within the capabilities of one skilled in the art.

[0061] As shown, the decoder 2020 is equipped with several interfaces for receiving and transmitting data, in particular an MPEG tuner and demultiplexer 2040 for receiving broadcast MPEG transmissions, a serial interface 2041, a parallel interface 2042, and a modem 2028 for sending and receiving data via the telephone network. In this embodiment, the decoder also includes a first and second smart card reader 2030 and 2031, the first reader 2030 for accepting a subscription smart card containing decryption keys associated with the system and the second reader 2031 for accepting bank and other cards. As will be described, the use of a two-slot decoder, adapted to read bank cards, is an important aspect in the implementation of the gaming system of Figure 4.

[0062] The decoder also includes a receiver 2043 for receiving infra-red control signals from the handset remote control 2044 and a Peritel output for sending audiovisual signals to a television 2022 connected to the decoder. In certain cases it may be desired that the infra-red signals transmitted from the handset 2044 to receiver 2043 are subject to a simple scrambling/descrambling process to ensure that no useful information may be obtained by any third party monitoring the transmission.

[0063] Such algorithms will not be described in any detail, but may comprise, for example a symmetric algorithmic key known to both handset 2044 and receiver/decoder 2020. This may be varied from time to time, for example, by means of a modulating random number chosen by the receiver/decoder 2020 and displayed by the television 2022, the user then programming the handset 2044 with this number to ensure that the handset scrambles entered data using an encryption algorithm key equivalent to that used the receiver/decoder to decrypt the received infra-red signals.

[0064] Processing of digital signals received via the interfaces and generation of digital output signals is handled by a central control unit 2045. The software architecture of the control unit within the decoder may correspond to that used in a known decoder and will not be described here in any detail. It may be based, for example, on a virtual machine interacting via an interface layer with a lower level operating system implemented in the hardware components of the decoder. In terms of the hardware architecture, the decoder will be equipped with a processor, memory elements such as ROM, RAM, FLASH memory etc. as in known decoders.

[0065] Applications processed by the control unit 2045 may be resident applications stored in the ROM or FLASH of the decoder or applications broadcast and downloaded via the MPEG interface 2 of the decoder. Applications can include program guide applications, games, interactive services, teleshopping applications, as well as initiating applications to enable the decoder

to be immediately operational upon start-up and applications for configuring the decoder. Applications are stored in memory locations in the decoder and represented as resource files comprising graphic object description files, unit files, variables block files, instruction sequence files, application files, data files etc.

[0066] Conventionally, applications downloaded into the decoder via the broadcast link are divided into modules, each module corresponding to one or more MPEG tables. Each MPEG table may be divided into a number of sections. For data transfer via the serial and parallel ports, modules are also split into tables and sections, the size of the section depending on the channel used.

[0067] In the case of broadcast transmission, modules are transported in the form of data packets within respective types of data stream, for example, the video data stream, the audio data stream, a text data stream. In accordance with MPEG standards each packet is preceded by a Packet Identifier (PID) of 13 bits, one PID for every packet transported in the MPEG stream. A programme map table (PMT) contains a list of the different streams and defines the content of each stream according to the respective PID. A PID may alert the device to the presence of applications in the data stream, the PID being identified by the PMT table.

Gaming System Architecture

[0068] Referring now to Figure 4, there will now be described the elements and functioning of a gaming system according to an embodiment of the present invention. The gaming system includes the elements of the digital television system described and shown in Figures 1 and 2, which have been assigned the same reference numerals . Some elements, such as the digital compressor 2002 shown in Figure 1, have been omitted in order to focus on those aspects of the system which are pertinent to the present invention.

[0069] As shown, the gaming system additionally comprises a source of audiovisual information 4001 regarding the event which will form the subject of betting etc within the system. In the present case, the event has been represented as a horse race, and the present system is indeed particular adapted to gaming activities centred around televised live action sporting events. However, as will be understood, the present system may equally used to permit gambling in relation to other events, such as casino-type games, as well as computer generated games, pre-recorded events etc.

[0070] The system further comprises a central gaming computer means in the form of a gaming system server 4002, together with associated operating terminal or terminals 4003, adapted to generate odds, calculate winnings etc in relation to the gaming event. The gaming server 4002 is adapted to communicate with a receiver/decoder 2020 via the intermediate communication server or servers 3022. The connection between the gaming server 4002 and communication server 3022 may be implemented by an X25 Transpac link or via a dedicated line. The network link for the server is indicated broadly at 4010.

[0071] As described above, the communication server 3022 communicates with the receiver/decoder 2020 by means of a telephone link using the in-built modem of the receiver/decoder.

[0072] The gaming server may be equally adapted to send information to the receiver/decoder 2020 via a satellite link, indicated broadly at 4011, by injection of information into the multiplexer 2004 for subsequent integration in the transmitted MPEG stream.

[0073] As will be understood, all communications from the receiver/decoder 2020 to the gaming server 4002 are via the receiver/decoder modem and communication server 3022. In the case of communications from the gaming server 4002 to the receiver/decoder 2020, the choice of communication channel and communication means (MPEG satellite transmission or communication server/modem connection) may depend on the nature of the information to be transmitted.

[0074] Typically, the satellite link 4011 will be used to send data or information that may be updated on a daily basis or which may be received by any number of receiver/decoders in the park (odds for tomorrow's races etc). In particular, the satellite link may be used to download the application that needs to be installed in the receiver/decoder to enable the receiver/decoder to function in the gaming system.

[0075] In contrast, the modem link 4010 may be preferred for data that changes on a minute-by-minute basis or that is specific to a particular user (results of last race, current state of the account of the user etc).

[0076] In addition to handling gaming activities resulting from bets placed via the receiver/decoder 2020, for example as programmed in using the remote control 2044, the gaming server 4002 may also be adapted to manage bets to be placed by other input means, for example as placed by a phone service or as received by a "Minitel" type system, as used in France and other countries.

[0077] The gaming system server 4002 is additionally connected to a bank server network 4003 comprising one or more bank servers 4005, 4006. The bank server network may correspond to an existing network used to handle electronic payment transactions. The level of security and encryption in the communications between each of the elements of the gaming system will be described in more detail below in relation to the operation of the system.

Gaming System Operation

[0078] As mentioned in the introduction of the present application, gaming systems used in interactive television systems proposed to date have tended to use relatively laborious methods for settling accounts between the viewer and the central gaming authority, requiring

the viewer either to pay by a conventional method (cheque, telephone credit transfer etc) or to physically purchase an "electronic purse" in the form of a smart card or key containing a number of pre-paid credits that may be gambled.

[0079] The present embodiment differs from such systems in proposing a system architecture that enables a viewer to pay by means of a credit or debit card inserted in the decoder and by entering data into the system by means of the hand-held remote control. As mentioned above, the provision of a decoder provided with two distinct card readers 2030, 2031 enables the decoder to simultaneously hold a subscription card containing the viewers access rights (eg to the gaming channel) as well as interacting with a credit/debit card inserted in the decoder.

[0080] In order to comply with regulations concerning the use of credit/debit cards in gambling transactions, two different types of transactions need to be distinguished: (i) opening or re-crediting an account managed by the gaming system server and (ii) gambling the sums in this account.

Opening an account

[0081] In the present case, the card reader 2031 functions in a similar manner to a standard card reader used in banking terminals and the like to read and write data on a smart card presented in the reader. As with all card readers used in the banking field, communication between the terminal (in this case the decoder) and external servers is prohibited during the time that the card is being accessed by the terminal, i.e. for the time that the memory zones on the card are "open".

[0082] In order to open and credit an account with the gaming system server, the following steps are carried out during a first phase:

a) Using the handheld remote control, and as guided by the application loaded in the receiver/decoder, the user selects the option "open an account" and enters the sum of money that he wishes to transfer to this account.

b) After having introduced his credit card into the card reader slot 2031, the viewer is invited to enter his personal PIN code. The user has a maximum of two opportunities to enter the code, after which the receiver/decoder will refuse to accept any further entries and the transaction will be abandoned.

Note that in the case of sensitive information communicated to the receiver/decoder by the handset (in particular the PIN code) the data entered by the user on the key pad of the handset may be scrambled before transmission between the handset and decoder so as to prevent interception of this information by any third party. See above.

c) Assuming the code is correct, the smart card downloads certain information in response to a request from the receiver/decoder, including details of the last transactions, to enable the decoder to verify that the sum of transactions during a certain period is within, for example, the transaction limit of the card holder for that period.

d) The receiver/decoder then passes to the smart card information regarding the current transaction including the amount of the transaction, the date and time of the transaction, the details of the bank account to be credited in the transaction and so on. (The details of the account to be credited can be obtained by the decoder prior to the interrogation of the card from the gaming system server or the intermediate communications system server).

e) In the conventional manner, the smart card then calculates a first numeric certificate using this information, which is communicated to the receiver/decoder. The receiver/decoder writes the present transaction in the card and a second numeric certificate is calculated and communicated to the receiver/decoder. The memory zones of the smart card are then closed off.

The generation of a pair of numeric certificates is a specific security measure associated with the use of a receiver/decoder as transaction terminal.

Once the above steps have been carried out, the system then moves to a second phase involving communication between the receiver/decoder 2020, the intermediate communication server 3022 and the bank server 4005.

f) Before transferring any information, the receiver/decoder 2020 verifies the identity of the communication server 3022 by means of a public/private key system (eg using the RSA algorithm). In particular, the receiver/decoder generates a random number, which is transmitted to the server for encryption by a private key and returned to the receiver/decoder, which checks the encrypted value using the equivalent public key.

A simple handshake signal may also be provided by the decoder 2020 to identify itself to the server 3022.

g) Assuming the identity of the communication server is verified, the receiver/decoder 2020 sends to the communication server 3022 the details of the transaction to be carried out, including the first and second numeric certificate generated by the smart card.

h) The communication server 3022 then sends the transaction details to the first bank server 4005, which verifies the account of the user, and author-

ises (or not) the transaction and sends an acknowledgement of the transaction to the communication server. The transfer of money between the user's account and that of the central gaming authority will then be handled within the bank network 4004.

i) Once the communication server 3022 has received acknowledgement of the acceptation of the monetary transfer, a message will be sent to the receiver/decoder 2020 of the completion of the transfer and the operation will proceed to the next phase.

Note that the same steps a) to i) as used in the first two phases will also be carried out in the event that the user wishes to increase the credit in an existing gaming account.

The next phase in the opening of a gaming account involves communication between the receiver/decoder 2020, the communication server 3022 (and the SAS and SMS servers 3002, 3004) and the gaming server 4002. The information communicated between these servers is largely non-sensitive and may be communicated in clear, with the exception of the code word chosen by the user to obtain access to his gaming account.

j) Using the information (name, address etc) on the user held in the SAS and SMS servers 3002, 3004, the communication server prepares a request for opening of an account with the gaming system server 4002. This information has been gathered in the SMS server during the original procedure carried out when the user originally subscribed to the television service. The user is thus spared the inconvenience of repeating all this information when subscribing to the gaming service.

Note that in the event that SMS database reveals, for example, that the subscriber is in debt with the television service, the communication server may abort the opening of an account with the gaming service. This extra verification step may be carried out earlier, for example, at step g).

k) In one embodiment, the communication server 3022 may send the subscriber information to the receiver/decoder 2020 where it is displayed on the television 2022 for verification by the user. Once verified, the information is sent to the gaming system server 4002 where a gambling account is created by the server 4002.

l) The account information (account number etc) is then sent from the gaming server 4002, via the communication server 3022, to the receiver/decoder 2022. The user is then invited to choose a suitable code word for the account which will be demanded by the system at every opening of a gaming session. As for the PIN number, the infrared signal containing this information and sent between the remote control and the decoder may be scrambled by the remote to avoid interception and descrambled by decoder.

m) The code word is then encrypted by a public key of a public/private key pair held in the receiver/decoder 2020 and sent to the communication server 3022, where it is decrypted by the corresponding private key. In this case, for example, the same RSA key pair as used for the verification of the communication server may be used.

n) The code word is then re-encrypted by the communication server 3022 and sent to the gaming system server 4002 where it is decrypted and assigned to the user's account. In this case, a symmetric key algorithm, such as DES, may be advantageously used, for example, to permit two-way encrypted communication between the communication server 3022 and gaming server 4002.

Gambling with an existing gaming account

[0083] Once the user has set up and credited a gaming account with the gaming server 4002, all future gambling transactions will be handled between the receiver/decoder 2020 and the gaming system server 4002. At the start of every gaming session, the system server 4002 will demand the user's assigned code word, which will be communicated between the receiver/decoder and the gaming server, via the communications server, as described above.

[0084] For simplicity, and in order to permit a relatively rapid dialogue, all questions and responses between the user and the gaming system in order to place a bet and receive the results are preferably passed via the telephone/modem link and the communication server 3022. Certain data, such as the format of the screens displayed by the receiver/decoder in gaming mode and/or slowly changing or universal data (details of that day's races, the horses taking part etc) may be passed via the satellite uplink in order to take advantage of the bandwidth of this channel.

[0085] Other embodiments, in which data is shared between the two communication channels in alternative ways may nevertheless be envisaged, for example, where all communication from the receiver/decoder to the gaming system server passes via the modem link, whilst all communications from the server to the receiver/decoder pass via the satellite link.

[0086] As mentioned above, the present system may be used with a number of interactive gaming applications, for example, with computer games such as blackjack, poker or the like, in which the user places a bet on the outcome of a game managed by the gaming server. However, in view of the use of television broadcast technology, the system is particularly adapted to permit

gaming in relation to live action sporting events, such as televised horse, dog or camel racing.

[0087] Figure 5 is a flow diagram of the steps involved in the placing of a bet in relation to one or more broadcast horse races. In the present case, the bet is to be placed in respect of the present day's races, i.e. in "real time", and the odds quoted for the horses may depend on the time at which the bet is taken. In alternative embodiments, bets may be placed the day or week before the race or races in question.

[0088] Firstly, at step 5000, the user enters his code word and opens a betting session. At steps 5001 and 5002, he chooses the racecourse he is interested in and one of the races running at that racecourse, respectively. Depending on which race is running, the user may be offered a number of different standard types of bet, from a simple bet to more complex bets, including main and side bets.

[0089] As will be appreciated, the bet types offered may be determined according to the wishes of the gaming authority and may be based on any of the usual types of bet offered for an event of this type.

[0090] At step 5003, the user chooses the type of bet he wishes to place. In the case of a simple bet on one horse, the next step will be step 5004 where the user chooses the formula of the bet, ie whether the horse will win or be placed in the first three or four positions. At step 5005, the user chooses the horse he wishes to bet on.

[0091] In the case of a complex bet, the user then chooses from a combination of win, place or win/place at step 5007 and from one of a number of types of bet (single, combined, reduced field, full field) at step 5007. The user may decide, for example to choose one horse to win and/or one horse to be placed in the top three or four. Other combinations may be made presented to reflect the choice of bet normally available. At step 5008 the user chooses the horses he wishes to bet on.

[0092] At step 5009 the user chooses his stake, i.e. the sum to be extracted from the money deposited in his gaming account. At step 5010 confirmation of the stake to be gambled is demanded. At this time, the system may also indicate the overall odds for the bet or bets placed and the sum of money to be won. Assuming that the user confirms the bet, the bet is registered at step 5011.

[0093] Following the results of the race, the gaming system server calculates the winnings or losses for the user. These will be subtracted or added automatically to his gaming account. The user may demand at any time the position of his account.

[0094] In the event that the user eventually wishes to close the account or to transfer some of his winnings to his bank account, a message to this end may be sent by the user from the receiver/decoder 2020 to the gaming system server 4002 (Figure 4). At that time, the server 4002 will communicate with the bank server 4006 to organise a credit transfer to the user's bank account. Since the identity and bank details of the owner of the receiver/decoder are already known, the server will only transfer money from the gaming account of the user to the bank account originally used in the setting up of the gaming account.

[0095] It will be understood that the present invention has been described above purely by way of example, and modifications of detail can be made within the scope of the invention.

[0096] Each feature disclosed in the description, and (where appropriate) the claims and drawings may be provided independently or in any appropriate combination.

[0097] In the aforementioned preferred embodiments, certain features of the present invention have been implemented using computer software. However, it will of course be clear to the skilled man that any of these features may be implemented using hardware. Furthermore, it will be readily understood that the functions performed by the hardware, the computer software, and such like are performed on or using electrical and like signals.

Claims

1. An interactive gaming and audiovisual transmission system comprising a central gaming computer means for processing gaming data, a decoder adapted to receive gaming data from the central gaming computer together with transmitted audiovisual data, the decoder further including a card reading device for interacting with a user's bank card in order to credit a gaming account held by the central gaming computer means in response to a transfer of credit from the user's bank account.

2. An interactive gaming and audiovisual transmission system as claimed in claim 1, in which the decoder is equipped with a card reading device in the form of a smart card reader.

3. An interactive gaming and audiovisual transmission system as claimed in claim 1 or 2, in which the decoder is further equipped with a second card reading device

4. An interactive gaming and audiovisual transmission system as claimed in any preceding claim in which the decoder is adapted to obtain transfer of credit information in the form of an electronic certificate generated by the bank card in response to transaction data submitted by the decoder.

5. An interactive gaming and audiovisual transmission system as claimed in any preceding claim in which the decoder is provided with a handheld remote control, some or all of the data sent to the decoder being encrypted by the handheld remote

control and subsequently decrypted by the decoder.

6. An interactive gaming and audiovisual transmission system as claimed in any preceding claim in which the decoder is adapted to transmit transfer of credit information from the decoder to a bank server via a network communication link.

21. A gaming system as claimed in Claim 19 or 20, adapted to communicate with the decoder and the bank server via a communications server.

22. A gaming system as claimed in Claim 21, adapted to receive encrypted information from the communications server.

23. A gaming system as claimed in any of Claims 19 to 22, adapted to transmit gaming data related to a real-time sporting event.

24. An interactive gaming and audiovisual transmission system comprising a gaming system as claimed in any of Claims 19 to 23, said user's decoder, and said bank server.

Fig.1.

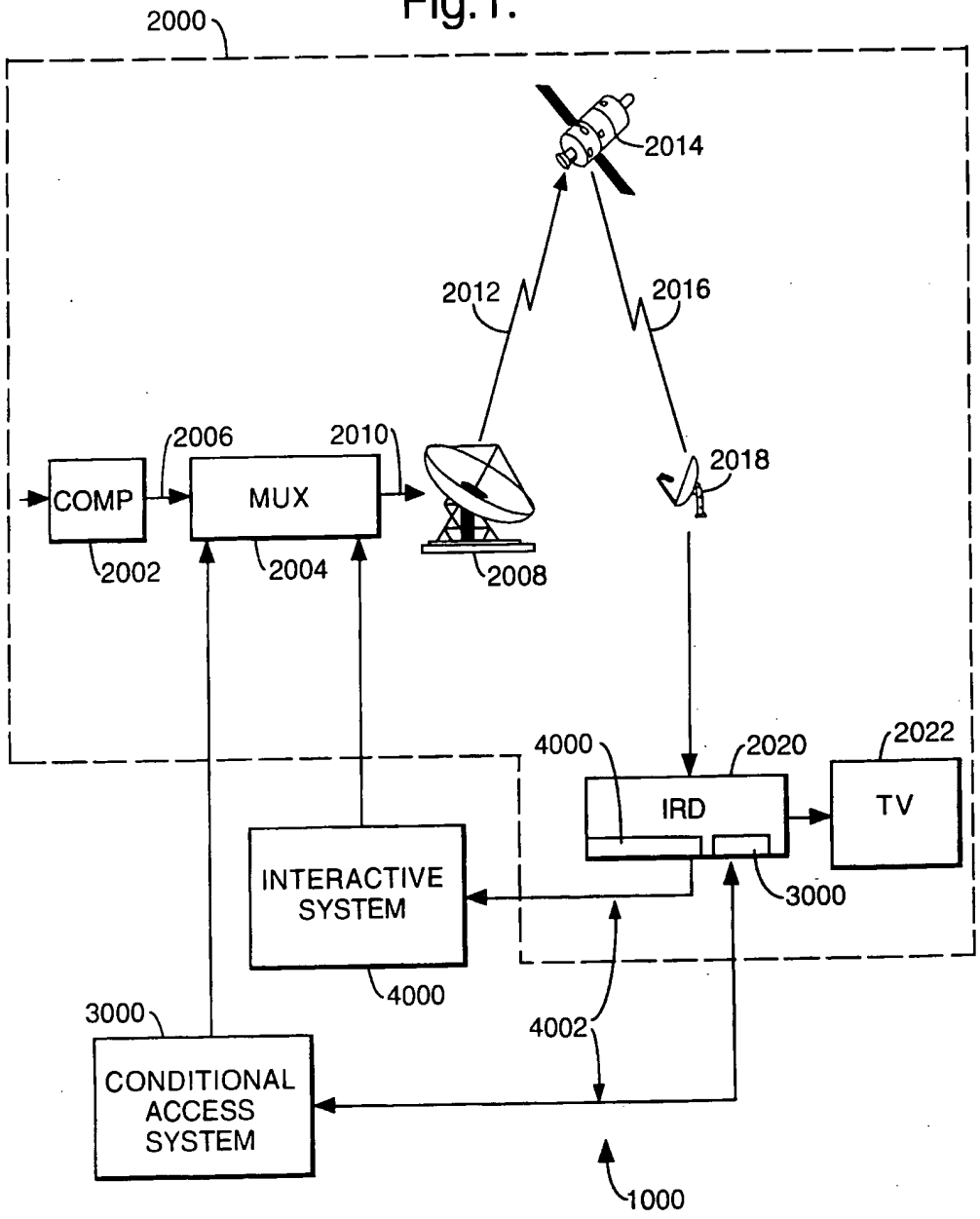


Fig.2.

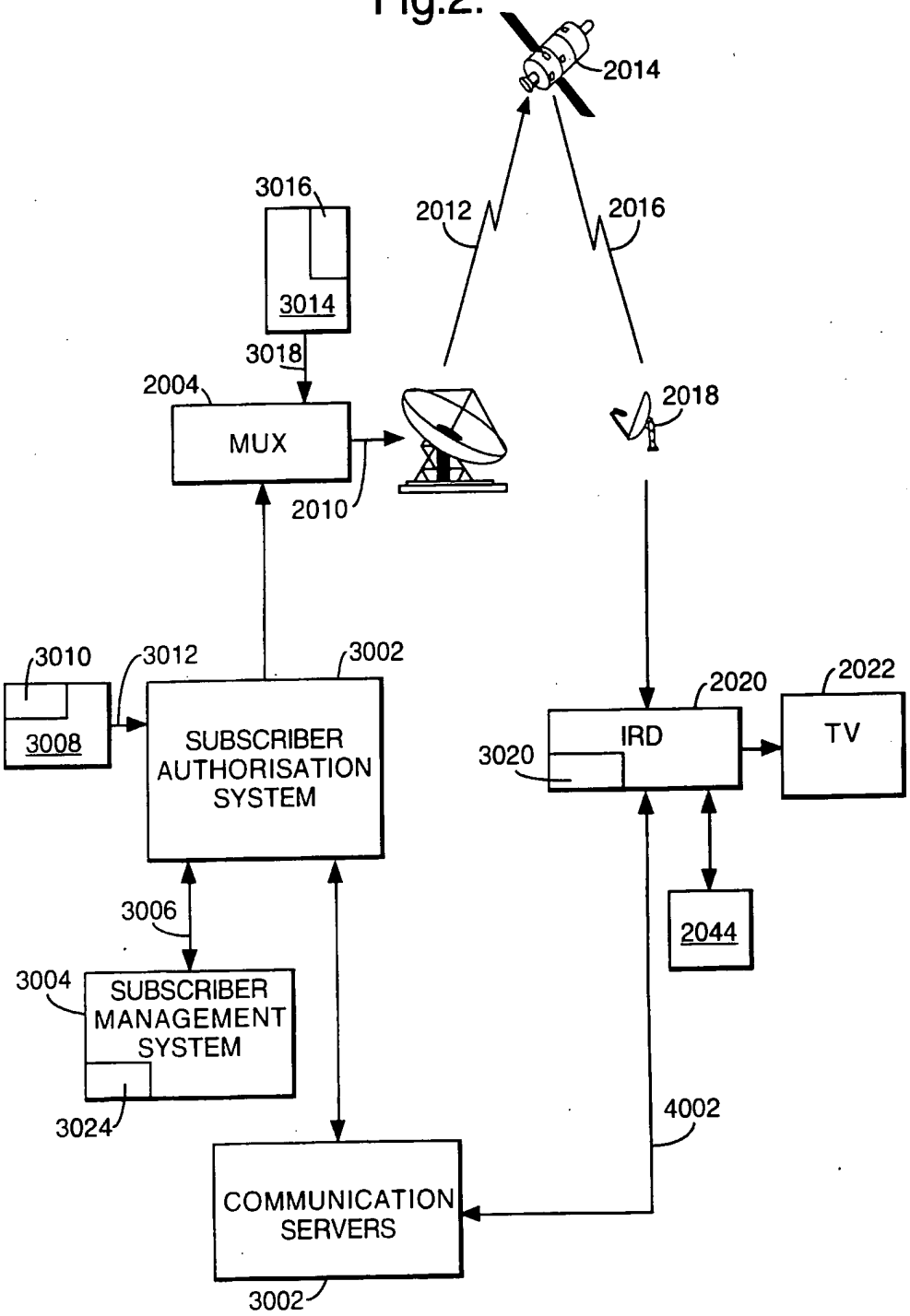
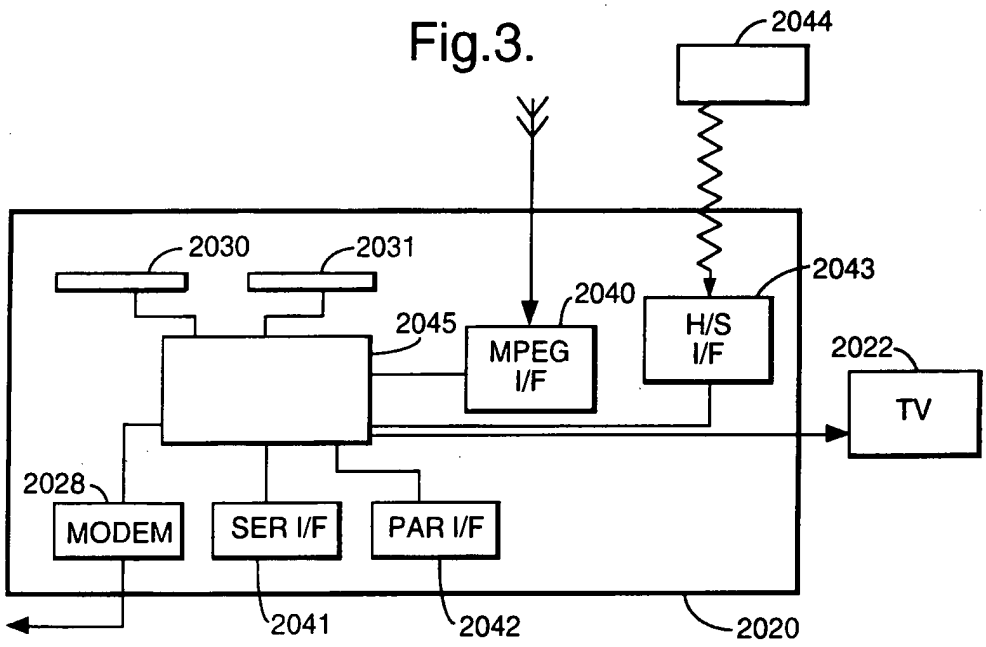


Fig.3.



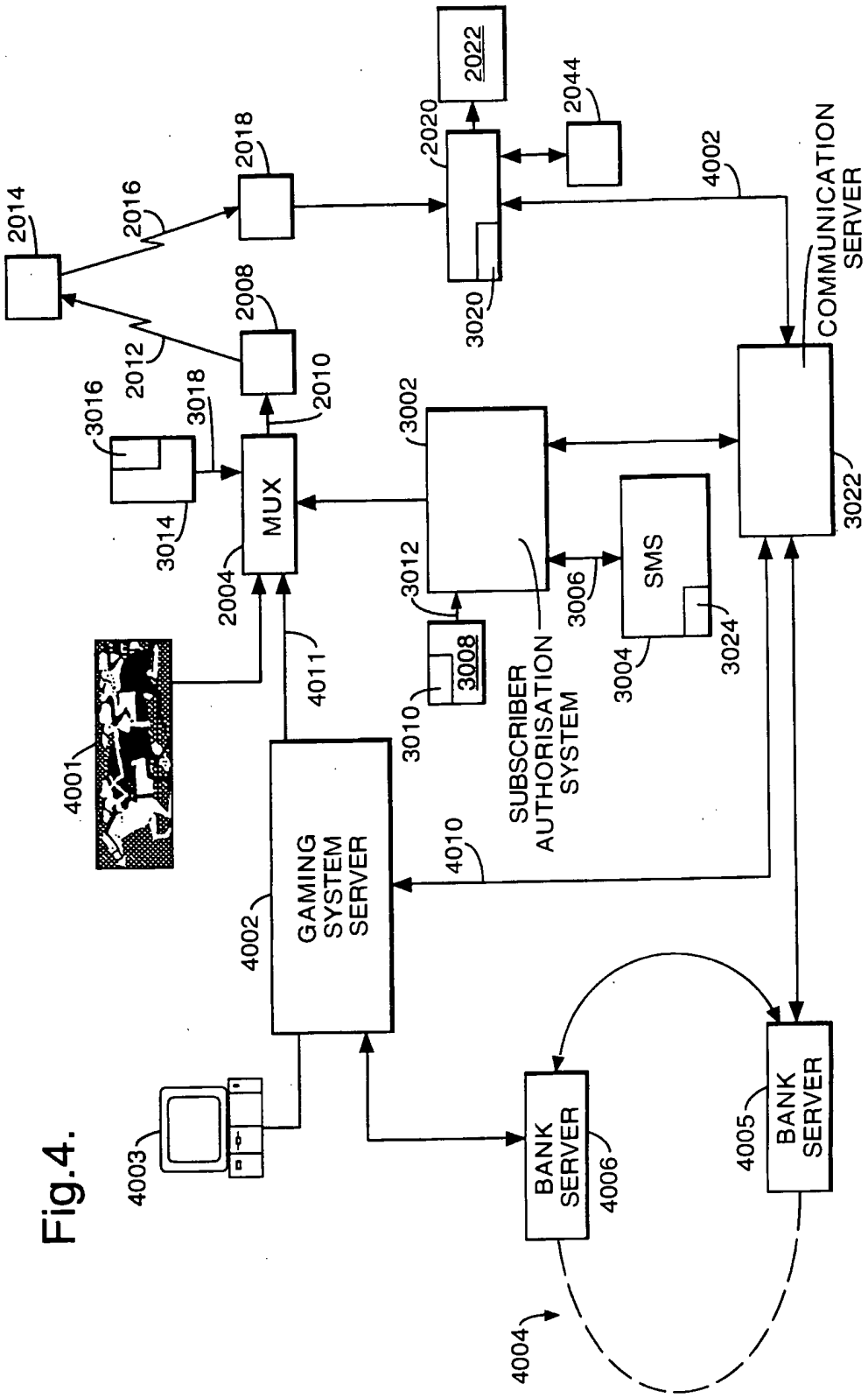


Fig.4.

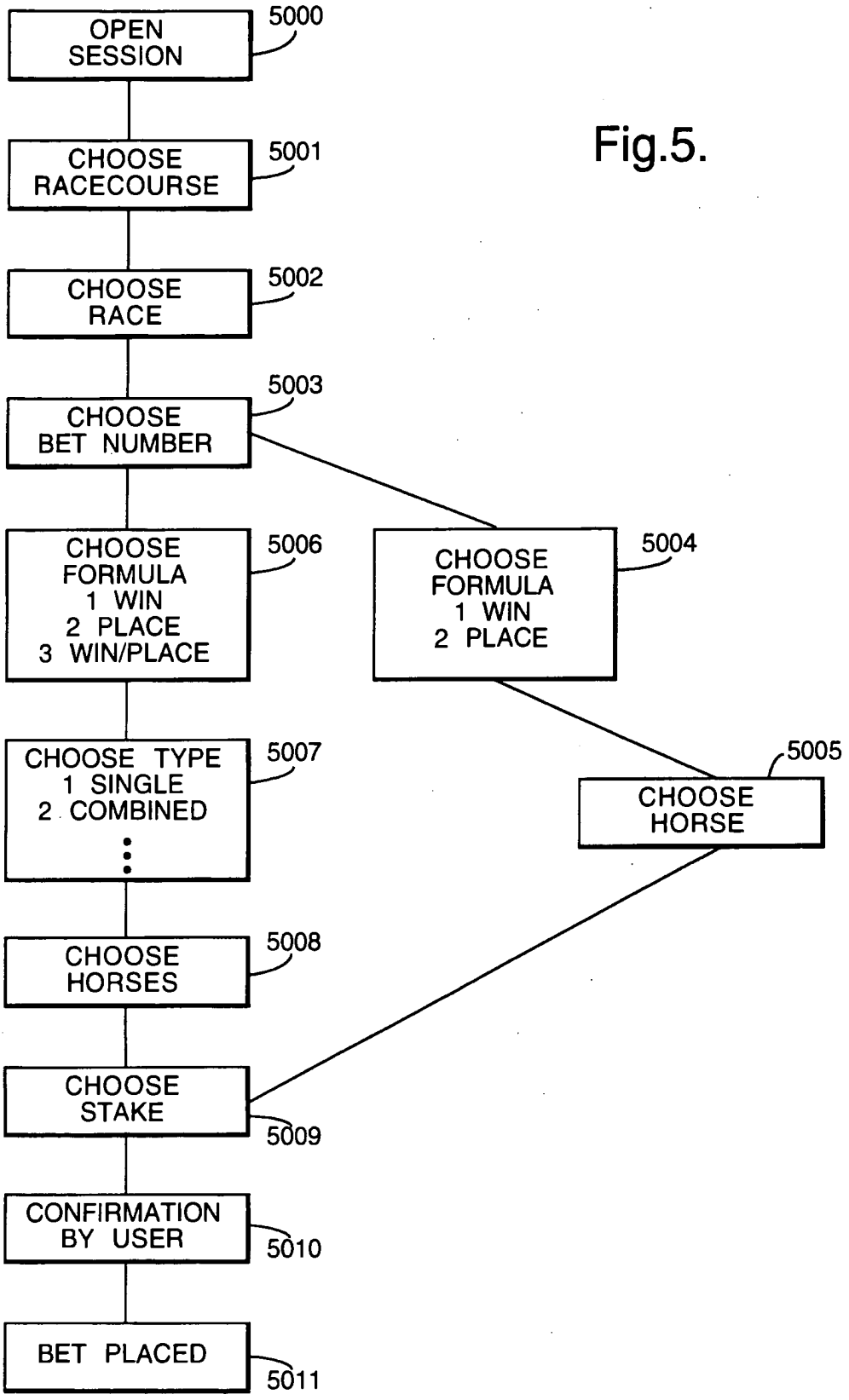


Fig.5.

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 98 40 0285

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
X Y A	US 5 539 822 A (LETT DAVID B) 23 July 1996 * column 7, line 13 - line 33 * * column 9, line 58 - column 10, line 11 * * column 11, line 22 - line 42 * * column 15, line 11 - line 44 * * column 18, line 44 - column 20, line 33 * * figures 3E,3I *	1-3,10, 12, 15-19,23 4-9,11, 14,20, 21,24 5	A63F9/22
X	US 4 815 741 A (SMALL MAYNARD E) 28 March 1989 * column 4, line 41 - column 5, line 5 * * column 5, line 27 - line 38 *	1,19	
Y A	US 5 634 848 A (TSUDA YOICHIRO ET AL) 3 June 1997 * column 1, line 42 - line 64 * * column 3, line 6 - column 4, line 2 * * column 8, line 44 - column 9, line 21 *	6-9,14, 20,21,24 1,19	TECHNICAL FIELDS SEARCHED (Int.Cl.6) A63F H04N G07F G06F
Y	WO 95 01060 A (LINCOLN MINT HONG KONG LTD) 5 January 1995 * page 1, line 19 - line 29 * * page 3, line 17 - page 4, line 35 * * page 12, line 36 - page 13, line 35 * * page 14, line 31 - page 15, line 2 * * page 18, line 22 - line 27 * * page 20, line 8 - line 17 * * page 27, line 21 - page 28, line 6 * * page 29, line 4 - line 23 * * page 40, line 13 - page 42, line 2 *	4,5,11	

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	26 August 1998	Sindic, G

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 946 022 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
29.09.1999 Bulletin 1999/39

(51) Int. Cl.[6]: **H04L 12/58**, H04L 29/06, H04L 12/22

(21) Application number: **99105140.0**

(22) Date of filing: **26.03.1999**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **26.03.1998 JP 7983798**
18.06.1998 JP 17193098
07.08.1998 JP 22486198
05.11.1998 JP 31517298

(71) Applicant:
Nippon Telegraph and Telephone Corporation
Tokyo (JP)

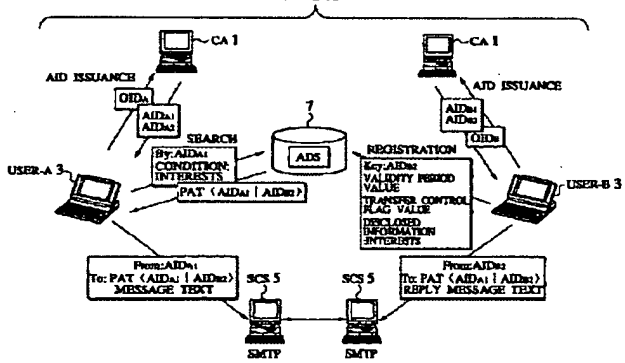
(72) Inventors:
- **Hisada, Yusuke**
  Nippon Telegraph Telephone Corp
  Shinjuku-ku, Tokyo 163-14 (JP)
- **Ono, Satoshi**
  Nippon Telegraph Telephone Corp
  Shinjuku-ku, Tokyo 163-14 (JP)
- **Ichikawa, Haruhisa**
  Nippon Telegraph Telephone Corp
  Shinjuku-ku, Tokyo 163-14 (JP)

(74) Representative: **HOFFMANN - EITLE**
Patent- und Rechtsanwälte
Arabellastrasse 4
81925 München (DE)

(54) Email access control scheme for communication network using identification concealment mechanism

(57) An email access control scheme capable of resolving problems of the real email address and enabling a unique identification of the identity of the user while concealing the user identification is disclosed. A personalized access ticket containing a sender's identification and a recipient's identification in correspondence is to be presented by a sender who wishes to send an email to a recipient so as to specify the recipient as an intended destination of the email. Then, accesses between the sender and the recipient by verifying an access right of the sender with respect to the recipient according to the personalized access ticket at a secure communication service. Also, an official identification of each user by which each user is uniquely identifiable by a certification authority, and an anonymous identification of each user containing at least one fragment of the official identification are defined, and each user is identified by the anonymous identification of each user in communications for emails on a communication network.

FIG.1



EP 0 946 022 A2

Description

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

[0001] The present invention relates to an email access control scheme for controlling transmission and reception of emails by controlling accesses for communications from other users whose identifications on the communication network are concealed while concealing an identification of a recipient on the communication network.

DESCRIPTION OF THE BACKGROUND ART

[0002] In conjunction with the spread of the Internet, the SPAM and the harassment using emails are drastically increasing. The SPAM is a generic name for emails or news that are unilaterally sent without any consideration to the recipient's time consumption, economical and mental burdens. The SPAM using emails are also known as UBE (Unsolicited Bulk Emails) or UCE (Unsolicited Commercial Emails).

[0003] The SPAM is sent indiscriminately regardless of the recipient's age, sex, interests, etc., so that the SPAM often contains an uninteresting or unpleasant content for the recipient. Moreover, the time consumption load and the economical load required for receiving the SPAM is not so small. For the business user, the SPAM can cause the lowering of the working efficiency as it becomes hard to find important mails that are buried among the SPAM. Also, as the SPAM is sent to a huge number of users, the SPAM wastes the network resources and in the worst case the SPAM can cause the overloading. As a result, there case be cases where mails that are important for the user may be lost. Also, the SPAM is sent either anonymously or by pretending someone else so that there is a need to provide some human resources to handle complaints.

[0004] On the other hand, the harassment is an act for keep sending mails with unpleasant contents for the user continually on the purpose of causing mental agony or exerting economical and time consumption burdens to the specific user. Similarly as the SPAM, the harassment mails are sent by pretending an actual or virtual third person, so that the identification of the sender is quite difficult. Also, there are cases where a large capacity mail is sent or a large amount of mails are sent in short period of time so that there is a danger of causing the system breakdown.

[0005] In order to deal with the SPAM and the harassment, the mail system is required to satisfy the following requirements.

- Security

  It is necessary to detect the pretending by the sender and refuse the delivery from the pretending sender.

- Strength

  It is necessary to limit the mail capacity in order to circumvent the system breakdown due to the large capacity mail. It is also necessary to limit the number of transmissions in order to circumvent the system breakdown due to the large amount transmission.

- Compatibility

  It is necessary not to require a considerable change to the implementation of the existing mail system.

- Handling

  It is necessary not to require a considerable change to the handling of the existing mail system.

The MTA (message Transfer Agent) such as sendmail and qmail detects the forgery of the envelope information and the header information and refuses the delivery. The MTA also refuses mail receiving from a mail server which is a source of the SPAM by referring to the so called black list such as MAPS RBL. The MTA also detects the transmission using someone else's real email address and refuses the delivery by carrying out the signature verification using PGP, S/MIME, TLS, etc. The MTA also limits the message length by partial deletion of the message text.

One of the causes of the SPAM and the harassment is the real email address, and the real email address is associated with the following problems.

- User's identity can be guessed from real email address:

  The real email address contains an information useful in guessing the identity so that it can be used in selecting the harassment target. For example, the place of employment can be identified from the real domain. Also, the name and the sex can be guessed from the user name.

- Real email address can be guessed from user's identity:

  The real email address has a universal format of [user name]@[domain name] so that the real email address can be guessed if the user's identity is known, without an explicit knowledge of the real email address itself. For example, if the user's real name is known, the candidates for the user name can be enumerated. Also, if the user's affiliation is known, the candidates for the domain name can be enumerated. Even in the case where the user name is given by a character string which is totally unrelated to the real name, if the naming rule for the user name is known, the user name can be guessed by trial and error transmissions.

- Real email address is transferrable:

  The real email address can be transferred from one person to another, so that mails can be transmitted even if the real email address is not taught by the holder himself. The transfer of real email

address through mails includes the following cases. By specifying the other's real email address in the cc: line of the mail, that real email address can be transferred to all the recipients specified in the To: line of the mail. Also, by forwarding the mail that contains the real email address of the recipient specified in the To: line in the message text to a third person, that real email address can be transferred to the third person.

- Real email address is hard to cancel:

  It is difficult to cancel the real email address because if the real email address is cancelled it becomes impossible to read not only the SPAM and the harassment mails but also the important mails as well.

[0006] Cypherpunk remailers and Mixmaster remailers which are collectively known as Anonymous remailers use a scheme for delivering mails after encrypting the real email address and the real domain of the sender. This scheme is called the reply block. The encryption and decryption of the reply block uses a public key and a secret key of the Anonymous remailer so that it is difficult to identify the real email address and the real domain of the sender for any users other than the sender.

[0007] The Anonymous remailers also make it difficult to transfer the real email address because it is difficult to identify the real email address. However, the reply block is transferrable, so that reply mails can be returned to the sender from users other than the recipient.

[0008] AS-Node and nym.alias.net which are collectively known as Pseudonymous servers use mail transmission and reception using a pseudonym account uniquely corresponding to the real email address of the user. The pseudonym account can be arbitrarily created at the user side so that the user can have a pseudonym account from which the real email address is hard to guess. In addition, by the use of the reply block, it is also possible to conceal the real email address and the real domain of the user to the Pseudonymous server. By combining these means, it can be made difficult to identify the real email address and the real domain of the sender for any users other than the sender. Also, the pseudonym account is cancellable so that there is no need to cancel the real email address.

[0009] The Pseudonymous servers also make it difficult to transfer the real email address because it is difficult to identify the real email address. However, the pseudonym account is transferrable so that reply mails can be returned to the sender from users other than the recipient.

[0010] In addition, in order to protect a recipient from the SPAM and the harrassment, it is also necessary to reject a connection request from a sender who are exercising such action. For this reason, it is necessary for the communication system to be capable of uniquely identifying the identity of the sender.

[0011] In view of these factors, the communication system is required to be capable of uniquely identifying the identity of the user while concealing the real email address of the user (that is while guaranteeing the anonymity of the user), but in the conventional communication system, it has been difficult to meet both of these requirements simultaneously.

[0012] In order to identify the identity of the user in the mail system, the real email address of that user is necessary. On the other hand, the Anonymous remailers deliver a mail after either encrypting or deleting the real email address of the sender in order to guarantee the anonymity of the sender. In order to identify the identity of the sender under this condition, it is necessary to trace the delivery route of the mail using the traffic analysis. However, the Anonymous remailers may delay the mail delivery or interchange the delivery orders of mails. Also, The Mixmaster remailers deliver the mail by dividing it into plural blocks. For this reason, it is difficult to trace the delivery route by the traffic analysis, and therefore the identification of the identity of the sender is also difficult.

[0013] The Pseudonymous servers also utilize the Anonymous remailers for the mail delivery, so that it is possible to guarantee the anonymity of the sender but it is also difficult to uniquely identify the identity of the sender.

[0014] On the other hand, the German Digital Signature Law allows entry of a pseudonym instead of a real name into a digital certificate for generating the digital signature to be used in communication services. The digital certificate is uniquely assigned to the user so that the identity of the user can be uniquely identified even if the pseudonym is entered. Also, the right for naming the pseudonym is given to the user side so that it is possible to enter the pseudonym from which it is difficult to guess the real name.

SUMMARY OF THE INVENTION

[0015] It is therefore an object of the present invention to provide an email access control scheme in a communication network which is capable of resolving the above described problems of the real email address which is one of the causes of the SPAM and the harassment.

[0016] It is another object of the present invention to provide an email access control scheme in a communication network which is capable of enabling a unique identification of the identity of the user while concealing the user identification.

[0017] In order to resolve the problems associated with the transfer and the cancellation of the real email address, the present invention employs the email access control scheme using a personalized access ticket (PAT). In order to resolve the problem associated with the transfer of the real email address, the destination is specified by the PAT which contains both the real email address of the sender and a real email address of

the recipient. Also, in order to resolve the problem associated with the cancellation of the real email address, a validity period is set in the PAT by a Trusted Third Party. Then, the mail delivery from the sender who presented the PAT with the expired validity period will be refused. Also, instead of cancelling the real email address, the PAT is registered at a secure storage device managed by a secure communication service.

[0018] In other words, the present invention controls accesses in units in which the real email address of the sender and the real email address of the recipient is paired. For this reason, even when the real email address is transferred, it is possible to avoid receiving mails from users to which the real email address has been transferred as long as the PAT is not acquired by these users.

[0019] Also, in the present invention, it is possible to refuse receiving mails without cancelling the real email address because the mail delivery from the sender who presented the PAT with the expired validity period or the PAT that is registered in a database by the recipient will be refused.

[0020] Also, in the present invention, the mail receiving can be resumed without re-acquiring the real email address because the mail receiving can be resumed by deleting the PAT from the above described storage device.

[0021] Also, in the present invention, the time consumption and economical loads required for the mail receiving or downloading at the user side can be reduced because the transmission of mails are refused at the server side.

[0022] In addition, the present invention employs the email access control scheme using an official identification (OID) and an anonymous identification (AID) in order to make it possible to identify the identity of the user while guaranteeing the anonymity of the user.

[0023] Namely, in the present invention, a certificate in which the personal information is signed by a secret key of the Trusted Third Party is assigned to each user in order to uniquely identify each user. This certificate will be referred to as OID. Also, a certificate which contains fragments of the OID information is assigned to each user as a user identifier on a communication network in order to make it possible to identify the identity while guaranteeing the anonymity of the user. This certificate will be referred to as AID.

[0024] Also, in the present invention, the OID is reconstructed by judging the identity of a plurality of AIDs in order to identify the identity of the user. Also, the AID is contained in the PAT and the PAT is authenticated at a secure communication service (SCS) in order to resolve the problems associated with the transfer and the cancellation of the AID.

[0025] Also, in the present invention, the AID is managed in a directory which is accessible for search by unspecified many and which outputs the PAT containing the AID as a destination, in order to meet the user side demand for being able to admit accesses from unspecified many without revealing the own identity.

[0026] In this way, in the present invention, the identity of the user can be concealed in the mail transmission and reception because the AID only contains fragments of the OID. Also, the identity of the user can be concealed from unspecified many even when the AID is registered at the directory service which is accessible from unspecified many.

[0027] Also, in the present invention, the identity of the user can be identified probabilistically by reconstructing the OID by judging the identity of a plurality of AIDs. For this reason, it is possible to provide a measure against the SPAM and the harassment without revealing the identity.

[0028] Also, in the present invention, it is possible to admit accesses from unspecified many without revealing the identity, by managing the AID rather than the real email address at the directory and outputting the PAT containing the AID as a destination at the directory.

[0029] More specifically, according to one aspect of the present invention there is provided a method of email access control, comprising the steps of: receiving a personalized access ticket containing a sender's identification and a recipient's identification in correspondence, which is presented by a sender who wishes to send an email to a recipient so as to specify the recipient as an intended destination of the email, at a secure communication service for connecting communications between the sender and the receiver; and controlling accesses between the sender and the recipient by verifying an access right of the sender with respect to the recipient according to the personalized access ticket at the secure communication service.

[0030] Also, in this aspect of the present invention, at the controlling step the secure communication service authenticates the personalized access ticket presented by the sender, and refuses a delivery of the email when the personalized access ticket presented by the sender has been altered.

[0031] Also, in this aspect of the present invention, the personalized access ticket is signed by a secret key of a secure processing device which issued the personalized access ticket, and at the controlling step the secure communication service authenticates the personalized access ticket by verifying a signature of the secure processing device in the personalized access ticket using a public key of the secure processing device.

[0032] Also, in this aspect of the present invention, at the receiving step the secure communication service also receives the sender's identification presented by the sender along with the personalized access ticket, and at the controlling step the secure communication service checks whether the sender's identification presented by the sender is contained in the personalized access ticket presented by the sender, and refuses a delivery of the email when the sender's identification presented by the sender is not contained in the person-

alized access ticket presented by the sender.

[0033] Also, in this aspect of the present invention, the personalized access ticket also contains a validity period indicating a period for which the personalized access ticket is valid, and at the controlling step the secure communication service checks the validity period contained in the personalized access ticket presented by the sender and refuses a delivery of the email when the personalized access ticket presented by the sender contains the validity period that has already been expired.

[0034] Also, in this aspect of the present invention, the validity period of the personalized access ticket is set by a trusted third party.

[0035] Also, in this aspect of the present invention, the method can further comprise the step of: issuing the personalized access ticket to the sender at a directory service for managing an identification of each registrant and a disclosed information of each registrant which has a lower secrecy than a personal information, in a state which is accessible for search by unspecified many, in response to search conditions specified by the sender, by using an identification of a registrant whose disclosed information matches the search conditions as the recipient's identification and the sender's identification specified by the sender along with the search conditions.

[0036] Also, in this aspect of the present invention, the method can further comprise the step of: registering in advance the personalized access ticket containing an identification of a specific user from which a delivery of emails to a specific registrant is to be refused as the sender's identification and an identification of the specific registrant as the recipient's identification, at the secure communication service; wherein the controlling step the secure communication service refuses a delivery of the email from the sender when the personalized access ticket presented by the sender is registered therein in advance at the registering step.

[0037] Also, in this aspect of the present invention, the method can further comprise the step of: deleting the personalized access ticket registered at the secure communication service upon request from the specific registrant who registered the personalized access ticket at the registering step.

[0038] Also, in this aspect of the present invention, the personalized access ticket also contains a transfer control flag indicating whether or not the sender should be authenticated by the secure communication service, and at the controlling step, when the transfer control flag contained in the personalized access ticket indicates that the sender should be authenticated, the secure communication service authenticates the sender's identification presented by the sender and refuses a delivery of the email when an authentication of the sender's identification fails.

[0039] Also, in this aspect of the present invention, the authentication of the sender's identification is realized by a challenge/response procedure between the sender and the secure communication service.

[0040] Also, in this aspect of the present invention, the transfer control flag of the personalized access ticket is set by a trusted third party.

[0041] Also, in this aspect of the present invention, the sender's identification and the recipient's identification in the personalized access ticket can be given by real email addresses of the sender and the recipient.

[0042] Also, in this aspect of the present invention, the sender's identification and the recipient's identification in the personalized access ticket can be given by anonymous identifications of the sender and the recipient, where an anonymous identification of each user contains at least one fragment of an official identification of each user by which each user is uniquely identifiable by a certification authority.

[0043] Also, in this aspect of the present invention, the anonymous identification of each user is an information containing the at least one fragment of the official identification of each user which is signed by the certification authority using a secret key of the certification authority.

[0044] Also, in this aspect of the present invention, the official identification of each user is a character string uniquely assigned to each user by the certification authority and a public key of each user which are signed by a secret key of the certification authority.

[0045] Also, in this aspect of the present invention, the method can further comprise the step of: probabilistically identifying an identity of the sender by reconstructing the official identification of the sender by judging identity of a plurality of anonymous identifications of the sender contained in a plurality of personalized access tickets used by the sender.

[0046] Also, in this aspect of the present invention, an anonymous identification of each user that contains at least one fragment of an official identification of each user by which each user is uniquely identifiable by a certification authority and a link information of each anonymous identification by which each anonymous identification can be uniquely identified can be defined, and the sender's identification and the recipient's identification in the personalized access ticket can be given by a link information of the anonymous identification of the sender and a link information of the anonymous identification of the recipient.

[0047] Also, in this aspect of the present invention, the link information of each anonymous identification is an identifier uniquely assigned to each anonymous identification by the certification authority.

[0048] Also, in this aspect of the present invention, the method can further comprise the step of: probabilistically identifying an identity of the sender by reconstructing the official identification of the sender by judging identity of a plurality of anonymous identifications of the sender corresponding to the link information contained in a plurality of personalized access tickets used by the sender.

[0049] Also, in this aspect of the present invention, the personalized access ticket can contain a single sender's identification and a single recipient's identification in 1-to-1 correspondence.

[0050] Also, in this aspect of the present invention, the personalized access ticket can contain a single sender's identification and a plurality of recipient's identifications in 1-to-N correspondence, where N is an integer greater than 1.

[0051] Also, in this aspect of the present invention, one identification among the single sender's identification and the plurality of recipient's identifications is a holder identification for identifying a holder of the personalized access ticket while other identifications among the single sender's identification and the plurality of recipient's identifications are member identifications for identifying members of a group to which the holder belongs.

[0052] Also, in this aspect of the present invention, the method can further comprise the step of: issuing an identification of each user and an enabler of the identification of each user indicating a right to change the personalized access ticket containing the identification of each user as the holder identification, to each user at a certification authority, such that prescribed processing on the personalized access ticket can be carried out at a secure processing device only by a user who presented both the holder identification contained in the personalized access ticket and the enabler corresponding to the holder identification to the secure processing device.

[0053] Also, in this aspect of the present invention, the certification authority issues the enabler of the identification of each user as an information indicating that it is the enabler and the identification of each user itself which are signed by a secret key of the certification authority.

[0054] Also, in this aspect of the present invention, the prescribed processing includes a generation of a new personalized access ticket, a merging of a plurality of personalized access tickets, a splitting of one personalized access ticket into a plurality of personalized access tickets, a changing of the holder of the personalized access ticket, changing of a validity period of the personalized access ticket, and a changing of a transfer control flag of the personalized access ticket.

[0055] Also, in this aspect of the present invention, a special identification and a special enabler corresponding to the special identification which are known to all users can be defined such that the generation of a new personalized access ticket and the changing of the holder of the personalized access ticket can be carried out by the holder of the personalized access ticket by using the special identification and the special enabler without using an enabler of a member identification.

[0056] Also, in this aspect of the present invention, the special identification is defined to be capable of being used only as the holder identification of the personalized access ticket.

[0057] Also, in this aspect of the present invention, a special identification which is known to all users can be defined such that a read only attribute can be set to the personalized access ticket by using the special identification.

[0058] Also, in this aspect of the present invention, at the controlling step, when the access right of the sender with respect to the recipient is verified according to the personalized access ticket, the secure communication service takes out the recipient's identification from the personalized access ticket by using the sender's identification presented by the sender, converts the mail by using a taken out recipient's identification into a format that can be interpreted by a mail transfer function for actually carrying out a mail delivery processing, and gives the mail after conversion to the mail transfer function by attaching the personalized access ticket.

[0059] According to another aspect of the present invention there is provided a method of email access control, comprising the steps of: defining an official identification of each user by which each user is uniquely identifiable by a certification authority, and an anonymous identification of each user containing at least one fragment of the official identification; and identifying each user by the anonymous identification of each user in communications for emails on a communication network.

[0060] Also, in this aspect of the present invention, the anonymous identification of each user is an information containing the at least one fragment of the official identification of each user which is signed by the certification authority using a secret key of the certification authority.

[0061] Also, in this aspect of the present invention, the official identification of each user is a character string uniquely assigned to each user by the certification authority and a public key of each user which are signed by a secret key of the certification authority.

[0062] Also, in this aspect of the present invention, the method can further comprise the steps of: receiving a personalized access ticket containing a sender's anonymous identification and a recipient's anonymous identification in correspondence, which is presented by a sender who wishes to send an email to a recipient so as to specify the recipient as an intended destination of the email, at a secure communication service for connecting communications between the sender and the receiver; and controlling accesses between the sender and the recipient by verifying an access right of the sender with respect to the recipient according to the personalized access ticket at the secure communication service.

[0063] Also, in this aspect of the present invention, the method can further comprises the step of: probabilistically identifying an identity of the sender at the secure communication service by reconstructing the official identification of the sender while judging identity of a plurality of anonymous identifications of the sender con-

tained in a plurality of personalized access tickets used by the sender.

[0064] Also, in this aspect of the present invention, the defining step can also define a link information of each anonymous identification by which each anonymous identification can be uniquely identified, and each anonymous identification can also contain the link information of each anonymous identification.

[0065] Also, in this aspect of the present invention, the link information of each anonymous identification is an identifier uniquely assigned to each anonymous identification by the certification authority.

[0066] Also, in this aspect of the present invention, the method can further comprises the steps of: receiving a personalized access ticket containing a link information of a sender's anonymous identification and a link information of a recipient's anonymous identification in correspondence, which is presented by a sender who wishes to send an email to a recipient so as to specify the recipient as an intended destination of the email, at a secure communication service for connecting communications between the sender and the receiver; and controlling accesses between the sender and the recipient by verifying an access right of the sender with respect to the recipient according to the personalized access ticket at the secure communication service.

[0067] Also, in this aspect of the present invention, the method can further comprises the step of: probabilistically identifying an identity of the sender by reconstructing the official identification of the sender while judging identity of a plurality of anonymous identifications of the sender corresponding to the link information contained in a plurality of personalized access tickets used by the sender.

[0068] According to another aspect of the present invention there is provided a communication system realizing email access control, comprising: a communication network to which a plurality of user terminals are connected; and a secure communication service device for connecting communications between the sender and the receiver on the communication network, by receiving a personalized access ticket containing a sender's identification and a recipient's identification in correspondence, which is presented by a sender who wishes to send an email to a recipient so as to specify the recipient as an intended destination of the email, and controlling accesses between the sender and the recipient by verifying an access right of the sender with respect to the recipient according to the personalized access ticket.

[0069] Also, in this aspect of the present invention, the secure communication service device authenticates the personalized access ticket presented by the sender, and refuses a delivery of the email when the personalized access ticket presented by the sender has been altered.

[0070] Also, in this aspect of the present invention, the system further comprises: a secure processing device for issuing the personalized access ticket which is signed by a secret key of the secure processing device; wherein the secure communication service device authenticates the personalized access ticket by verifying a signature of the secure processing device in the personalized access ticket using a public key of the secure processing device.

[0071] Also, in this aspect of the present invention, the secure communication service device also receives the sender's identification presented by the sender along with the personalized access ticket, checks whether the sender's identification presented by the sender is contained in the personalized access ticket presented by the sender, and refuses a delivery of the email when the sender's identification presented by the sender is not contained in the personalized access ticket presented by the sender.

[0072] Also, in this aspect of the present invention, the personalized access ticket also contains a validity period indicating a period for which the personalized access ticket is valid, and the secure communication service device checks the validity period contained in the personalized access ticket presented by the sender and refuses a delivery of the email when the personalized access ticket presented by the sender contains the validity period that has already been expired.

[0073] Also, in this aspect of the present invention, the system further comprises: a trusted third party for setting the validity period of the personalized access ticket.

[0074] Also, in this aspect of the present invention, the system can further comprise: a directory service device for managing an identification of each registrant and and a disclosed information of each registrant which has a lower secrecy than a personal information, in a state which is accessible for search by unspecified many, and issuing the personalized access ticket to the sender in response to search conditions specified by the sender, by using an identification of a registrant whose disclosed information matches the search conditions as the recipient's identification and the sender's identification specified by the sender along with the search conditions.

[0075] Also, in this aspect of the present invention, the secure communication service device can register in advance the personalized access ticket containing an identification of a specific user from which a delivery of emails to a specific registrant is to be refused as the sender's identification and an identification of the specific registrant as the recipient's identification, and refuse a delivery of the email from the sender when the personalized access ticket presented by the sender is registered therein in advance.

[0076] Also, in this aspect of the present invention, the secure communication service device can delete the personalized access ticket registered therein upon request from the specific registrant who registered the personalized access ticket.

[0077] Also, in this aspect of the present invention, the

personalized access ticket also contains a transfer control flag indicating whether or not the sender should be authenticated by the secure communication service, and when the transfer control flag contained in the personalized access ticket indicates that the sender should be authenticated, the secure communication service device authenticates the sender's identification presented by the sender and refuses a delivery of the email when an authentication of the sender's identification fails.

[0078] Also, in this aspect of the present invention, the authentication of the sender's identification is realized by a challenge/response procedure between the sender and the secure communication service device.

[0079] Also, in this aspect of the present invention, the system further comprises a trusted third party for setting the transfer control flag of the personalized access ticket.

[0080] Also, in this aspect of the present invention, the sender's identification and the recipient's identification in the personalized access ticket can be given by real email addresses of the sender and the recipient.

[0081] Also, in this aspect of the present invention, the system can further comprise: a certification authority device for issuing an anonymous identification of each user which contains at least one fragment of an official identification of each user by which each user is uniquely identifiable by the certification authority device; wherein the sender's identification and the recipient's identification in the personalized access ticket can be given by anonymous identifications of the sender and the recipient.

[0082] Also, in this aspect of the present invention, the anonymous identification of each user is an information containing the at least one fragment of the official identification of each user which is signed by the certification authority device using a secret key of the certification authority device.

[0083] Also, in this aspect of the present invention, the official identification of each user is a character string uniquely assigned to each user by the certification authority device and a public key of each user which are signed by a secret key of the certification authority device.

[0084] Also, in this aspect of the present invention, the secure communication service device can probabilistically identify an identity of the sender by reconstructing the official identification of the sender while judging identity of a plurality of anonymous identifications of the sender contained in a plurality of personalized access tickets used by the sender.

[0085] Also, in this aspect of the present invention, the system can further comprise: a certification authority device for issuing an anonymous identification of each user which contains at least one fragment of an official identification of each user by which each user is uniquely identifiable by the certification authority device and a link information of each anonymous identification by which each anonymous identification can be uniquely identified; wherein the sender's identification and the recipient's identification in the personalized access ticket can be given by a link information of the anonymous identification of the sender and a link information of the anonymous identification of the recipient.

[0086] Also, in this aspect of the present invention, the link information of each anonymous identification is an identifier uniquely assigned to each anonymous identification by the certification authority device.

[0087] Also, in this aspect of the present invention, the secure communication service device can probabilistically identify an identity of the sender by reconstructing the official identification of the sender while judging identity of a plurality of anonymous identifications of the sender corresponding to the link information contained in a plurality of personalized access tickets used by the sender.

[0088] Also, in this aspect of the present invention, the personalized access ticket can contain a single sender's identification and a single recipient's identification in 1-to-1 correspondence.

[0089] Also, in this aspect of the present invention, the personalized access ticket can contain a single sender's identification and a plurality of recipient's identifications in 1-to-N correspondence, where N is an integer greater than 1.

[0090] Also, in this aspect of the present invention, one identification among the single sender's identification and the plurality of recipient's identifications is a holder identification for identifying a holder of the personalized access ticket while other identifications among the single sender's identification and the plurality of recipient's identifications are member identifications for identifying members of a group to which the holder belongs.

[0091] Also, in this aspect of the present invention, the system can further comprises: a certification authority device for issuing to each user an identification of each user and an enabler of the identification of each user indicating a right to change the personalized access ticket containing the identification of each user as the holder identification; and a secure processing device at which prescribed processing on the personalized access ticket can be carried out only by a user who presented both the holder identification contained in the personalized access ticket and the enabler corresponding to the holder identification to the secure processing device.

[0092] Also, in this aspect of the present invention, the certification authority device issues the enabler of the identification of each user as an information indicating that it is the enabler and the identification of each user itself which are signed by a secret key of the certification authority device.

[0093] Also, in this aspect of the present invention, the prescribed processing includes a generation of a new personalized access ticket, a merging of a plurality of

personalized access tickets, a splitting of one personalized access ticket into a plurality of personalized access tickets, a changing of the holder of the personalized access ticket, changing of a validity period of the personalized access ticket, and a changing of a transfer control flag of the personalized access ticket.

[0094] Also, in this aspect of the present invention, a special identification and a special enabler corresponding to the special identification which are known to all users can be defined such that the generation of a new personalized access ticket and the changing of the holder of the personalized access ticket can be carried out by the holder of the personalized access ticket by using the special identification and the special enabler without using an enabler of a member identification.

[0095] Also, in this aspect of the present invention, the special identification is defined to be capable of being used only as the holder identification of the personalized access ticket.

[0096] Also, in this aspect of the present invention, a special identification which is known to all users can be defined such that a read only attribute can be set to the personalized access ticket by using the special identification.

[0097] Also, in this aspect of the present invention, when the access right of the sender with respect to the recipient is verified according to the personalized access ticket, the secure communication service device takes out the recipient's identification from the personalized access ticket by using the sender's identification presented by the sender, converts the mail by using a taken out recipient's identification into a format that can be interpreted by a mail transfer function for actually carrying out a mail delivery processing, and gives the mail after conversion to the mail transfer function by attaching the personalized access ticket.

[0098] According to another aspect of the present invention there is provided a communication system realizing email access control, comprising: a certification authority device for defining an official identification of each user by which each user is uniquely identifiable by the certification authority device, and an anonymous identification of each user which contains at least one fragment of the official identification; and a communication network on which each user is identified by the anonymous identification of each user in communications for emails on the communication network.

[0099] Also, in this aspect of the present invention, the anonymous identification of each user is an information containing the at least one fragment of the official identification of each user which is signed by the certification authority device using a secret key of the certification authority device.

[0100] Also, in this aspect of the present invention, the official identification of each user is a character string uniquely assigned to each user by the certification authority device and a public key of each user which are signed by a secret key of the certification authority device.

[0101] Also, in this aspect of the present invention, the system can further comprises: a secure communication service device for connecting communications between the sender and the receiver on the communication network, by receiving a personalized access ticket containing a sender's anonymous identification and a recipient's anonymous identification in correspondence, which is presented by a sender who wishes to send an email to a recipient so as to specify the recipient as an intended destination of the email, and controlling accesses between the sender and the recipient by verifying an access right of the sender with respect to the recipient according to the personalized access ticket.

[0102] Also, in this aspect of the present invention, the secure communication service device can probabilistically identify an identity of the sender by reconstructing the official identification of the sender while judging identity of a plurality of anonymous identifications of the sender contained in a plurality of personalized access tickets used by the sender.

[0103] Also, in this aspect of the present invention, the certification authority device can also define a link information of each anonymous identification by which each anonymous identification can be uniquely identified, and each anonymous identification can also contain the link information of each anonymous identification.

[0104] Also, in this aspect of the present invention, the link information of each anonymous identification is an identifier uniquely assigned to each anonymous identification by the certification authority device.

[0105] Also, in this aspect of the present invention, the system can further comprise: a secure communication service device for connecting communications between the sender and the receiver on the communication network, by receiving a personalized access ticket containing a link information of a sender's anonymous identification and a link information of a recipient's anonymous identification in correspondence, which is presented by a sender who wishes to send an email to a recipient so as to specify the recipient as an intended destination of the email, and controlling accesses between the sender and the recipient by verifying an access right of the sender with respect to the recipient according to the personalized access ticket.

[0106] Also, in this aspect of the present invention, the secure communication service device can probabilistically identify an identity of the sender by reconstructing the official identification of the sender while judging identity of a plurality of link informations of anonymous identifications of the sender contained in a plurality of personalized access tickets used by the sender.

[0107] According to another aspect of the present invention there is provided a secure communication service device for use in a communication system realizing email access control, comprising: a computer hardware; and a computer software for causing the computer hardware to connect communications

between the sender and the receiver, by receiving a personalized access ticket containing a sender's identification and a recipient's identification in correspondence, which is presented by a sender who wishes to send an email to a recipient so as to specify the recipient as an intended destination of the email, and controlling accesses between the sender and the recipient by verifying an access right of the sender with respect to the recipient according to the personalized access ticket.

[0108] Also, in this aspect of the present invention, the computer software causes the computer hardware to authenticate the personalized access ticket presented by the sender, and refuse a delivery of the email when the personalized access ticket presented by the sender has been altered.

[0109] Also, in this aspect of the present invention, the personalized access ticket is signed by a secret key of a secure processing device which issued the personalized access ticket, and the computer software causes the computer hardware to authenticate the personalized access ticket by verifying a signature of the secure processing device in the personalized access ticket using a public key of the secure processing device.

[0110] Also, in this aspect of the present invention, the computer software causes the computer hardware to also receive the sender's identification presented by the sender along with the personalized access ticket, check whether the sender's identification presented by the sender is contained in the personalized access ticket presented by the sender, and refuse a delivery of the email when the sender's identification presented by the sender is not contained in the personalized access ticket presented by the sender.

[0111] Also, in this aspect of the present invention, the personalized access ticket also contains a validity period indicating a period for which the personalized access ticket is valid, and the computer software causes the computer hardware to check the validity period contained in the personalized access ticket presented by the sender and refuse a delivery of the email when the personalized access ticket presented by the sender contains the validity period that has already been expired.

[0112] Also, in this aspect of the present invention, the computer software can cause the computer hardware to register in advance the personalized access ticket containing an identification of a specific user from which a delivery of emails to a specific registrant is to be refused as the sender's identification and an identification of the specific registrant as the recipient's identification, at the secure communication service device, and refuse a delivery of the email from the sender when the personalized access ticket presented by the sender is registered at the secure communication service device in advance.

[0113] Also, in this aspect of the present invention, the computer software can cause the computer hardware to delete the personalized access ticket registered at the secure communication service device upon request from the specific registrant who registered the personalized access ticket.

[0114] Also, in this aspect of the present invention, the personalized access ticket also contains a transfer control flag indicating whether or not the sender should be authenticated by the secure communication service device, and when the transfer control flag contained in the personalized access ticket indicates that the sender should be authenticated, the computer software causes the computer hardware to authenticate the sender's identification presented by the sender and refuse a delivery of the email when an authentication of the sender's identification fails.

[0115] Also, in this aspect of the present invention, the computer software causes the computer hardware to realize the authentication of the sender's identification by a challenge/response procedure between the sender and the secure communication service device.

[0116] Also, in this aspect of the present invention, the sender's identification and the recipient's identification in the personalized access ticket can be given by anonymous identifications of the sender and the recipient, where an anonymous identification of each user contains at least one fragment of an official identification of each user by which each user is uniquely identifiable by a certification authority, and the computer software can also cause the computer hardware to probabilistically identify an identity of the sender by reconstructing the official identification of the sender by judging identity of a plurality of anonymous identifications of the sender contained in a plurality of personalized access tickets used by the sender.

[0117] Also, in this aspect of the present invention, an anonymous identification of each user that contains at least one fragment of an official identification of each user by which each user is uniquely identifiable by a certification authority and a link information of each anonymous identification by which each anonymous identification can be uniquely identified can be defined, the sender's identification and the recipient's identification in the personalized access ticket can be given by a link information of the anonymous identification of the sender and a link information of the anonymous identification of the recipient, and the computer software can also cause the computer hardware to probabilistically identify an identity of the sender by reconstructing the official identification of the sender by judging identity of a plurality of anonymous identifications of the sender corresponding to the link information contained in a plurality of personalized access tickets used by the sender.

[0118] Also, in this aspect of the present invention, when the access right of the sender with respect to the recipient is verified according to the personalized access ticket, the computer software causes the computer hardware to take out the recipient's identification from the personalized access ticket by using the sender's identification presented by the sender, convert

the mail by using a taken out recipient's identification into a format that can be interpreted by a mail transfer function for actually carrying out a mail delivery processing, and give the mail after conversion to the mail transfer function by attaching the personalized access ticket.

[0119] According to another aspect of the present invention there is provided a secure processing device for use in a communication system realizing email access control, comprising: a computer hardware; and a computer software for causing the computer hardware to receive a request for a personalized access ticket from a user, and issue a personalized access ticket containing a sender's identification and a recipient's identification in correspondence, which is signed by a secret key of the secure processing device.

[0120] According to another aspect of the present invention there is provided a directory service device for use in a communication system realizing email access control, comprising: a computer hardware; and a computer software for causing the computer hardware to manage an identification of each registrant and a disclosed information of each registrant which has a lower secrecy than a personal information, in a state which is accessible for search by unspecified many, and issue a personalized access ticket containing a sender's identification and a recipient's identification in correspondence, to the sender in response to search conditions specified by the sender, by using an identification of a registrant whose disclosed information matches the search conditions as the recipient's identification and the sender's identification specified by the sender along with the search conditions.

[0121] According to another aspect of the present invention there is provided a certification authority device for use in a communication system realizing email access control, comprising: a computer hardware; and a computer software for causing the computer hardware to issue to each user an official identification of each user by which each user is uniquely identifiable by the certification authority device, and an anonymous identification of each user which contains at least one fragment of the official identification.

[0122] According to another aspect of the present invention there is provided a certification authority device for use in a communication system realizing email access control, comprising: a computer hardware; and a computer software for causing the computer hardware to issue to each user an identification of each user and an enabler of the identification of each user indicating a right to change any personalized access ticket that contains the identification of each user as a holder identification, where the personalized access ticket generally contains a sender's identification and a plurality of recipient's identifications in correspondence, and one of the sender's identification and the recipient's identifications is a holder identification.

[0123] According to another aspect of the present invention there is provided a secure processing device for use in a communication system realizing email access control, comprising: a computer hardware; and a computer software for causing the computer hardware to receive from a user a request for prescribed processing on a personalized access ticket containing a sender's identification and a plurality of recipient's identifications in correspondence, where one of the sender's identification and the recipient's identifications is a holder identification, and execute the prescribed processing on the personalized access ticket when the user presented both the holder identification contained in the personalized access ticket and an enabler corresponding to the holder identification which indicates a right to change the personalized access ticket containing the identification of the user as the holder identification.

[0124] According to another aspect of the present invention there is provided a computer usable medium having computer readable program code means embodied therein for causing a computer to function as a secure communication service device for use in a communication system realizing email access control, the computer readable program code means includes: first computer readable program code means for causing said computer to receive a personalized access ticket containing a sender's identification and a recipient's identification in correspondence, which is presented by a sender who wishes to send an email to a recipient so as to specify the recipient as an intended destination of the email; and second computer readable program code means for causing said computer to control accesses between the sender and the recipient by verifying an access right of the sender with respect to the recipient according to the personalized access ticket, so as to connect communications between the sender and the receiver on the communication network.

[0125] Also, in this aspect of the present invention, the second computer readable program code means causes said computer to authenticate the personalized access ticket presented by the sender, and refuse a delivery of the email when the personalized access ticket presented by the sender has been altered.

[0126] Also, in this aspect of the present invention, the personalized access ticket is signed by a secret key of a secure processing device which issued the personalized access ticket, and the second computer readable program code means causes said computer to authenticate the personalized access ticket by verifying a signature of the secure processing device in the personalized access ticket using a public key of the secure processing device.

[0127] Also, in this aspect of the present invention, the first computer readable program code means causes said computer to also receive the sender's identification presented by the sender along with the personalized access ticket, and the second computer readable program code means causes said computer to check

whether the sender's identification presented by the sender is contained in the personalized access ticket presented by the sender and refuse a delivery of the email when the sender's identification presented by the sender is not contained in the personalized access ticket presented by the sender.

[0128] Also, in this aspect of the present invention, the personalized access ticket also contains a validity period indicating a period for which the personalized access ticket is valid, and the second computer readable program code means causes said computer to check the validity period contained in the personalized access ticket presented by the sender and refuse a delivery of the email when the personalized access ticket presented by the sender contains the validity period that has already been expired.

[0129] Also, in this aspect of the present invention, the second computer readable program code means can cause said computer to register in advance the personalized access ticket containing an identification of a specific user from which a delivery of emails to a specific registrant is to be refused as the sender's identification and an identification of the specific registrant as the recipient's identification, at the secure communication service device, and refuse a delivery of the email from the sender when the personalized access ticket presented by the sender is registered at the secure communication service device in advance.

[0130] Also, in this aspect of the present invention, the second computer readable program code means can cause said computer to delete the personalized access ticket registered at the secure communication service device upon request from the specific registrant who registered the personalized access ticket.

[0131] Also, in this aspect of the present invention, the personalized access ticket also contains a transfer control flag indicating whether or not the sender should be authenticated by the secure communication service device, and when the transfer control flag contained in the personalized access ticket indicates that the sender should be authenticated, the second computer readable program code means causes said computer to authenticate the sender's identification presented by the sender and refuse a delivery of the email when an authentication of the sender's identification fails.

[0132] Also, in this aspect of the present invention, the second computer readable program code means causes said computer to realize the authentication of the sender's identification by a challenge/response procedure between the sender and the secure communication service device.

[0133] Also, in this aspect of the present invention, the sender's identification and the recipient's identification in the personalized access ticket can be given by anonymous identifications of the sender and the recipient, where an anonymous identification of each user contains at least one fragment of an official identification of each user by which each user is uniquely identifiable by a certification authority, and the second computer readable program code means can also cause said computer to probabilistically identify an identity of the sender by reconstructing the official identification of the sender by judging identity of a plurality of anonymous identifications of the sender contained in a plurality of personalized access tickets used by the sender.

[0134] Also, in this aspect of the present invention, an anonymous identification of each user that contains at least one fragment of an official identification of each user by which each user is uniquely identifiable by a certification authority and a link information of each anonymous identification by which each anonymous identification can be uniquely identified can be defined, the sender's identification and the recipient's identification in the personalized access ticket can be given by a link information of the anonymous identification of the sender and a link information of the anonymous identification of the recipient, and the second computer readable program code means can also cause said computer to probabilistically identify an identity of the sender by reconstructing the official identification of the sender by judging identity of a plurality of anonymous identifications of the sender corresponding to the link information contained in a plurality of personalized access tickets used by the sender.

[0135] Also, in this aspect of the present invention, when the access right of the sender with respect to the recipient is verified according to the personalized access ticket, the second computer readable program code means causes said computer to take out the recipient's identification from the personalized access ticket by using the sender's identification presented by the sender, convert the mail by using a taken out recipient's identification into a format that can be interpreted by a mail transfer function for actually carrying out a mail delivery processing, and give the mail after conversion to the mail transfer function by attaching the personalized access ticket.

[0136] According to another aspect of the present invention there is provided a computer usable medium having computer readable program code means embodied therein for causing a computer to function as a secure processing device for use in a communication system realizing email access control, the computer readable program code means includes: first computer readable program code means for causing said computer to receive a request for a personalized access ticket from a user; and second computer readable program code means for causing said computer to issue the personalized access ticket containing a sender's identification and a recipient's identification in correspondence, which is signed by a secret key of the secure processing device.

[0137] According to another aspect of the present invention there is provided a computer usable medium having computer readable program code means embodied therein for causing a computer to function as

a directory service devicer for use in a communication system realizing email access control, the computer readable program code means includes: first computer readable program code means for causing said computer to manage an identification of each registrant and a disclosed information of each registrant which has a lower secrecy than a personal information, in a state which is accessible for search by unspecified many, and second computer readable program code means for causing said computer to issue a personalized access ticket containing a sender's identification and a recipient's identification in correspondence, to the sender in response to search conditions specified by the sender, by using an identification of a registrant whose disclosed information matches the search conditions as the recipient's identification and the sender's identification specified by the sender along with the search conditions.

[0138] According to another aspect of the present invention there is provided a computer usable medium having computer readable program code means embodied therein for causing a computer to function as a certification authority device for use in a communication system realizing email access control, the computer readable program code means includes: first computer readable program code means for causing said computer to issue to each user an official identification of each user by which each user is uniquely identifiable by the certification authority device; and second computer readable program code means for causing said computer to issue to each user an anonymous identification of each user which contains at least one fragment of the official identification.

[0139] According to another aspect of the present invention there is provided a computer usable medium having computer readable program code means embodied therein for causing a computer to function as a certification authority device for use in a communication system realizing email access control, the computer readable program code means includes: first computer readable program code means for causing said computer to issue to each user an identification of each user; and second computer readable program code means for causing said computer to issue to each user an enabler of the identification of each user indicating a right to change any personalized access ticket that contains the identification of each user as a holder identification, where the persnalized access ticket generally contains a sender's identification and a plurality of recipient's identifications in correspondence, and one of the sender's identification and the recipient's identifications is a holder identification.

[0140] According to another aspect of the present invention there is provided a computer usable medium having computer readable program code means embodied therein for causing a computer to function as a secure processing device for use in a communication system realizing email access control, the computer readable program code means includes: first computer readable program code means for causing said computer to receive from a user a request for prescribed processing on a personalized access ticket containing a sender's identification and a plurality of recipient's identifications in correspondence, where one of the sender's identification and the recipient's identifications is a holder identification; and second computer readable program code means for causing said computer to execute the prescribed processing on the personalized access ticket when the user presented both the holder identification contained in the personalized access ticket and an enabler corresponding to the holder identification which indicates a right to change the personalized access ticket containing the identification of the user as the holder identification.

[0141] Other features and advantages of the present invention will become apparent from the following description taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0142]

Fig. 1 is a diagram showing an overall configuration of a communication system according to the first embodiment of the present invention.

Fig. 2 is a diagram showing exemplary data structures of an official identification, an anonymous identification, and a 1-to-1 personalized access ticket according to the first embodiment of the present invention.

Fig. 3 is a flow chart for an anonymous identification generation processing at a certification authority according to the first embodiment of the present invention.

Fig. 4 is a flow chart for a personalized access ticket generation processing at an anonymous directory service according to the first embodiment of the present invention.

Fig. 5 is a flow chart for a mail access control processing at a secure communication service according to the first embodiment of the present invention.

Fig. 6 is a flow chart for an anonymous identification identity judgement processing at a secure communication service according to the first embodiment of the present invention.

Fig. 7 is a diagram showing exemplary data structures of data used in the anonymous identification identity judgement processing of Fig. 6.

Fig. 8 is a diagram showing exemplary data structures of an official identification, an anonymous identification, and a 1-to-N personalized access ticket according to the second embodiment of the present invention.

Fig. 9 is a diagram showing exemplary data struc-

tures of an anonymous identification and an enabler according to the second embodiment of the present invention.

Fig. 10 is a diagram showing a definition of a processing rule (MakePAT) used in the second embodiment of the present invention.

Fig. 11 is a diagram showing a definition of a processing rule (MergePAT) used in the second embodiment of the present invention.

Fig. 12 is a diagram showing a definition of a processing rule (SplitPAT) used in the second embodiment of the present invention.

Fig. 13 is a diagram showing a definition of a processing rule (TransPAT) used in the second embodiment of the present invention.

Fig. 14 is a first exemplary system configuration that can be used in the second embodiment of the present invention.

Fig. 15 is a second exemplary system configuration that can be used in the second embodiment of the present invention.

Fig. 16 is a third exemplary system configuration that can be used in the second embodiment of the present invention.

Fig. 17 is a fourth exemplary system configuration that can be used in the second embodiment of the present invention.

Fig. 18 is a fifth exemplary system configuration that can be used in the second embodiment of the present invention.

Fig. 19 is a sixth exemplary system configuration that can be used in the second embodiment of the present invention.

Fig. 20 is a seventh exemplary system configuration that can be used in the second embodiment of the present invention.

Fig. 21 is a flow chart showing an overall processing flow of MakePAT, MergePAT or TransPAT processing according to the second embodiment of the present invention.

Fig. 22 is a flow chart showing an overall processing flow of SplitPAT processing according to the second embodiment of the present invention.

Fig. 23 is a flow chart for an anonymous identification list generation processing (for MakePAT, MergePAT, SplitPAT and TransPAT) according to the second embodiment of the present invention.

Fig. 24 is an enabler authenticity verification processing (for MakePAT, MergePAT, SplitPAT and TransPAT) according to the second embodiment of the present invention.

Fig. 25 is a diagram showing an exemplary data structure of Null-AID used in the third embodiment of the present invention.

Fig. 26 is a diagram showing an exemplary data structure of Enabler of Null-AID used in the third embodiment of the present invention.

Fig. 27 is a diagram showing a first exemplary application of the third embodiment of the present invention.

Fig. 28 is a diagram showing a second exemplary application of the third embodiment of the present invention.

Fig. 29 is a diagram showing an exemplary data structure of God-AID used in the fourth embodiment of the present invention.

Fig. 30 is a diagram showing a first exemplary application of the fourth embodiment of the present invention.

Fig. 31 is a diagram showing a second exemplary application of the fourth embodiment of the present invention.

Fig. 32 is a flow chart for a member anonymous identification checking processing according to the fifth embodiment of the present invention.

Fig. 33 is a diagram showing an overall configuration of a communication system according to the sixth embodiment of the present invention.

Fig. 34 is a diagram showing exemplary data structures of an official identification, a link information attached anonymous identification, and a link specifying 1-to-1 personalized access ticket according to the sixth embodiment of the present invention.

Fig. 35 is a flow chart for a link information attached anonymous identification generation processing at a certification authority according to the sixth embodiment of the present invention.

Fig. 36 is a flow chart for a link specifying 1-to-1 personalized access ticket generation processing at an anonymous directory service according to the sixth embodiment of the present invention.

Fig. 37 is a flow chart for a mail access control processing at a secure communication service according to the sixth embodiment of the present invention.

Fig. 38 is a flow chart for an anonymous identification identity judgement processing at a secure communication service according to the sixth embodiment of the present invention.

Fig. 39 is a diagram showing exemplary data structures of data used in the anonymous identification identity judgement processing of Fig. 38.

Fig. 40 is a diagram showing exemplary data structures of an official identification, a link information attached anonymous identification, and a link specifying 1-to-N personalized access ticket according to the seventh embodiment of the present invention.

Fig. 41 is a diagram showing exemplary data structures of a link information attached anonymous identification and an enabler according to the seventh embodiment of the present invention.

Fig. 42 is a first exemplary system configuration that can be used in the seventh embodiment of the present invention.

Fig. 43 is a second exemplary system configuration

that can be used in the seventh embodiment of the present invention.

Fig. 44 is a third exemplary system configuration that can be used in the seventh embodiment of the present invention.

Fig. 45 is a fourth exemplary system configuration that can be used in the seventh embodiment of the present invention.

Fig. 46 is a fifth exemplary system configuration that can be used in the seventh embodiment of the present invention.

Fig. 47 is a sixth exemplary system configuration that can be used in the seventh embodiment of the present invention.

Fig. 48 is a seventh exemplary system configuration that can be used in the seventh embodiment of the present invention.

Fig. 49 is a flow chart for a link specifying anonymous identification list generation processing (for MakePAT, MergePAT, SplitPAT and TransPAT) according to the seventh embodiment of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0143] Referring now to Fig. 1 to Fig. 7, the first embodiment of the email access control scheme according to the present invention will be described in detail.

[0144] The email access control scheme of the present invention enables bidirectional communications between a sender and a recipient appropriately while maintaining anonymity of a sender and a recipient on a communication network. Basically, this is realized by disclosing only information indicative of characteristics of recipients in a state of concealing true identifiers of the recipients, and assigning limited access rights with respect to those who wish to carry out communications while maintaining the anonymity according to the disclosed information.

[0145] More specifically, an Anonymous Identification (abbreviated hereafter as AID) that functions as a role identifier in which a personal information is concealed is assigned to a user, and this AID is disclosed on the network in combination with an information indicative of characteristics of the user such as his/her interests, age, job, etc., which cannot be used in identifying the user on the network but which can be useful for a sender in judging whether or not it is worth communicating with that user.

[0146] Also, the sender can search out a recipient with whom he/she wishes to communicate by reading or searching through the disclosed information. Namely, in the case where the sender wishes to communicate with a recipient while maintaining his/her own anonymity, the sender specifies the AID of that recipient and acquires a Personalized Access Ticket (abbreviated hereafter as PAT). The PAT contains the AIDs of the sender and the recipient as well as information regarding a transfer control flag and a validity period. The transfer control flag is used in order to determine whether a Secure Communication Service (abbreviated hereafter as SCS) to be described below carries out the authentication with respect to the sender. Namely, when the transfer control flag is set ON, the SCS will carry out the authentication such as signature verification for example, with respect to the sender at a time of the connection request. On the other hand, when the transfer control flag is set OFF, the SCS will give the connection request to a physical communication network to which the SCS is connected, without carrying out the authentication. In other words, the transfer control is used in order to verify whether or not the AID is properly utilized by the user to whom it is allocated by a Certification Authority (abbreviated hereafter as CA).

[0147] In the communication network realizing the email access control scheme of the present invention, the assignment of AIDs with respect to users, the maintenance of information disclosed in combination with AIDs, the issuance of PATs, and the email access control based on PATs are realized by separate organizations. This is because it is more convenient to realize them by separate organizations from a perspective of maintaining the security of the entire network, since security levels to be maintained in relation to respective actions are different. Note however that the maintenance of the disclosed information and the issuance of PATs may be realized by the same organization.

[0148] Fig. 1 shows an overall configuration of a communication system in this first embodiment, which is directed to the email service on Internet or Intranet.

[0149] In Fig. 1, the CA (Certification Authority) 1 has a right to authenticate an Official Identification (abbreviated hereafter as OID) that identifies each individual and a right to issue AIDs, and functions to generate AIDs from OIDs and allocate AIDs to users 3.

[0150] The SCS (Secure Communication Service) 5 judges whether or not to admit a connection in response to a connection request by an email from a user 3, according to the PAT (Personalized Access Ticket) presented from a user 3. The SCS 5 also rejects a connection request by an email according to a request from a user 3. The SCS 5 also judges the identity of OIDs according to a request from a user 3.

[0151] An Anonymous Directory Service (abbreviated hereafter as ADS) 7 is a database for managing the AID, the transfer control flag value, the validity period value, and the disclosed information (such as interests, which can be regarded as requiring a lower secrecy compared with a personal information such as name, telephone number, and real email address) of each user 3. The ADS 7 has a function to generate the PAT from the AID of a user 3 who presented search conditions, the AID of a user 3 who has been registering the disclosed information that matches the search conditions

in the ADS 7, the transfer control flag value given from a user 3 or administrators of the ADS, and the validity period value given from a user 3 or administrators of the ADS, and then allocate the PAT to a user 3 who presented the search conditions.

[0152] First, a series of processing from generating the AID from the OID according to a request from a user until allocating the AID to that user will be described.

[0153] Fig. 2 shows exemplary formats of the OID, the AID, and the PAT. As shown in a part (a) of Fig. 2, the OID is an information comprising an arbitrary character string according to a rule by which the CA 1 can uniquely identify the user and a public key, which is signed by the CA 1 using a secret key of the CA 1.

[0154] Also, as shown in a part (b) of Fig. 2, the AID is an information comprising fragments of the OID and their position information, redundant character strings, and an SCS information given by an arbitrary character string (host name, real domain name, etc.) by which a host or a domain that is operating the SCS 5 can be uniquely identified on the network, which is signed by the CA 1 using the secret key of the CA 1.

[0155] Also, as shown in a part (c) of Fig. 2, the PAT is an information comprising the transfer control flag, AID_{ϕ}, AID_1, and the validity period, which is signed by the ADS 7 using a secret key of the ADS 7. Here, the transfer control flag value is defined to take either 0 or 1. Also, the validity period is defined by any one or combination of the number of times for which the PAT is available, the absolute time (UTC) by which the PAT becomes unavailable, the absolute time (UTC) by which the PAT becomes available, and the relative time (lifetime) since the PAT becomes available until it becomes unavailable.

[0156] Note that, as will be explained in the subsequent embodiments described below, in addition to the 1-to-1 PAT which sets one sender and one recipient in correspondence as described above, the present invention can also use a 1-to-N PAT which sets one sender and N recipients, as well as a link specifying PAT which specifies the AID by a link information that is capable of specifying the AID instead of specifying the AID itself in the PAT. The link specifying PAT can be either a link specifying 1-to-1 PAT or a link specifying 1-to-N PAT depending on the correspondence relationship between the sender and the recipients as described above. Namely, the PAT of the present invention can be given in four types: 1-to-1 PAT, 1-to-N PAT, link specifying 1-to-1 PAT, and link specifying 1-to-N PAT.

[0157] Next, a procedure by which the user 3 requests the AID to the CA 1 will be described. The user 3 generates a pair of a secret key and a public key. Then, the user 3 and the CA 1 carries out the bidirectional authentication using the OID of the user 3 and the certificate of the CA 1, and the user 3 transmits the public key to the CA 1 by arbitrary means. Here, there can be cases where communications between the user 3 and the CA 1 are to be encrypted.

[0158] Next, a procedure by which the CA 1 issues the AID to the user 3 in response to a request for the AID as described above will be described. Upon receiving the public key from the user 3, the CA 1 generates the AID. Then, the CA 1 transmits the AID to the user 3 by arbitrary means. Upon receiving the AID from the CA 1, the user 3 stores the received AID into its storage device. Here, there can be cases where communications between the user 3 and the CA 1 are to be encrypted.

[0159] Next, the AID generation processing at the CA will be described with reference to Fig. 3.

[0160] In the procedure of Fig. 3, the CA 1 generates an information of a length equal to the total length L of the OID, and sets this information as a tentative AID (step S911). Then, in order to carry out the partial copying of the OID, values of parameters p_i and ℓ_i for specifying a copying region are determined using arbitrary means such as random number generation respectively (step S913). Here, L is equal to the total length L of the OID, and ℓ_i is an arbitrarily defined value within a range in which a relationship of $0 \leq \ell_i \leq L$ holds. Then, an information in a range between a position p_i to a position $p_i + \ell_i$ from the top of the OID is copied to the same positions in the tentative AID (step S915). In other words, this OID fragment will be copies to a range between a position p_i and a position $p_i + \ell_i$ from the top of the tentative AID. Then, the values of p_i and ℓ_i are written into a prescribed range in the tentative AID into which the OID has been partially copied, in a form encrypted by an arbitrary means (step S917). Then, an SCS information given by an arbitrary character string (host name, real domain, etc.) that can uniquely identify a host or a domain that is operating the SCS 5 on the network is written into a prescribed range in the tentative AID into which these values are written (step S919). Then, the tentative AID into which the above character string is written is signed using a secret key of the CA 1 (step S921).

[0161] Next, a procedure for registering the AID of a user-B 3 and the disclosed information into the ADS 7 will be described. First, the bidirectional authentication by arbitrary means using the AID of the user-B 3 and the certificate of the ADS 7 is carried out between the user-B 3 who is a registrant and the ADS 7. Then, the user-B 3 transmits the transfer control flag value, the validity period value, and the disclosed information such as interests to the ADS 7. Then, the ADS 7 stores the transfer control flag value, the validity period value, and the entire disclosed information in relation to the AID of the user-B 3 in its storage device. Here, there can be cases where communications between the user-B 3 who is the registrant and the ADS 7 are to be encrypted.

[0162] Next, a procedure by which a user-A 3 searches through the disclosed information that is registered in the ADS 7 will be described. First, the bidirectional authentication by arbitrary means using the AID of the user-A 3 and the certificate of the ADS 7 is carried out between the user-A 3 who is a searcher and the

ADS 7. Then, the user-A 3 transmits arbitrary search conditions to the ADS 7. Then, the ADS 7 presents all the received search conditions to its storage device, and extracts the AID of a registrant which satisfies these search conditions. Then, the ADS 7 generates the PAT from the AID of the user-A 3, the AID of the registrant who satisfied all the search conditions, the transfer control flag value, and the validity period value. Then, the ADS 7 transmits the generated PAT to the user-A 3. Here, there can be cases where communications between the user-A 3 who is a searcher and the ADS 7 are to be encrypted. Note that the 1-to-1 PAT is generated as a search result of the ADS 7.

[0163] Next, the 1-to-1 PAT generation processing at the ADS 7 will be described with reference to Fig. 4.

[0164] First, an information of a prescribed length is generated, and this information is set as a tentative PAT (step S1210). Then, the AID of the user-A 3 who is a searcher and the AID of the user-B 3 who is a registrant are copied into a prescribed region of the tentative PAT (step S1215). Then, the transfer control flag value and the validity period value are written into respective prescribed regions of the tentative PAT into which the AIDs are copied (step S1217). Then, the tentative PAT into which these values are written is signed using a secret key of the ADS 7 (step S1219).

[0165] Next, the transfer control using the 1-to-1 PAT will be described. The transfer control is a function for limiting accesses to a user who has a proper access right from a third person to whom the PAT has been transferred or who has eavesdropped the PAT (a user who originally does not have the access right).

[0166] The ADS 7 and the user-B 3 of the registrant AID can prohibit a connection to the user-B 3 from a third person who does not have the access right, by setting a certain value in to the transfer control flag of the PAT.

[0167] When the transfer control flag value is set to be 1, the sender's AID is authenticated between the SCS 5 and the sender according to an arbitrary challenge/response process, so that even if the sender gives both the sender's AID and the PAT to another user other than the sender, that another user will not be able to make a connection to the registrant of the ADS 7 through the SCS 5.

[0168] On the other hand, when the transfer control flag value is set to be 0, no challenge/response process will be carried out between the SCS 5 and the sender, so that if the sender gives both the sender's AID and the PAT to another user other than the sender, that another user will also be able to make a connection to the registrant of the ADS 7 through the SCS 5.

[0169] Next, the email access control method at the SCS 5 will be described with reference to Fig. 5.

[0170] The sender specifies "[sender's AID]@[real domain of SCS of sender]" in From: line, and "[PAT]@[real domain of SCS of sender]" in To: line.

[0171] The SCS 5 acquires a mail received by an MTA (Message Transfer Agent) such as SMTP (Simple Mail Transfer Protocol), and executes the processing of Fig. 5 as follows.

(1) The signature of the PAT is verified using a public key of the ADS 7 (step S1413).

When the PAT is found to have been altered (step S1415 YES), the mail is discarded and the processing is terminated (step S1416).

When the PAT is found to have been not altered (step S1415 NO), the following processing (2) is executed.

(2) The search is carried out by presenting the sender's AID to the PAT (steps S1417, S1419, S1421).

When an AID that completely matches with the sender's AID is not contained in the PAT (step S1423 NO), the mail is discarded and the processing is terminated (step S1416).

When an AID that completely matches with the sender's AID is contained in the PAT (step S1423 YES), the following processing (3) is executed.

(3) The validity period value of the PAT is evaluated (steps S1425, S1427).

When the PAT is outside the validity period (step S1427 NO), the mail is discarded and the processing is terminated (step S1416).

When the PAT is within the validity period (step S1427 YES), the following processing (4) is executed.

(4) Whether or not to authenticate the sender is determined by referring to the transfer control flag value of the PAT (steps S1431, S1433).

When the value is 1 (step S1433 YES), the challenge/response authentication between the SCS 5 and the sender is carried out, and the signature of the sender is verified (step S1435). When the signature is valid, the recipient is specified and the PAT is attached (step S1437). When the signature is invalid, the mail is discarded and the processing is terminated (step S1416).

When the value is 0 (step S1433 NO), the recipient is specified and the PAT is attached without executing the challenge/response authentication (step S1437).

[0172] Next, an exemplary challenge/response authentication between the SCS 5 and the sender will be described.

[0173] First, the SCS 5 generates an arbitrary information such as a timestamp, for example, and transmits the generated information to the sender.

[0174] Then, the sender signs the received information using a secret key of the sender's AID and transmits it along with a public key of the sender's AID.

[0175] The SCS 5 then verifies the signature of the received information using the public key of the sender's AID. When the signature is valid, the recipient is speci-

fied and the PAT is attached. When the signature is invalid, the mail is discarded and the processing is terminated.

[0176] Next, a method for specifying the recipient at the SCS 5 will be described. First, the SCS 5 carries out the search by presenting the sender's AID to the PAT, so as to acquire all the AIDs which do not completely match the sender's AID. All these acquired AIDs will be defined as recipient's AIDs hereafter. Then, for every recipient's AID, the real domain of SCS of recipient is taken out from the recipient's AID. Then, the recipient is specified in a format of "[recipient's AID]@[real domain of SCS of recipient]". Finally, the SCS 5 changes the sender from a format of "[sender's AID]@[real domain of SCS of sender]" to a format of "sender's AID".

[0177] Next, a method for attaching the PAT at the SCS 5 will be described. The SCS 5 attaches the PAT to an arbitrary position in the mail. The SCS 5 gives the mail to the MTA after specifying the sender and the recipient and attaching the PAT.

[0178] Note that all the processings described above are the same in the case of the 1-to-N PAT.

[0179] Next, a method of receiving refusal with respect to the PAT at the SCS 5 will be described.

[0180] Receiving refusal setting: The bidirectional authentication is carried out by an arbitrary means between the user and the SCS 5. Then, the user transmits a registration command, his/her own AID, and arbitrary PATs to the SCS 5. Then, the SCS 5 verifies the signature of the received AID. If the signature is invalid, the processing of the SCS 5 is terminated. If the signature is valid, the SCS 5 next verifies the signature of each received PAT using a public key of the ADS. Those PATs with the invalid signature are discarded by the SCS 5. When the signature is valid, the SCS 5 carries out the search by presenting the received AID to each PAT. For each of those PATs which contain the AID that completely matches with the received AID, the SCS 5 presents the registration command and the PAT to the storage device such that the PAT is registered into the storage device. Those PATs which do not contain the AID that completely matches with the received AID are discarded by the SCS 5 without storing them into the storage device. Here, there can be cases where communications between the user and the SCS 5 are to be encrypted.

[0181] Receiving refusal execution: The SCS 5 carries out the search by presenting the PAT to the storage device. When a PAT that completely matches the presented PAT is registered in the storage device, the mail is discarded. When a PAT that completely matches the present PAT is not registered in the storage device, the mail is not discarded.

[0182] Receiving refusal cancellation: The bidirectional authentication is carried out by an arbitrary means between the user and the SCS 5. Then, the user presents his/her own AID to the SCS 5. Then, the SCS 5 verifies the signature of the received AID. If the signature is invalid, the processing of the SCS 5 is terminated. If the signature is valid, the SCS 5 next presents the presented AID as a search condition to the storage device and acquire all the PATs that contain the presented AID, and then presents all the acquired PATs to the user. Then, the user selects all the PATs for which the receiving refusal is to be cancelled by referring to all the PATs presented from the SCS 5, and transmits all the selected PATs along with a deletion command to the SCS 5. Upon receiving the deletion command and all the PATs for which the receiving refusal is to be cancelled, the SCS 5 presents the deletion command and all the PATs received from the user to the storage device, such that all the received PATs are deleted from the storage device.

[0183] Note that the method of receiving refusal with respect to the 1-to-N PAT at the SCS 5 is the same as the method of receiving refusal with respect to the 1-to-1 PAT described above.

[0184] Note also the the case of returning of a mail from the user-B to the user-A is the same as in the case of transmitting a mail from the user-A to the user-B.

[0185] Next, the judgement of identity will be described with reference to Fig. 6 and Fig. 7.

(1) An initial value of a variable OID_M is defined as a bit sequence with a length equal to the total length L of the OID and all values equal to "0". Also, an initial value of a variable OID_V is defined as a bit sequence with a length equal to the total length of the OID and all values equal to "0" (step S2511).

(2) One AID is selected from a set of processing target AIDs, and the following bit processing is carried out (step S2513).

(a) Values of variables AID_M and AID_V are determined according to the position information contained in the AID (step S2515). Here, AID_M is defined as a bit sequence with a length equal to the total length L of the OID and a value of a position at which the OID information is defined is "1" while a value of a position at which the OID information is not defined is "0" (see Fig. 7). Also, AID_V is defined as a bit sequence with a length equal to the total length L of the OID and a value of a position at which the OID information is defined is an actual value of the OID information while a value of a position at which the OID information is not defined is 0 (see Fig. 7).

(b) AND processing of OID_M and AID_M is carried out and its result is substituted into a variable OVR_M (step S2517).

(c) AND processing of OVR_M and AID_M as well as AND processing of OVR_M and OID_M are carried out and their results are compared (step S2519). When they coincide, OR processing of OID_M and AID_M is carried out

and its result is substituted into OID_M (step S2521), while OR processing of OID_V and AID_V is also carried out and its result is substituted into OID_M (step S2523). On the other hand, when they do not coincide, the processing proceeds to the step S2525.

(d) An AID to be processed next is selected from a set of processing target AIDs. When at least one another AID is contained in the set, the steps S2513 to S2523 are executed for that another AID. When no other AID is contained in the set, the processing proceeds to the step S2527.

(e) Values of OID_M and OID_V are outputted (step S2527).

[0186] The value of OID_M that is eventually obtained indicates all positions of the OID information that can be recovered from the set of processing target AIDs. Also, the value of OID_V that is eventually obtained indicates all the OID information that can be recovered from the set of processing target AID. In other words, by using the values of OID_M and OID_V, it is possible to obtain the OID albeit probabilistically when the value of OID_V is used as a search condition, and it is possible to quantitatively evaluate a precision of the above search by a ratio OID_M/L with respect to the total length L of the OID.

[0187] As described above, in this first embodiment, the CA 1 which is a Trusted Third Party with high secrecy and credibility generates the AID in which the personal information is concealed, from the OID that contains the highly secret personal information such as name, telephone number, real email address, etc., according to a user request, and issues the AID to the user. By identifying the user by this AID on the communication network as well as in various services provided on the communication network, it becomes possible to provide both the anonymity guarantee and the identity guarantee for the user. In other words, it becomes possible for the user to communicate with another user without revealing the own real name, telephone number, email address, etc., to that another user, and it also becomes possible to disclose the disclosed information to unspecified many through the ADS 7 as will be described below.

[0188] The user registers the disclosed information, that is an information which is supposed to have a low secrecy compared with the personal information at the ADS 7. In the case of searching the disclosed information and the registrant AID, the searcher presents the AID of the searcher and arbitrary search conditions to the ADS 7. The ADS 7 then extracts the registrant AID that satisfies these search conditions, and generates the PAT from the AID of the searcher and the AID of the registrant who satisfied the search conditions, the transfer control flag value, and the validity period value.

[0189] In this 1-to-1 PAT, the transfer control flag value and the validity period value are set as shown a part (c) of Fig. 2, and by setting up this validity period in advance, it is possible to limit connections from the sender.

[0190] It is also possible to prohibit connections from a third person who does not have the access right, by using the transfer control flag value. Namely, when the transfer control flag value is set to be 1, the sender's AID is authenticated between the SCS 5 and the sender according to an arbitrary challenge/response process, so that even if the sender gives both the sender's AID and the PAT to another user other than the sender, that another user will not be able to make a connection to the registrant of the ADS 7 through the SCS 5. On the other hand, when the transfer control flag value is set to be 0, no challenge/response process will be carried out between the SCS 5 and the sender, so that if the sender gives both the sender's AID and the PAT to another user other than the sender, that another user will also be able to make a connection to the registrant of the ADS 7 through the SCS 5.

[0191] It is also possible to make a connection request to the communication network such that a call for which the recipient is specified by the 1-to-1 PAT will be received by the recipient's AID or the sender's AID defined within the PAT. In addition, it is also possible to refuse receiving calls with the 1-to-1 PAT selected by the recipient among calls which are specified by the 1-to-1 PAT. It is also possible to cancel the receiving refusal of the calls with the 1-to-1 PAT selected by the recipient. In addition, as a measure against the sender who repeats the personal attach using a plurality of sender's AIDs by taking an advantage of the anonymity, it is possible to judge the identity of the OID from these plurality of sender's AIDs and it is possible to extract that OID at some probability.

[0192] Next, with references to Fig. 8 to Fig. 24, the second embodiment of the email access control scheme according to the present invention will be described in detail.

[0193] In contrast to the first embodiment described above which is directed to the case where a sender and a recipient are set in 1-to-1 correspondence, this second embodiment is directed to the case where a sender and recipients are set in 1-to-N correspondence and a generation of a new PAT and a content change of the existing PAT can be made by the initiative of a user. Here, the sender is either a holder of the PAT or a member of the PAT. Similarly, the recipient is either a holder of the PAT or a member of the PAT.

[0194] In general, a membership of a group communication (mailing list, etc.) is changing dynamically so that it is necessary for a host of the group communication to manage information on a point of contact such as telephone number, email address, etc., of each member. In contrast, in the case where it is only possible to newly generate a 1-to-1 PAT as in the first embodiment, the management of a point of contact is difficult. For example, it is difficult to manage the group collectively, and

even if it is given to the others for the purpose of the transfer control, it does not function as an address of the group communication such as mailing list.

[0195] In this second embodiment, in order to resolve such a problem, it is made possible to carry out a generation of a new 1-to-N PAT and a content change of the existing 1-to-N PAT by the initiative of a user.

[0196] First, the definition of various identifications used in this second embodiment will be described with references to Fig. 8 and Fig. 9.

[0197] As shown in a part (a) of Fig. 8, the OID is an information comprising an arbitrary character string (telephone number, email address, etc.) according to a rule by which the CA 1 can uniquely identify the user and a public key, which is signed by the CA 1.

[0198] Also, as shown in a part (b) of Fig. 8, the AID is an information comprising fragments of the OID and their position information, redundant character strings, and an SCS information given by an arbitrary character string (host name, real domain name, etc.) by which a host or a domain that is operating the SCS 5 can be uniquely identified on the network, which is signed by the CA 1.

[0199] Also, as shown in a part (c) of Fig. 8, the 1-to-N PAT is an information comprising two or more AIDs, a holder index, the validity period, the transfer control flag, and a PAT processing device identifier, which is signed using a secret key of the PAT processing device.

[0200] Here, one of the AIDs is a holder AID of this PAT, where the change of the information contained in the PAT such as an addition of AID to the PAT, a deletion of AID from the PAT, a change of the validity period in the PAT, a change of the transfer control flag value in the PAT, etc., can be made by presenting the holder AID and a corresponding Enabler to the PAT processing device.

[0201] On the other hand, the AIDs other than the holder AID that are contained in the PAT are all member AIDs, where a change of the information contained in the PAT cannot be made even when the member AID and a corresponding Enabler are presented to the PAT processing device.

[0202] The holder index is a numerical data for identifying the holder AID, which is defined to take a value 1 when the holder AID is a top AID in the AID list formed from the holder AID and the member AIDs, a value 2 when the holder AID is a second AID from the top of the AID list, or a value n when the holder AID is an n-th AID from the top of the AID list.

[0203] The transfer control flag value is defined to take either 0 or 1 similarly as in the case of the 1-to-1 PAT.

[0204] The holder AID is defined to be an AID which is written at a position of the holder index value in the AID list. The member AIDs are defined to be all the AIDs other than the holder AID.

[0205] The validity period is defined by any one or combination of the number of times for which the PAT is available, the absolute time (UTC) by which the PAT becomes unavailable, the absolute time (UTC) by which the PAT becomes available, and the relative time (lifetime) since the PAT becomes available until it becomes unavailable.

[0206] The identifier of a PAT processing device (or a PAT processing object on the network) is defined as a serial number of the PAT processing device (or an distinguished name of the PAT processing object on the network). The secret key of the PAT processing device (or the PAT processing object on the network) is defined to be uniquely corresponding to the identifier.

[0207] Also, in this second embodiment, an Enabler is introduced as an identifier corresponding to the AID. As shown in Fig. 9, the Enabler is an information comprising a character string uniquely indicating that it is an Enabler and an AID itself, which is signed by the CA 1.

[0208] Next, the operations for a generation of a new PAT and a content change of the existing PAT will be described. Here, the following operations are defined at a secure PAT processing device on the communication terminal or a PAT processing object on the CA or on a network which is properly requested from the CA (which will also be referred to as a PAT processing device hereafter).

1. Editing of AID list:
   A list of AIDs (referred hereafter as an AID list) contained in the PAT is edited using AIDs and Enabler. Else, the AID list is newly generated.
2. Setting of the validity period and the transfer control flag:
   The validity period value and the transfer control flag value contained in the PAT are changed using an AID and Enabler. Also, a new validity period value and a new transfer control flag value are set in the newly generated AID list.

[0209] A user who presented the holder AID and the Enabler corresponding to this holder AID to the PAT processing device can edit the list of AIDs contained in the PAT. In this case, the following processing rules are used.

(1) Generating a new PAT (MakePAT) (see Fig. 10):
The AID list (ALIST<holder AID | member AID_1, member AID_2, ・・・・・・・・・, member AID_n>) is newly generated, and the validity period value and the transfer control flag value are set with respect to the generated ALIST.

AID_A + AID_B + Enabler of AID_B + Enabler of AID_A

→ ALIST<AID_A | AID_B>

ALIST<AID_A | AID_B > + Enabler of AID_A

+ validity period value

+ transfer control flag value

→ PAT<AID_A | AID_B >

(2) Merging PATs (MergePAT) (see Fig. 11):

A plurality of ALISTs of the same holder AID are merged and the validity period value and the transfer control flag value are set with respect to the merged ALIST.

ALIST<AID_A | AID_{B1}, AID_{B2}, · · · · · · · · >

+ ALIST<AID_A | AID_{C1}, AID_{C2}, · · · · · · · · >

+ Enabler of AID_A

→ ALIST<AID_A | AID_{B1}, AID_{B2}, · · · · · · · · , AID_{C1}, AID_{C2}, · · · · · · · · >

ALIST<AID_A | AID_{B1}, AID_{B2}, · · · · · · · · , AID_{C1}, AID_{C2}, · · · · · · · · >

+ Enabler of AID_A + validity period value

+ transfer control flag value

→ PAT<AID_A | AID_{B1}, AID_{B2}, · · · · · · · · , AID_{C1}, AID_{C2}, · · · · · · · · >

(3) Splitting a PAT (SplitPAT) (see Fig. 12):

The ALIST is split into a plurality of ALISTs of the same holder AID, and the respective validity period value and transfer control flag value are set with respect to each one of the split ALISTs.

ALIST<AID_A | AID_{B1}, AID_{B2}, · · · · · · · · , AID_{C1}, AID_{C2}, · · · · · · · · >

+ Enabler of AID_A

→ ALIST<AID_A | AID_{B1}, AID_{B2}, · · · · · · · · >

+ ALIST<AID_A | AID_{C1}, AID_{C2}, · · · · · · · · >

ALIST<AID_A | AID_{C1}, AID_{C2}, · · · · · · · · >

+ Enabler of AID_A + validity period value

+ transfer control flag value

→ PAT<AID_A | AID_{C1}, AID_{C2}, · · · · · · · · >

(4) Changing a holder of a PAT (TransPAT) (see Fig. 13):

The holder AID of the ALIST is changed, and the validity period value and the transfer control flag value are set with respect to the changed ALIST.

ALIST<AID_A | AID_B > + ALIST<AID_A | AID_{C1}, AID_{C2}, · · · · · · · · >

+ Enabler of AID_A + Enabler of AID_B

→ ALIST<AID_B | AID_{C1}, AID_{C2}, · · · · · · · · >

ALIST<AID_B | AID_{C1}, AID_{C2}, · · · · · · · · >

+ Enabler of AID_B + validity period value

+ transfer control flag value

→ PAT<AID_B | AID_{C1}, AID_{C2}, · · · · · · · · >

[0210] In the operation for setting the validity period value, in order to permit the setting of the validity period value only to a user who holds both the holder AID and the corresponding Enabler, the following operation is defined.

PAT<AID_A | AID_B > + Enabler of AID_A

+ validity period value

→ PAT<AID_A | AID_B >

[0211] In the operation for setting the transfer control flag value, in order to permit the setting of the transfer control flag value only to a user who holds both the holder AID and the corresponding Enabler, the following operation is defined.

PAT<AID_A | AID_B > + Enabler of AID_A

+ transfer control flag value

→ PAT<AID_A | AID_B >

[0212] Next, with references to Fig. 14 to Fig. 20, the overall system configuration of this second embodiment will be described. In Fig. 14 to Fig. 20, the user-A who has AID_A allocated from the CA stores AID_A and Enabler of AID_A in a computer of the user-A, and the input/output devices such as floppy disk drive, CD-ROM drive, communication board, microphone, speaker, etc., are connected. Else, AID_A and Enabler of AID_A are stored in a communication terminal (telephone, cellular phone, etc.) which has a storage device and a data input/output function.

[0213] Similarly, the user-B who has AID_B allocated from the CA stores AID_B and Enabler of AID_B in a computer of the user-B, and the input/output devices such as floppy disk drive, CD-ROM drive, communication board, microphone, speaker, etc., are connected. Else, AID_B and Enabler of AID_B are stored in a communication terminal (telephone, cellular phone, etc.) which has

a storage device and a data input/output function.

[0214] In the following, a procedure by which the user-A generates PAT<AID_A | AID_B > will be described.

(1) The user-A acquires AID_B and Enabler of AID_B using any of the following means.

- AID_B and Enabler of AID_B are registered at the ADS 7, and it is waited until the user-A acquires them as a search result (Fig. 14).
- AID_B and Enabler of AID_B are directly transmitted to the user-A by the email, signaling, etc. (Figs. 15, 16).
- AID_B and Enabler of AID_B are stored in a magnetic, optic, or electronic medium such as floppy disk, CD-ROM, MO, IC card, etc., and this medium is given to the user-A. Else, it is waited until the user acquires them by reading this medium (Figs. 17, 18).
- AID_B and Enabler of AID_B are printed on a paper medium such as book, name card, etc., and this medium is given to the user-A. Else, it is waited until the user-A acquire them by reading this medium (Figs. 19, 20).

(2) The user-A who has acquired AID_B and Enabler of AID_B by any of the means described in the above (1) issues the MakePAT command to the PAT processing device. This procedure is common to Fig. 14 to Fig. 20, and defined as follows.

(a) The user-A requests the issuance of the MakePAT command by setting AID_A, Enabler of AID_A, AID_B, Enabler of AID_B, the validity period value, and the transfer control flag value into the communication terminal of the user-A.
(b) The communication terminal of the user-A generates the MakePAT command.
(c) The communication terminal of the user-A transmits the generated MakePAT command to the PAT processing device by means such as the email, signaling, etc. (the issuance of the MakePAT command).
(d) The PAT processing device generates PAT<AID_A | AID_B > by processing the received MakePAT command according to Fig. 21 and Fig. 23. More specifically, this is done as follows.

AID_A + AID_B + Enabler of AID_B + Enabler of AID_A

→ ALIST<AID_A | AID_B >

ALIST<AID_A | AID_B > + Enabler of AID_A

+ validity period value + transfer control flag value

→ PAT<AID_A | AID_B >

(e) The PAT processing device transmits the generated PAT<AID_A | AID_B > to the communication terminal of the user-A, or to the communication terminal of the user-B according to the need, by means such as the email, signaling, etc.
(f) The communication terminal of the user-A (or the user-B) stores the received PAT<AID_A | AID_B > in the storage device of the communication terminal of the user-A.

[0215] The merging of PATs (MergePAT, Fig. 21, Fig. 23), the splitting of a PAT (SplitPAT, Fig. 22, Fig. 23), and the changing of a holder of a PAT (TransPAT, Fig. 21, Fig. 23) are also carried out by the similar procedure.

[0216] Next, the procedure of MakePAT, MergePAT and TransPAT will be described with reference to Fig. 21.

(1) The holder AID is specified (step S4411).
(2) All the member AIDs are specified (step S4412).
(3) The AID list is generated from the specified holder AID and all the specified member AIDs (step S4413). More specifically, the specified holder AID and all the specified member AIDs are concatenated using arbitrary means.
(4) A tentative PAT is generated using arbitrary means, similarly as in the case of a tentative AID (step S4414).
(5) The generated AID list is copied to a prescribed region of the generated tentative PAT (step S4415).
(6) The holder index value is written into the tentative pat to which the AID list has been copied (step S4416).
(7) The transfer control flag value is written into the tentative PAT into which the holder index value has been written (step S4417).
(8) The validity period value is written into the tentative PAT into which the transfer control flag value has been written (step S4418).
(9) The PAT processing device identifier is written into the tentative PAT into which the validity period value has been written (step S4419).
(10) The tentative PAT into which the PAT processing device identifier has been written is signed using the secret key of the PAT processing device (step S4420).

[0217] Next, the procedure of SplitPAT will be described with reference to Fig. 22.

(1) The holder AID is specified (step S4511).
(2) All the AIDs to be the member AIDs of the PATs after the splitting are specified (step S4512).
(3) The AID list is generated from the specified holder AID and all the specified member AIDs (step

S4513). More specifically, the specified holder AID and all the specified member AIDs are concatenated using arbitrary means.

(4) A tentative PAT is generated using arbitrary means, similarly as in the case of a tentative AID (step S4514).

(5) The generated AID list is copied to a prescribed region of the generated tentative PAT (step S4515).

(6) The holder index value is written into the tentative pat to which the AID list has been copied (step S4516).

(7) The transfer control flag value is written into the tentative PAT into which the holder index value has been written (step S4517).

(8) The validity period value is written into the tentative PAT into which the transfer control flag value has been written (step S4518).

(9) The PAT processing device identifier is written into the tentative PAT into which the validity period value has been written (step S4519).

(10) The tentative PAT into which the PAT processing device identifier has been written is signed using the secret key of the PAT processing device (step S4520).

(11) In the case of continuing the splitting (step S4521 YES), the procedure returns to (2), and repeats (2) to (10) sequentially.

[0218] Note that, in the procedures of Fig. 21 and Fig. 22, the AID list generation is carried out according to Fig. 23 as follows. Namely, a buffer length is determined first (step S4611) and a buffer is generated (step S4612). Then, the holder AID is copied to a vacant region of the generated buffer (step S4613). Then, the member AID is copied to a vacant region of the resulting buffer (step S4614), and if the next member AID exists (step S4615 YES), the step S4614 is repeated.

[0219] Next, the determination of the holder AID will be described. Each of the MakePAT, the MergePAT, the SplitPAT, and the TransPAT commands is defined to have two or more arguments, where AID, PAT, or Enabler can be specified as an argument. In this case, the PAT processing device specifies the holder AID of the PAT to be outputted after executing each command according to the following rules.

* Case of the MakePAT:

  For the MakePAT command, it is defined that AIDs are to be specified for the first argument to the N-th argument (N = 2, 3, ·····) and Enablers are to be specified for the N+1-th and subsequent arguments. For example, they can be specified as follows.

  MakePAT AID_1, AID_2, ·····, AID_N, Enabler of AID_1, Enabler of AID_2, Enabler of AID_N

  The PAT processing device interprets the AID of the first argument of the MakePAT command as the holder AID.

  Only when one of the Enablers of the N+1-th and subsequent arguments corresponds to the AID of the first argument, the PAT processing device specifies this AID (that is the AID of the first argument) as the holder AID of the PAT to be outputted after executing the MakePAT command.

* Case of the MergePAT:

  For the MergePAT command, it is defined that PATs are to be specified for the first argument to the N-th argument (N = 2, 3, ·····) and Enabler is to be specified for the N+1-th argument. Namely, they can be specified as follows.

  MergePAT PAT_1 PAT_2 ····· PAT_N Enabler of AID

  The PAT processing device interprets the holder AID of the PAT of the first argument of the MergePAT command as the holder AID of the PAT to be outputted after executing the MergePAT command.

  Only when the Enabler of the N+1-th argument corresponds to the holder AID of the PAT of the first argument, the PAT processing device specifies this AID (that is the holder AID of the PAT of the first argument) as the holder AID of the PAT to be outputted after executing the MergePAT command.

* Case of the SplitPAT:

  For the SplitPAT command, it is defined that PAT is to be specified for the first argument, a set of one or more AIDs grouped together by some prescribed symbols (assumed to be parentheses () in this example) are to be specified for the second argument to the N-th argument (N = 3, 4, ·····), and Enabler is to be specified for the N+1-th argument. Namely, they can be specified as follows.

  SplitPAT PAT_1 (AID_{11}) $(AID_{21}$ $AID_{22})$ ····· $(AID_{N1}$ AID_{N2} ····· $AID_{NM})$ Enabler of AID

  The PAT processing device interprets the holder AID of the PAT of the first argument of the SplitPAT command as the holder AID of the PAT to be outputted after executing the SplitPAT command.

  Only when the Enabler of the N+1-th argument corresponds to the holder AID of the PAT of the first argument, the PAT processing device specifies this AID (that is the holder AID of the PAT of the first argument) as the holder AID of the PAT to be outputted after executing the SplitPAT command.

* Case of the TransPAT:

  For the TransPAT command, it is defined that

PATs are to be specified for the first argument and the second argument, AID is to be specified for the third argument, and Enablers are to be specified for the fourth argument and the fifth argument. Namely, they can be specified as follows.

TransPAT PAT_1 PAT_2 AID Enabler of AID_1 Enabler of AID_2

The PAT processing device interprets the AID of the third argument as the holder AID of the PAT to be outputted after executing the TransPAT command provided that the AID of the third argument of the TransPAT command is contained in the PAT of the second argument.

Only when the Enabler of the fourth argument corresponds to both the PAT of the first argument and the PAT of the second argument and the Enabler of the fifth argument corresponds to the AID of the third argument, the PAT processing device specifies the AID of the third argument as the holder AID of the PAT to be outputted after executing the TransPAT command.

Next, the determination of the member AIDs will be described. The definitions of the MakePAT, the MergePAT, the SplitPAT, and the TransPAT commands are as described above. The PAT processing device specifies the member AIDs of the PAT to be outputted after executing each command according to the following rules.

* Case of the MakePAT:

Only when the holder AID of the PAT to be outputted after executing the MakePAT command is formally determined, the PAT processing device interprets all the AIDs of the second and subsequent arguments of the MakePAT command as the member AIDs of the PAT to be outputted after executing the MakePAT command.

The PAT processing device specifies only those AIDs among all the AIDs of the second and subsequent arguments which correspond to the Enablers specified by the N+1-th and subsequent arguments as the member AIDs of the PAT to be outputted after executing the MakePAT command.

* Case of the MergePAT:

Only when the holder AID of the PAT to be outputted after executing the MergePAT command is formally determined, the PAT processing device specifies the member AIDs of all the PATs specified by the first to N-th arguments of the MergePAT as the member AIDs of the PAT to be outputted after executing the MergePAT command.

* Case of the SplitPAT:

Only when the holder AID of the PAT to be outputted after executing the SplitPAT command is formally determined, the PAT processing device specifies the member AID of the PAT specified by the first argument of the SplitPAT command as the member AID of the PAT to be outputted after executing the SplitPAT command. At this point, the member AIDs are distributed into different PATs in units of parentheses (). For example, in the case of:

SplitPAT PAT (AID_{11}) (AID_{21} AID_{22}) ..... (AID_{N1} AID_{N2} ..... AID_{NM}) Enabler of AID

(AID_{11}), (AID_{21} AID_{22}) and (AID_{N1} AID_{N2} ..... AID_{NM}) will be the member AIDs of different PATs having a common holder AID.

* Case of TransPAT:

Only when the holder AID of the PAT to be outputted after executing the TransPAT command is formally determined, the PAT processing device specifies all the member AIDs remaining after excluding the member AID that is scheduled to be a new holder AID from all the member AIDs of the PAT specified by the first argument of the TransPAT command and the member AIDs of the PAT specified by the second argument as the member AIDs of the PAT to be outputted after executing the TransPAT command.

[0220] Next, the verification of the properness of the Enabler will be described. This verification of the properness of the Enabler is common to the MakePAT, the MergePAT, the SplitPAT and the TransPAT, and carried out according to Fig. 24 as follows.

(1) AID and Enabler are entered (step S5511).
(2) Each of these entered AID and Enabler is verified using the public key of the CA 1 (step S5512). If at least one of them is altered (step S5513 YES), the processing is terminated.
(3) A character string for certifying that it is Enabler is entered (step S5514).
(4) The top field of the Enabler of the step S5511 and the character string of the step S5514 are compared (step S5515). If they do not match (step S5516 NO), the processing is terminated.
(5) If they match (step S5516 YES), the AID of the step S5511 and the AID within the Enabler are compared (step S5517).
(6) A comparison result is outputted (step S5519).

[0221] Next, with references to Fig. 25 to Fig. 28, the third embodiment of the email access control scheme according to the present invention will be described in detail.

[0222] In the generation of a new PAT (MakePAT) and the PAT holder change (TransPAT) of the above described embodiment, it is necessary to give member AIDs and Enablers of member AIDs to the holder of the PAT, but when they are given to the holder, it becomes possible for that holder to participate the group communications hosted by the other holders by using the

acquired member AIDs. Namely, there arises a problem that the pretending using the member AIDs become possible. Moreover, if that holder places the acquired member AIDs and Enablers of member AIDs on a medium that is readable by unspecified many, these member AIDs become accessible to anyone so that there arises a problem that the harassment to the users of the member AIDs may occur and the pretending using the member AIDS by a third person also become possible.

[0223] For this reason, in this third embodiment, it is made possible to carry out the MakePAT and the Trans-PAT without giving the Enablers of member AIDs to the holder.

[0224] To this end, in this third embodiment, the generation of a new PAT and the content change of the existing PAT are carried out by using Null-AID (AID_{Null}) and Enabler of Null-AID (Enabler of AID_{Null}).

[0225] Here, the processing involving the Null-AID obeys all of the following rules:

(a) the processing rules of MakePAT, MergePAT, SplitPAT and TransPAT as in the above described embodiment; and
(b) the rules applicable only to the Null-AID, including:

(i) Null-AID is known to every user, and
(ii) Enabler of Null-AID is known to every user.

[0226] Here, the processing rules as defined in the above described embodiment in the case of this third embodiment will be described.

(1) Making a PAT from plural AIDs (MakePAT):

$AID_{holder} + AID_{member1} + AID_{member2} + \cdots\cdots\cdots + AID_{memberN}$

+ Enabler of $AID_{member1}$ + Enabler of $AID_{member2}$ + $\cdots\cdots\cdots$

+ Enabler of $AID_{memberN}$ + Enabler of AID_{holder}

$\rightarrow$ PAT<AID_{holder} | $AID_{member1}$, $AID_{member2}$, $\cdots\cdots\cdots$, $AID_{memberN}$ >

(2) Merging plural PATs of the same holder (MergePAT):

PAT<AID_{holder} | $AID_{membera1}$, $AID_{membera2}$, $\cdots\cdots\cdots$, $AID_{memberaM}$ >

+ PAT<AID_{holder} | $AID_{memberb1}$, $AID_{memberb2}$, $\cdots\cdots\cdots$, $AID_{memberbN}$ >

+ Enabler of AID_{holder}

$\rightarrow$ PAT<AID_{holder} | $AID_{membera1}$, $AID_{membera2}$, $\cdots\cdots\cdots$, $AID_{memberaM}$, $AID_{memberb1}$, $AID_{memberb2}$, $\cdots\cdots\cdots$, $AID_{memberbN}$ >

(3) Splitting a PAT into plural PATs of the same holder (SplitPAT):

PAT<AID_{holder} | $AID_{membera1}$, $AID_{membera2}$, $\cdots\cdots\cdots$, $AID_{memberaM}$, $AID_{memberb1}$, $AID_{memberb2}$, $\cdots\cdots\cdots$, $AID_{memberbN}$ >

+ Enabler of AID_{holder}

$\rightarrow$ PAT<AID_{holder} | $AID_{membera1}$, $AID_{membera2}$, $\cdots\cdots\cdots$, $AID_{memberaM}$ >

+ PAT<AID_{holder} | $AID_{memberb1}$, $AID_{memberb2}$, $\cdots\cdots\cdots$, $AID_{memberbN}$ >

(4) Changing a holder AID of a PAT (TransPAT):

PAT<AID_{holder} | $AID_{membera1}$, $AID_{membera2}$, $\cdots\cdots\cdots$, $AID_{memberaM}$ > + PAT<AID_{holder} | $AID_{newholder}$ >

+ Enabler of AID_{holder} + Enabler of $AID_{newholder}$

$\rightarrow$ PAT<$AID_{newholder}$ | $AID_{membera1}$, $AID_{membera2}$, $\cdots\cdots\cdots$, $AID_{memberaM}$ >

[0227] The method for specifying the validity period value and the transfer control flag value in the PAT containing the Null-AID is similar to the method for specifying the validity period value and the transfer control flag value in the second embodiment described above. Next, the exemplary processings involving the Null-AID will be described.

(1) Case of producing PAT<AID_{Null} | AID_A > from AID_A and Enabler of AID_A:

(a) According to the above described rules (b)(i) and (b)(ii) of the Null-AID, AID_{Null} and Enabler of AID_{Null} are known.
(b) Using MakePAT,

$AID_{Null} + AID_A$ + Enabler of AID_A + Enabler of AID_{Null}

$\rightarrow$ PAT<AID_{Null} | AID_A >.

(2) Case of producing PAT<AID_{Null} | AID_A, AID_B > from PAT<AID_{Null} | AID_A > and PAT<AID_{Null} | AID_B >:

(a) According to the above described rules (b)(i) and (b)(ii) of the Null-AID, AID_{Null} and Enabler of AID_{Null} are known.

(b) Using MergePAT,

$PAT<AID_{Null} \mid AID_A> + PAT<AID_{Null} \mid AID_B>$

+ Enabler of AID_{Null}

$\rightarrow PAT<AID_{Null} \mid AID_A, AID_B>$.

(3) Case of producing $PAT<AID_A \mid AID_B>$ from $PAT<AID_{Null} \mid AID_A>$, $PAT<AID_{Null} \mid AID_B>$ and Enabler of AID_A:

(a) According to the above described rules (b)(i) and (b)(ii) of the Null-AID, AID_{Null} and Enabler of AID_{Null} are known.
(b) Using TransPAT,

$PAT<AID_{Null} \mid AID_A> + PAT<AID_{Null} \mid AID_B>$

+ Enabler of AID_{Null} + Enabler of AID_A

$\rightarrow PAT<AID_A \mid AID_B>$.

[0228] As shown in Fig. 25, the data structure of the Null-AID comprises a character string uniquely indicating that it is Null-AID (a character string defined by the CA, for example), which is signed by the CA using the secret key of the CA.

[0229] Also, as shown in Fig. 26, the data structure of the Enabler of Null-AID comprises a character string uniquely indicating that it is Enabler (a character string defined by the CA, for example) and the Null-AID itself, which is signed by the CA using the secret key of the CA.

[0230] Note that the Null-AID and the Enabler of Null-AID are maintained at secure PAT processing devices and secure PAT certification authority.

[0231] Next, the first exemplary application of this third embodiment will be described with reference to Fig. 27, which includes the following operations.

(1) The user-B (PAT member) generates $PAT<AID_{Null} \mid AID_B>$ by executing the above described exemplary processing (1) involving the Null-AID at the secure PAT processing device which is connected with the terminal of the user-B, and gives it to the user-A (PAT holder) by arbitrary means.
(2) The user-A who received $PAT<AID_{Null} \mid AID_B>$ carries out the following operations at the secure PAT processing device which is connected with the terminal of the user-A.

(a) $PAT<AID_{Null} \mid AID_A>$ is produced by executing the above described exemplary processing (1) involving the Null-AID.
(b) $PAT<AID_A \mid AID_B>$ is produced by executing the above described exemplary processing (3) involving the Null-AID.

(3) The user-A gives the generated $PAT<AID_A \mid AID_B>$ to the user-B by arbitrary means.

[0232] Note that the method for determining the validity period is the same as described above so that it will not be repeated here. Also, the processing involving the Null-AID is the same as described above so that it will not be repeated here.

[0233] In the case of giving $PAT<AID_{Null} \mid AID_A, AID_B>$ to the user-B, the above described exemplary processing (2) involving the Null-AID will be executed in the operation (2) described above.

[0234] Next, the second exemplary application of this third embodiment will be described with reference to Fig. 28, which includes the following operations.

(1) The user-B (PAT member) produces $PAT<AID_{Null} \mid AID_B>$ by executing the above described exemplary processing (1) involving the Null-AID at the secure PAT processing device which is connected with the terminal of the user-B, and registers it along arbitrary disclosed information at the ADS.
(2) The user-A produces $PAT<AID_{Null} \mid AID_A>$ by executing the above described exemplary processing (1) involving the Null-AID at the secure PAT processing device which is connected with the terminal of the user-A, and presents it along arbitrary search conditions to the ADS.
(3) When the personal information of the user-B satisfies the search conditions presented by the user-A, the secure PAT processing device connected with the ADS carries out the following operations.

(a) $PAT<AID_{Null} \mid AID_A, AID_B>$ is produced by executing the above described exemplary processing (2) involving the Null-AID.
(b) The produced $PAT<AID_{Null} \mid AID_A, AID_B>$ is given to the ADS.

(4) The ADS gives $PAT<AID_{Null} \mid AID_A, AID_B>$ produced by the PAT processing device to the user-A.
(5) The user-A who received $PAT<AID_{Null} \mid AID_A, AID_B>$ produces $PAT<AID_A \mid AID_B>$ by executing the following TransPAT processing at the secure PAT processing device which is connected with the terminal of the user-A.

$PAT<AID_{Null} \mid AID_A> + PAT<AID_{Null} \mid AID_A, AID_B>$

+ Enabler of AID_{Null} + Enabler of AID_A

$\rightarrow PAT<AID_A \mid AID_B>$.

[0235] Note that the method for determining the validity period is the same as described above so that it will not be repeated here. Also, the processing involving the Null-AID is the same as described above so that it will not be repeated here.

[0236] In the case of generating PAT<AID_A | AID_B > at the PAT processing device connected with the ADS, Enabler of AID_A will be given to that PAT processing device, and the above described exemplary processing (3) involving the Null-AID will be executed in the operation (3) described above.

[0237] In the case of generating PAT<AID_B | AID_A > at the PAT processing device connected with the ADS and giving it to the user-B, Enabler of AID_B will be given to that PAT processing device, and the above described exemplary processing (3) involving the Null-AID will be executed in the operation (3) described above.

[0238] Next, with references to Fig. 29 to Fig. 31, the fourth embodiment of the email access control scheme according to the present invention will be described in detail.

[0239] In the group communication, a situation where it is desired to fix the participants is frequently encountered, but the above described embodiment does not have a function for making it impossible to change the PAT so that the participants cannot be fixed. Namely, in the above described embodiment, whether or not to fix the participants is left to the judgement of the holder of the PAT.

[0240] For this reason, in this fourth embodiment, a read only attribute is set up in the PAT. More specifically, in this fourth embodiment, the read only attribute is set up in the PAT by using God-AID (AID_{God}).

[0241] Here, the processing involving the God-AID obeys all of the following rules:

(a) God-AID is known to every user, and
(b) the processing involving God-AID is allowed only in the following cases:

(i) a case where the AID_{holder} is neither AID_{Null} nor AID_{God}:

PAT<AID_{holder} | $AID_{member1}$, $AID_{member2}$, ·····, $AID_{memberN}$ > + Enabler of AID_{holder}

→ PAT<AID_{god} | AID_{holder}, $AID_{member1}$, $AID_{member2}$, ·····, $AID_{memberN}$ >

(ii) a case where AID_{holder} is AID_{Null}:

PAT<AID_{Null} | $AID_{member1}$, $AID_{member2}$, ·····, $AID_{memberN}$ >

+ Enabler of AID_{Null}

→ PAT<AID_{god} | $AID_{member1}$, $AID_{member2}$, ·····, $AID_{memberN}$ >

[0242] As shown in Fig. 29, the data structure of the God-AID comprises a character string uniquely indicating that it is God-AID (a character string defined by the CA, for example), which is signed by the CA using the secret key of the CA. The God-AID is maintained at the secure PAT processing devices and the secure PAT certification authority described above.

[0243] The processings of a PAT that contains the Null-AID are according to Fig. 21 to Fig. 24. When the holder AID is neither Null-AID nor God-AID, the God-AID is appended to the AID list and the holder index value is specified to be a position of the God-AID in the AID list after appending the God-AID. When the holder AID is Null-AID, the Null-AID is deleted from the AID list, the God-AID is appended to the AID list, and then the holder index value is specified to be a position of the God-AID in the AID list after appending the God-AID.

[0244] Next, the exemplary application of this fourth embodiment will be described with reference to Fig. 30.

[0245] In the case of producing PAT<AID_{God} | AID_A, AID_B > from PAT<AID_{Null} | AID_A > and PAT<AID_{Null} | AID_B >, the following processing is executed at the secure PAT processing device which is connected with the terminal of the PAT holder (user-A in Fig. 30).

(1) Using MergePAT,

PAT<AID_{Null} | AID_A > + PAT<AID_{Null} | AID_B >

+ Enabler of AID_{Null}

→ PAT<AID_{Null} | AID_A, AID_B >.

(2) According to the above described rule (a) of the God-AID, AID_{God} is known.
(3) According to the above described rule (b)(ii) of the God-AID,

PAT<AID_{Null} | AID_A, AID_B > + Enabler of AID_{Null}

→ PAT<AID_{god} | AID_A, AID_B >

[0246] The above processing is also executed at the secure PAT processing device connected with a computer (search engine, etc.) of the third person (Fig. 31) or at the secure PAT certification authority.

[0247] Next, with reference to Fig. 32, the fifth embodiment of the email access control scheme according to the present invention will be described in detail.

[0248] When the Null-AID is added as described in the third embodiment, there arises a problem that it becomes possible for the holder of the PAT (the user of the holder AID) to transfer the access right with respect to the member (the user of the member AID) to the third person, and moreover this transfer can be done without a permission of the member, as will be described now.

(1) The holder-A of PAT<AID_A | AID_B > (for the member-B) produces PAT<AID_{Null} | AID_B > by using PAT<AID_A | AID_B >, AID_A and Enabler of AID_A. Here, it is assumed that the holder-A knows all of AID_A, Enabler of AID_A, AID_{Null}, and Enabler of AID_{Null} in addition to PAT<AID_A | AID_B >.

(a) The holder-A produces PAT<AID_A | AID_{Null} > using the MakePAT as follows.

AID_A + AID_{Null} + Enabler of AID_{Null} + Enabler of AID_A

→ PAT<AID_A | AID_{Null} >

(b) The holder-A produces PAT<AID_{Null} | AID_B > using the TransPAT as follows.

PAT<AID_A | AID_B > + PAT<AID_A | AID_{Null} >

+ Enabler of AID_A + Enabler of AID_{Null}

→ PAT<AID_{Null} | AID_B >

After the above described operation (1)(b), the holder-A gives PAT<AID_{Null} | AID_B > to the third person-C, the following operation (2) becomes possible.

(2) The third person-C produces PAT<AID_C | AID_B > by using PAT<AID_{Null} | AID_B >. Here, it is assumed that the third person-C knows all of AID_C, Enabler of AID_C, AID_{Null}, and Enabler of AID_{Null} in addition to PAT<AID_{Null} | AID_B >.

(a) The third person-C produces PAT<AID_{Null} | AID_C > using the MakePAT as follows.

AID_{Null} + AID_C + Enabler of AID_C + Enabler of AID_{Null}

→ PAT<AID_{Null} | AID_C >

(b) The third person-C produces PAT<AID_C | AID_B > using the TransPAT as follows.

PAT<AID_{Null} | AID_B > + PAT<AID_{Null} | AID_C >

+ Enabler of AID_{Null} + Enabler of AID_C

→ PAT<AID_C | AID_B >

[0249] As a result of the above described operation (2)(b), the third person-C obtains PAT<AID_C | AID_B > so that accesses to the member-B become possible.

[0250] For this reason, in this fifth embodiment, it is made impossible for the holder of PAT<AID_{holder} | AID_{member} > to produce PAT<AID_{Null} | AID_{member} > from this PAT<AID_{holder} | AID_{member} > as long as the holder does not know Enabler of AID_{member}.

[0251] In the third embodiment described above, in order for the PAT holder to produce PAT<AID_{Null} | AID_{member} > without using Enabler of AID_{member}, it is necessary to produce PAT<AID_{holder} | AID_{Null} >.

[0252] To this end, in this fifth embodiment, for the Null-AID described in the third embodiment, the following rule is added:

* the Null-AID can be used only as the holder AID of the PAT (the Null-AID cannot be used as the member AID).

That is, PAT<AID_{Null} | $AID_{member1}$, $AID_{member2}$, ·····, $AID_{memberN}$ > is allowed, but PAT<AID_{holder} | AID_{Null}, $AID_{member1}$, $AID_{member2}$, ·····, $AID_{memberN}$ > is not allowed.

Each of the secure PAT processing devices and the secure PAT certification authority is additionally equipped with a function for checking whether the Null-AID is contained as the member AID or not. This member AID checking processing is carried out according to Fig. 32 as follows.

(1) Null-AID and PAT are entered (step S6911).
(2) All the member AIDs are taken out from the PAT entered at the step S6911 (step S6913).
(3) Each of the taken out member AIDs is compared with the Null-AID entered at the step S6911 (step S6915).

If all the member AIDs do not completely match with the Null-AID (step S6917 NO, step S6919 NO), the processing proceeds to the MergePAT, SplitPAT or TransPAT processing (Fig. 21 or Fig. 22) (step S6921).

If there is a member AID that completely matches with the Null-AID (step S6917 YES), the processing is terminated.

[0253] Next, with reference to Fig. 33 to Fig. 39, the sixth embodiment of the email access control scheme according to the present invention will be described in detail.

[0254] This sixth embodiment differs from the first embodiment described above in that a link information is added to the AID of Fig. 2 used in the first embodiment, as shown in a part (b) of Fig. 34, while a link information of the AID is set instead of the AID itself that is contained in the 1-to-1 PAT of Fig. 2, as shown in a part (c) of Fig. 34, such that the AID is uniquely identified by the link information.

[0255] Note that such an AID to which the link information is added will be referred to as a link information attached AID, and a 1-to-1 PAT having the link information of the AID will be referred to as a link specifying 1-to-1 PAT. Also, the link information is an information

capable of uniquely identifying the AID, which is given by a kind of data generally known as identifier such as a serial number uniquely assigned to the AID by the CA for example.

[0256] Fig. 33 shows an overall configuration of a communication system in this sixth embodiment.

[0257] In Fig. 33, the CA (Certification Authority) 1 has a right to authenticate OIDs and a right to issue AIDs, and functions to allocate AIDs to users 3.

[0258] The SCS (Secure Communication Service) 5 transfers emails among the users 3, carries out the receiving refusal and the identity judgement and the extraction of the OID according to the need.

[0259] The ADS (Anonymous Directory Service) 7 is a database for managing the AID, the transfer control flag value, the validity period value, and the disclosed information of each user 3. The ADS 7 has a function to generate the PAT from the AID of a searcher and the AID of a registrant who satisfies the search conditions, and issue it to the searcher.

[0260] A series of processing from generating the AID from the OID according to a request from a user until allocating the AID to that user is basically the same as in the first embodiment, except that the link information is to be added, which will now be described with reference to Fig. 34.

[0261] Fig. 34 shows exemplary formats of the OID, the link information attached AID, and the link specifying 1-to-1 PAT. As shown in a part (a) of Fig. 34, the OID is an information comprising an arbitrary character string according to a rule by which the CA 1 can uniquely identify the user and a public key, which is signed by the CA 1.

[0262] Also, as shown in a part (b) of Fig. 34, the link information attached AID is an information comprising fragments of the OID and their position information, redundant character strings, an SCS information given by an arbitrary character string (host name, real domain name, etc.) by which a host or a domain that is operating the SCS 5 can be uniquely identified on the network, and the link information, which is signed by the CA 1.

[0263] Also, as shown in a part (c) of Fig. 34, the link specifying 1-to-1 PAT is an information comprising the transfer control flag, the link information of AID_ϕ, the link information of AID_1, and the validity period, which is signed by the ADS 7 using a secret key of the ADS 7.

[0264] A procedure by which the user 3 requests the link information attached AID to the CA 1 is the same as that of the first embodiment. A procedure by which the CA 1 issues the link information attached AID to the user 3 in response to a request for the AID is also the same as that of the first embodiment.

[0265] Next, the link information attached AID generation processing at the CA will be described with reference to Fig. 35.

[0266] In the procedure of Fig. 35, the CA 1 generates an information of a length equal to the total length L of the OID, and sets this information as a tentative AID (step S7211). Then, in order to carry out the partial copying of the OID, values of parameters p_i and ℓ_i for specifying a copying region are determined using arbitrary means such as random number generation respectively (step S7213). Here, L is equal to the total length L of the OID, and ℓ_i is an arbitrarily defined value within a range in which a relationship of $0 \leq \ell_i \leq L$ holds. Then, an information in a range between a position p_i to a position $p_i + \ell_i$ from the top of the OID is copied to the same positions in the tentative AID (step S7215). In other words, this OID fragment will be copies to a range between a position p_i and a position $p_i + \ell_i$ from the top of the tentative AID. Then, the values of p_i and ℓ_i are written into a prescribed range in the tentative AID into which the OID has been partially copied, in a form encrypted by an arbitrary means (step S7217). Then, an SCS information given by an arbitrary character string (host name, real domain, etc.) that can uniquely identify a host or a domain that is operating the SCS 5 on the network is written into a prescribed range in the tentative AID into which these values are written (step S7219). Then, the link information is written (step S7220). Then, the tentative AID into which the above character string and the link information are written is signed using a secret key of the CA 1 (step S7221).

[0267] Next, a procedure for registering the AID of a user-B 3 and the disclosed information into the ADS 7 will be described. First, the bidirectional authentication by arbitrary means using the AID of the user-B 3 and the certificate of the ADS 7 is carried out between the user-B 3 who is a registrant and the ADS 7. Then, the user-B 3 transmits the transfer control flag value, the validity period value, and the disclosed information such as interests to the ADS 7. Then, the ADS 7 stores the transfer control flag value, the validity period value, and the entire disclosed information in relation to the AID of the user-B 3 in its storage device. Here, there can be cases where communications between the user-B 3 who is the registrant and the ADS 7 are to be encrypted.

[0268] Next, a procedure by which a user-A 3 searches through the disclosed information that is registered in the ADS 7 will be described. First, the bidirectional authentication by arbitrary means using the AID of the user-A 3 and the certificate of the ADS 7 is carried out between the user-A 3 who is a searcher and the ADS 7. Then, the user-A 3 transmits arbitrary search conditions to the ADS 7. Then, the ADS 7 presents all the received search conditions to its storage device, and extracts the AID of a registrant which satisfies these search conditions. Then, the ADS 7 generates the link specifying 1-to-1 PAT from the link information of the AID of the user-A 3 and the link information of the AID of the registrant who satisfied the search conditions, the transfer control flag value, and the validity period value. Then, the ADS 7 transmits the generated PAT to the user-A 3. Here, there can be cases where communications between the user-A 3 who is a searcher and the ADS 7 are to be encrypted. Note that the link specifying

1-to-1 PAT is generated as a search result of the ADS 7.

[0269] Next, the link specifying 1-to-1 PAT generation processing at the ADS 7 will be described with reference to Fig. 36.

[0270] First, an information of a prescribed length is generated, and this information is set as a tentative PAT (step S7510). Then, the link information of the AID of the user-A 3 who is a searcher and the link information of the AID of the user-B 3 who is a registrant are copied into a prescribed region of the tentative PAT (step S7516). Then, the transfer control flag value and the validity period value are written into respective prescribed regions of the tentative PAT into which the link informations of the AIDs are copied (step S7517). Then, the tentative PAT into which these values are written is signed using a secret key of the ADS 7 (step S7519).

[0271] Next, the transfer control using the link specifying 1-to-1 PAT will be described. The transfer control is a function for limiting accesses to a user who has a proper access right from a third person to whom the PAT has been transferred or who has eavesdropped the PAT (a user who originally does not have the access right).

[0272] The ADS 7 and the user-B 3 of the registrant AID can prohibit a connection to the user-B 3 from a third person who does not have the access right, by setting a certain value in to the transfer control flag of the PAT.

[0273] When the transfer control flag value is set to be 1, the sender's AID is authenticated between the SCS 5 and the sender according to an arbitrary challenge/response process, so that even if the sender gives both the sender's AID and the PAT to another user other than the sender, that another user will not be able to make a connection to the registrant of the ADS 7 through the SCS 5.

[0274] On the other hand, when the transfer control flag value is set to be 0, no challenge/response process will be carried out between the SCS 5 and the sender, so that if the sender gives both the sender's AID and the PAT to another user other than the sender, that another user will also be able to make a connection to the registrant of the ADS 7 through the SCS 5.

[0275] Next, the email access control method at the SCS 5 will be described with reference to Fig. 37.

[0276] The sender specifies "[sender's AID]@[real domain of SCS of sender]" in From: line, and "[PAT]@[real domain of SCS of sender]" in To: line.

[0277] The SCS 5 acquires a mail received by an MTA (Message Transfer Agent) such as SMTP (Simple Mail Transfer Protocol), and executes the processing of Fig. 37 as follows.

(1) The signature of the PAT is verified using a public key of the ADS 7 (step S7713).

When the PAT is found to have been altered (step S7715 YES), the mail is discarded and the processing is terminated (step S7716).

When the PAT is found to have been not altered (step S7715 NO), the following processing (2) is executed.

(2) The search is carried out by presenting the link information of the sender's AID to the PAT (steps S7717, S7720, S7722).

When a link information that completely matches with the link information of the sender's AID is not contained in the PAT (step S7723 NO), the mail is discarded and the processing is terminated (step S7716).

When a link information that completely matches with the link information of the sender's AID is contained in the PAT (step S7723 YES), the following processing (3) is executed.

(3) The validity period value of the PAT is evaluated (steps S7725, S7727).

When the PAT is outside the validity period (step S7727 NO), the mail is discarded and the processing is terminated (step S7716).

When the PAT is within the validity period (step S7727 YES), the following processing (4) is executed.

(4) Whether or not to authenticate the sender is determined by referring to the transfer control flag value of the PAT (steps S7731, S7733).

When the value is 1 (step S7733 YES), the SCS 5 acquires the sender's AID itself and the public key of the sender's AID by presenting the link information to the CA 1, and then the challenge/response authentication between the SCS 5 and the sender is carried out, and the signature of the sender is verified (step S7735). When the signature is valid, the recipient is specified and the PAT is attached (step S7737). When the signature is invalid, the mail is discarded and the processing is terminated (step S7716).

When the value is 0 (step S7733 NO), the recipient is specified and the PAT is attached without executing the challenge/response authentication (step S7737).

[0278] The challenge/response authentication between the SCS 5 and the sender is the same as that for the 1-to-1 PAT described above.

[0279] Next, a method for specifying the recipient at the SCS 5 will be described. First, the SCS 5 carries out the search by presenting the link information of the sender's AID to the PAT, so as to acquire all the link informations which do not completely match the link information of the sender's AID. Then, the search is carried out by presenting all these acquired link informations to the CA 1 so as to acquire the AIDs. All these acquired AIDs will be defined as recipient's AIDs hereafter. Then, for every recipient's AID, the real domain of SCS of recipient is taken out from the recipient's AID. Then, the recipient is specified in a format of "[recipient's AID]@[real domain of SCS of recipient]". Finally, the SCS 5 changes the sender from a format of

"[sender's AID]@[real domain of SCS of sender]" to a format of "sender's AID".

[0280] The method for attaching the PAT at the SCS 5 is the same as that for the 1-to-1 PAT described above.

[0281] Next, a method of receiving refusal with respect to the PAT at the SCS 5 will be described.

[0282] Receiving refusal setting: The bidirectional authentication is carried out by an arbitrary means between the user and the SCS 5. Then, the user transmits a registration command, his/her own AID, and arbitrary PATs to the SCS 5. Then, the SCS 5 verifies the signature of the received AID. If the signature is invalid, the processing of the SCS 5 is terminated. If the signature is valid, the SCS 5 next verifies the signature of each received PAT using a public key of the ADS. Those PATs with the invalid signature are discarded by the SCS 5. When the signature is valid, the SCS 5 takes out the link information from the received AID, and then carries out the search by presenting the taken out link information to each PAT. For each of those PATs which contain the link information that completely matches with the link information of the received AID, the SCS 5 presents the registration command and the PAT to the storage device such that the PAT is registered into the storage device. Those PATs which do not contain the link information that completely matches with the link information of the received AID are discarded by the SCS 5 without storing them into the storage device. Here, there can be cases where communications between the user and the SCS 5 are to be encrypted.

[0283] Receiving refusal execution: The SCS 5 carries out the search by presenting the PAT to the storage device. When a PAT that completely matches the presented PAT is registered in the storage device, the mail is discarded. When a PAT that completely matches the present PAT is not registered in the storage device, the mail is not discarded.

[0284] Receiving refusal cancellation: The bidirectional authentication is carried out by an arbitrary means between the user and the SCS 5. Then, the user presents his/her own AID to the SCS 5. Then, the SCS 5 verifies the signature of the received AID. If the signature is invalid, the processing of the SCS 5 is terminated. If the signature is valid, the SCS 5 next takes out the link information from the presented AID, and presents the taken out link information as a search condition to the storage device and acquire all the PATs that contain the presented link information, and then presents all the acquired PATs to the user. Then, the user selects all the PATs for which the receiving refusal is to be cancelled by referring to all the PATs presented from the SCS 5, and transmits all the selected PATs along with a deletion command to the SCS 5. Upon receiving the deletion command and all the PATs for which the receiving refusal is to be cancelled, the SCS 5 presents the deletion command and all the PATs received from the user to the storage device, such that all the received PATs are deleted from the storage device.

[0285] Note that the method of receiving refusal with respect to the link specifying 1-to-N PAT at the SCS 5 is the same as the method of receiving refusal with respect to the link specifying 1-to-1 PAT described above.

[0286] Next, the judgement of identity will be described with reference to Fig. 38 and Fig. 39.

(1) An initial value of a variable OID_M is defined as a bit sequence with a length equal to the total length L of the OID and all values equal to "0". Also, an initial value of a variable OID_V is defined as a bit sequence with a length equal to the total length of the OID and all values equal to "0" (step S7911).

(2) One link information attached AID is selected from a set of processing target link information attached AIDs, and the following bit processing is carried out (step S7913).

(a) Values of variables AID_M and AID_V are determined according to the position information contained in the link information attached AID (step S7915). Here, AID_M is defined as a bit sequence with a length equal to the total length L of the OID and a value of a position at which the OID information is defined is "1" while a value of a position at which the OID information is not defined is "0" (see Fig. 39). Also, AID_V is defined as a bit sequence with a length equal to the total length L of the OID and a value of a position at which the OID information is defined is an actual value of the OID information while a value of a position at which the OID information is not defined is 0 (see Fig. 39).

(b) AND processing of OID_M and AID_M is carried out and its result is substituted into a variable OVR_M (step S7917).

(c) AND processing of OVR_M and AID_M as well as AND processing of OVR_M and OID_M are carried out and their results are compared (step S7919). When they coincide, OR processing of OID_M and AID_M is carried out and its result is substituted into OID_M (step S7921), while OR processing of OID_V and AID_V is also carried out and its result is substituted into OID_M (step S7923). On the other hand, when they do not coincide, the processing proceeds to the step S7925.

(d) A link information attached AID to be processed next is selected from a set of processing target link information attached AIDs. When at least one another link information attached AID is contained in the set, the steps S7913 to S7923 are executed for that another link information attached AID. When no other link information attached AID is contained in the set, the

processing proceeds to the step S7927.

(e) Values of OID_M and OID_V are outputted (step S7927).

[0287] The value of OID_M that is eventually obtained indicates all positions of the OID information that can be recovered from the set of processing target link information attached AIDs. Also, the value of OID_V that is eventually obtained indicates all the OID information that can be recovered from the set of processing target link information attached AID. In other words, by using the values of OID_M and OID_V, it is possible to obtain the OID albeit probabilistically when the value of OID_V is used as a search condition, and it is possible to quantitatively evaluate a precision of the above search by a ratio OID_M/L with respect to the total length L of the OID.

[0288] As described above, in this sixth embodiment, the CA 1 which is a Trusted Third Party with high secrecy and credibility generates the link information attached AID in which the personal information is concealed, from the OID that contains the highly secret personal information such as name, telephone number, real email address, etc., according to a user request, and issues the AID to the user. By identifying the user by this AID on the communication network as well as in various services provided on the communication network, it becomes possible to provide both the anonymity guarantee and the identity guarantee for the user. In other words, it becomes possible for the user to communicate with another user without revealing the own real name, telephone number, email address, etc., to that another user, and it also becomes possible to disclose the disclosed information to unspecified many through the ADS 7 as will be described below.

[0289] The user registers the disclosed information, that is an information which is supposed to have a low secrecy compared with the personal information at the ADS 7. In the case of searching the disclosed information and the registrant AID, the searcher presents the link information attached AID of the searcher and arbitrary search conditions to the ADS 7. The ADS 7 then extracts the registrant link information attached AID that satisfies these search conditions, and generates the link specifying 1-to-1 PAT from the link information of the AID of the searcher and the link information of the AID of the registrant who satisfied the search conditions, the transfer control flag value, and the validity period value.

[0290] In this link specifying 1-to-1 PAT, the transfer control flag value and the validity period value are set as shown a part (c) of Fig. 34, and by setting up this validity period in advance, it is possible to limit connections from the sender.

[0291] It is also possible to prohibit connections from a third person who does not have the access right, by using the transfer control flag value. Namely, when the transfer control flag value is set to be 1, the sender's AID is authenticated between the SCS 5 and the sender according to an arbitrary challenge/response process, so that even if the sender gives both the sender's AID and the PAT to another user other than the sender, that another user will not be able to make a connection to the registrant of the ADS 7 through the SCS 5. On the other hand, when the transfer control flag value is set to be 0, no challenge/response process will be carried out between the SCS 5 and the sender, so that if the sender gives both the sender's AID and the PAT to another user other than the sender, that another user will also be able to make a connection to the registrant of the ADS 7 through the SCS 5.

[0292] It is also possible to make a connection request to the communication network such that a call for which the recipient is specified by the link specifying 1-to-1 PAT will be received by the recipient's AID or the sender's AID specified by the link information of the link specifying 1-to-1 PAT. In addition, it is also possible to refuse receiving calls with the link specifying 1-to-1 PAT selected by the recipient among calls which are specified by the link specifying 1-to-1 PAT. It is also possible to cancel the receiving refusal of the calls with the link specifying 1-to-1 PAT selected by the recipient. In addition, as a measure against the sender who repeats the personal attack using a plurality of sender's AIDs by taking an advantage of the anonymity, it is possible to judge the identity of the OID from these plurality of sender's AIDs and it is possible to extract that OID at some probability.

[0293] Next, with references to Fig. 40 to Fig. 49, the seventh embodiment of the email access control scheme according to the present invention will be described in detail.

[0294] In contrast to the sixth embodiment described above which is directed to the case where a sender and a recipient are set in 1-to-1 correspondence, this seventh embodiment is directed to the case where a sender and recipients are set in 1-to-N correspondence and a generation of a new link specifying 1-to-N PAT and a content change of the existing link specifying 1-to-N PAT can be made by the initiative of a user, similarly as in the second embodiment described above. Here, the sender is either a holder of the PAT or a member of the PAT. Similarly, the recipient is either a holder of the PAT or a member of the PAT.

[0295] As described in the second embodiment, in general, a membership of a group communication (mailing list, etc.) is changing dynamically so that it is necessary for a host of the group communication to manage information on a point of contact such as telephone number, email address, etc., of each member. In contrast, in the case where it is possible to newly generate a 1-to-1 PAT as in the sixth embodiment, the management of a point of contact is difficult. For example, it is difficult to manage the group collectively, and even if it is given to the others for the purpose of the transfer control, it does not function as an address of the group communication such as mailing list.

[0296] In this seventh embodiment, in order to resolve

such a problem, it is made possible to carry out a generation of a new link specifying 1-to-N PAT and a content change or the existing link specifying 1-to-N PAT by the initiative of a user.

[0297] First, the definition of various identifications used in this seventh embodiment will be described with references to Fig. 40 and Fig. 41.

[0298] As shown in a part (a) of Fig. 40, the OID is an information comprising an arbitrary character string (telephone number, email address, etc.) according to a rule by which the CA 1 can uniquely identify the user and a public key, which is signed by the CA 1.

[0299] Also, as shown in a part (b) of Fig. 40, the link information attached AID is an information comprising fragments of the OID and their position information, redundant character strings, an SCS information given by an arbitrary character string (host name, real domain name, etc.) by which a host or a domain that is operating the SCS 5 can be uniquely identified on the network, and a link information, which is signed by the CA 1. Note that the AID may be encrypted at the SCS 5 or the CA 1. The link information is the same as in the sixth embodiment.

[0300] Also, as shown in a part (c) of Fig. 40, the link specifying 1-to-N PAT is an information comprising two or more link informations of AIDs, a holder index, the validity period, the transfer control flag, and a PAT processing device identifier, which is signed using a secret key of the PAT processing device.

[0301] Here, one of the link informations of AIDs is the link information of the holder AID of this PAT, where the change of the information contained in the PAT such as an addition of the link information of AID to the PAT, a deletion of the link information of AID from the PAT, a change of the validity period in the PAT, a change of the transfer control flag value in the PAT, etc., can be made by presenting the link information of the holder AID and a corresponding Enabler to the PAT processing device.

[0302] On the other hand, the link informations of AIDs other than the link information of the holder AID that are contained in the PAT are all link information of member AIDs, where a change of the information contained in the PAT cannot be made even when the link information of the member AID and a corresponding Enabler are presented to the PAT processing device.

[0303] The holder index is a numerical data for identifying the link information of the holder AID, which is defined to take a value 1 when the link information of the holder AID is a top link information of AID in the link specifying AID list formed from the link information of the holder AID and the link informations of the member AIDs, a value 2 when the link information of the holder AID is a second link information of AID from the top of the link specifying AID list, or a value n when the link information of the holder AID is an n-th link information of AID from the top of the link specifying AID list.

[0304] The transfer control flag value is defined to take either 0 or 1 similarly as in the case of the link specifying 1-to-1 PAT.

[0305] The link information of the holder AID is defined to be a link information of AID which is written at a position of the holder index value in the link specifying AID list. The link informations of the member AIDs are defined to be all the link informations of AIDs other than the link information of the holder AID.

[0306] The validity period is defined by any one or combination of the number of times for which the PAT is available, the absolute time (UTC) by which the PAT becomes unavailable, the absolute time (UTC) by which the PAT becomes available, and the relative time (lifetime) since the PAT becomes available until it becomes unavailable.

[0307] The identifier of a PAT processing device (or a PAT processing object on the network) is defined as a serial number of the PAT processing device (or an distinguished name of the PAT processing object on the network). The secret key of the PAT processing device (or the PAT processing object on the network) is defined to be uniquely corresponding to the identifier.

[0308] Also, in this second embodiment, an Enabler is introduced as an identifier corresponding to the AID. As shown in Fig. 41, the Enabler is an information comprising a character string uniquely indicating that it is an Enabler and a link information attached AID itself, which is signed by the CA 1.

[0309] Next, the operations for a generation of a new PAT and a content change of the existing PAT will be described. Here, the following operations are defined at a secure PAT processing device on the communication terminal or a PAT processing object on the CA or on a network which is properly requested from the CA (which will also be referred to as a PAT processing device hereafter). These operations are similar to those of the second embodiment described above so that they will be described by referring to Fig. 10 to Fig. 13 but it is assumed that each occurrence of AID in Fig. 10 to Fig. 13 should be replaced by the link information of AID in the following.

1. Editing of link specifying AID list:

A link specifying AID list, which is a list of link informations of AIDs contained in the PAT, is edited using link information attached AIDs and Enabler. Else, the link specifying AID list is newly generated.

2. Setting of the validity period and the transfer control flag:

The validity period value and the transfer control flag value contained in the PAT are changed using a link information attached AID and Enabler. Also, a new validity period value and a new transfer control flag value are set in the newly generated link specifying AID list.

[0310] A user who presented the holder AID and the Enabler corresponding to this holder AID to the PAT processing device can edit the list of link informations of

AIDs contained in the PAT. In this case, the following processing rules are used.

(1) Generating a new PAT (MakePAT) (see Fig. 10):

The link specifying AID list (LALIST<(link)holder AID | (link)member AID_1, (link)member AID_2, ·····, (link)member AID_n >) where (link)AID_x denotes the link information of AID_x is newly generated, and the validity period value and the transfer control flag value are set with respect to the generated LALIST.

(link)AID_A + (link)AID_B + Enabler of AID_B

+ Enabler of AID_A

→ LALIST<(link)AID_A | (link)AID_B >

LALIST<(link)AID_A | (link)AID_B > + Enabler of AID_A

+ validity period value

+ transfer control flag value

→ PAT<(link)AID_A | (link)AID_B >

(2) Merging PATs (MergePAT) (see Fig. 11):

A plurality of LALISTs of the same holder AID are merged and the validity period value and the transfer control flag value are set with respect to the merged LALIST.

LALIST<(link)AID_A | (link)AID_{B1}, (link)AID_{B2}, ····· >

+ LALIST<(link)AID_A | (link)AID_{C1}, (link)AID_{C2}, ····· >

+ Enabler of AID_A

→ LALIST<(link)AID_A | (link)AID_{B1}, (link)AID_{B2}, ·····, (link)AID_{C1}, (link)AID_{C2}, ····· >

LALIST<(link)AID_A | (link)AID_{B1}, (link)AID_{B2}, ·····, (link)AID_{C1}, (link)AID_{C2}, ····· >

+ Enabler of AID_A + validity period value

+ transfer control flag value

→ PAT<(link)AID_A | (link)AID_{B1}, (link)AID_{B2}, ·····, (link)AID_{C1}, (link)AID_{C2}, ····· >

(3) Splitting a PAT (SplitPAT) (see Fig. 12):

The LALIST is split into a plurality of LALISTs of the same holder AID, and the respective validity period value and transfer control flag value are set with respect to each one of the split LALISTs.

LALIST<(link)AID_A | (link)AID_{B1}, (link)AID_{B2}, ·····, (link)AID_{C1}, (link)AID_{C2}, ····· >

+ Enabler of AID_A

→ LALIST<(link)AID_A | (link)AID_{B1}, (link)AID_{B2}, ····· >

+ LALIST<(link)AID_A | (link)AID_{C1}, (link)AID_{C2}, ····· >

LALIST<(link)AID_A | (link)AID_{C1}, (link)AID_{C2}, ····· >

+ Enabler of AID_A + validity period value

+ transfer control flag value

→ PAT<(link)AID_A | (link)AID_{C1}, (link)AID_{C2}, ····· >

(4) Changing a holder of a PAT (TransPAT) (see Fig. 13):

The holder AID of the LALIST is changed, and the validity period value and the transfer control flag value are set with respect to the changed LALIST.

LALIST<(link)AID_A | (link)AID_B >

+ LALIST<(link)AID_A | (link)AID_{C1}, (link)AID_{C2}, ····· >

+ Enabler of AID_A + Enabler of AID_B

→ LALIST<(link)AID_B | (link)AID_{C1}, (link)AID_{C2}, ····· >

LALIST<(link)AID_B | (link)AID_{C1}, (link)AID_{C2}, ····· >

+ Enabler of AID_B + validity period value

+ transfer control flag value

→ PAT<(link)AID_B | (link)AID_{C1}, (link)AID_{C2}, ····· >

[0311] In the operation for setting the validity period value, in order to permit the setting of the validity period value only to a user who holds both the holder AID and the corresponding Enabler, the following operation is defined.

PAT<(link)AID_A | (link)AID_B > + Enabler of AID_A

+ validity period value

→ PAT<(link)AID_A | (link)AID_B >

[0312] In the operation for setting the transfer control flag value, in order to permit the setting of the transfer control flag value only to a user who holds both the holder AID and the corresponding Enabler, the following operation is defined.

PAT<(link)AID_A | (link)AID_B > + Enabler of AID_A

+ transfer control flag value

→ PAT<(link)AID_A | (link)AID_B >

[0313] Next, with references to Fig. 42 to Fig. 48, the overall system configuration of this seventh embodiment will be described. In Fig. 42 to Fig. 48, the user-A who has AID_A allocated from the CA stores AID_A and Enabler of AID_A in a computer of the user-A, and the input/output devices such as floppy disk drive, CD-ROM drive, communication board, microphone, speaker, etc., are connected. Else, AID_A and Enabler of AID_A are stored in a communication terminal (telephone, cellular phone, etc.) which has a storage device and a data input/output function.

[0314] Similarly, the user-B who has AID_B allocated from the CA stores AID_B and Enabler of AID_B in a computer of the user-B, and the input/output devices such as floppy disk drive, CD-ROM drive, communication board, microphone, speaker, etc., are connected. Else, AID_B and Enabler of AID_B are stored in a communication terminal (telephone, cellular phone, etc.) which has a storage device and a data input/output function.

[0315] In the following, a procedure by which the user-A generates PAT<(link)AID_A | (link)AID_B > will be described.

(1) The user-A acquires AID_B and Enabler of AID_B using any of the following means.

- AID_B and Enabler of AID_B are registered at the ADS 7, and it is waited until the user-A acquires them as a search result (Fig. 42).
- AID_B and Enabler of AID_B are directly transmitted to the user-A by the email, signaling, etc. (Figs. 43, 44).
- AID_B and Enabler of AID_B are stored in a magnetic, optic, or electronic medium such as floppy disk, CD-ROM, MO, IC card, etc., and this medium is given to the user-A. Else, it is waited until the user acquires them by reading this medium (Figs. 45, 46).
- AID_B and Enabler of AID_B are printed on a paper medium such as book, name card, etc., and this medium is given to the user-A. Else, it is waited until the user-A acquire them by reading this medium (Figs. 47, 48).

(2) The user-A who has acquired AID_B and Enabler of AID_B by any of the means described in the above (1) issues the MakePAT command to the PAT processing device. This procedure is common to Fig. 42 to Fig. 48, and defined as follows.

(a) The user A requests the issuance of the MakePAT command by setting AID_A , Enabler of AID_A, AID_B, Enabler of AID_B , the validity period value, and the transfer control flag value into the communication terminal of the user-A.
(b) The communication terminal of the user-A generates the MakePAT command.
(c) The communication terminal of the user-A transmits the generated MakePAT command to the PAT processing device by means such as the email, signaling, etc. (the issuance of the MakePAT command).
(d) The PAT processing device generates PAT<(link)AID_A | (link)AID_B > by processing the received MakePAT command according to Fig. 21 and Fig. 49. More specifically, this is done as follows.

(link)AID_A + (link)AID_B

+ Enabler of AID_B + Enabler of AID_A

→ LALIST<(link)AID_A | (link)AID_B >

LALIST<(link)AID_A | (link)AID_B > + Enabler of AID_A

+ validity period value + transfer control flag value

→ PAT<(link)AID_A | (link)AID_B>

(e) The PAT processing device transmits the generated PAT<(link)AID_A | (link)AID_B > to the communication terminal of the user-A, or to the communication terminal of the user-B according to the need, by means such as the email, signaling, etc.
(f) The communication terminal of the user-A (or the user-B) stores the received PAT<(link)AID_A | (link)AID_B > in the storage device of the communication terminal of the user-A.

[0316] The merging of PATs (MergePAT, Fig. 21, Fig. 49), the splitting of a PAT (SplitPAT, Fig. 22, Fig. 49), and the changing of a holder of a PAT (TransPAT, Fig. 21, Fig. 49) are also carried out by the similar procedure.