

specified as the string "fire fly" or "fire flies". Content type B is associated with the following rule set:

1. A use event that specifies that the use may only be by the software agent or a routing agent. The software agent has read only permission, the routing agent has read/write access to the information. There are no charges associated with using the information, but two meters; one by read and one by write are kept to track use of the information by various steps in the process.
2. Audits of usage are required and will be stored in object 300w under control information specified in that object.

After container 300y and its control sets are specified, they are constructed and embedded in the smart object container 300.

Container 300x is specified as a content object that is empty of content. It contains a control set that contains the following rules:

1. A write_without_billing event that specifies a meter and a general budget that limits the value of writing to \$15.00.
2. Audits of usage are required and will be stored in object 300w under control information specified in that object.
3. An empty use control set that may be filled in by the owner of the information using predefined methods (method options).

After container 300x and its control sets are specified, they are constructed and embedded in the smart object container 300.

Container 300w is specified as an empty administrative object with a control set that contains the following rules:

1. A use event that specifies that the information contained in the administrative object may only be released to the creator of smart object container 300.
2. No other rules may be attached to the administrative content in container 300w.

After container 300w and its control sets are specified, they are constructed and embedded in the smart object container 300.

At this point, the smart object has been constructed and is ready to be dispatched to a remote VDE site. The smart object is sent to a remote VDE site (e.g., using electronic mail or another transport mechanism) that contains an information locator service 3012 via path 3014. The smart object is registered at the remote site 3012 for the "item locator service." The control set in container related to "item locator service" is selected and the rules contained within it activated at the remote site 3012. The remote site 3012 then reads the contents of container 300y under the control of rule set 806f and 300y(l), and permits writes of a list of location information into container 300y pursuant to these rules. The item locator service writes a list of three items into the smart object, and then "deregisters" the smart object (now containing the location information) and sends it to a site 3016 specified in the list written to the smart object via path 3018. In this example, the user may have specified electronic mail for transport and a list of remote sites that may have the desired information is stored as a forwarding list.

The smart object 3000, upon arriving at the second remote site 3016, is registered with that second site. The site 3016 provides agent execution and software description list services

compatible with VDE as a service to smart objects. It publishes these services and specifies that it requires \$10.00 to start the agent and \$20/piece for all information returned. The registration process compares the published service information against the rules stored within the object and determines that an acceptable overlap does not exist. Audit information for all these activities is written to the administrative object 300w. The registration process then fails (the object is not registered), and the smart object is forwarded by site 3016 to the next VDE site 3020 in the list via path 3022.

The smart object 3000, upon arriving at the third remote site 3020, is registered with that site. The site 3020 provides agent execution and software description list services compatible with VDE as a service to smart objects. It publishes these services and specifies that it requires \$1.00 to start the agent and \$0.50/piece for all information returned. The registration process compares the published service information against the rules stored within the object and determines that an acceptable overlap exists. The registration process creates a URT that specifies the agreed upon control information. This URT is used in conjunction with the other control information to execute the software agent under VDE control.

The agent software starts and reads its parameters out of container 300y. It then starts searching the database and obtains 253 "hits" in the database. The list of hits is written to container 300x along with a completed control set that specifies the granularity of each item and that each item costs \$0.50. Upon completion of the search, the budget for use of the service is incremented by \$1.00 to reflect the use charge for the service. Audit information for all these activities is written to the administrative object 300w.

The remote site 3020 returns the now "full" smart object 3000 back to the original sender (the user) at their VDE node 3010 via path 3024. Upon arrival, the smart object 3000 is registered and the database records are available. The control information specified in container 300x is now a mix of the original control information and the control information specified by the service regarding remote release of their information. The user then extracts 20 records from the smart object 3000 and has \$10.00 charged to her VISA budget at the time of extraction.

In the above smart agent VDE examples, a certain organization of smart object 3000 and its constituent containers is described. Other organizations of VDE and smart object related control information and parameter data may be created

and may be used for the same purposes as those ascribed to object 3000 in the above example.

Negotiation and Electronic Contracts

An electronic contract is an electronic form of an agreement including rights, restrictions, and obligations of the parties to the agreement. In many cases, electronic agreements may surround the use of digitally provided content; for example, a license to view a digitally distributed movie. It is not required, however, that an electronic agreement be conditioned on the presence or use of electronic content by one or more parties to the agreement. In its simplest form, an electronic agreement contains a right and a control that governs how that right is used.

Electronic agreements, like traditional agreements, may be negotiated between their parties (terms and conditions submitted by one or more parties may simply be accepted (cohesion contract) by one or more other parties and/or such other parties may have the right to select certain of such terms and conditions (while others may be required)). Negotiation is defined in the dictionary as "the act of bringing together by mutual agreement." The preferred embodiment provides electronic negotiation processes by which one or more rights and associated controls can be established through electronic automated negotiation of terms.

Negotiations normally require a precise specification of rights and controls associated with those rights. PERC and URT structures provide a mechanism that may be used to provide precise electronic representations of rights and the controls associated with those rights. VDE thus provides a "vocabulary" and mechanism by which users and creators may specify their desires. Automated processes may interpret these desires and negotiate to reach a common middle ground based on these desires. The results of said negotiation may be concisely described in a structure that may be used to control and enforce the results of the electronic agreement. VDE further enables this process by providing a secure execution space in which the negotiation process(es) are assured of integrity and confidentiality in their operation. The negotiation process(es) may also be executed in such a manner that inhibits external tampering with the negotiation.

A final desirable feature of agreements in general (and electronic representations of agreements in particular) is that they be accurately recorded in a non-repudiatable form. In traditional terms, this involves creating a paper document (a contract) that describes the rights, restrictions, and obligations of all parties involved. This document is read and then signed by all parties as being an accurate representation of the agreement. Electronic agreements, by their nature, may not be initially

rendered in paper. VDE enables such agreements to be accurately electronically described and then electronically signed to prevent repudiation. In addition, the preferred embodiment provides a mechanism by which human-readable descriptions of terms of the electronic contract can be provided.

VDE provides a concise mechanism for specifying control sets that are VDE site interpretable. Machine interpretable mechanisms are often not human readable. VDE often operates the negotiation process on behalf of at least one human user. It is thus desirable that the negotiation be expressible in "human readable form." VDE data structures for objects, methods, and load modules all have provisions to specify one or more DTDs within their structures. These DTDs may be stored as part of the item or they may be stored independently. The DTD describes one or more data elements (MDE, UDE, or other related data elements) that may contain a natural language description of the function of that item. These natural language descriptions provide a language independent, human readable description for each item. Collections of items (for example, a BUDGET method) can be associated with natural language text that describes its function and forms a term of an electronically specified and enforceable contract. Collections of terms (a control set) define a contract associated with a specific right. VDE thus permits the

electronic specification, negotiation, and enforcement of electronic contracts that humans can understand and adhere to.

VDE 100 enables the negotiation and enforcement of electronic contracts in several ways:

- it enables a concise specification of rights and control information that permit a common vocabulary and procedure for negotiation,
- it provides a secure processing environment within which to negotiate,
- it provides a distributed environment within which rights and control specifications may be securely distributed,
- it provides a secure processing environment in which negotiated contracts may be electronically rendered and signed by the processes that negotiate them, and
- it provides a mechanism that securely enforces a negotiated electronic contract.

Types of Negotiations

A simple form of a negotiation is a demand by one party to form an "adhesion" contract. There are few, if any, options that may be chosen by the other party in the negotiation. The recipient of the demand has a simple option; she may accept or reject the terms and conditions (control information) in the demand. If she accepts the conditions, she is granted rights subject to the specified control information. If she rejects the conditions, she is not granted the rights. PERC and URT structures may support negotiation by demand; a PERC or control set from a PERC may be presented as a demand, and the recipient may accept or reject the demand (selecting any permitted method options if they are presented).

A common example of this type of negotiation today is the purchase of software under the terms of a "shrink-wrap license." Many widely publicized electronic distribution schemes use this type of negotiation. CompuServe is an example of an on-line service that operates in the same manner. The choice is simple: either pay the specified charge or don't use the service or software. VDE supports this type of negotiation with its capability to provide PERCs and URTs that describe rights and control information, and by permitting a content owner to provide a REGISTER method that allows a user to select from a set of predefined method options. In this scenario, the REGISTER

method may contain a component that is a simplified negotiation process.

A more complex form of a negotiation is analogous to "haggling." In this scenario, most of the terms and conditions are fixed, but one or more terms (e.g., price or payment terms) are not. For these terms, there are options, limits, and elements that may be negotiated over. A VDE electronic negotiation between two parties may be used to resolve the desired, permitted, and optional terms. The result of the electronic negotiation may be a finalized set of rules and control information that specify a completed electronic contract. A simple example is the scenario for purchasing software described above adding the ability of the purchaser to select a method of payment (VISA, Mastercard, or American Express). A more complex example is a scenario for purchasing information in which the price paid depends on the amount of information about the user that is returned along with a usage audit trail. In this second example, the right to use the content may be associated with two control sets. One control set may describe a fixed ("higher") price for using the content. Another control set may describe a fixed ("lower") price for using the content with additional control information and field specifications requiring collection and return the user's personal information. In both of these cases, the optional and permitted fields and control sets in a PERC may describe the options that

may be selected as part of the negotiation. To perform the negotiation, one party may propose a control set containing specific fields, control information, and limits as specified by a PERC; the other party may pick and accept from the control sets proposed, reject them, or propose alternate control sets that might be used. The negotiation process may use the permitted, required, and optional designations in the PERC to determine an acceptable range of parameters for the final rule set. Once an agreement is reached, the negotiation process may create a new PERC and/or URT that describes the result of the negotiation. The resulting PERCs and/or URTs may be "signed" (e.g., using digital signatures) by all of the negotiation processes involved in the negotiation to prevent repudiation of the agreement at a later date.

Additional examples of negotiated elements are: electronic cash, purchase orders, purchase certificates (gift certificates, coupons), bidding and specifications, budget "rollbacks" and reconciliation, currency exchange rates, stock purchasing, and billing rates.

A set of PERCs that might be used to support the second example described above is presented in Figures 75A (PERC sent by the content owner), 75B (PERC created by user to represent their selections and rights), and 75C (PERC for controlling the

negotiation process). These PERCs might be used in conjunction with any of the negotiation process(es) and protocols described later in this section.

Figure 75A shows an example of a PERC 3100 that might be created by a content provider to describe their rights options. In this example, the PERC contains information regarding a single USE right. Two alternate control sets 3102a, 3102b are presented for this right in the example. Control set 3102a permits the use of the content without passing back information about the user, and another control set 3102b permits the use of the content and collects "response card" type information from the user. Both control sets 3102a, 3102b may use a common set of methods for most of the control information. This common control information is represented by a CSR 3104 and CS0 3106.

Control set 3102a in this PERC 3100 describes a mechanism by which the user may obtain the content without providing any information about its user to the content provider. This control set 3102a specifies a well-known vending control method and set of required methods and method options. Specifically, in this example, control set 3102a defines a BUDGET method 3108 (e.g., one of VISA, Mastercard, or American Express) and it defines a BILLING method 3110 that specifies a charge (e.g., a one-time charge of \$100.00).

Control set 3102b in this PERC 3100 describes another mechanism by which the user may obtain the content. In this example, the control set 3102b specifies a different vending control method and a set of required methods and method options. This second control set 3102b specifies a BUDGET method 3112 (e.g., one of VISA, Mastercard, or American Express), a BILLING method 3116 that specifies a charge (e.g., a lesser one-time charge such as \$25.00) and an AUDIT method 3114 that specifies a set of desired and required fields. The required and desired field specification 3116 may take the form of a DTD specification, in which, for example, the field names are listed.

The content creator may "prefer" one of the two control sets (e.g., control set 2) over the other one. If so, the "preferred" control set may be "offered" first in the negotiation process, and withdrawn in favor of the "non-preferred" control set if the other party to the negotiation "rejects" the "preferred" control set.

In this example, these two control sets 3102a, 3102b may share a common BUDGET method specification. The BUDGET method specification may be included in the CSR 3104 or CS0 3106 control sets if desired. Selecting control set 3102a (use with no information passback) causes a unique component assembly to be assembled as specified by the PERC 3100. Specifically, in this

example it selects the "Vending" CONTROL method 3118, the BILLING method 3110 for a \$100 fixed charge, and the rest of the control information specified by CSR 3104 and CS0 3106. It also requires the user to specify her choice of acceptable BUDGET method (e.g., from the list including VISA, Mastercard, and American Express). Selecting control set 3102b assembles a different component assembly using the "Vending with 'response card" CONTROL method 3120, the BILLING method 3116 (e.g., for a \$25 fixed charge), an AUDIT method 3114 that requires the fields listed in the Required Fields DTD 3116. The process may also select as many of the fields listed in the Desired Fields DTD 3116 as are made available to it. The rest of the control information is specified by CSR 3104 and CS0 3106. The selection of control set 3102b also forces the user to specify their choice of acceptable BUDGET methods (e.g., from the list including VISA, Mastercard, and American Express).

Figure 75B shows an example of a control set 3125 that might be used by a user to specify her desires and requirements in a negotiation process. This control set has a USE rights section 3127 that contains an aggregated CSR budget specification 3129 and two optional control sets 3131a, 3131b for use of the content. Control set 3131a requires the use of a specific CONTROL method 3133 and AUDIT method 3135. The specified AUDIT method 3135 is parameterized with a list of

fields 3137 that may be released in the audit trail. Control set 3131a also specifies a BILLING method 3139 that can cost no more than a certain amount (e.g., \$30.00). Control set 3131b in this example describes a specific CONTROL method 3141 and may reference a BILLING method 3143 that can cost no more than a certain amount (e.g., \$150.00) if this option is selected.

Figure 75E shows a more high-level view of an electronic contract 3200 formed as a "result" of a negotiation process as described above. Electronic contract 3200 may include multiple clauses 3202 and multiple digital signatures 3204. Each clause 3202 may comprise a PERC/URT such as item 3160 described above and shown in Figure 75D. Each "clause" 3202 of electronic contract 3200 thus corresponds to a component assembly 690 that may be assembled and executed by a VDE electronic appliance 600. Just as in normal contracts, there may be as many contract clauses 3202 within electronic contract 3200 as is necessary to embody the "agreement" between the "parties." Each of clauses 3202 may have been electronically negotiated and may thus embody a part of the "agreement" (e.g., a "compromise") between the parties. Electronic contract 3200 is "self-executing" in the sense that it may be literally executed by a machine, i.e., a VDE electronic appliance 600 that assembles component assemblies 690 as specified by various electronic clauses 3202. Electronic contract 3200 may be automatically

"enforced" using the same VDE mechanisms discussed above that are used in conjunction with any component assembly 690. For example, assuming that a clause 3202(2) corresponds to a payment or BILLING condition or term, its corresponding component assembly 690 when assembled by a user's VDE electronic appliance 600 may automatically determine whether conditions are right for payment and, when they are, automatically access an appropriate payment mechanism (e.g., a virtual "credit card" object for the user) to arrange that payment to be made. As another example, assuming that electronic contract clause N 3202(N) corresponds to a user's obligation to provide auditing information to a particular VDE participant, electronic contract 3200 will cause VDE electronic appliance 600 to assemble a corresponding component assembly 690 that may, for example, access the appropriate audit trails within secure database 610 and provide them in an administrative object to the correct participant. Figure 75F shows that clause 3202(N) may, for example, specify a component assembly 690 that arranges for multiple steps in a transaction 3206 to occur. Some of these steps (e.g., step 3208(4), 3208(5)) may be conditional on a test (e.g., 3208(3)) such as, for example, whether content usage has exceeded a certain amount, whether a certain time period has expired, whether a certain calendar date has been reached, etc.

Digital signatures 3204 shown in the Figure 75E electronic contract 3200 may comprise, for example, conventional digital signatures using public key techniques as described above. Some electronic contracts 3200 may not bear any digital signatures 3204. However, it may be desirable to require the electronic appliance 600 of the user who is a party to the electronic contract 3200 to digitally "sign" the electronic contract so that the user cannot later repudiate the contract, for evidentiary purposes, etc. Multiple parties to the same contract may each digitally "sign" the same electronic contract 3200 similarly to the way multiple parties to a contract memorialized in a written instrument use an ink pen to sign the instrument.

Although each of the clauses 3202 of electronic contract 3200 may ultimately correspond to a collection of data and code that may be executed by a PPE 650, there may in some instances be a need for rendering a human readable version of the electronic contract. This need can be accommodated by, as mentioned above, providing text within one or more DTDs associated with the component assembly or assemblies 690 used to "self-execute" the contract. Such text might, for example, describe from a functional point of view what the corresponding electronic contract clause 3202 means or involves, and/or might describe in legally enforceable terms what the legal obligation under the contract is or represents. "Templates" (described

elsewhere herein) might be used to supply such text from a text library. An expert system and/or artificial intelligence capability might be used to impose syntax rules that bind different textual elements together into a coherent, humanly readable contract document. Such text could, if necessary, be reviewed and modified by a "human" attorney in order to customize it for the particular agreement between the parties and/or to add further legal obligations augmenting the "self-executing" electronic obligations embodied within and enforced by the associated component assemblies 690 executing on a VDE electronic appliance 600. Such text could be displayed automatically or on demand upon execution of the electronic contract, or it could be used to generate a printed humanly-readable version of the contract at any time. Such a document version of the electronic contract 3200 would not need to be signed in ink by the parties to the agreement (unless desired) in view of the fact that the digital signatures 3204 would provide a sufficiently secure and trusted evidentiary basis for proving the parties' mutual assent to all the terms and conditions within the contract.

In the preferred embodiment, the negotiation process executes within a PPE 650 under the direction of a further PERC that specifies the process. Figure 75C shows an example of a PERC 3150 that specifies a negotiation process. The PERC 3150 contains a single right 3152 for negotiation, with two permitted control sets 3154a, 3154b described for that right. The first

control set 3154a may be used for a "trusted negotiation"; it references the desired negotiation CONTROL method ("Negotiate") 3156 and references (in fields 3157a, 3157b) two UDEs that this CONTROL method will use. These UDEs may be, for example, the PERCs 3100, 3125 shown in Figures 75A and 75B. The second control set 3154b may be used by "multiple negotiation" processes to manage the negotiation, and may provide two negotiation methods: "Negotiate1," and "Negotiate2". Both negotiation processes may be described as required methods ("Negotiate1" and "Negotiate2") 3156, 3158 that take respective PERCs 3100, 3125 as their inputs. The CONTROL method 3158 for this control set in this example may specify the name of a service that the two negotiation processes will use to communicate with each other, and may also manage the creation of the URT resulting from the negotiation.

When executed, the negotiation process(es) specified by the PERC 3150 shown in Figure 75C may be provided with the PERCs 3100, 3125 as input that will be used as the basis for negotiation. In this example, the choice of negotiation process type (trusted or multiple) may be made by the executing VDE node. The PERC 3150 shown in Figure 75C might be, for example, created by a REGISTER method in response to a register request from a user. The process specified by this PERC

3150 may then be used by a REGISTER method to initiate negotiation of the terms of an electronic contract.

During this example negotiation process, the PERCs 3100, 3125 shown in Figures 75A and 75B act as input data structures that are compared by a component assembly created based on PERC 3150 shown in Figure 35C. The component assembly specified by the control sets may be assembled and compared, starting with required "terms," and progressing to preferred/desired "terms" and then moving on to permitted "terms," as the negotiation continues. Method option selections are made using the desired method and method options specified in the PERCs 3100, 3125. In this example, a control set for the PERC 3100 shown in Figure 75A may be compared against the PERC 3125 shown in Figure 75B. If there is a "match," the negotiation is successfully concluded and "results" are generated.

In this embodiment, the results of such negotiation will generally be written as a URT and "signed" by the negotiation process(es) to indicate that an agreement has been reached. These electronic signatures provide the means to show that a (virtual) "meeting of minds" was reached (one of the traditional legal preconditions for a contract to exist). An example of the URT 3160 that would have been created by the above example is shown in Figure 75D.

This URT 3160 (which may itself be a PERC 808) includes a control set 3162 that reflects the "terms" that were "agreed upon" in the negotiation. In this example, the "agreed upon" terms must "match" terms required by input PERCs 3100, 3125 in the sense that they must be "as favorable as" the terms required by those PERCs. The negotiation result shown includes, for example, a "negotiated" control set 3162 that in some sense corresponds to the control set 3102a of the Figure 75A PERC 3100 and to the control set 3131a of the Figure 75B control set 3125. Resulting "negotiated" control set 3162 thus includes a required BUDGET method 3164 that corresponds to the control set 3125 desired BUDGET method 3142 but which is "within" the range of control sets allowed by control set 3100 required BUDGET method 3112. Similarly, resulting negotiated control set 3162 includes a required AUDIT method 3166 that complies with the requirements of both PERC 3100 required AUDIT method 3114 and PERC 3125 required AUDIT method 3135. Similarly, resulting negotiated control set 3162 includes a required BILLING method 3170 that "matches" or complies with each of PERC 3100 required BILLING method 3116 and PERC 3125 required BILLING method 3170.

Another class of negotiation is one under which no rules are fixed and only the desired goals are specified. The negotiation processes for this type of negotiation may be very

complex. It may utilize artificial intelligence, fuzzy logic, and/or related algorithms to reach their goals. VDE supports these types of processes by providing a mechanism for concisely specifying rights, control information, fields and goals (in the form of desired rights, control information, and fields). Goals for these types of processes might be specified as one more control sets that contain specific elements that are tagged as optional, permitted, or desired.

Types of Negotiations

Negotiations in the preferred embodiment may be structured in any of the following ways:

1. shared knowledge
2. trusted negotiator
3. "zero-based" knowledge

"Shared knowledge" negotiations are based on all parties knowing all of the rules and constraints associated with the negotiation. Demand negotiations are a simple case of shared knowledge negotiations; the demander presents a list of demands that must be accepted or rejected together. The list of demands comprises a complete set of knowledge required to accept or reject each item on the list. VDE enables this class of negotiation to occur electronically by providing a mechanism by which demands may be encoded, securely passed, and securely processed between

and with secure VDE subsystems using VDE secure processing, and communication capabilities. Other types of shared knowledge negotiations employed by VDE involve the exchange of information between two or more negotiating parties; the negotiation process(es) can independently determine desired final outcome(s) based on their independent priorities. The processes can then negotiate over any differences. Shared knowledge negotiations may require a single negotiation process (as in a demand type negotiation) or may involve two or more cooperative processes. Figures 76A and 76B illustrate scenarios in which one and two negotiation processes are used in a shared knowledge negotiation.

Figure 76A shows a single negotiation process 3172 that takes any number of PERCs 808 (e.g., supplied by different parties) as inputs to the negotiation. The negotiation process 3172 executes at a VDE node under supervision of "Negotiation Process Rules and Control information" that may be supplied by a further PERC (e.g., PERC 3150 shown in Figure 75C). The process 3172 generates one or more PERCs/URTs 3160 as results of the negotiation.

Figure 76B shows multiple negotiation processes 3172A-3172N each of which takes as input a PERC 808 from a party and a further PERC 3150 that controls the negotiation process,

and each of which generates a negotiated "result" PERC/URT 3160 as output. Processes 3172A-3172N may execute at the same or different VDE nodes and may communicate using a "negotiation protocol."

Single and multiple negotiation processes may be used for specific VDE sites. The negotiation processes are named, and can be accessed using well known method names. PERCs and URTs may be transported in administrative or smart objects to remote VDE sites for processing at that site, as may the control PERCs and REGISTER method that controls the negotiation.

Multiple negotiation processes require the ability to communicate between these processes 3172; including secure communication between secure processes that are present at physically separate VDE sites (secure subsystems). VDE generalizes the inter-process communication into a securely provided service that can be used if the configuration requires it. The inter-process communication uses a negotiation protocol to exchange information about rule sets between processes 3172. An example of a negotiation protocol includes the following negotiation "primitives":

WANT	Want a set of terms and conditions
ACCEPT	Accept a set of terms and conditions
REJECT	Reject a set of terms and conditions

OFFER	Offer a set of terms and conditions in exchange for other terms and conditions
HAVE	Assert a set of terms and conditions are possible or desirable
QUIT	Assert the end of the negotiation without reaching an agreement
AGREEMENT	Conclude the negotiation and pass the rule set for signature

The WANT primitive takes rights and control set (or parts of control sets) information, and asserts to the other process(es) 3172 that the specified terms are desired or required. Demand negotiations are a simple case of a WANT primitive being used to assert the demand. This example of a protocol may introduce a refined form of the WANT primitive, REQUIRE. In this example, REQUIRE allows a party to set terms that she decides are necessary for a contract to be formed, WANT may allow the party to set terms that are desirable but not essential. This permits a distinction between "must have" and "would like to have."

In this example, WANT primitives must always be answered by an ACCEPT, REJECT, or OFFER primitive. The ACCEPT primitive permits a negotiation process 3172 to accept a set of terms and conditions. The REJECT primitive permits a process 3172 to reject an offered set of terms and conditions.

Rejecting a set of required terms and conditions may terminate the negotiation. OFFER permits a counter-offer to be made.

The HAVE, QUIT, and AGREEMENT primitives permit the negotiation protocols to pass information about rule sets. Shared knowledge negotiations may, for example, start with all negotiation processes 3172A-3172N asserting HAVE (my PERC) to the other processes. HAVE is also used when an impasse is reached and one process 3172 needs to let the other process 3172 know about permitted options. QUIT signals an unsuccessful end of the negotiation without reaching an agreement, while AGREEMENT signals a successful end of an agreement and passes the resulting "negotiated" PERC/URT 3160 to the other process(es) 3172 for signature.

In "trusted negotiator" negotiations, all parties provide their demands and preferences to a "trusted" negotiator and agree to be bound by her decision. This is similar to binding arbitration in today's society. VDE enables this mode of negotiation by providing an environment in which a "trusted" negotiation service may be created. VDE provides not only the mechanism by which demands, desires, and limits may be concisely specified (e.g., in PERCs), but in which the PERCs may be securely transferred to a "trusted" negotiation service along with a rule set that specifies how the negotiation will be

conducted, and by providing a secure execution environment so that the negotiation process may not be tampered with. Trusted negotiator services can be used at VDE sites where the integrity of the site is well known. Remote trusted negotiation services can be used by VDE sites that do not possess sufficient computing resources to execute one or more negotiation process(es); they can establish a communication link to a VDE site that provides this service and permits the service to handle the negotiation on their behalf.

"Zero-based" knowledge negotiations share some characteristics of the zero-based knowledge protocols used for authentication. It is well understood in the art how to construct a protocol that can determine if a remote site is the holder of a specific item without exchanging or exposing the item. This type of protocol can be constructed between two negotiation processes operating on at least one VDE site using a control set as their knowledge base. The negotiation processes may exchange information about their control sets, and may make demands and counter proposals regarding using their individual rule sets. For example, negotiation process A may communicate with negotiation process B to negotiate rights to read a book. Negotiation process A specifies that it will pay not more than \$10.00 for rights to read the book, and prefers to pay between \$5.00 and \$6.00 for this right. Process A's rule set also specifies

that for the \$5.00 option, it will permit the release of the reader's name and address. Process B's rule set specifies that it wants \$50.00 for rights to read the book, and will provide the book for \$5.50 if the user agrees to release information about himself. The negotiation might go something like this:

Process A	<--- >	Process B
Want (right to read, unrestricted)	---->	
	<----	Have(right to read, unrestricted, \$50)
Offer (right to read, tender user info)	---->	
	<----	Have(right to read, tender user info, \$5.50)
Accept(right to read, tender user info, \$5.50)	----- >	

In the above example, process A first specifies that it desires the right to read the book without restrictions or other information release. This starting position is specified as a rights option in the PERC that process A is using as a rule. Process B checks its rules and determines that an unrestricted right to read is indeed permitted for a price of \$50. It replies to process A that these terms are available. Process A receives this reply and checks it against the control set in the PERC it uses as a rule base. The \$50 is outside the \$10 limit specified for this control set, so Process A cannot accept the offer. It makes a counter offer

(as described in another optional rights option) of an unrestricted right to read coupled with the release of the reader's name and address. The name and address fields are described in a DTD referenced by Process A's PERC. Process B checks its rules PERC and determines that an unrestricted right to read combined with the release of personal information is a permitted option. It compares the fields that would be released as described in the DTD provided by Process A against the desired fields in a DTD in its own PERC, and determines an acceptable match has occurred. It then sends an offer for unrestricted rights with the release of specific information for the cost of \$5.50 to Process A. Process A compares the right, restrictions, and fields against its rule set and determines that \$5.50 is within the range of \$5-\$6 described as acceptable in its rule set. It accepts the offer as made. The offer is sealed by both parties "signing" a new PERC that describes the results of the final negotiation (unrestricted rights, with release of user information, for \$5.50). The new PERC may be used by the owner of Process A to read the content (the book) subject to the described terms and conditions.

Further Chain of Handling Model

As described in connection with Figure 2, there are four (4) "participant" instances of VDE 100 in one example of a VDE chain of handling and control used, for example, for content distribution. The first of these participant instances, the content

creator 102, is manipulated by the publisher, author, rights owner or distributor of a literary property to prepare the information for distribution to the consumer. The second participant instance, VDE rights distributor 106, may distribute rights and may also administer and analyze customers' use of VDE authored information. The third participant instance, content user 112, is operated by users (included end-users and distributors) when they use information. The fourth participant instance, financial clearinghouse 116 enables the VDE related clearinghouse activities. A further participant, a VDE administrator, may provide support to keep VDE 100 operating properly. With appropriate authorizations and Rights Operating System components installed, any VDE electronic appliance 600 can play any or all of these participant roles.

Literary property is one example of raw material for VDE 100. To transfer this raw material into finished goods, the publisher, author, or rights owner uses tools to transform digital information (such as electronic books, databases, computer software and movies) into protected digital packages called "objects." Only those consumers (or others along the chain of possession such as a redistributor) who receive permission from a distributor 106 can open these packages. VDE packaged content can be constrained by "rules and control information" provided by content creator 102 and/or content distributor 106—or by other

VDE participants in the content's distribution pathway, i.e., normally by participants "closer" to the creation of the VDE secured package than the participant being constrained.

Once the content is packaged in an "object," the digital distribution process may begin. Since the information packages themselves are protected, they may be freely distributed on CD-ROM disks, through computer networks, or broadcast through cable or by airwaves. Informal "out of channel" exchange of protected packages among end-users does not pose a risk to the content property rights. This is because only authorized individuals may use such packages. In fact, such "out of channel" distribution may be encouraged by some content providers as a marginal cost method of market penetration. Consumers with usage authorizations (e.g., a VISA clearinghouse budget allowing a certain dollar amount of usage) may, for example, be free to license classes of out of channel VDE protected packages provided to them, for example, by a neighbor.

To open a VDE package and make use of its content, an end-user must have permission. Distributors 106 can grant these permissions, and can very flexibly (if permitted by senior control information) limit or otherwise specify the ways in which package contents may be used. Distributors 106 and financial clearinghouses 116 also typically have financial responsibilities

(they may be the same organization in some circumstances if desired). They ensure that any payments required from end-users fulfill their own and any other participant's requirements. This is achieved by auditing usage.

Distributors 106 using VDE 100 may include software publishers, database publishers, cable, television, and radio broadcasters, and other distributors of information in electronic form. VDE 100 supports all forms of electronic distribution, including distribution by broadcast or telecommunications, or by the physical transfer of electronic storage media. It also supports the delivery of content in homogeneous form, seamlessly integrating information from multiple distribution types with separate delivery of permissions, control mechanisms and content.

Distributors 106 and financial clearinghouses 116 may themselves be audited based on secure records of their administrative activities and a chain of reliable, "trusted" processes ensures the integrity of the overall digital distribution process. This allows content owners, for example, to verify that they are receiving appropriate compensation based on actual content usage or other agreed-upon bases.

Since the end-user 112 is the ultimate consumer of content in this example, VDE 100 is designed to provide protected

content in a seamless and transparent way—so long as the end-user stays within the limits of the permissions she has received. The activities of end-user 112 can be metered so that an audit can be conducted by distributors 106. The auditing process may be filtered and/or generalized to satisfy user privacy concerns. For example, metered, recorded VDE content and/or appliance usage information may be filtered prior to reporting it to distributor 106 to prevent more information than necessary from being revealed about content user 112 and/or her usage.

VDE 100 gives content providers the ability to recreate important aspects of their traditional distribution strategies in electronic form and to innovatively structure new distribution mechanisms appropriate to their individual needs and circumstances. VDE 100 supports relevant participants in the chain of distribution, and also enables their desired pricing strategies, access and redistribution permissions, usage rules, and related administrative and analysis procedures. The reusable functional primitives of VDE 100 can be flexibly combined by content providers to reflect their respective distribution objectives. As a result, content providers can feed their information into established distribution channels and also create their own personalized distribution channels.

A summary of the roles of the various participants of virtual distribution environment 100 is set forth in the table below:

Role	Description
"Traditional" Participants	
Content creator	Packager and initial distributor of digital information
Content owner	Owner of the digital information.
Distributors	Provide rights distribution services for budgets and/or content.
Auditor	Provides services for processing and reducing usage based audit trails.
Clearinghouse	Provides intermediate store and forward services for content and audit information. Also, typically provides a platform for other services, including third party financial providers and auditors.
Network provider	Provides communication services between sites and other participants.
Financial providers	Provider of third party sources of electronic funds to end-users and distributors. Examples of this class of users are VISA, American Express, or a government.
End Users	Consumers of information.
Other Participants	
Redistributor	Redistributes rights to use content based on chain of handling restrictions from content providers and/or other distributors.
VDE Administrator	Provider of trusted services for support of VDE nodes.

Role	Description
Independent Audit Processor	Provider of services for processing and summarizing audit trail data. Provides anonymity to end-users while maintaining the comprehensive audit capabilities required by the content providers.
Agents	Provides distributed presence for end-users and other VDE participants.

Of these various VDE participants, the "redistributor," "VDE Administrator," "independent audit processor" and "agents" are, in certain respects "new" participants that may have no counterpart in many "traditional" business models. The other VDE participants (i.e., content provider, content owner, distributors, auditor, clearinghouse, network provider and financial providers) have "traditional" business model counterparts in the sense that traditional distribution models often included non-electronic participants performing some of the same business roles they serve in the virtual distribution environment 100.

VDE distributors 106 may also include "end-users" who provide electronic information to other end-users. For example, Figure 77 shows a further example of a virtual distribution environment 100 chain of handling and control provided by the present invention. As compared to Figure 2, Figure 77 includes a new "client administrator" participant 700. In addition, Figure

77 shows several different content users 112(1), 112(2), . . . , 112(n) that may all be subject to the "jurisdiction" of the client administrator 700. Client administrator 700 may be, for example, a further rights distributor within a corporation or other organization that distributes rights to employees or other organization participant units (such as divisions, departments, networks, and or groups, etc.) subject to organization-specific "rules and control information." The client administrator 700 may fashion rules and control information for distribution, subject to "rules and control" specified by creator 102 and/or distributor 106.

As mentioned above, VDE administrator 116b is a trusted VDE node that supports VDE 100 and keeps it operating properly. In this example, VDE administrator 116b may provide, among others, any of all of the following:

- VDE appliance initialization services
- VDE appliance reinitialization/update services
- Key management services
- "Hot lists" of "rogue" VDE sites
- Certification authority services
- Public key registration
- Client participant unit content budgets and other authorizations

All participants of VDE 100 have the innate ability to participate in any role. For example, users may gather together existing protected packages, add (create new content) packages of their own, and create new products. They may choose to serve as their own distributor, or delegate this responsibility to others. These capabilities are particularly important in the object oriented paradigm which is entering the marketplace today. The production of compound objects, object linking and embedding, and other multi-source processes will create a need for these capabilities of VDE 100. The distribution process provided by VDE 100 is symmetrical; any end-user may redistribute information received to other end-users, provided they possess permission from and follow the rules established by the distribution chain VDE control information governing redistribution. End-users also may, within the same rules and permissions restriction, encapsulate content owned by others within newly published works and distribute these works independently. Royalty payments for the new works may be accessed by the publisher, distributors, or end-users, and may be tracked and electronically collected at any stage of the chain.

Independent financial providers can play an important role in VDE 100. The VDE financial provider role is similar to the role played by organizations such as VISA in traditional distribution scenarios. In any distribution model, authorizing

payments for use of products or services and auditing usage for consistency and irregularities, is critical. In VDE 100, these are the roles filled by independent financial providers. The independent financial providers may also provide audit services to content providers. Thus, budgets or limits on use, and audits, or records of use, may be processed by (and may also be put in place by) clearinghouses 116, and the clearinghouses may then collect usage payments from users 112. Any VDE user 112 may assign the right to process information or perform services on their behalf to the extent allowed by senior control information. The arrangement by which one VDE participant acts on behalf of another is called a "proxy." Audit, distribution, and other important rights may be "proxied" if permitted by the content provider. One special type of "proxy" is the VDE administrator 116b. A VDE administrator is an organization (which may be acting also as a financial clearinghouse 116) that has permission to manage (for example, "intervene" to reset) some portion or all of VDE secure subsystem control information for VDE electronic appliances. This administration right may extend only to admitting new appliances to a VDE infrastructure and to recovering "crashed" or otherwise inoperable appliances, and providing periodic VDE updates.

More On Object Creation, Distribution Methods, Budgets, and Audits

VDE node electronic appliances 600 in the preferred embodiment can have the ability to perform object creation, distribution, audit collection and usage control functions provided by the present invention. Incorporating this range of capabilities within each of many electronic appliances 600 provided by the preferred embodiment is important to a general goal of creating a single (or prominent) standard for electronic transactions metering, control, and billing, that, in its sum of installations, constitutes a secure, trusted, virtual transaction/distribution management environment. If, generally speaking, certain key functions were generally or frequently missing, at least in general purpose VDE node electronic appliances 600, then a variety of different products and different standards would come forth to satisfy the wide range of applications for electronic transaction/distribution management; a single consistent set of tools and a single "rational," trusted security and commercial distribution environment will not have been put in place to answer the pressing needs of the evolving "electronic highway." Certain forms of certain electronic appliances 600 containing VDE nodes which incorporate embedded dedicated VDE microcontrollers such as certain forms of video cassette players, cable television converters and the like may not necessarily have or need full VDE capabilities. However, the preferred

embodiment provides a number of distributed, disparately located electronic appliances 600 each of which desirably include authoring, distribution, extraction, audit, and audit reduction capabilities, along with object authoring capabilities.

The VDE object authoring capabilities provided by the preferred embodiment provides an author, for example, with a variety of menus for incorporating methods in a VDE object 300, including:

- menus for metering and/or billing methods which define how usage of the content portion of a VDE object is to be controlled,
- menus related to extraction methods for limiting and/or enabling users of a VDE object from extracting information from that object, and may include placing such information in a newly created and/or pre-existing VDE content container,,
- menus for specifying audit methods—that is, whether or not certain audit information is to be generated and communicated in some secure fashion back to an object provider, object creator, administrator, and/or clearinghouse, and

- menus for distribution methods for controlling how an object is distributed, including for example, controlling distribution rights of different participant's "down" a VDE chain of content container handling.

The authoring capabilities may also include procedures for distributing administrative budgets, object distribution control keys, and audit control keys to distributors and other VDE participants who are authorized to perform distribution and/or auditing functions on behalf of the author, distributors, and/or themselves. The authoring capabilities may also include procedures for selecting and distributing distribution methods, audit methods and audit reduction methods, including for example, securely writing and/or otherwise controlling budgets for object redistribution by distributors to subsequent VDE chain of content handling participants.

The content of an object 300 created by an author may be generated with the assistance of a VDE aware application program or a non-VDE aware application program. The content of the object created by an author in conjunction with such programs may include text, formatted text, pictures, moving pictures, sounds, computer software, multimedia, electronic games, electronic training materials, various types of files, and so on, without limitation. The authoring process may encapsulate

content generated by the author in an object, encrypt the content with one or more keys, and append one or more methods to define parameters of allowed use and/or required auditing of use and/or payment for use of the object by users (and/or by authorized users only). The authoring process may also include some or all aspects of distributing the object.

In general, in the preferred embodiment, an author can:

- A. Specify what content is to be included in an object.
- B. Specify content oriented methods including:
 - Information--typically abstract, promotional, identifying, scheduling, and/or other information related to the content and/or author
 - Content--e.g. list of files and/or other information resources containing content, time variables, etc.
- C. Specify control information (typically a collection of methods related to one another by one or more permissions records, including any method defining variables) and any initial authorized user list including, for example:
 - Control information over Access & Extraction

Control information over Distribution

Control information over Audit Processing

A VDE node electronic appliance 600 may, for example, distribute an object on behalf of an object provider if a VDE node receives from an object provider administrative budget information for distributing the object and associated distribution key information.

A VDE node electronic appliance 600 may receive and process audit records on behalf of an object provider if that VDE node receives any necessary administrative budget, audit method, and audit key information (used, for example, to decrypt audit trails), from the object provider. An auditing-capable VDE electronic appliance 600 may control execution of audit reduction methods. "Audit reduction" in the preferred embodiment is the process of extracting information from audit records and/or processes that an object provider (e.g., any object provider along a chain of handling of the object) has specified to be reported to an object's distributors, object creators, client administrators, and/or any other user of audit information. This may include, for example, advertisers who may be required to pay for a user's usage of object content. In one embodiment, for example, a clearinghouse can have the ability to "append" budget, audit method, and/or audit key information to an object or class or other grouping of objects located at a user site or located at an

object provider site to ensure that desired audit processes will take place in a "trusted" fashion. A participant in a chain of handling of a VDE content container and/or content container control information object may act as a "proxy" for another party in a chain of handling of usage auditing information related to usage of object content (for example a clearinghouse, an advertiser, or a party interested in market survey and/or specific customer usage information). This may be done by specifying, for that other party, budget, audit method, and/or key information that may be necessary to ensure audit information is gathered and/or provided to, in a proper manner, said additional party. For example, employing specification information provided by said other party.

Object Creation and Initial Control Structures

The VDE preferred embodiment object creation and control structure design processes support fundamental configurability of control information. This enables VDE 100 to support a full range of possible content types, distribution pathways, usage control information, auditing requirements, and users and user groups. VDE object creation in the preferred embodiment employs VDE templates whose atomic elements represent at least in part modular control processes. Employing VDE creation software (in the preferred embodiment a GUI programming process) and VDE templates, users may create VDE objects 300

by, for example, partitioning the objects, placing "meta data" (e.g., author's name, creation date, etc.) into them, and assigning rights associated with them and/or object content to, for example, a publisher and/or content creator. When an object creator runs through this process, she normally will go through a content specification procedure which will request required data. The content specification process, when satisfied, may proceed by, for example, inserting data into a template and encapsulating the content. In addition, in the preferred embodiment, an object may also automatically register its presence with the local VDE node electronic appliance 600 secure subsystem, and at least one permissions record 808 may be produced as a result of the interaction of template instructions and atomic methods, as well as one or more pieces of control structure which can include one or more methods, budgets, and/or etc. A registration process may require a budget to be created for the object. If an object creation process specifies an initial distribution, an administrative object may also be created for distribution. The administrative object may contain one or more permission records 808, other control structures, methods, and/or load modules.

Permissions records 808 may specify various control relationships between objects and users. For example, VDE 100 supports both single access (e.g., one-to-one relationship between a user and a right user) and group access (any number of people

may be authorized as a group). A single permissions record 808 can define both single and group access. VDE 100 may provide "sharing," a process that allows multiple users to share a single control budget as a budget. Additional control structure concepts include distribution, redistribution, and audit, the latter supporting meter and budget information reduction and/or transfer. All of these processes are normally securely controlled by one or more VDE secure subsystems.

Templates and Classes

VDE templates, classes, and flexible control structures support frameworks for organizations and individuals that create, modify, market, distribute, redistribute, consume, and otherwise use movies, audio recordings and live performances, magazines, telephony based retail sales, catalogs, computer software, information databases, multimedia, commercial communications, advertisements, market surveys, infomercials, games, CAD/CAM services for numerically controlled machines, and the like. As the context surrounding these classes changes or evolves, the templates provided by the preferred embodiment of the present invention can be modified to meet these changes for broad use, or more focused activities.

VDE 100 authoring may provide three inputs into a create process: Templates, user input and object content. Templates

act as a set of control instructions and/or data for object control software which are capable of creating (and/or modifying) VDE objects in a process that interacts with user instructions and provided content to create a VDE object. Templates are usually specifically associated with object creation and/or control structures. Classes represent user groups which can include "natural" groups within an organization, such as department members, specific security clearance levels, etc., or ad hoc lists of individual's and/or VDE nodes.

For example, templates may be represented as text files defining specific structures and/or component assemblies. Templates, with their structures and/or component assemblies may serve as VDE object authoring or object control applications. A creation template may consist of a number of sub-templates, which, at the lowest level, represent an "atomic level" of description of object specification. Templates may present one or more models that describe various aspects of a content object and how the object should be created including employing secure atomic methods that are used to create, alter, and/or destroy permissions records 808 and/or associated budgets, etc.

Templates, classes (including user groups employing an object under group access), and flexible control structures including object "independent" permissions records (permissions

that can be associated with a plurality of objects) and structures that support budgeting and auditing as separate VDE processes, help focus the flexible and configurable capabilities inherent within authoring provided by the present invention in the context of specific industries and/or businesses and/or applications. VDE rationalizes and encompasses distribution scenarios currently employed in a wide array of powerful industries (in part through the use of application or industry specific templates). Therefore, it is important to provide a framework of operation and/or structure to allow existing industries and/or applications and/or businesses to manipulate familiar concepts related to content types, distribution approaches, pricing mechanisms, user interactions with content and/or related administrative activities, budgets, and the like.

The VDE templates, classes, and control structures are inherently flexible and configurable to reflect the breadth of information distribution and secure storage requirements, to allow for efficient adaptation into new industries as they evolve, and to reflect the evolution and/or change of an existing industry and/or business, as well as to support one or more groups of users who may be associated with certain permissions and/or budgets and object types. The flexibility of VDE templates, classes, and basic control structures is enhanced through the use of VDE aggregate and control methods which have a compound,

conditional process impact on object control. Taken together, and employed at times with VDE administrative objects and VDE security arrangements and processes, the present invention truly achieves a content control and auditing architecture that can be configured to most any commercial distribution embodiment. Thus, the present invention fully supports the requirements and biases of content providers without forcing them to fit a predefined application model. It allows them to define the rights, control information, and flow of their content (and the return of audit information) through distribution channels.

Modifying Object Content (Adding, Hiding, Modifying, Removing, and/or Extending)

Adding new content to objects is an important aspect of authoring provided by the present invention. Providers may wish to allow one or more users to add, hide, modify, remove and/or extend content that they provide. In this way, other users may add value to, alter for a new purpose, maintain, and/or otherwise change, existing content. The ability to add content to an empty and/or newly created object is important as well.

When a provider provides content and accompanying control information, she may elect to add control information that enables and/or limits the addition, modification, hiding and/or deletion of said content. This control information may concern:

- the nature and/or location of content that may be added, hidden, modified, and/or deleted;
- portions of content that may be modified, hidden, deleted and/or added to;
- required secure control information over subsequent VDE container content usage in a chain of control and/or locally to added, hidden, and/or modified content;
- requirements that provider-specified notices and/or portions of content accompany added, hidden, deleted and/or modified content and/or the fact that said adding, hiding, modification and/or deletion occurred;
- secure management of limitations and/or requirements concerning content that may be removed, hidden and/or deleted from content, including the amount and/or degree of addition, hiding, modification and/or deletion of content;
- providing notice to a provider or providers that modification, hiding, addition and/or deletion has occurred and/or the nature of said occurrence; and
- other control information concerned with modification, addition, hiding, and/or deleting provider content.

A provider may use this control information to establish an opportunity for other users to add value to and/or maintain existing content in a controlled way. For example, a provider of software development tools may allow other users to add commentary and/or similar and/or complementary tools to their provided objects. A provider of movies may allow commentary and/or promotional materials to be added to their materials. A provider of CAD/CAM specifications to machine tool owners may allow other users to modify objects containing instructions associated with a specification to improve and/or translate said instructions for use with their equipment. A database owner may allow other users to add and/or remove records from a provided database object to allow flexibility and/or maintenance of the database.

Another benefit of introducing control information is the opportunity for a provider to allow other users to alter content for a new purpose. A provider may allow other users to provide content in a new setting.

To attach this control information to content, a provider may be provided with, or if allowed, design and implement, a method or methods for an object that govern addition, hiding, modification and/or deletion of content. Design and implementation of such one or more methods may be performed

using VDE software tools in combination with a PPE 650. The provider may then attach the method(s) to an object and/or provide them separately. A permissions record 808 may include requirements associated with this control information in combination with other control information, or a separate permissions record 808 may be used.

An important aspect of adding or modifying content is the choice of encryption/decryption keys and/or other relevant aspects of securing new or altered content. The provider may specify in their method(s) associated with these processes a technique or techniques to be used for creating and/or selecting the encryption/decryption keys and/or other relevant aspect of securing new and/or altered content. For example, the provider may include a collection of keys, a technique for generating new keys, a reference to a load module that will generate keys, a protocol for securing content, and/or other similar information.

Another important implication is the management of new keys, if any are created and/or used. A provider may require that such keys and reference to which keys were used must be transmitted to the provider, or she may allow the keys and/or securing strategy to remain outside a provider's knowledge and/or control. A provider may also choose an intermediate

course in which some keys must be transmitted and others may remain outside her knowledge and/or control.

An additional aspect related to the management of keys is the management of permissions associated with an object resulting from the addition, hiding, modification and/or deletion of content. A provider may or may not allow a VDE chain of control information user to take some or all of the VDE rules and control information associated with granting permissions to access and/or manipulate VDE managed content and/or rules and control information associated with said resulting object. For example, a provider may allow a first user to control access to new content in an object, thereby requiring any other user of that portion of content to receive permission from the first user. This may or may not, at the provider's discretion, obviate the need for a user to obtain permission from the provider to access the object at all.

Keys associated with addition, modification, hiding and/or deletion may be stored in an independent permissions record or records 808. Said permissions record(s) 808 may be delivered to a provider or providers and potentially merged with an existing permissions record or records, or may remain solely under the control of the new content provider. The creation and content of an initial permissions record 808 and any control information

over the permissions record(s) are controlled by the method(s) associated with activities by a provider. Subsequent modification and/or use of said permission record(s) may involve a provider's method(s), user action, or both. A user's ability to modify and/or use permissions record(s) 808 is dependent on, at least in part, the senior control information associated with the permissions record(s) of a provider.

Distribution Control information

To enable a broad and flexible commercial transaction environment, providers should have the ability to establish firm control information over a distribution process without unduly limiting the possibilities of subsequent parties in a chain of control. The distribution control information provided by the present invention allow flexible positive control. No provider is required to include any particular control, or use any particular strategy, except as required by senior control information. Rather, the present invention allows a provider to select from generic control components (which may be provided as a subset of components appropriate to a provider's specific market, for example, as included in and/or directly compatible with, a VDE application) to establish a structure appropriate for a given chain of handling/control. A provider may also establish control information on their control information that enable and limit modifications to their control information by other users.

The administrative systems provided by the present invention generate administrative "events." These "events" correspond to activities initiated by either the system or a user that correspond to potentially protected processes within VDE. These processes include activities such as copying a permissions record, copying a budget, reading an audit trail record, copying a method, updating a budget, updating a permissions record, updating a method, backing up management files, restoring management files, and the like. Reading, writing, modifying, updating, processing, and/or deleting information from any portion of any VDE record may be administrative events. An administrative event may represent a process that performs one or more of the aforementioned activities on one or more portions of one or more records.

When a VDE electronic appliance 600 encounters an administrative event, that event is typically processed in conjunction with a VDE PPE 650. As in the case of events generally related to access and/or use of content, in most cases administrative events are specified by content providers (including, for example, content creators, distributors, and/or client administrators) as an aspect of a control specified for an object, group and/or class of objects.

For example, if a user initiates a request to distribute permission to use a certain object from a desktop computer to a notebook computer, one of the administrative events generated may be to create a copy of a permissions record that corresponds to the object. When this administrative event is detected by ROS 602, an EVENT method for this type of event may be present. If an EVENT method is present, there may also be a meter, a billing, and a budget associated with the EVENT method. Metering, billing, and budgeting can allow a provider to enable and limit the copying of a permissions record 808.

For example, during the course of processing a control program, a meter, a billing, and a budget and/or audit records may be generated and/or updated. Said audit records may record information concerning circumstances surrounding an administrative event and processing of said event. For example, an audit record may contain a reference to a user and/or system activity that initiated an event, the success or failure of processing said event, the date and/or time, and/or other relevant information.

Referring to the above example of a user with both a desktop and notebook computer, the provider of a permissions record may require an audit record each time a meter for copying said permissions record is processed. The audit record provides a

flexible and configurable control and/or recording environment option for a provider.

In some circumstances, it may be desirable for a provider to limit which aspects of a control component may be modified, updated, and/or deleted. "Atomic element definitions" may be used to limit the applicability of events (and therefore the remainder of a control process, if one exists) to certain "atomic elements" of a control component. For example, if a permissions record 808 is decomposed into "atomic elements" on the fields described in Figure 26, an event processing chain may be limited, for example, to a certain number of modifications of expiration date/time information by specifying only this field in an atomic element definition. In another example, a permissions record 808 may be decomposed into atomic elements based on control sets. In this example, an event chain may be limited to events that act upon certain control sets.

In some circumstances, it may be desirable for a provider to control how administrative processes are performed. The provider may choose to include in distribution records stored in secure database 610 information for use in conjunction with a component assembly 690 that controls and specifies, for example, how processing for a given event in relation to a given method and/or record should be performed. For example, if a provider

wishes to allow a user to make copies of a permissions record 808, she may want to alter the permissions record internally. For example, in the earlier example of a user with a desktop and a notebook computer, a provider may allow a user to make copies of information necessary to enable the notebook computer based on information present in the desktop computer, but not allow any further copies of said information to be made by the notebook VDE node. In this example, the distribution control structure described earlier would continue to exist on the desktop computer, but the copies of the enabling information passed to the notebook computer would lack the required distribution control structure to perform distribution from the notebook computer. Similarly, a distribution control structure may be provided by a content provider to a content provider who is a distributor in which a control structure would enable a certain number of copies to be made of a VDE content container object along with associated copies of permissions records, but the permissions records would be altered (as per specification of the content provider, for example) so as not to allow end-users who received distributor created copies from making further copies for distribution to other VDE nodes.

Although the preceding example focuses on one particular event (copying) under one possible case, similar processes may be used for reading, writing, modifying, updating, processing,

and/or deleting information from records and/or methods under any control relationship contemplated by the present invention. Other examples include: copying a budget, copying a meter, updating a budget, updating a meter, condensing an audit trail, and the like.

Creating Custom Methods

In the preferred embodiment of the present invention, methods may be created "at will," or aliased to another method. These two modes contribute to the superior configurability, flexibility, and positive control of the VDE distribution process. Generally, creating a method involves specifying the required attributes or parameters for the data portion of the method, and then "typing" the method. The typing process typically involves choosing one or more load modules to process any data portions of a method. In addition to the method itself, the process of method creation may also result in a method option subrecord for inclusion in, or modification of, a permissions record, and a notation in the distribution records. In addition to any "standard" load module(s) required for exercise of the method, additional load modules, and data for use with those load modules, may be specified if allowed. These event processing structures control the distribution of the method.

For example, consider the case of a security budget. One form of a typical budget might limit the user to 10Mb of decrypted data per month. The user wishes to move their rights to use the relevant VDE content container object to their notebook. The budget creator might have limited the notebook to the same amount, half the original amount, a prorated amount based on the number of moves budgeted for an object, etc. A distribute method (or internal event processing structure) associated with the budget allows the creator of the budget to make a determination as to the methodology and parameters involved. Of course, different distribution methods may be required for the case of redistribution, or formal distribution of the method. The aggregate of these choices is stored in a permissions record for the method.

An example of the process steps used for the move of a budget record might look something like this:

- 1) Check the move budget (e.g., to determine the number of moves allowed)
- 2) Copy static fields to new record (e.g., as an encumbrance)
- 3) Decrement the Decr counter in the old record (the original budget)
- 4) Increment the Encumbrance counter in the old record

- 5) Write a distribution record
- 6) Write a Distribution Event Id to the new record
- 7) Increment the move meter
- 8) Decrement the move budget
- 9) Increment the Decr counter in the new record

Creating a Budget

In the preferred embodiment, to create a budget, a user manipulates a Graphical User Interface budget distribution application (e.g., a VDE template application). The user fills out any required fields for type(s) of budget, expiration cycle(s), auditor(s), etc. A budget may be specified in dollars, deutsche marks, yen, and/or in any other monetary or content measurement schema and/or organization. The preferred embodiment output of the application, normally has three basic elements. A notation in the distribution portion of secure database 610 for each budget record created, the actual budget records, and a method option record for inclusion in a permissions record. Under some circumstances, a budget process may not result in the creation of a method option since an existing method option may be being used. Normally, all of this output is protected by storage in secure database 610 and/or in one or more administrative objects.

There are two basic modes of operation for a budget distribution application in the preferred embodiment. In the first case, the operator has an unlimited ability to specify budgets. The budgets resulting from this type of activity may be freely used to control any aspect of a distribution process for which an operator has rights, including for use with "security" budgets such as quantities limiting some aspect of usage. For example, if the operator is a "regular person," he may use these budgets to control his own utilization of objects based on a personal accounting model or schedule. If the operator is an authorized user at VISA, the resulting budgets may have broad implications for an entire distribution system. A core idea is that this mode is controlled strictly by an operator.

The second mode of operation is used to create "alias" budgets. These budgets are coupled to a preexisting budget in an operator's system. When an operator fills a budget, an encumbrance is created on the aliased budget. When these types of budgets are created, the output includes two method option subrecords coupled together: the method option subrecord for the aliased budget, and a method option subrecord for the newly created budget. In most cases, the alias budget can be used in place of the original budget if the budget creator is authorized to modify the method options within the appropriate required method record of a permissions record.

For example, assume that a user (client administrator) at a company has the company's VISA budget on her electronic appliance 600. She wants to distribute budget to a network of company users with a variety of preexisting budgets and requirements. She also wants to limit use of the company's VISA budget to certain objects. To do this, she aliases a company budget to the VISA budget. She then modifies (if so authorized) the permissions record for all objects that the company will allow their users to manipulate so that they recognize the company budget in addition to, or instead of, the VISA budget. She then distributes the new permissions records and budgets to her users. The audit data from these users is then reduced against the encumbrance on the company's VISA budget to produce a periodic billing.

In another example, a consumer wants to control his family's electronic appliance use of his VISA card, and prevent his children from playing too many video games, while allowing unlimited use of encyclopedias. In this case, he could create two budgets. The first budget can be aliased to his VISA card, and might only be used with encyclopedia objects (referenced to individual encyclopedia objects and/or to one or more classes of encyclopedia objects) that reference the aliased budget in their explicitly modified permissions record. The second budget could be, for example, a time budget that he redistributes to the family

for use with video game objects (video game class). In this instance, the second budget is a "self-replenishing" security/control budget, that allows, for example, two hours of use per day. The first budget operates in the same manner as the earlier example. The second budget is added as a new required method to permissions records for video games. Since the time budget is required to access the video games, an effective control path is introduced for requiring the second budget -- only permissions records modified to accept the family budget can be used by the children for video games and they are limited to two hours per day.

Sharing and Distributing Rights and Budgets

Move

The VDE "move" concept provided by the preferred embodiment covers the case of "friendly sharing" of rights and budgets. A typical case of "move" is a user who owns several machines and wishes to use the same objects on more than one of them. For example, a user owns a desktop and a notebook computer. They have a subscription to an electronic newspaper that they wish to read on either machine, i.e., the user wishes to move rights from one machine to the other.

An important concept within "move" is the idea of independent operation. Any electronic appliance 600 to which

rights have been moved may contact distributors or clearinghouses independently. For example, the user mentioned above may want to take their notebook on the road for an extended period of time, and contact clearinghouses and distributors without a local connection to their desktop.

To support independent operation, the user should be able to define an account with a distributor or clearinghouse that is independent of the electronic appliance 600 she is using to connect. The transactions must be independently traceable and reconcilable among and between machines for both the end user and the clearinghouse or distributor. The basic operations of moving rights, budgets, and bitmap or compound meters between machines is also supported.

Redistribution

Redistribution forms a UDE middle ground between the "friendly sharing" of "move," and formal distribution. Redistribution can be thought of as "anonymous distribution" in the sense that no special interaction is required between a creator, clearinghouse, or distributor and a redistributor. Of course, a creator or distributor does have the ability to limit or prevent redistribution.

Unlike the "move" concept, redistribution does not imply independent operation. The redistributor serves as one point of contact for users receiving redistributed rights and/or budgets, etc. These users have no knowledge of, or access to, the clearinghouse (or and/or distributor) accounts of the redistributor. The redistributor serves as an auditor for the rights and/or budgets, etc. that they redistribute, unless specifically overridden by restrictions from distributors and/or clearinghouses. Since redistributees (recipients of redistributed rights and/or budgets, etc.) would place a relatively unquantifiable workload on clearinghouses, and furthermore, since a redistributor would be placing himself at an auditable risk (responsible for all redistributed rights and/or budgets, etc.), the audit of rights, budgets, etc. of redistributees by redistributors is assumed as the default case in the preferred embodiment.

Distribution

Distribution involves three types of entity. Creators usually are the source of distribution. They typically set the control structure "context" and can control the rights which are passed into a distribution network. Distributors are users who form a link between object (content) end users and object (content) creators. They can provide a two-way conduit for rights and audit data. Clearinghouses may provide independent

financial services, such as credit and/or billing services, and can serve as distributors and/or creators. Through a permissions and budgeting process, these parties collectively can establish fine control over the type and extent of rights usage and/or auditing activities.

Encumbrance

An "encumbrance" is a special type of VDE budget. When that a budget distribution of any type occurs, an "encumbrance" may be generated. An encumbrance is indistinguishable from an original budget for right exercise (e.g., content usage payment) purposes, but is uniquely identified within distribution records as to the amount of the encumbrance, and all necessary information to complete a shipping record to track the whereabouts of an encumbrance. For right exercise purposes, an encumbrance is identical to an original budget; but for tracking purposes, it is uniquely identifiable.

In the preferred embodiment of the present invention, a Distribution Event ID will be used by user VDE nodes and by clearinghouse services to track and reconcile encumbrances, even in the case of asynchronous audits. That is, the "new" encumbrance budget is unique from a tracking point of view, but indistinguishable from a usage point of view.

Unresolved encumbrances are a good intermediate control for a VDE distribution process. A suitable "grace period" can be introduced during which encumbrances must be resolved. If this period elapses, an actual billing or payment may occur. However, even after the interval has expired and the billing and/or payment made, an encumbrance may still be outstanding and support later reconciliation. In this case, an auditor may allow a user to gain a credit, or a user may connect to a VDE node containing an encumbered budget, and resolve an amount as an internal credit. In some cases, missing audit trails may concern a distributor sufficiently to revoke redistribution privileges if encumbrances are not resolved within a "grace period," or if there are repeated grace period violations or if unresolved encumbrances are excessively large.

Encumbrances can be used across a wide variety of distribution modes. Encumbrances, when used in concert with aliasing of budgets, opens important additional distribution possibilities. In the case of aliasing a budget, the user places himself in the control path for an object -- an aliased budget may only be used in conjunction with permissions records that have been modified to recognize it. An encumbrance has no such restrictions.

For example, a user may want to restrict his children's use of his electronic, VDE node VISA budget. In this case, the user can generate an encumbrance on his VISA budget for the children's family alias budget, and another for his wife that is a transparent encumbrance of the original VISA budget. BigCo may use a similar mechanism to distribute VISA budget to department heads, and aliased BigCo budget to users directly.

Account Numbers and User IDs

In the preferred embodiment, to control access to clearinghouses, users are assigned account numbers at clearinghouses. Account numbers provide a unique "instance" value for a secure database record from the point of view of an outsider. From the point of view of an electronic appliance 600 site, the user, group, or group/user ids provide the unique instance of a record. For example, from the point of view of VISA, your Gold Card belongs to account number #123456789. From the point of view of the electronic appliance site (for example, a server at a corporation), the Gold card might belong to user id 1023. In organizations which have plural users and/or user groups using a VDE node, such users and/or user groups will likely be assigned unique user IDs. differing budgets and/or other user rights may be assigned to different users and/or user groups and/or other VDE control information may be applied on a differing manner to electronic content and/or appliance usage by

users assigned with different such IDs. Of course, both a clearinghouse and a local site will likely have both pieces of information, but "used data" versus the "comment data" may differ based on perspective.

In the preferred embodiment case of "move," an account number stored with rights stays the same. In the preferred embodiment of other forms of distribution, a new account number is required for a distributee. This may be generated automatically by the system, or correspond to a methodology developed by a distributor or redistributor. Distributors maintain account numbers (and associated access secrets) in their local name services for each distributee. Conversely, distributees' name services may store account numbers based on user id for each distributor. This record usually is moved with other records in the case of move, or is generated during other forms of distribution.

Organizations (including families) may automatically assign unique user IDs when creating control information (e.g., a budget) for a new user or user group.

Requirements Record

In order to establish the requirements, and potentially options, for exercising a right associated with a VDE content

container object before one or more required permissions records are received for that object, a requirements record may exist in the private header of such an object. This record will help the user establish what they have, and what they need from a distributor prior to forming a connection. If the requirements or possibilities for exercising a particular right have changed since such an object was published, a modified requirements record may be included in a container with an object (if available and allowed), or a new requirements record may be requested from a distributor before registration is initiated. Distributors may maintain "catalogs" online, and/or delivered to users, of collections of requirements records and/or descriptive information corresponding to objects for which they may have ability to obtain and/or grant rights to other users.

Passing an Audit

In the preferred embodiment of VDE there may be at least two types of auditing. In the case of budget distribution, billing records that reflect consumption of a budget generally need to be collected and processed. In the case of permissions distribution, usage data associated with an object are also frequently required.

In order to effect control over an object, a creator may establish the basic control information associated with an object.

This is done in the formulation of permissions, the distribution of various security, administrative and/or financial budgets, and the level of redistribution that is allowed, etc. Distributors (and redistributors) may further control this process within the rights, budgets, etc. (senior control information) they have received.

For example, an object creator may specify that additional required methods may be added freely to their permissions records, establish no budget for this activity, and allow unlimited redistribution of this right. As an alternative example, a creator may allow moving of usage rights by a distributor to half a dozen subdistributors, each of whom can distribute 10,000 copies, but with no redistribution rights being allowed to be allocated to subdistributors' (redistributors') customers. As another example, a creator may authorize the moving of usage rights to only 10 VDE nodes, and to only one level of distribution (no redistribution). Content providers and other contributors of control information have the ability through the use of permissions records and/or component assemblies to control rights other users are authorized to delegate in the permissions records they send to those users, so long as such right to control one, some, or all such rights of other users is either permitted or restricted (depending on the control information distribution model). It is possible and often desirable, using VDE, to construct a mixed model in which a distributor is restricted from

controlling certain rights of subsequent users and is allowed to control other rights. VDE control of rights distribution in some VDE models will in part or whole, at least for certain one or more "levels" of a distribution chain, be controlled by electronic content control information providers who are either not also providers of the related content or provide only a portion of the content controlled by said content control information. for example, in certain models, a clearinghouse might also serve as a rights distribution agent who provides one or more rights to certain value chain participants, which one or more rights may be "attached" to one or more rights to use the clearinghouse's credit (if said clearinghouse is, at least in part, a financial clearinghouse (such a control information provider may alternatively, or in addition, restrict other users' rights.

A content creator or other content control information provider may budget a user (such as a distributor) to create an unlimited number of permissions records for a content object, but revoke this right and/or other important usage rights through an expiration/termination process if the user does not report his usage (provide an audit report) at some expected one or more points in time and/or after a certain interval of time (and/or if the user fails to pay for his usage or violates other aspects of the agreement between the user and the content provider). This termination (or suspension or other specified consequence) can be

enforced, for example, by the expiration of time-aged encryption keys which were employed to encrypt one or more aspects of control information. This same termination (or other specified consequence such as budget reduction, price increase, message displays on screen to users, messages to administrators, etc.) can also be the consequence of the failure by a user or the users VDE installation to complete a monitored process, such as paying for usage in electronic currency, failure to perform backups of important stored information (e.g., content and/or appliance usage information, control information, etc.), failure to use a repeated failure to use the proper passwords or other identifiers, etc.).

Generally, the collection of audit information that is collected for reporting to a certain auditor can be enforced by expiration and/or other termination processes. For example, the user's VDE node may be instructed (a) from an external source to no longer perform certain tasks, (b) carries within its control structure information informing it to no longer perform certain tasks, or (c) is otherwise no longer able to perform certain tasks. The certain tasks might comprise one or more enabling operations due to a user's (or installation's) failure to either report said audit information to said auditor and/or receive back a secure confirmation of receipt and/or acceptance of said audit information. If an auditor fails to receive audit information from

a user (or some other event fails to occur or occur properly), one or more time-aged keys which are used, for example, as a security component of an embodiment of the present invention, may have their aging suddenly accelerated (completed) so that one or more processes related to said time-aged keys can no longer be performed.

Authorization Access Tags and Modification Access Tags

In order to enable a user VDE installation to pass audit information to a VDE auditing party such as a Clearinghouse, VDE allows a VDE auditing party to securely, electronically communicate with the user VDE installation and to query said installation for certain or all information stored within said installation's secure sub-system, depending on said auditing party's rights (said party shall normally be unable to access securely stored information that said party is not expressly authorized to access, that is one content provider will normally not be authorized to access content usage information related to content provided by a different content provider). The auditing party asserts a secure secret (e.g., a secure tag) that represents the set of rights of the auditor to access certain information maintained by said subsystem. If said subsystem validates said tag, the auditing party may then receive auditing information that it is allowed to request and receive.

Great flexibility exists in the enforcement of audit trail requirements. For example, a creator (or other content provider or control information provider or auditor in an object's or audit report's chain of handling) may allow changes by an auditor for event trails, but not allow anyone but themselves to read those trails, and limit the redistribution of this right to, for example, six levels. Alternatively, a creator or other controlling party may give a distributor the right to process, for example, 100,000 audit records (and/or, for example, the right to process 12 audit records from a given user) before reporting their usage. If a creator or other controlling party desires, he may allow (and/or require) separate (and containing different, a subset of, overlapping, or the same information) audit "packets" containing audit information, certain of said audit information to be processed by a distributor and certain other of said audit information to be passed back to the creator and/or other auditors (each receiving the same, overlapping, a subset of, or different audit information). Similarly, as long as allowed by, for example, an object creator, a distributor (or other content and/or control information provider) may require audit information to be passed back to it, for example, after every 50,000 audit records are processed (or any other unit of quantity and/or after a certain time interval and/or at a certain predetermined date) by a redistributor. In the preferred embodiment, audit rules, like other control structures, may be stipulated at any stage of a

distribution chain of handling as long as the right to stipulate said rules has not been restricted by a more "senior" object and/or control information distributing (such as an auditing) participant.

Audit information that is destined for different auditors may be encrypted by different one or more encryption keys which have been securely provided by each auditor's VDE node and communicated for inclusion in a user's permissions record(s) as a required step, for example, during object registration. This can provide additional security to further ensure (beyond the use of passwords and/or other identification information and other VDE security features) that an auditor may only access audit information to which he is authorized. In one embodiment, encrypted (and/or unencrypted) "packets" of audit information (for example, in the form of administrative objects) may be bound for different auditors including a clearinghouse and/or content providers and/or other audit information users (including, for example, market analysts and/or list purveyors). The information may pass successively through a single chain of handling, for example, user to clearinghouse to redistributor to distributor to publisher/object creator, as specified by VDE audit control structures and parameters. Alternatively, encrypted (or, normally less preferably, unencrypted) audit packets may be required to be dispersed directly from a user to a plurality of

auditors, some one or more who may have the responsibility to "pass along" audit packets to other auditors. In another embodiment, audit information may be passed, for example, to a clearinghouse, which may then redistribute all and/or appropriate subsets of said information (and/or some processed result) to one or more other parties, said redistribution employing VDE secure objects created by said clearinghouse.

An important function of an auditor (receiver of audit information) is to pass administrative events back to a user VDE node in acknowledgement that audit information has been received and/or "recognized." In the preferred embodiment, the receipt and/or acceptance of audit information may be followed by two processes. The first event will cause the audit data at a VDE node which prepared an audit report to be deleted, or compressed into, or added to, one or more summary values. The second event, or set of events, will "inform" the relevant security (for example, termination and/or other consequence) control information (for example, budgets) at said VDE node of the audit receipt, modify expiration dates, provide key updates, and/or etc. In most cases, these events will be sent immediately to a site after an audit trail is received. In some cases, this transmission may be delayed to, for example, first allow processing of the audit trail and/or payment by a user to an auditor or other party.

In the preferred embodiment, the administrative events for content objects and independently distributed methods/component assemblies are similar, but not necessarily identical. For example, key updates for a budget may control encryption of a billing trail, rather than decryption of object content. The billing trail for a budget is in all respects a method event trail. In one embodiment, this trail must include sufficient references into distribution records for encumbrances to allow reconciliation by a clearinghouse. This may occur, for example, if a grace period elapses and the creator of a budget allows unresolved encumbrances to ultimately yield automatic credits if an expired encumbrance is "returned" to the creator.

Delivery of audit reports through a path of handling may be in part insured by an inverse (return of information) audit method. Many VDE methods have at least two pieces: a portion that manages the process of producing audit information at a user's VDE node; and a portion that subsequently acts on audit data. In an example of the handling of audit information bound for a plurality of auditors, a single container object is received at a clearinghouse (or other auditor). This container may contain (a) certain encrypted audit information that is for the use of the clearinghouse itself, and (b) certain other encrypted audit information bound for other one or more auditor parties. The two sets of information may have the same, overlapping and in part

different, or entirely different, information content.

Alternatively, the clearinghouse VDE node may be able to work with some or all of the provided audit information. The audit information may be, in part, or whole, in some summary and/or analyzed form further processed at the clearinghouse and/or may be combined with other information to form a, at least in part, derived set of information and inserted into one or more at least in part secure VDE objects to be communicated to said one or more (further) auditor parties. When an audit information container is securely processed at said clearinghouse VDE node by said inverse (return) audit method, the clearinghouse VDE node can create one or more VDE administrative objects for securely carrying audit information to other auditors while separately processing the secure audit information that is specified for use by said clearinghouse. Secure audit processes and credit information distribution between VDE participants normally takes place within the secure VDE "black box," that is processes are securely processed within secure VDE PPE650 and audit information is securely communicated between the VDE secure subsystems of vDE participants employing VDE secure communication techniques (e.g., public key encryption, and authentication).

This type of inverse audit method may specify the handling of returned audit information, including, for example, the local

processing of audit information and/or the secure passing along of audit information to one or more auditor parties. If audit information is not passed to one or more other auditor parties as may be required and according to criteria that may have been set by said one or more other auditor parties and/or content providers and/or control information providers during a permissions record specification and/or modification process, the failure to, for example, receive notification of successful transfer of required audit information by an auditor party, e.g., a content provider, can result in the disablement of at least some capability of the passing through party's VDE node (for example, disablement of the ability to further perform certain one or more VDE managed business functions that are related to object(s) associated with said audit or party). In this preferred embodiment example, when an object is received by an auditor, it is automatically registered and permissions record(s) contents are entered into the secure management database of the auditor's VDE node.

One or more permissions records that manage the creation and use of an audit report object (and may manage other aspects of object use as well) may be received by a user's system during an audit information reporting exchange (or other electronic interaction between a user and an auditor or auditor agent). Each received permissions record may govern the creation of the

next audit report object. After the reporting of audit information, a new permissions record may be required at a user's VDE node to refresh the capability of managing audit report creation and audit information transfer for the next audit reporting cycle. In our above example, enabling an auditor to supply one or more permissions records to a user for the purpose of audit reporting may require that an auditor (such as a clearinghouse) has received certain, specified permissions records itself from "upstream" auditors (such as, for example, content and/or other content control information providers). Information provided by these upstream permissions records may be integrated into the one or more permissions records at an auditor VDE (e.g., clearinghouse) installation that manage the permissions record creation cycle for producing administrative objects containing permissions records that are bound for users during the audit information reporting exchange. If an upstream auditor fails to receive, and/or is unable to process, required audit information, this upstream auditor may fail to provide to the clearinghouse (in this example) the required permissions record information which enables a distributor to support the next permission record creation/auditing cycle for a given one or more objects (or class of objects). As a result, the clearinghouse's VDE node may be unable to produce the next cycle's permissions records for users, and/or perform some other important process. This VDE audit reporting control process may be entirely electronic process

management involving event driven VDE activities at both the intended audit information receiver and sender and employing both their secure PPE650 and secure VDE communication techniques.

In the preferred embodiment, each time a user registers a new object with her own VDE node, and/or alternatively, with a remote clearinghouse and/or distributor VDE node, one or more permissions records are provided to, at least in part, govern the use of said object. The permissions records may be provided dynamically during a secure UDE registration process (employing the VDE installation secure subsystem), and/or may be provided following an initial registration and received at some subsequent time, e.g. through one or more separate secure VDE communications, including, for example, the receipt of a physical arrangement containing or otherwise carrying said information. At least one process related to the providing of the one or more permissions records to a user can trigger a metering event which results in audit information being created reflecting the user's VDE node's, clearinghouse's, and/or distributor's permissions records provision process. This metering process may not only record that one or more permissions records have been created. It may also record the VDE node name, user name, associated object identification information, time, date, and/or other identification information. Some or all of this information can

become part of audit information securely reported by a clearinghouse or distributor, for example, to an auditing content creator and/or other content provider. This information can be reconciled by secure VDE applications software at a receiving auditor's site against a user's audit information passed through by said clearinghouse or distributor to said auditor. For each metered one or more permissions records (or set of records) that were created for a certain user (and/or VDE node) to manage use of certain one or more VDE object(s) and/or to manage the creation of VDE object audit reports, it may be desirable that an auditor receive corresponding audit information incorporated into an, at least in part, encrypted audit report. This combination of metering of the creation of permissions records; secure, encrypted audit information reporting processes; secure VDE subsystem reconciliation of metering information reflecting the creation of registration and/or audit reporting permissions with received audit report detail; and one or more secure VDE installation expiration and/or other termination and/or other consequence processes; taken together significantly enhances the integrity of the VDE secure audit reporting process as a trusted, efficient, commercial environment.

Secure Document Management Example

VDE 100 may be used to implement a secure document management environment. The following are some examples of how this can be accomplished.

In one example, suppose a law firm wants to use VDE 100 to manage documents. In this example, a law firm that is part of a litigation team might use VDE in the following ways:

1. to securely control access to, and/or other usage of, confidential client records,
2. to securely control access, distribution, and/or other rights to documents and memoranda created at the law firm,
3. to securely control access and other use of research materials associated with the case,
4. to securely control access and other use, including distribution of records, documents, and notes associated with the case,
5. to securely control how other firms in the litigation team may use, including change, briefs that have been distributed for comment and review,
6. to help manage client billing.

The law firm may also use VDE to electronically file briefs with the court (presuming the court is also VDE capable) including providing secure audit verification of the ID (e.g., digital signature) of filers and other information pertinent to said filing procedure.

In this example, the law firm receives in VDE content containers documents from their client's VDE installation secure subsystem(s). Alternatively, or in addition, the law firm may receive either physical documents which may be scanned into electronic form, and/or they receive electronic documents which have not yet been placed in VDE containers. The electronic form of a document is stored as a VDE container (object) associated with the specific client and/or case. The VDE container mechanism supports a hierarchical ordering scheme for organizing files and other information within a container; this mechanism may be used to organize the electronic copies of the documents within a container. A VDE container is associated with specific access control information and rights that are described in one or more permissions control information sets (PERCs) associated with that container. In this example, only those members of the law firm who possess a VDE instance, an appropriate PERC, and the VDE object that contains the desired document, may use the document. Alternatively or in addition, a law firm member may use a VDE instance which has been

installed on the law firm's network server. In this case, the member must be identified by an appropriate PERC and have access to the document containing VDE object (in order to use the server VDE installation). Basic access control to electronic documents is enabled using the secure subsystem of one or more user VDE installations.

VDE may be used to provide basic usage control in several ways. First, it permits the "embedding" of multiple containers within a single object. Embedded objects permit the "nesting" of control structures within a container. VDE also extends usage control information to an arbitrary granular level (as opposed to a file based level provided by traditional operating systems) and provides flexible control information over any action associated with the information which can be described as a VDE controlled process. For example, simple control information may be associated with viewing the one or more portions of documents and additional control information may be associated with editing, printing and copying the same and/or different one or more portions of these same documents.

In this example, a "client" container contains all documents that have been provided by the client (documents received in other containers can be securely extracted and embedded into the VDE client container using VDE extraction and embedding

capabilities). Each document in this example is stored as an object within the parent, client VDE container. The "client" container also has several other objects embedded within it; one for each attorney to store their client notes, one (or more) for research results and related information, and at least one for copies of letters, work papers, and briefs that have been created by the law firm. The client container may also contain other information about the client, including electronic records of billing, time, accounting, and payments. Embedding VDE objects within a parent VDE content container provides a convenient way to securely categorize and/or store different information that shares similar control information. All client provided documents may, for example, be subject to the same control structures related to use and non-disclosure. Attorney notes may be subject to control information, for example, their use may be limited to the attorney who created the notes and those attorneys to whom such creating attorney expressly grants access rights. Embedded containers also provide a convenient mechanism to control collections of dissimilar information. For example, the research object(s) may be stored in the form of (or were derived from) VDE "smart objects" that contain the results of research performed by that object. Research results related to one aspect of the case retrieved from a VDE enabled LEXIS site might be encapsulated as one smart object; the results of another session related to another (or the same) aspect of the case may be encapsulated as a

different object. Smart objects are used in this example to help show that completely disparate and separately delivered control information may be incorporated into a client container as desired and/or required to enforce the rights of providers (such as content owners).

Control structures may be employed to manage any variety of desired granularities and/or logical document content groupings; a document, page, paragraph, topically related materials, etc. In this example, the following assumptions are made: client provided documents are controlled at the page level, attorney notes are controlled at the document level on an attorney by attorney basis, court filings and briefs are controlled at a document level, research information is controlled at whatever level the content provider specifies at the time the research was performed, and certain highly confidential information located in various of the above content may be identified as subject to display and adding comments only, and only by the lead partner attorneys, with only the creator and/or embedder of a given piece of content having the right to be otherwise used (printed, extracted, distributed, etc).

In general, container content in this example is controlled with respect to distribution of rights. This control information are associated at a document level for all internally generated

documents, at a page level for client level documents, and at the level specified by the content provider for research documents.

VDE control information can be structured in either complex or simple structures, depending on the participant's desires. In some cases, a VDE creator will apply a series of control structure definitions that they prefer to use (and that are supported by the VDE application managing the specification of rules and control information, either directly, or through the use of certified application compatible VDE component assemblies.

In this example, the law firm sets up a standard VDE client content container for a new client at the time they accept the case. A law firm VDE administrator would establish a VDE group for the new client and add the VDE IDs of the attorneys at the firm that are authorized to work on the case, as well as provide, if appropriate, one or more user template applications. These templates provide, for example, one or more user interfaces and associated control structures for selection by users of additional and/or alternative control functions (if allowed by senior control information), entry of control parameter data, and/or performing user specific administrative tasks. The administrator uses a creation tool along with a predefined creation template to create the container. This creation template specifies the document usage (including distribution control

information) for documents as described above. Each electronic document from the client (including letters, memoranda, E-mail, spreadsheet, etc.) are then added to the container as separate embedded objects. Each new object is created using a creation template that satisfies that the default control structures specified with the container as required for each new object of a given type.

As each attorney works on the case, they may enter notes into an object stored within the client's VDE container. These notes may be taken using a VDE aware word processor already in use at the law firm. In this example, a VDE redirector handles the secure mapping of the word processor file requests into the VDE container and its objects through the use of VDE control processes operating with one or more VDE PPEs. Attorney note objects are created using the default creation template for the document type with assistance from the attorney if the type cannot be automatically determined from the content. This permits VDE to automatically detect and protect the notes at the predetermined level, e.g. document, page, paragraph.

Research can be automatically managed using VDE. Smart objects can be used to securely search out, pay for if necessary, and retrieve information from VDE enabled information resources on the information highway.

Examples of such resources might include LEXIS, Westlaw, and other related legal databases. Once the information is retrieved, it may be securely embedded in the VDE content client container. If the smart object still contains unreleased information, the entire smart object may be embedded in the client's VDE container. This places the unreleased information under double VDE control requirements: those associated with releasing the information from smart object (such as payment and/or auditing requirements) and those associated with access to, or other usage of, client information of the specified type.

Briefs and other filings may be controlled in a manner similar to that for attorney notes. The filings may be edited using the standard word processors in the law firm; with usage control structures controlling who may review, change, and/or add to the document (or, in a more sophisticated example, a certain portion of said document). VDE may also support electronic filing of briefs by providing a trusted source for time/date stamping and validation of filed documents.

When the client and attorney want to exchange confidential information over electronic mail or other means, VDE can play an important role in ensuring that information exchanged under privilege, properly controlled, and not

inappropriately released and/or otherwise used. The materials (content) stored in a VDE content container object will normally be encrypted. Thus wrapped, a VDE object may be distributed to the recipient without fear of unauthorized access and/or other use. The one or more authorized users who have received an object are the only parties who may open that object and view and/or manipulate and/or otherwise modify its contents and VDE secure auditing ensures a record of all such user content activities. VDE also permits the revocation of rights to use client/attorney privileged information if such action becomes necessary, for example, after an administrator review of user usage audit information.

Large Organization Example

In a somewhat more general example, suppose an organization (e.g., a corporation or government department) with thousands of employees and numerous offices disposed throughout a large geographic area wishes to exercise control over distribution of information which belongs to said organization (or association). This information may take the form of formal documents, electronic mail messages, text files, multimedia files, etc., which collectively are referred to as "documents."

Such documents may be handled by people (referred to as "users") and/or by computers operating on behalf of users. The documents may exist both in electronic form for storage and transmission and in paper form for manual handling.

These documents may originate wholly within the organization, or may be created, in whole or in part, from information received from outside the organization. Authorized persons within the organization may choose to release documents, in whole or in part, to entities outside the organization. Some such entities may also employ VDE 100 for document control, whereas others may not.

Document Control Policies

The organization as a whole may have a well-defined policy for access control to, and/or other usage control of documents. This policy may be based on a "lattice model" of information flow, in which documents are characterized as having one or more hierarchical "classification" security attributes 9903 and zero or more non-hierarchical "compartment" security attributes, all of which together comprise a sensitivity security attribute.

The classification attributes may designate the overall level of sensitivity of the document as an element of an ordered set. For example, the set "unclassified," "confidential," "secret,"

“top secret” might be appropriate in a government setting, and the set “public,” “internal,” “confidential,” “registered confidential” might be appropriate in a corporate setting.

The compartment attributes may designate the document's association with one or more specific activities within the organization, such as departmental subdivisions (e.g., “research,” “development,” “marketing”) or specific projects within the organization.

Each person using an electronic appliance 600 would be assigned, by an authorized user, a set of permitted sensitivity attributes to designate those documents, or one or more portions of certain document types, which could be processed in certain one or more ways, by the person's electronic appliance. A document's sensitivity attribute would have to belong to the user's set of permitted sensitivity values to be accessible.

In addition, the organization may desire to permit users to exercise control over specific documents for which the user has some defined responsibility. As an example, a user (the “originating user”) may wish to place an “originator controlled” (“ORCON”) restriction on a certain document, such that the document may be transmitted and used only by those specific other users whom he designates (and only in certain, expressly

authorized ways). Such a restriction may be flexible if the "distribution list" could be modified after the creation of the document, specifically in the event of someone requesting permission from the originating user to transmit the document outside the original list of authorized recipients. The originating user may wish to permit distribution only to specific users, defined groups of users, defined geographic areas, users authorized to act in specific organizational roles, or a combination of any or all such attributes.

In this example, the organization may also desire to permit users to define a weaker distribution restriction such that access to a document is limited as above, but certain or all information within the document may be extracted and redistributed without further restriction by the recipients.

The organization and/or originating users may wish to know to what uses or geographic locations a document has been distributed. The organization may wish to know where documents with certain protection attributes have been distributed, for example, based on geographic information stored in site configuration records and/or name services records.

A user may wish to request a "return receipt" for a distributed document, or may wish to receive some indication of

how a document has been handled by its recipients (e.g., whether it has been viewed, printed, edited and/or stored), for example, by specifying one or more audit requirements (or methods known to have audit requirements) in a PERC associated with such document(s).

User Environment

In an organization (or association) such as that described above, users may utilize a variety of electronic appliances 600 for processing and managing documents. This may include personal computers, both networked and otherwise, powerful single-user workstations, and servers or mainframe computers. To provide support for the control information described in this example, each electronic appliance that participates in use and management of VDE-protected documents may be enhanced with a VDE secure subsystem supporting an SPE 503 and/or HPE 655.

In some organizations, where the threats to secure operation are relatively low, an HPE 655 may suffice. In other organizations (e.g., government defense), it may be necessary to employ an SPE 503 in all situations where VDE-protected documents are processed. The choice of enhancement environment and technology may be different in different of the organization. Even if different types of PPE 650 are used within

an organization to serve different requirements, they may be compatible and may operate on the same types (or subsets of types) of documents.

Users may employ application programs that are customized to operate in cooperation with the VDE for handling of VDE-protected documents. Examples of this may include VDE-aware document viewers, VDE aware electronic mail systems, and similar applications. Those programs may communicate with the PPE 650 component of a user's electronic appliance 600 to make VDE-protected documents available for use while limiting the extent to which their contents may be copied, stored, viewed, modified, and/or transmitted and/or otherwise further distributed outside the specific electronic appliance.

Users may wish to employ commercial, off-the-shelf ("COTS") operating systems and application programs to process the VDE-protected documents. One approach to permit the use of COTS application programs and operating systems would be to allow such use only for documents without restrictions on redistribution. The standard VDE operating system redirector would allow users to access VDE-protected documents in a manner equivalent to that for files. In such an approach, however, a chain of control for metering and/or auditing use may

be "broken" to some extent at the point that the protected object was made available to the COTS application. The fingerprinting (watermarking) techniques of VDE may be used to facilitate further tracking of any released information.

A variety of techniques may be used to protect printing of protected documents, such as, for example: server-based decryption engines, special fonts for "fingerprinting," etc.

Another approach to supporting COTS software would use the VDE software running on the user's electronic appliance to create one or more "virtual machine" environments in which COTS operating system and application programs may run, but from which no information may be permanently stored or otherwise transmitted except under control of VDE. Such an environment would permit VDE to manage all VDE-protected information, yet may permit unlimited use of COTS applications to process that information within the confines of a restricted environment. The entire contents of such an environment could be treated by VDE 100 as an extension to any VDE-protected documents read into the environment. Transmission of information out of the environment could be governed by the same rules as the original document(s).

"Coarse-Grain" Control Capabilities

As mentioned above, an organization may employ VDE-enforced control capabilities to manage the security, distribution, integrity, and control of entire documents. Some examples of these capabilities may include:

- 1) A communication channel connecting two or more electronic appliances 600 may be assigned a set of permitted sensitivity attributes. Only documents whose sensitivity attributes belong to this set would be permitted to be transmitted over the channel. This could be used to support the Device Labels requirement of the Trusted Computer System Evaluation Criteria (TCSEC).
- 2) A writable storage device (e.g., fixed disk, diskette, tape drive, optical disk) connected to or incorporated in an electronic appliance 600 may be assigned a set of permitted sensitivity attributes. Only documents whose sensitivity attributes belong to this set would be permitted to be stored on the device. This could be used to support the TCSEC Device Labels requirement.

- 3) A document may have a list of users associated with it representing the users who are permitted to "handle" the document. This list of users may represent, for example, the only users who may view the document, even if other users receive the document container, they could not manipulate the contents. This could be used to support the standard ORCON handling caveat.
- 4) A document may have an attribute designating its originator and requiring an explicit permission to be granted by an originator before the document's content could be viewed. This request for permission may be made at the time the document is accessed by a user, or, for example, at the time one user distributes the document to another user. If permission is not granted, the document could not be manipulated or otherwise used.
- 5) A document may have an attribute requiring that each use of the document be reported to the document's originator. This may be used by an originator to gauge the distribution of the document. Optionally, the report may be required to have been made successfully before any use of the document is

permitted, to ensure that the use is known to the controlling party at the time of use. Alternatively, for example, the report could be made in a deferred ("batch") fashion.

- 6) A document may have an attribute requiring that each use of the document be reported to a central document tracking clearinghouse. This could be used by the organization to track specific documents, to identify documents used by any particular user and/or group of users to track documents with specific attributes (e.g., sensitivity), etc. Optionally, for example, the report may be required to have been made successfully before any use of the document is permitted.

- 7) A VDE protected document may have an attribute requiring that each use of the document generate a "return receipt," to an originator. A person using the document may be required to answer specific questions in order to generate a return receipt, for example by indicating why the document is of interest, or by indicating some knowledge of the document's contents (after reading it). This may be used as assurance that the document had been

handled by a person, not by any automated software mechanism.

- 8) A VDE protected document's content may be made available to a VDE-unaware application program in such a way that it is uniquely identifiable (traceable) to a user who caused its release. Thus, if the released form of the document is further distributed, its origin could be determined. This may be done by employing VDE "fingerprinting" for content release. Similarly, a printed VDE protected document may be marked in a similar, VDE fingerprinted unique way such that the person who originally printed the document could be determined, even if copies have since been made.

- 9) Usage of VDE protected documents could be permitted under control of budgets that limit (based on size, time of access, etc.) access or other usage of document content. This may help prevent wholesale disclosure by limiting the number of VDE documents accessible to an individual during a fixed time period. For example, one such control might permit a user, for some particular class of documents, to view at most 100 pages/day, but only print 10

pages/day and permit printing only on weekdays between nine and five. As a further example, a user might be restricted to only a certain quantity of logically related, relatively "contiguous" and/or some other pattern (such as limiting the use of a database's records based upon the quantity of records that share a certain identifier in field) of VDE protected document usage to identify, for example, the occurrence of one or more types of excessive database usage (under normal or any reasonable circumstances). As a result, VDE content providers can restrict usage of VDE content to acceptable usage characteristics and thwart and/or identify (for example, by generating an exception report for a VDE administrator or organization supervisor) user attempts to inappropriately use, for example, such an information database resource.

These control capabilities show some examples of how VDE can be used to provide a flexible, interactive environment for tracking and managing sensitive documents. Such an environment could directly trace the flow of a document from person to person, by physical locations, by organizations, etc. It would also permit specific questions to be answered such as "what persons outside the R&D department have received any

R&D-controlled document.” Because the control information is carried with each copy of a VDE protected document, and can ensure that central registries are updated and/or that originators are notified of document use, tracking can be prompt and accurate.

This contrasts with traditional means of tracking paper documents: typically, a paper-oriented system of manually collected and handled receipts is used. Documents may be individually copy-numbered and signed for, but once distributed are not actively controlled. In a traditional paper-oriented system, it is virtually impossible to determine the real locations of documents; what control can be asserted is possible only if all parties strictly follow the handling rules (which are at best inconvenient).

The situation is no better for processing documents within the context of ordinary computer and network systems. Although said systems can enforce access control information based on user identity, and can provide auditing mechanisms for tracking accesses to files, these are low-level mechanisms that do not permit tracking or controlling the flow of content. In such systems, because document content can be freely copied and manipulated, it is not possible to determine where document content has gone, or where it came from. In addition, because the

control mechanisms in ordinary computer operating systems operate at a low level of abstraction, the entities they control are not necessarily the same as those that are manipulated by users. This particularly causes audit trails to be cluttered with voluminous information describing uninteresting activities.

Fine-Grain[®] Control Capabilities

In addition to controlling and managing entire documents, users may employ customized VDE-aware application software to control and manage individual modifications to documents.

Examples of these capabilities include the following:

- 1) A VDE content user may be permitted to append further information to a VDE document to indicate a proposed alternative wording. This proposed alteration would be visible to all other users (in addition to the original text) of the document but would (for example) be able to be incorporated into the actual text only by the document's owner.

- 2) A group of VDE users could be permitted to modify one or more parts of a document in such a way that each individual alteration would be unambiguously traceable to the specific user who performed it. The rights to modify certain portions of a document, and

the extension of differing sets of rights to different users, allows an organization or secure environment to provide differing permissions enabling different rights to users of the same content.

- 3) A group of users could create a VDE document incrementally, by building it from individual contributions. These contributions would be bound together within a single controlled document, but each would be individually identified, for example, through their incorporation in VDE content containers as embedded container objects.
- 4) VDE control and management capabilities could be used to track activities related to individual document areas, for instance recording how many times each section of a document was viewed.

Example - VDE Protected Content Repository

As the "Digital Highway" emerges, there is increased discussion concerning the distribution of content across networks and, in particular, public networks such as the Internet. Content may be made available across public networks in several ways including:

- “mailing” content to a user in response to a request or advance purchase (sending a token representing the commitment of electronic funds or credit to purchase an item);
- supporting content downloadable from an organization’s own content repository, such a repository comprising, for example, a store of products (such as software programs) and/or a store of information resources, normally organized into one or more databases; and
- supporting a public repository into which other parties can deposit their products for redistribution to customers (normally by making electronic copies for distribution to a customer in response to a request).

One possible arrangement of VDE nodes involves use of one or more "repositories." A repository, for example, may serve as a location from which VDE participants may retrieve VDE content containers. In this case, VDE users may make use of a network to gain access to a "server" system that allows one or more VDE users to access an object repository containing VDE content containers.

Some VDE participants may create or provide content and/or VDE content container objects, and then store content and/or content objects at a repository so that other participants may access such content from a known and/or efficiently organized (for retrieval) location. For example, a VDE repository (portion of a VDE repository, multiple VDE repositories, and/or providers of content to such repositories) may advertise the availability of certain types of VDE protected content by sending out email to a list of network users. If the network users have secure VDE subsystems in their electronic appliances, they may then choose to access such a repository directly, or through one or more smart agents and, using an application program for example, browse (and/or electronically search) through the offerings of VDE managed content available at the repository, download desirable VDE content containers, and make use of such containers. If the repository is successful in attracting users who have an interest in such content, VDE content providers may determine that such a repository is a desirable location(s) to make their content available for easy access by users. If a repository, such as CompuServe, stores content in non-encrypted (plaintext) form, it may encrypt "outgoing" content on an "as needed" basis through placing such content in VDE content containers with desired control information, and may employ VDE secure communications techniques for content communication to VDE participants.

VDE repositories may also offer other VDE services. For example, a repository may choose to offer financial services in the form of credit from the repository that may be used to pay fees associated with use of VDE objects obtained from the repository. Alternatively or in addition, a VDE repository may perform audit information clearinghouse services on behalf of VDE creators or other participants (e.g. distributors, redistributors, client administrators, etc.) for usage information reported by VDE users. Such services may include analyzing such usage information, creating reports, collecting payments, etc.

A "full service" VDE repository may be very attractive to both providers and users of VDE managed content. Providers of VDE managed content may desire to place their content in a location that is well known to users, offers credit, and/or performs audit services for them. In this case, providers may be able to focus on creating content, rather than managing the administrative processes associated with making content available in a "retail" fashion, collecting audit information from many VDE users, sending and receiving bills and payments, etc. VDE users may find the convenience of a single location (or an integrated arrangement of repositories) appealing as they are attempting to locate content of interest. In addition, a full service VDE repository may serve as a single location for the reporting of usage information generated as a consequence of their use of

VDE managed content received from a VDE repository and/or, for example, receiving updated software (e.g. VDE-aware applications, load modules, component assemblies, non VDE-aware applications, etc.) VDE repository services may be employed in conjunction with VDE content delivery by broadcast and/or on physical media, such as CD-ROM, to constitute an integrated array of content resources that may be browsed, searched, and/or filtered, as appropriate, to fulfill the content needs of VDE users.

A public repository system may be established and maintained as a non-profit or for-profit service. An organization offering the service may charge a service fee, for example, on a per transaction basis and/or as a percentage of the payments by, and/or cost of, the content to users. A repository service may supply VDE authoring tools to content creators, publishers, distributors, and/or value adding providers such that they may apply rules and controls that define some or all of the guidelines managing use of their content and so that they may place such content into VDE content container objects.

A repository may be maintained at one location or may be distributed across a variety of electronic appliances, such as a variety of servers (e.g. video servers, etc.) which may be at different locations but nonetheless constitute a single resource. A

VDE repository arrangement may employ VDE secure communications and VDE node secure subsystems ("protected processing environments"). The content comprising a given collection or unit of information desired by a user may be spread across a variety of physical locations. For example, content representing a company's closing stock price and the activity (bids, lows, highs, etc.) for the stock might be located at a World Wide Web server in New York, and content representing an analysis of the company (such as a discussions of the company's history, personnel, products, markets, and/or competitors) might be located on a server in Dallas. The content might be stored using VDE mechanisms to secure and audit use. The content might be maintained in clear form if sufficient other forms of security are available at such one or more of sites (e.g. physical security, password, protected operating system, data encryption, or other techniques adequate for a certain content type). In the latter instances, content may be at least in part encrypted and placed in VDE containers as it streams out of a repository so as to enable secure communication and subsequent VDE usage control and usage consequence management.

A user might request information related to such a company including stock and other information. This request might, for example, be routed first through a directory or a more sophisticated database arrangement located in Boston. This

arrangement might contain pointers to, and retrieve content from, both the New York and Dallas repositories. This information content may, for example, be routed directly to the user in two containers (e.g. such as a VDE content container object from Dallas and a VDE content container object from New York). These two containers may form two VDE objects within a single VDE container (which may contain two content objects containing the respective pieces of content from Dallas and New York) when processed by the user's electronic appliance.

Alternatively, such objects might be integrated together to form a single VDE container in Boston so that the information can be delivered to the user within a single container to simplify registration and control at the user's site. The information content from both locations may be stored as separate information objects or they may be joined into a single, integrated information object (certain fields and/or categories in an information form or template may be filled in by one resource and other fields and/or categories may be filled by information provided by a different resource). A distributed database may manage such a distributed repository resource environment and use VDE to secure the storing, communicating, auditing, and/or use of information through VDE's electronic enforcement of VDE controls. VDE may then be used to provide both consistent content containers and content control services.

An example of one possible repository arrangement 3300 is shown in Figure 78. In this example, a repository 3302 is connected to a network 3304 that allows authors 3306A, 3306B, 3306C, and 3306D; a publisher 3308; and one or more end users 3310 to communicate with the repository 3302 and with each other. A second network 3312 allows the publisher 3308, authors 3306E and 3306F, an editor 3314, and a librarian 3316 to communicate with each other and with a local repository 3318. The publisher 3308 is also directly connected to author 3306E. In this example, the authors 3306 and publisher 3308 connect to the repository 3302 in order to place their content into an environment in which end users 3310 will be able to gain access to a broad selection of content from a common location.

In this example, the repository has two major functional areas: a content system 3302A and a clearinghouse system 3302B. The content system 3302A is comprised of a user/author registration system 3320, a content catalog 3322, a search mechanism 3324, content storage 3326, content references 3328, and a shipping system 3330 comprised of a controls packager 3322, a container packager 3334, and a transaction system 3336. The clearinghouse system 3302B is comprised of a user/author registration system 3338; template libraries 3340; a control structure library 3342; a disbursement system 3344; an authorization system 3346 comprised of a financial system 3348

and a content system 3350; a billing system 3352 comprised of a paper system 3354, a credit card system 3356, and an electronic funds transfer (EFT) system 3358; and an audit system 3360 comprised of a receipt system 3362, a response system 3364, a transaction system 3366, and an analysis system 3368.

In this example, author 3306A creates content in electronic form that she intends to make broadly available to many end users 3310, and to protect her rights through use of VDE. Author 3306A transmits a message to the repository 3302 indicating her desire to register with the repository to distribute her content. In response to this message, the user/author registration system 3320 of the content system 3302A, and the user/author registration system 3338 of the clearinghouse system 3302B transmit requests for registration information to author 3306A using the network 3304. These requests may be made in an on-line interactive mode; or they may be transmitted in a batch to author 3306A who then completes the requested information and transmits it as a batch to the repository 3302; or some aspects may be handled on-line (such as basic identifying information) and other information may be exchanged in a batch mode.

Registration information related to the content system 3302A may, for example, include:

- a request that Author 3306A provide information concerning the types and/or categories of content proposed for storage and access using the repository,
- the form of abstract and/or other identifying information required by the repository—in addition to providing author 3306A with an opportunity to indicate whether or not author 3306A generally includes other information with content submissions (such as promotional materials, detailed information regarding the format of submitted content, any equipment requirements that should or must be met for potential users of submitted content to successfully exploit its value, etc.),
- requests for information from author 3306A concerning where the content is to be located (stored at the repository, stored at author 3306A's location, stored elsewhere, or some combination of locations),
- what general search characteristics should be associated with content submissions (e.g. whether abstracts should be automatically indexed for searches by users of the repository, the manner in which content titles, abstracts, promotional

materials, relevant dates, names of performers and/or authors, or other information related to content submissions may or should be used in lists of types of content and/or in response to searches, etc.), and/or

- how content that is stored at and/or passed through the repository should be shipped (including any container criteria, encryption requirements, transaction requirements related to content transmissions, other control criteria, etc.)

The information requested from author 3306A by the user/author registration system of the clearinghouse may, for example, consist of:

- VDE templates that author 3306A may or must make use of in order to correctly format control information such that, for example, the audit system 3360 of the clearinghouse system 3302B is properly authorized to receive and/or process usage information related to content submitted by author 3306A,
- VDE control information available from the clearinghouse 3302B that may or must be used by

author 3306A (and/or included by reference) in some or all of the VDE component assemblies created and/or used by author 3306A associated with submitted content,

- the manner in which disbursement of any funds associated with usage of content provided by, passed through, or collected by the repository clearinghouse system 3302B should be made,
- the form and/or criteria of authorizations to use submitted content and/or financial transactions associated with content,
- the acceptable forms of billing for use of content and/or information associated with content (such as analysis reports that may be used by others),
- how VDE generated audit information should be received,
- how responses to requests from users should be managed,

- how transactions associated with the receipt of audit information should be formatted and authorized,
- how and what forms of analysis should be performed on usage information, and/or
- under what circumstances (if any) usage information and/or analysis results derived from VDE controlled content usage information should be managed (including to whom they may or must be delivered, the form of delivery, any control information that may be associated with use of such information, etc.)

The repository 3302 receives the completed registration information from author 3306A and uses this information to build an account profile for author 3306A. In addition, software associated with the authoring process may be transmitted to author 3306A. This software may, for example, allow author 3306A to place content into a VDE content container with appropriate controls in such a way that many of the decisions associated with creating such containers are made automatically to reflect the use of the repository 3302 as a content system and/or a clearinghouse system (for example, the location of content, the party to contact for updates to content and/or controls associated with content, the party or parties to whom

audit information may and/or must be transmitted and the pathways for such communication, the character of audit information that is collected during usage, the forms of payment that are acceptable for use of content, the frequency of audit transmissions required, the frequency of billing, the form of abstract and/or other identifying information associated with content, the nature of at least a portion of content usage control information, etc.)

Author 3306A makes use of a VDE authoring application to specify the controls and the content that she desires to place within a VDE content container, and produces such a container in accordance with any requirements of the repository 3302. Such a VDE authoring application may be, for example, an application provided by the repository 3302 which can help ensure adherence to repository content control requirements such as the inclusion of one or more types of component assemblies or other VDE control structures and/or required parameter data, an application received from another party, and/or an application created by author 3306A in whole or in part. Author 3306A then uses the network 3304 to transmit the container and any deviations from author 3306A's account profile that may relate to such content to the repository 3302. The repository 3302 receives the submitted content, and then -- in accordance with any account profile requirements, deviations and/or desired options in

this example—makes a determination as to whether the content was produced within the boundaries of any content and/or control information requirements of the repository and therefore should be placed within content storage or referenced by a location pointer or the like. In addition to placing the submitted content into content storage or referencing such content's location, the repository 3302 may also make note of characteristics associated with such submitted content in the search mechanism 3324, content references 3328, the shipping system 3330, and/or the relevant systems of the clearinghouse system 3302B related to templates and control structures, authorizations, billing and/or payments, disbursements, and/or audits of usage information.

During an authoring process, author 3306A may make use of VDE templates. Such templates may be used as an aspect of a VDE authoring application. For example, such templates may be used in the construction of a container as described above. Alternatively or in addition, such templates may also be used when submitted content is received by the repository 3302. References to such templates may be incorporated by author 3306A as an aspect of constructing a container for submitted content (in this sense the container delivered to the repository may be in some respects "incomplete" until the repository "completes" the container through use of indicated templates). Such references may be required for use by the repository 3302

(for example, to place VDE control information in place to fulfill an aspect of the repository's business or security models such as one or more map tables corresponding to elements of content necessary for interacting with other VDE control structures to accommodate certain metering, billing, budgeting, and/or other usage and/or distribution related controls of the repository).

For example, if content submitted by author 3306A consists of a periodical publication, a template delivered to the author by the repository 3302 when the author registers at the repository may be used as an aspect of an authoring application manipulated by the author in creating a VDE content container for such a periodical. Alternatively or in addition, a template designed for use with periodical publications may be resident at the repository 3302, and such a template may be used by the repository to define, in whole or in part, control structures associated with such a container. For example, a VDE template designed to assist in formulating control structures for periodical publications might indicate (among other things) that:

- usage controls should include a meter method that records each article within a publication that a user opens,

- a certain flat rate fee should apply to opening the periodical regardless of the number of articles opened, and/or
- a record should be maintained of every advertisement that is viewed by a user.

If content is maintained in a known and/or identifiable format, such a template may be used during initial construction of a container without author 3306A's intervention to identify any map tables that may be required to support such recording and billing actions. If such a VDE template is unavailable to author 3306A, she may choose to indicate that the container submitted should be reconstructed (e.g. augmented) by the repository to include the VDE control information specified in a certain template or class of templates. If the format of the content is known and/or identifiable by the repository, the repository may be able to reconstruct (or "complete") such a container automatically.

One factor in a potentially ongoing financial relationship between the repository and author 3306A may relate to usage of submitted content by end users 3310. For example, author 3306A may negotiate an arrangement with the repository wherein the repository is authorized to keep 20% of the total revenues generated from end users 3310 in exchange for

maintaining the repository services (e.g. making content available to end users 3310, providing electronic credit, performing billing activities, collecting fees, etc.) A financial relationship may be recorded in control structures in flexible and configurable ways. For example, the financial relationship described above could be created in a VDE container and/or installation control structure devised by author 3306A to reflect author 3306A's financial requirements and the need for a 20% split in revenue with the repository wherein all billing activities related to usage of submitted content could be processed by the repository, and control structures representing reciprocal methods associated with various component assemblies required for use of author 3306A's submitted content could be used to calculate the 20% of revenues. Alternatively, the repository may independently and securely add and/or modify control structures originating from author 3306A in order to reflect an increase in price. Under some circumstances, author 3306A may not be directly involved (or have any knowledge of) the actual price that the repository charges for usage activities, and may concern herself only with the amount of revenue and character of usage analysis information that she requires for her own purposes, which she specifies in VDE control information which governs the use, and consequences of use, of VDE controlled content.

Another aspect of the relationship between authors and the repository may involve the character of transaction recording requirements associated with delivery of VDE controlled content and receipt of VDE controlled content usage audit information. For example, author 3306A may require that the repository make a record of each user that receives a copy of content from the repository. Author 3306A may further require collection of information regarding the circumstances of delivery of content to such users (e.g. time, date, etc.) In addition, the repository may elect to perform such transactions for use internally (e.g. to determine patterns of usage to optimize systems, detect fraud, etc.)

In addition to recording information regarding delivery of such VDE controlled content, author 3306A may have required or requested the repository to perform certain VDE container related processes. For example, author 3306A may want differing abstract and/or other descriptive information delivered to different classes of users. In addition, author 3306A may wish to deliver promotional materials in the same container as submitted content depending on, for example, the character of usage exhibited by a particular user (e.g. whether the user has ever received content from author 3306A, whether the user is a regular subscriber to author 3306A's materials, and/or other patterns that may be relevant to author 3306A and/or the end

user that are used to help determine the mix of promotional materials delivered to a certain VDE content end user.) In another example, author 3306A may require that VDE fingerprinting be performed on such content prior to transmission of content to an end user.

In addition to the form and/or character of content shipped to an end user, authors may also require certain encryption related processes to be performed by the repository as an aspect of delivering content. For example, author 3306A may have required that the repository encrypt each copy of shipped content using a different encryption key or keys in order to help maintain greater protection for content (e.g. in case an encryption key was "cracked" or inadvertently disclosed, the "damage" could be limited to the portion(s) of that specific copy of a certain content deliverable). In another example, encryption functions may include the need to use entirely different encryption algorithms and/or techniques in order to fulfill circumstantial requirements (e.g. to comply with export restrictions). In a further example, encryption related processes may include changing the encryption techniques and/or algorithms based on the level of trustedness and/or tamper resistance of the VDE site to which content is delivered.

In addition to transaction information gathered when content is shipped from a VDE repository to an end user, the repository may be required to keep transaction information related to the receipt of usage information, requests, and/or responses to and/or from end users 3310. For example, author 3306A may require the repository to keep a log of some or all connections made by end users 3310 related to transmissions and or reception of information related to the use of author 3306A's content (e.g. end user reporting of audit information, end user requests for additional permissions information, etc.)

Some VDE managed content provided to end users 3310 through the repository may be stored in content storage. Other information may be stored elsewhere, and be referenced through the content references. In the case where content references are used, the repository may manage the user interactions in such a manner that all repository content, whether stored in content storage or elsewhere (such as at another site), is presented for selection by end users 3310 in a uniform way, such as, for example, a consistent or the same user interface. If an end user requests delivery of content that is not stored in content storage, the VDE repository may locate the actual storage site for the content using information stored in content references (e.g. the network address where the content may be located, a URL, a filesystem reference, etc.) After the content is located, the

content may be transmitted across the network to the repository or it may be delivered directly from where it is stored to the requesting end user. In some circumstances (e.g. when container modification is required, when encryption must be changed, if financial transactions are required prior to release, etc.), further processing may be required by the repository in order to prepare such VDE managed content and/or VDE content container for transmission to an end user.

In order to provide a manageable user interface to the content available to VDE repository end users 3310 and to provide administrative information used in the determination of control information packaged in VDE content containers shipped to end users 3310, the repository in this example includes a content catalog 3322. This catalog is used to record information related to the VDE content in content storage, and/or content available through the repository reflected in content references. The content catalog 3322 may consist of titles of content, abstracts, and other identifying information. In addition, the catalog may also indicate the forms of electronic agreement and/or agreement VDE template applications (offering optional, selectable control structures and/or one or more opportunities to provide related parameter data) that are available to end users 3310 through the repository for given pieces of content in deciding, for example, options and/or requirements for: what

type(s) of information is recorded during such content's use, the charge for certain content usage activities, differences in charges based on whether or not certain usage information is recorded and/or made available to the repository and/or content provider, the redistribution rights associated with such content, the reporting frequency for audit transmissions, the forms of credit and/or currency that may be used to pay certain fees associated with use of such content, discounts related to certain volumes of usage, discounts available due to the presence of rights associated with other content from the same and/or different content providers, sales, etc. Furthermore, a VDE repository content catalog 3322 may indicate some or all of the component assemblies that are required in order to make use of content such that the end user's system and the repository can exchange messages to help ensure that any necessary VDE component assemblies or other VDE control information is identified, and if necessary and authorized, are delivered along with such content to the end user (rather than, for example, being requested later after their absence has been detected during a registration and/or use attempt).

In order to make use of the VDE repository in this example, an end user must register with the repository. In a manner similar to that indicated above in the case of an author, a VDE end user transmits a message from her VDE installation to

the repository across the network indicating that she wishes to make use of the services provided by the repository (e.g. access content stored at and/or referenced by the repository, use credit provided by the repository, etc.) In response to this message, the user/author registration systems of the content system 3302A and the clearinghouse system 3302B of the repository transmit requests for information from the end user (e.g. in an on-line and/or batch interaction). The information requested by the user/author registration system of the content system 3302A may include type(s) of content that the user wishes to access, the characteristics of the user's electronic appliance 600, etc. The information requested by the user/author registration system of the clearinghouse system 3302B may include whether the user wishes to establish a credit account with the clearinghouse system 3302B, what other forms of credit the user may wish to use for billing purposes, what other clearinghouses may be used by the end user in the course of interacting with content obtained from the repository, any general rules that the user has established regarding their preferences for release and handling of usage analysis information, etc. Once the end user has completed the registration information and transmitted it to the repository, the repository may construct an account profile for the user. In this example, such requests and responses are handled by secure VDE communications between secure VDE subsystems of both sending and receiving parties.

In order to make use of the repository, the end user may operate application software. In this example, the end user may either make use of a standard application program (e.g. a World Wide Web browser such as Mosaic), or they may make use of application software provided by the repository after completion of the registration process. If the end user chooses to make use of the application software provided by the repository, they may be able to avoid certain complexities of interaction that may occur if a standard package is used. Although standardized packages are often relatively easy to use, a customized package that incorporates VDE aware functionality may provide an easier to use interface for a user. In addition, certain characteristics of the repository may be built in to the interface to simplify use of the services (e.g. similar to the application programs provided by America Online).

The end user may connect to the repository using the network. In this example, after the user connects to the repository, an authentication process will occur. This process can either be directed by the user (e.g. through use of a login and password protocol) or may be established by the end user's electronic appliance secure subsystems interacting with a repository electronic appliance in a VDE authentication. In either event, the repository and the user must initially ensure that they are connected to the correct other party. In this

example, if secured information will flow between the parties, a VDE secured authentication must occur, and a secure session must be established. On the other hand, if the information to be exchanged has already been secured and/or is available without authentication (e.g. certain catalog information, containers that have already been encrypted and do not require special handling, etc.), the "weaker" form of login/password may be used.

Once an end user has connected to the VDE repository and authentication has occurred, the user may begin manipulating and directing their user interface software to browse through a repository content catalog 3322 (e.g. lists of publications, software, games, movies, etc.), use the search mechanism to help locate content of interest, schedule content for delivery, make inquiries of account status, availability of usage analysis information, billing information, registration and account profile information, etc. If a user is connecting to obtain content, the usage requirements for that content may be delivered to them. If the user is connecting to deliver usage information to the repository, information related to that transmission may be delivered to them. Some of these processes are described in more detail below.

In this example, when an end user requests content from the VDE repository (e.g. by selecting from a menu of available

options), the content system 3302A locates the content either in the content references and/or in content storage. The content system 3302A may then refer to information stored in the content catalog 3322, the end user's account profile, and/or the author's account profile to determine the precise nature of container format and/or control information that may be required to create a VDE content container to fulfill the end user's request. The shipping system then accesses the clearinghouse system 3302B to gather any necessary additional control structures to include with the container, to determine any characteristics of the author's and/or end user's account profiles that may influence either the transaction(s) associated with delivering the content to the end user or with whether the transaction may be processed. If the transaction is authorized, and all elements necessary for the container are available, the controls packager forms a package of control information appropriate for this request by this end user, and the container packager takes this package of control information and the content and forms an appropriate container (including any permissions that may be codeliverable with the container, incorporating any encryption requirements, etc.) If required by the repository or the author's account profile, transactions related to delivery of content are recorded by the transaction system of the shipping system. When the container and any transactions related to delivery have been completed, the container is transmitted across the network to the end user.

An end user may make use of credit and/or currency securely stored within the end user's VDE installation secure subsystem to pay for charges related to use of VDE content received from the repository, and/or the user may maintain a secure credit and/or currency account remotely at the repository, including a "virtual" repository where payment is made for the receipt of such content by an end user. This later approach may provide greater assurance for payment to the repository and/or content providers particularly if the end user has only an HPE based secure subsystem. If an end user electronic credit and/or currency account is maintained at the repository in this example, charges are made to said account based on end user receipt of content from the repository. Further charges to such a remote end user account may be made based on end user usage of such received content and based upon content usage information communicated to the repository clearinghouse system 3302B.

In this example, if an end user does not have a relationship established with a financial provider (who has authorized the content providers whose content may be obtained through use of the repository to make use of their currency and/or credit to pay for any usage fees associated with such provider's content) and/or if an end user desires a new source of such credit, the end user may request credit from the repository clearinghouse system 3302B. If an end user is approved for credit, the repository may

extend credit in the form of credit amounts (e.g. recorded in one or more UDEs) associated with a budget method managed by the repository. Periodically, usage information associated with such a budget method is transmitted by the end user to the audit system of the repository. After such a transmission (but potentially before the connection is terminated), an amount owing is recorded for processing by the billing system, and in accordance with the repository's business practices, the amount of credit available for use by the end user may be replenished in the same or subsequent transmission. In this example, the clearinghouse of the repository supports a billing system with a paper system for resolving amounts owed through the mail, a credit card system for resolving amounts owed through charges to one or more credit cards, and an electronic funds transfer system for resolving such amounts through direct debits to a bank account. The repository may automatically make payments determined by the disbursement system for monies owed to authors through use of similar means. Additional detail regarding the audit process is provided below.

As indicated above, end users 3310 in this example will periodically contact the VDE repository to transmit content usage information (e.g. related to consumption of budget, recording of other usage activities, etc.), replenish their budgets, modify their account profile, access usage analysis information, and perform

other administrative and information exchange activities. In some cases, an end user may wish to contact the repository to obtain additional control structures. For example, if an end user has requested and obtained a VDE content container from the repository, that container is typically shipped to the end user along with control structures appropriate to the content, the author's requirements and account profile, the end user's account profile, the content catalog 3322, and/or the circumstances of the delivery (e.g. the first delivery from a particular author, a subscription, a marketing promotion, presence and/or absence of certain advertising materials, requests formulated on behalf of the user by the user's local VDE instance, etc.) Even though, in this example, the repository may have attempted to deliver all relevant control structures, some containers may include controls structures that allow for options that the end user did not anticipate exercising (and the other criteria did not automatically select for inclusion in the container) that the end user nonetheless determines that they would like to exercise. In this case, the end user may wish to contact the repository and request any additional control information (including, for example, control structures) that they will need in order to make use of the content under such option.

For example, if an end user has obtained a VDE content container with an overall control structure that includes an

option that records of the number of times that certain types of accesses are made to the container and further bases usage fees on the number of such accesses, and another option within the overall control structure allows the end user to base the fees paid for access to a particular container based on the length of time spent using the content of the container, and the end user did not originally receive controls that would support this latter form of usage, the repository may deliver such controls at a later time and when requested by the user. In another example, an author may have made changes to their control structures (e.g. to reflect a sale, a new discounting model, a modified business strategy, etc.) which a user may or must receive in order to use the content container with the changed control structures. For example, one or more control structures associated with a certain VDE content container may require a "refresh" for continued authorization to employ such structures, or the control structures may expire. This allows (if desired) a VDE content provider to periodically modify and/or add to VDE control information at an end user's site (employing the local VDE secure subsystem).

Audit information (related to usage of content received from the repository) in this example is securely received from end users 3310 by the receipt system 3362 of the clearinghouse. As indicated above, this system may process the audit information and pass some or all of the output of such a process to the billing

system and/or transmit such output to appropriate content authors. Such passing of audit information employs secure VDE pathway of reporting information handling techniques. Audit information may also be passed to the analysis system in order to produce analysis results related to end user content usage for use by the end user, the repository, third party market researchers, and/or one or more authors. Analysis results may be based on a single audit transmission, a portion of an audit transmission, a collection of audit transmissions from a single end user and/or multiple end users 3310, or some combination of audit transmissions based on the subject of analysis (e.g. usage patterns for a given content element or collection of elements, usage of certain categories of content, payment histories, demographic usage patterns, etc.) The response system 3364 is used to send information to the end user to, for example, replenish a budget, deliver usage controls, update permissions information, and to transmit certain other information and/or messages requested and/or required by an end user in the course of their interaction with the clearinghouse. During the course of an end user's connections and transmissions to and from the clearinghouse, certain transactions (e.g. time, date, and/or purpose of a connection and/or transmission) may be recorded by the transaction system of the audit system to reflect requirements of the repository and/or authors.

Certain audit information may be transmitted to authors. For example, author 3306A may require that certain information gathered from an end user be transmitted to author 3306A with no processing by the audit system. In this case, the fact of the transmission may be recorded by the audit system, but author 3306A may have elected to perform their own usage analysis rather than (or in addition to) permitting the repository to access, otherwise process and/or otherwise use this information. The repository in this example may provide author 3306A with some of the usage information related to the repository's budget method received from one or more end users 3310 and generated by the payment of fees associated with such users' usage of content provided by author 3306A. In this case, author 3306A may be able to compare certain usage information related to content with the usage information related to the repository's budget method for the content to analyze patterns of usage (e.g. to analyze usage in light of fees, detect possible fraud, generate user profile information, etc.) Any usage fees collected by the clearinghouse associated with author 3306A's content that are due to author 3306A will be determined by the disbursement system of the clearinghouse. The disbursement system may include usage information (in complete or summary form) with any payments to author 3306A resulting from such a determination. Such payments and information reporting may be an entirely automated sequence of processes occurring within

the VDE pathway from end user VDE secure subsystems, to the clearinghouse secure subsystem, to the author's secure subsystem.

In this example, end users 3310 may transmit VDE permissions and/or other control information to the repository 3302 permitting and/or denying access to usage information collected by the audit system for use by the analysis system. This, in part, may help ensure end user's privacy rights as it relates to the usage of such information. Some containers may require, as an aspect of their control structures, that an end user make usage information available for analysis purposes. Other containers may give an end user the option of either allowing the usage information to be used for analysis, or denying some or all such uses of such information. Some users may elect to allow analysis of certain information, and deny this permission for other information. End users 3310 in this example may, for example, elect to limit the granularity of information that may be used for analysis purposes (e.g. an end user may allow analysis of the number of movies viewed in a time period but disallow use of specific titles, an end user may allow release of their ZIP code for demographic analysis, but disallow use of their name and address, etc.) Authors and/or the repository 3302 may, for example, choose to charge end users 3310 smaller fees if they agree to release certain usage information for analysis purposes.

In this example, the repository 3302 may receive content produced by more than one author. For example, author B, author C, and author D may each create portions of content that will be delivered to end users 3310 in a single container. For example, author B may produce a reference work. Author C may produce a commentary on author B's reference work, and author D may produce a set of illustrations for author B's reference work and author C's commentary. Author B may collect together author C's and author D's content and add further content (e.g. the reference work described above) and include such content in a single container which is then transmitted to the repository 3302. Alternatively, each of the authors may transmit their works to the repository 3302 independently, with an indication that a template should be used to combine their respective works prior to shipping a container to an end user. Still alternatively, a container reflecting the overall content structure may be transmitted to the repository 3302 and some or all of the content may be referenced in the content references rather than delivered to the repository 3302 for storage in content storage.

When an end user makes use of container content, their content usage information may, for example, be segregated in accordance with control structures that organize usage information based at least in part on the author who created that segment. Alternatively, the authors and/or the VDE repository

3302 may negotiate one or more other techniques for securely dividing and/or sharing usage information in accordance with VDE control information. Furthermore, control structures associated with a container may implement models that differentiate any usage fees associated with portions of content based on usage of particular portions, overall usage of the container, particular patterns of usage, or other mechanism negotiated (or otherwise agreed to) by the authors. Reports of usage information, analysis results, disbursements, and other clearinghouse processes may also be generated in a manner that reflects agreements reached by repository 3302 participants (authors, end users 3310 and/or the repository 3302) with respect to such processes. These agreements may be the result of a VDE control information negotiation amongst these participants.

In this example, one type of author is a publisher 3308. The publisher 3308 in this example communicates over an "internal" network with a VDE based local repository 3302 and over the network described above with the public repository 3302. The publisher 3308 may create or otherwise provide content and/or VDE control structure templates that are delivered to the local repository 3302 for use by other participants who have access to the "internal" network. These templates may be used to describe the structure of containers, and may further describe whom in the publisher 3308's organization may take which

actions with respect to the content created within the organization related to publication for delivery to (and/or referencing by) the repository 3302. For example, the publisher 3308 may decide (and control by use of said template) that a periodical publication will have a certain format with respect to the structure of its content and the types of information that may be included (e.g. text, graphics, multimedia presentations, advertisements, etc.), the relative location and/or order of presentation of its content, the length of certain segments, etc. Furthermore, the publisher 3308 may, for example, determine (through distribution of appropriate permissions) that the publication editor is the only party that may grant permissions to write into the container, and that the organization librarian is the only party that may index and/or abstract the content. In addition, the publisher 3308 may, for example, allow only certain one or more parties to finalize a container for delivery to the repository 3302 in usable form (e.g. by maintaining control over the type of permissions, including distribution permissions, that may be required by the repository 3302 to perform subsequent distribution activities related to repository end users 3310).

In this example, author 3306E is connected directly to the publisher 3308, such that the publisher 3308 can provide templates for that author that establish the character of containers for author 3306E's content. For example, if author

3306E creates books for distribution by the publisher 3308, the publisher 3308 may define the VDE control structure template which provides control method options for author 3306E to select from and which provides VDE control structures for securely distributing author 3306E's works. Author 3306E and the publisher 3308 may employ VDE negotiations for the template characteristics, specific control structures, and/or parameter data used by author 3306E. Author 3306E may then use the template(s) to create control structures for their content containers. The publisher 3308 may then deliver these works to the repository 3302 under a VDE extended agreement comprising electronic agreements between author 3306E and the publisher 3308 and the repository 3302 and the publisher 3308.

In this example, the publisher 3308 may also make author 3306E's work available on the local repository 3302. The editor may authorize (e.g. through distribution of appropriate permissions) author F to create certain portions of content for a publication. In this example, the editor may review and/or modify author F's work and further include it in a container with content provided by author 3306E (available on the local repository 3302). The editor may or may not have permissions from the publisher 3308 to modify author 3306E's content (depending on any negotiation(s) that may have occurred between the publisher 3308 and author 3306E, and the publisher

3308's decision to extend such rights to the editor if permissions to modify author 3306E's content are held in redistributable form by the publisher 3308). The editor may also include content from other authors by (a) using a process of granting permissions to authors to write directly into the containers and/or (b) retrieving containers from the local repository 3302 for inclusion. The local repository 3302 may also be used for other material used by the publisher 3308's organization (e.g. databases, other reference works, internal documents, draft works for review, training videos, etc.), such material may, given appropriate permissions, be employed in VDE container collections of content created by the editor.

The librarian in this example has responsibility for building and/or editing inverted indexes, keyword lists (e.g. from a restricted vocabulary), abstracts of content, revision histories, etc. The publisher 3308 may, for example, grant permissions to only the librarian for creating this type of content. The publisher 3308 may further require that this building and/or editing occur prior to release of content to the repository 3302.

Example -- Evolution and Transformation of VDE Managed Content and Control Information

The VDE content control architecture allows content control information (such as control information for governing content usage) to be shaped to conform to VDE control information requirements of multiple parties. Formulating such multiple party content control information normally involves securely deriving control information from control information securely contributed by parties who play a role in a content handling and control model (e.g. content creator(s), provider(s), user(s), clearinghouse(s), etc.). Multiple party control information may be necessary in order to combine multiple pieces of independently managed VDE content into a single VDE container object (particularly if such independently managed content pieces have differing, for example conflicting, content control information). Such secure combination of VDE managed pieces of content will frequently require VDE's ability to securely derive content control information which accommodates the control information requirements, including any combinatorial rules, of the respective VDE managed pieces of content and reflects an acceptable agreement between such plural control information sets.

The combination of VDE managed content pieces may result in a VDE managed composite of content. Combining VDE

managed content must be carried out in accordance with relevant content control information associated with said content pieces and processed through the use of one or more secure VDE sub-system PPEs 650. VDE's ability to support the embedding, or otherwise combining, of VDE managed content pieces, so as to create a combination product comprised of various pieces of VDE content, enables VDE content providers to optimize their VDE electronic content products. The combining of VDE managed content pieces may result in a VDE content container which "holds" consolidated content and/or concomitant, separate, nested VDE content containers.

VDE's support for creation of content containers holding distinct pieces of VDE content portions that were previously managed separately allows VDE content providers to develop products whose content control information reflects value propositions consistent with the objectives of the providers of content pieces, and further are consistent with the objectives of a content aggregator who may be producing a certain content combination as a product for commercial distribution. For example, a content product "launched" by a certain content provider into a commercial channel (such as a network repository) may be incorporated by different content providers and/or end-users into VDE content containers (so long as such incorporation is allowed by the launched product's content

control information). These different content providers and/or end-users may, for example, submit differing control information for regulating use of such content. They may also combine in different combinations a certain portion of launched content with content received from other parties (and/or produced by themselves) to produce different content collections, given appropriate authorizations.

VDE thus enables copies of a given piece of VDE managed content to be securely combined into differing consolidations of content, each of which reflects a product strategy of a different VDE content aggregator. VDE's content aggregation capability will result in a wider range of competitive electronic content products which offer differing overall collections of content and may employ differing content control information for content that may be common to such multiple products. Importantly, VDE securely and flexibly supports editing the content in, extracting content from, embedding content into, and otherwise shaping the content composition of, VDE content containers. Such capabilities allow VDE supported product models to evolve by progressively reflecting the requirements of "next" participants in an electronic commercial model. As a result, a given piece of VDE managed content, as it moves through pathways of handling and branching, can participate in many different content container and content control information commercial models.

VDE content, and the electronic agreements associated with said content, can be employed and progressively manipulated in commercial ways which reflect traditional business practices for non-electronic products (though VDE supports greater flexibility and efficiency compared with most of such traditional models). Limited only by the VDE control information employed by content creators, other providers, and other pathway of handling and control participants, VDE allows a "natural" and unhindered flow of, and creation of, electronic content product models. VDE provides for this flow of VDE products and services through a network of creators, providers, and users who successively and securely shape and reshape product composition through content combining, extracting, and editing within a Virtual Distribution Environment.

VDE provides means to securely combine content provided at different times, by differing sources, and/or representing differing content types. These types, timings, and/or different sources of content can be employed to form a complex array of content within a VDE content container. For example, a VDE content container may contain a plurality of different content container objects, each containing different content whose usage can be controlled, at least in part, by its own container's set of VDE content control information.

A VDE content container object may, through the use of a secure VDE sub-system, be "safely" embedded within a "parent" VDE content container. This embedding process may involve the creation of an embedded object, or, alternatively, the containing, within a VDE content container, of a previously independent and now embedded object by, at minimum, appropriately referencing said object as to its location.

An embedded content object within a parent VDE content container:

(1) may have been a previously created VDE content container which has been embedded into a parent VDE content container by securely transforming it from an independent to an embedded object through the secure processing of one or more VDE component assemblies within a VDE secure sub-system PPE 650. In this instance, an embedded object may be subject to content control information, including one or more permissions records associated with the parent container, but may not, for example, have its own content control information other than content identification information, or the embedded object may be more extensively controlled by its own content control information (e.g. permissions records).

(2) may include content which was extracted from another VDE content container (along with content control information, as may be applicable) for inclusion into a parent VDE content container in the form of an embedded VDE content container object. In this case, said extraction and embedding may use one or more VDE processes which run securely within a VDE secure sub-system PPE 650 and which may securely remove (or copy) the desired content from a source VDE content container and place such content in a new or existing container object, either of which may be or become embedded into a parent VDE content container.

(3) may include content which was first created and then placed in a VDE content container object. Said receiving container may already be embedded in a parent VDE content container and may already contain other content. The container in which such content is placed may be specified using a VDE aware application which interacts with content and a secure VDE subsystem to securely create such VDE container and place such content therein followed by securely embedding such container into the destination, parent container. Alternatively, content may be specified without the use of a VDE aware application, and then manipulated using a VDE aware

application in order to manage movement of the content into a VDE content container. Such an application may be a VDE aware word processor, desktop and/or multimedia publishing package, graphics and/or presentation package, etc. It may also be an operating system function (e.g. part of a VDE aware operating system or mini-application operating with an O/S such as a Microsoft Windows compatible object packaging application) and movement of content from "outside" VDE to within a VDE object may, for example, be based on a "drag and drop" metaphor that involves "dragging" a file to a VDE container object using a pointing device such as a mouse. Alternatively, a user may "cut" a portion of content and "paste" such a portion into a VDE container by first placing content into a "clipboard," then selecting a target content object and pasting the content into such an object. Such processes may, at the direction of VDE content control information and under the control of a VDE secure subsystem, put the content automatically at some position in the target object, such as at the end of the object or in a portion of the object that corresponds to an identifier carried by or with the content such as a field identifier, or the embedding process might pop-up a user interface that allows a user to browse a target object's contents and/or table of contents and/or other directories, indexes, etc. Such processes may further

allow a user to make certain decisions concerning VDE content control information (budgets limiting use, reporting pathway(s), usage registration requirements, etc.) to be applied to such embedded content and/or may involve selecting the specific location for embedding the content, all such processes to be performed as transparently as practical for the application.

(4) may be accessed in conjunction with one or more operating system utilities for object embedding and linking, such as utilities conforming to the Microsoft OLE standard. In this case, a VDE container may be associated with an OLE "link." Accesses (including reading content from, and writing content to) to a VDE protected container may be passed from an OLE aware application to a VDE aware OLE application that accesses protected content in conjunction with control information associated with such content.

A VDE aware application may also interact with component assemblies within a PPE to allow direct editing of the content of a VDE container, whether the content is in a parent or embedded VDE content container. This may include the use of a VDE aware word processor, for example, to directly edit (add to, delete, or otherwise modify) a VDE container's content. The

secure VDE processes underlying VDE container content editing may be largely or entirely transparent to the editor (user) and may transparently enable the editor to securely browse through (using a VDE aware application) some or all of the contents of, and securely modify one or more of the VDE content containers embedded in, a VDE content container hierarchy.

The embedding processes for all VDE embedded content containers normally involves securely identifying the appropriate content control information for the embedded content. For example, VDE content control information for a VDE installation and/or a VDE content container may securely, and transparently to an embedder (user), apply the same content control information to edited (such as modified or additional) container content as is applied to one or more portions (including all, for example) of previously "in place" content of said container and/or securely apply control information generated through a VDE control information negotiation between control sets, and/or it may apply control information previously applied to said content. Application of control information may occur regardless of whether the edited content is in a parent or embedded container. This same capability of securely applying content control information (which may be automatically and/or transparently applied), may also be employed with content that is embedded into a VDE container through extracting and embedding content,

or through the moving, or copying and embedding, of VDE container objects. Application of content control information normally occurs securely within one or more VDE secure sub-system PPEs 650. This process may employ a VDE template that enables a user, through easy to use GUI user interface tools, to specify VDE content control information for certain or all embedded content, and which may include menu driven, user selectable and/or definable options, such as picking amongst alternative control methods (e.g. between different forms of metering) which may be represented by different icons picturing (symbolizing) different control functions and apply such functions to an increment of VDE secured content, such as an embedded object listed on an object directory display.

Extracting content from a VDE content container, or editing or otherwise creating VDE content with a VDE aware application, provides content which may be placed within a new VDE content container object for embedding into said parent VDE container, or such content may be directly placed into a previously existing content container. All of these processes may be managed by processing VDE content control information within one or more VDE installation secure sub-systems.

VDE content container objects may be embedded in a parent object through control information referenced by a parent

object permissions record that resolves said embedded object's location and/or contents. In this case, little or no change to the embedded object's previously existing content control information may be required. VDE securely managed content which is relocated to a certain VDE content container may be relocated through the use of VDE sub-system secure processes which may, for example, continue to maintain relocated content as encrypted or otherwise protected (e.g. by secure tamper resistant barrier 502) during a relocation/embedding process.

Embedded content (and/or content objects) may have been contributed by different parties and may be integrated into a VDE container through a VDE content and content control information integration process securely managed through the use of one or more secure VDE subsystems. This process may, for example, involve one or more of:

- (1.) securely applying instructions controlling the embedding and/or use of said submitted content, wherein said instructions were securely put in place, at least in part, by a content provider and/or user of said VDE container. For example, said user and/or provider may interact with one or more user interfaces offering a selection of content embedding and/or control options (e.g. in the form of a VDE template). Such options may include which, and/or whether, one or more controls should

be applied to one or more portions of said content and/or the entry of content control parameter data (such a time period before which said content may not be used, cost of use of content, and/or pricing discount control parameters such as software program suite sale discounting). Once required and/or optional content control information is established by a provider and/or user, it may function as content control information which may be, in part or in full, applied automatically to certain, or all, content which is embedded in a VDE content container.

(2.) secure VDE managed negotiation activities, including the use of a user interface interaction between a user at a receiving VDE installation and VDE content control information associated with the content being submitted for embedding. For example, such associated control information may propose certain content information and the content receiver may, for example, accept, select from a plurality, reject, offer alternative control information, and/or apply conditions to the use of certain content control information (for example, accept a certain one or more controls if said content is used by a certain one or more users and/or if the volume of usage of certain content exceeds a certain level).

(3.) a secure, automated, VDE electronic negotiation process involving VDE content control information of the

receiving VDE content container and/or VDE installation and content control information associated with the submitted content (such as control information in a permissions record of a contributed VDE object, certain component assemblies, parameter data in one or more UDEs and/or MDEs, etc.).

Content embedded into a VDE content container may be embedded in the form of:

(1.) content that is directly, securely integrated into previously existing content of a VDE content container (said container may be a parent or embedded content container) without the formation of a new container object. Content control information associated with said content after embedding must be consistent with any pre-embedding content control information controlling, at least in part, the establishment of control information required after embedding. Content control information for such directly integrated, embedded content may be integrated into, and/or otherwise comprise a portion of, control information (e.g. in one or more permissions records containing content control information) for said VDE container, and/or

(2.) content that is integrated into said container in one or more objects which are nested within said VDE content container object. In this instance, control information for said content may

be carried by either the content control information for the parent VDE content container, or it may, for example, be in part or in full carried by one or more permissions records contained within and/or specifically associated with one or more content containing nested VDE objects. Such nesting of VDE content containing objects within a parent VDE content container may employ a number of levels, that is a VDE content container nested in a VDE content container may itself contain one or more nested VDE content containers.

VDE content containers may have a nested structure comprising one or more nested containers (objects) that may themselves store further containers and/or one or more types of content, for example, text, images, audio, and/or any other type of electronic information (object content may be specified by content control information referencing, for example, byte offset locations on storage media). Such content may be stored, communicated, and/or used in stream (such as dynamically accumulating and/or flowing) and/or static (fixed, such as predefined, complete file) form. Such content may be derived by extracting a subset of the content of one or more VDE content containers to directly produce one or more resulting VDE content containers. VDE securely managed content (e.g. through the use of a VDE aware application or operating system having extraction capability) may be identified for extraction from each of one or more

locations within one or more VDE content containers and may then be securely embedded into a new or existing VDE content container through processes executing VDE controls in a secure subsystem PPE 650. Such extraction and embedding (VDE "exporting") involves securely protecting, including securely executing, the VDE exporting processes.

A VDE activity related to VDE exporting and embedding involves performing one or more transformations of VDE content from one secure form to one or more other secure forms. Such transformation(s) may be performed with or without moving transformed content to a new VDE content container (e.g. by component assemblies operating within a PPE that do not reveal, in unprotected form, the results or other output of such transforming processes without further VDE processes governing use of at least a portion of said content). One example of such a transformation process may involve performing mathematical transformations and producing results, such as mathematical results, while retaining, none, some, or all of the content information on which said transformation was performed. Other examples of such transformations include converting a document format (such as from a WordPerfect format to a Word for Windows format, or an SGML document to a Postscript document), changing a video format (such as a QuickTime video format to a MPEG video format), performing an artificial

intelligence process (such as analyzing text to produce a summary report), and other processing that derives VDE secured content from other VDE secured content.

Figure 79 shows an example of an arrangement of commercial VDE users. The users in this example create, distribute, redistribute, and use content in a variety of ways. This example shows how certain aspects of control information associated with content may evolve as control information passes through a chain of handling and control. These VDE users and controls are explained in more detail below.

Creator A in this example creates a VDE container and provides associated content control information that includes references (amongst other things) to several examples of possible "types" of VDE control information. In order to help illustrate this example, some of the VDE control information passed to another VDE participant is grouped into three categories in the following more detailed discussion: distribution control information, redistribution control information, and usage control information. In this example, a fourth category of embedding control information can be considered an element of all three of the preceding categories. Other groupings of control information are possible (VDE does not require organizing control information in this way). The content control information associated with this

example of a container created by creator A is indicated on Figure 80 as C_A. Figure 80 further shows the VDE participants who may receive enabling control information related to creator A's VDE content container. Some of the control information in this example is explained in more detail below.

Some of the distribution control information (in this example, control information primarily associated with creation, modification, and/or use of control information by distributors) specified by creator A includes: (a) distributors will compensate creator A for each active user of the content of the container at the rate of \$10 per user per month, (b) distributors are budgeted such that they may allow no more than 100 independent users to gain access to such content (i.e. may create no more than 100 permissions records reflecting content access rights) without replenishing this budget, and (c) no distribution rights may be passed on in enabling control information (e.g. permissions records and associated component assemblies) created for distribution to other participants.

Some of the content redistribution control information (in this example, control information produced by a distributor within the scope permitted by a more senior participant in a chain of handling and control and passed to user/providers (in this example, user/distributors) and associated with controls

and/or other requirements associated with redistribution activities by such user/distributors) specified by creator A includes: (a) a requirement that control information enabling content access may be redistributed by user/distributors no more than 2 levels, and further requires that each redistribution decrease this value by one, such that a first redistributor is restricted to two levels of redistribution, and a second redistributor to whom the first redistributor delivers permissions will be restricted to one additional level of redistribution, and users receiving permissions from the second redistributor will be unable to perform further redistribution (such a restriction may be enforced, for example, by including as one aspect of a VDE control method associated with creating new permissions a requirement to invoke one or more methods that: (i) locate the current level of redistribution stored, for example, as an integer value in a UDE associated with such one or more methods, (ii) compare the level of redistribution value to a limiting value, and (iii) if such level of redistribution value is less than the limiting value, increment such level of redistribution value by one before delivering such a UDE to a user as an aspect of content control information associated with VDE managed content, or fail the process if such value is equal to such a limiting value), and (b) no other special restrictions are placed on redistributors.

Some of the usage control information (in this example, control information that a creator requires a distributor to provide in control information passed to users and/or user/distributors) specified by creator A may include, for example: (a) no moves (a form of distribution explained elsewhere in this document) of the content are permitted, and (b) distributors will be required to preserve (at a minimum) sufficient metering information within usage permissions in order to calculate the number of users who have accessed the container in a month and to prevent further usage after a rental has expired (e.g. by using a meter method designed to report access usages to creator A through a chain of handling and reporting, and/or the use of expiration dates and/or time-aged encryption keys within a permissions record or other required control information).

Some of the extracting and/or embedding control information specified by creator A in this example may include a requirement that no extracting and/or embedding of the content is or will be permitted by parties in a chain of handling and control associated with this control information, except for users who have no redistribution rights related to such VDE secured content provided by Creator A. Alternatively, or in addition, as regards different portions of said content, control information enabling certain extraction and/or embedding may be provided

along with the redistribution rights described in this example for use by user/distributors (who may include user content aggregators, that is they may provide content created by, and/or received from, different sources so as to create their own content products).

Distributor A in this example has selected a basic approach that distributor A prefers when offering enabling content control information to users and/or user/distributors that favors rental of content access rights over other approaches. In this example, some of the control information provided by creators will permit distributor A to fulfill this favored approach directly, and other control structures may disallow this favored approach (unless, for example, distributor A completes a successful VDE negotiation allowing such an approach and supporting appropriate control information). Many of the control structures received by distributor A, in this example, are derived from (and reflect the results of) a VDE negotiation process in which distributor A indicates a preference for distribution control information that authorizes the creation of usage control information reflecting rental based usage rights. Such distribution control information may allow distributor A to introduce and/or modify control structures provided by creators in such a way as to create control information for distribution to users and/or user/distributors that, in effect, "rent" access rights. Furthermore, distributor A in

this example services requests from user/distributors for redistribution rights, and therefore also favors distribution control information negotiated (or otherwise agreed to) with creators that permits distributor A to include such rights as an aspect of control information produced by distributor A.

In this example, distributor A and creator A may use VDE to negotiate (for example, VDE negotiate) for a distribution relationship. Since in this example creator A has produced a VDE content container and associated control information that indicates creator A's desire to receive compensation based on rental of usage rights, and such control information further indicates that creator A has placed acceptable restrictions in redistribution control information that distributor A may use to service requests from user/distributors, distributor A may accept creator A's distribution control information without any negotiated changes.

After receiving enabling distribution control information from creator A, distributor A may manipulate an application program to specify some or all of the particulars of usage control information for users and/or user/distributors enabled by distributor A (as allowed, or not prevented, by senior control information). Distributor A may, for example, determine that a price of \$15 per month per user would meet distributor A's

business objectives with respect to payments from users for creator A's container. Distributor A must specify usage control information that fulfill the requirements of the distribution control information given to distributor A by creator A. For example, distributor A may include any required expiration dates and/or time-aged encryption keys in the specification of control information in accordance with creator A's requirements. If distributor A failed to include such information (or to meet other requirements) in their specification of control information, the control method(s) referenced in creator A's permissions record and securely invoked within a PPE 650 to actually create this control information would, in this example, fail to execute in the desired way (e.g. based on checks of proposed values in certain fields, a requirement that certain methods be included in permissions, etc.) until acceptable information were included in distributor A's control information specification.

In this example, user A may have established an account with distributor A such that user A may receive VDE managed content usage control information from distributor A. User A may receive content usage control information from distributor A to access and use creator A's content. Since the usage control information has passed through (and been added to, and/or modified by) a chain of handling including distributor A, the usage control information requested from distributor A to make

use of creator A's content will, in this example, reflect a composite of control information from creator A and distributor A. For example, creator A may have established a meter method that will generate an audit record if a user accesses creator A's VDE controlled content container if the user has not previously accessed the container within the same calendar month (e.g. by storing the date of the user's last access in a UDE associated with an open container event referenced in a method core of such a meter method and comparing such a date upon subsequent access to determine if such access has occurred within the same calendar month). Distributor A may make use of such a meter method in a control method (e.g. also created and/or provided by creator A, or created and/or provided by distributor A) associated with opening creator A's container that invokes one or more billing and/or budget methods created, modified, referenced in one or more permissions records and/or parameterized by distributor A to reflect a charge for monthly usage as described above. If distributor A has specified usage and/or redistribution control information within the boundaries permitted by creator A's senior control information, a new set of control information (shown as $D_A(C_A)$ in Figure 80) may be associated with creator A's VDE content container when control information associated with that container by distributor A are delivered to users and/or user/distributors (user A, user B, and user/distributor A in this example).

In this example, user A may receive control information related to creator A's VDE content container from distributor A. This control information may represent an extended agreement between user A and distributor A (e.g. regarding fees associated with use of content, limited redistribution rights, etc.) and distributor A and creator A (e.g. regarding the character, extent, handling, reporting, and/or other aspects of the use and/or creation of VDE controlled content usage information and/or content control information received, for example, by distributor A from creator A, or vice versa, or in other VDE content usage information handling). Such an extended agreement is enforced by processes operating within a secure subsystem of each participant's VDE installation. The portion of such an extended agreement representing control information of creator A as modified by distributor A in this example is represented by $D_A(C_A)$, including, for example, (a) control structures (e.g. one or more component assemblies, one or more permissions records, etc.), (b) the recording of usage information generated in the course of using creator A's content in conformance with requirements stated in such control information, (c) making payments (including automatic electronic credit and/or currency payments "executed" in response to such usage) as a consequence of such usage (wherein such consequences may also include electronically, securely and automatically receiving a bill delivered through use of VDE, wherein such a bill is derived from

said usage), (d) other actions by user A and/or a VDE secure subsystem at user A's VDE installation that are a consequence of such usage and/or such control information.

In addition to control information $D_A(C_A)$, user A may enforce her own control information on her usage of creator A's VDE content container (within the limits of senior content control information). This control information may include, for example, (a) transaction, session, time based, and/or other thresholds placed on usage such that if such thresholds (e.g. quantity limits, for example, self imposed limits on the amount of expenditure per activity parameter) are exceeded user A must give explicit approval before continuing, (b) privacy requirements of user A with respect to the recording and/or transmission of certain usage related details relating to user A's usage of creator A's content, (c) backup requirements that user A places on herself in order to help ensure a preservation of value remaining in creator A's content container and/or local store of electronic credit and/or currency that might otherwise be lost due to system failure or other causes. The right to perform in some or all of these examples of user A's control information, in some examples, may be negotiated with distributor A. Other such user specified control information may be enforced independent of any control information received from any content provider and may be set in relationship to a user's, or more generally, a VDE installation's,

control information for one or more classes, or for all classes, of content and/or electronic appliance usage. The entire set of VDE control information that may be in place during user A's usage of creator A's content container is referred to on Figure 80 as $U_A(D_A(C_A))$. This set may represent the control information originated by creator A, as modified by distributor A, as further modified by user A, all in accordance with control information from value chain parties providing more senior control information, and therefore constitutes, for this example, a "complete" VDE extended agreement between user A, distributor A, and creator A regarding creator A's VDE content container. User B may, for example, also receive such control information $D_A(C_A)$ from distributor A, and add her own control information in authorized ways to form the set $U_B(D_A(C_A))$.

User/distributor A may also receive VDE control information from distributor A related to creator A's VDE content container. User/distributor A may, for example, both use creator A's content as a user and act as a redistributor of control information. In this example, control information $D_A(C_A)$ both enables and limits these two activities. To the extent permitted by $D_A(C_A)$, user/distributor A may create their own control information based on $D_A(C_A)$ -- $UD_A(D_A(C_A))$ -- that controls both user/distributor A's usage (in a manner similar to that described above in connection with user A and user B), and control

information redistributed by user/distributor A (in a manner similar to that described above in connection with distributor A). For example, if user/distributor A redistributes $UD_A(D_A(C_A))$ to user/distributor B, user/distributor B may be required to report certain usage information to user/distributor A that was not required by either creator A or distributor A. Alternatively or in addition, user/distributor B may, for example, agree to pay user/distributor A a fee to use creator A's content based on the number of minutes user/distributor B uses creator A's content (rather than the monthly fee charged to user/distributor A by distributor A for user/distributor B's usage).

In this example, user/distributor A may distribute control information $UD_A(D_A(C_A))$ to user/distributor B that permits user/distributor B to further redistribute control information associated with creator A's content. User/distributor B may make a new set of control information $UD_B(UD_A(D_A(C_A)))$. If the control information $UD_A(D_A(C_A))$ permits user/distributor B to redistribute, the restrictions on redistribution from creator A in this example will prohibit the set $UD_B(UD_A(D_A(C_A)))$ from including further redistribution rights (e.g. providing redistribution rights to user B) because the chain of handling from distributor A to user/distributor A (distribution) and the continuation of that chain from user/distributor A to user/distributor B (first level of redistribution) and the further

continuation of that chain to another user represents two levels of redistribution, and, therefore, a set $UD_B(UD_A(D_A(C_A)))$ may not, in this example, include further redistribution rights.

As indicated in Figure 79, user B may employ content from both user/distributor B and distributor A (amongst others). In this example, as illustrated in Figure 80, user B may receive control information associated with creator A's content from distributor A and/or user/distributor B. In either case, user B may be able to establish their own control information on $D_A(C_A)$ and/or $UD_B(UD_A(D_A(C_A)))$, respectively (if allowed by such control information. The resulting set(s) of control information, $U_B(D_A(C_A))$ and/or $U_B(UD_B(UD_A(D_A(C_A))))$ respectively, may represent different control scenarios, each of which may have benefits for user B. As described in connection with an earlier example, user B may have received control information from user/distributor B along a chain of handling including user/distributor A that bases fees on the number of minutes that user B makes use of creator A's content (and requiring user/distributor A to pay fees of \$15 per month per user to distributor A regardless of the amount of usage by user B in a calendar month). This may be more favorable under some circumstances than the fees required by a direct use of control information provided by distributor A, but may also have the disadvantage of an exhausted chain of redistribution and, for

example, further usage information reporting requirements included in $UD_B(UD_A(D_A(C_A)))$. If the two sets of control information $D_A(C_A)$ and $UD_B(UD_A(D_A(C_A)))$ permit (e.g. do not require exclusivity enforced, for example, by using a registration interval in an object registry used by a secure subsystem of user B's VDE installation to prevent deregistration and reregistration of different sets of control information related to a certain container (or registration of plural copies of the same content having different control information and/or being supplied by different content providers) within a particular interval of time as an aspect of an extended agreement for a chain of handling and control reflected in $D_A(C_A)$ and/or $UD_B(UD_A(D_A(C_A)))$), user B may have both sets of control information registered and may make use of the set that they find preferable under a given usage scenario.

In this example, creator B creates a VDE content container and associates a set of VDE control information with such container indicated in Figure 81 as C_B . Figure 81 further shows the VDE participants who may receive enabling control information related to creator B's VDE content container. In this example, control information may indicate that distributors of creator B's content: (a) must pay creator B \$0.50 per kilobyte of information decrypted by users and/or user/distributors authorized by such a distributor, (b) may allow users and/or

user/distributors to embed their content container in another container while maintaining a requirement that creator B receive \$0.50 per kilobyte of content decrypted, (c) have no restrictions on the number of enabling control information sets that may be generated for users and/or user/distributors, (d) must report information concerning the number of such distributed control information sets at certain time intervals (e.g. at least once per month), (e) may create control information that allows users and/or user/distributors to perform up to three moves of their control information, (f) may allow redistribution of control information by user/distributors up to three levels of redistribution, (g) may allow up to one move per user receiving redistributed control information from a user/distributor.

In this example, distributor A may request control information from creator B that enables distributor A to distribute control information to users and/or user/distributors that is associated with the VDE container described above in connection with creator B. As stated earlier, distributor A has established a business model that favors "rental" of access rights to users and user/distributors receiving such rights from distributor A. Creator B's distribution control information in this example does not force a model including "rental" of rights, but rather bases payment amounts on the quantity of content decrypted by a user or user/distributor. In this example,

distributor A may use VDE to negotiate with creator B to include a different usage information recording model allowed by creator B. This model may be based on including one or more meter methods in control structures associated with creator B's container that will record the number of bytes decrypted by end users, but not charge users a fee based on such decryptions; rather distributor A proposes, and creator B's control information agrees to allow, a "rental" model to charge users, and determines the amount of payments to creator B based on information recorded by the bytes decrypted meter methods and/or collections of payment from users.

Creator B may, for example, (a) accept such a new control model with distributor A acting as the auditor (e.g. trusting a control method associated with processing audit information received by distributor A from users of creator B's content using a VDE secure subsystem at distributor A's site, and further to securely calculate amounts owed by distributor A to creator B and, for example, making payments to creator B using a mutually acceptable budget method managing payments to creator B from credit and/or currency held by distributor A), (b) accept such a new control model based on distributor A's acceptance of a third party to perform all audit functions associated with this content, (c) may accept such a model if information associated with the one or more meter methods that

record the number of bytes decrypted by users is securely packaged by distributor B's VDE secure subsystem and is securely, employing VDE communications techniques, sent to creator B in addition to distributor A, and/or (d) other mutually acceptable conditions. Control information produced by distributor A based on modifications performed by distributor A as permitted by C_B are referred to in this example as $D_A(C_B)$.

User A may receive a set of control information $D_A(C_B)$ from distributor A. As indicated above in connection with content received from creator A via a chain of handling including distributor A, user A may apply their own control information to the control information $D_A(C_B)$, to the extent permitted by $D_A(C_B)$, to produce a set of control information $U_A(D_A(C_B))$. The set of control information $D_A(C_B)$ may include one or more meter methods that record the number of bytes of content from creator B's container decrypted by user A (in order to allow correct calculation of amounts owed by distributor A to creator B for user A's usage of creator B's content in accordance with the control information of C_B that requires payment of \$0.50 per kilobyte of decrypted information), and a further meter method associated with recording usage such that distributor A may gather sufficient information to securely generate billings associated with user A's usage of creator B's content and based on a "rental" model (e.g. distributor A may, for example, have included a meter

method that records each calendar month that user A makes use of creator B's content, and relates to further control information that charges user A \$10 per month for each such month during which user A makes use of such content.)

User/distributor A may receive control information C_B directly from creator B. In this case, creator B may use VDE to negotiate with user/distributor A and deliver a set of control information C_B that may be the same or differ from that described above in connection with the distribution relationship established between creator B and distributor A. For example, user/distributor A may receive control information C_B that includes a requirement that user/distributor A pay creator B for content decrypted by user/distributor A (and any participant receiving distributed and/or redistributed control information from user/distributor A) at the rate of \$0.50 per kilobyte. As indicated above, user/distributor A also may receive control information associated with creator B's VDE content container from distributor A. In this example, user/distributor A may have a choice between paying a "rental" fee through a chain of handling passing through distributor A, and a fee based on the quantity of decryption through a chain of handling direct to creator B. In this case, user/distributor A may have the ability to choose to use either or both of C_B and $D_A(C_B)$. As indicated earlier in connection with a chain of handling including creator A

and distributor A, user/distributor A may apply her own control information to the extent permitted by C_B and/or $D_A(C_B)$ to form the sets of control information $UD_A(C_B)$ and $UD_A(D_A(C_B))$, respectively.

As illustrated in Figure 81, in this example, user B may receive control information associated with creator B's VDE content container from six different sources: C_B directly from creator B, $D_A(C_B)$ from distributor A, $UD_B(UD_A(D_A(C_B)))$ and/or $UD_B(UD_A(C_B))$ from user/distributor B, $D_C(C_B)$ from distributor C, and/or $D_B(D_C(C_B))$ from distributor B. This represents six chains of handling through which user B may enter into extended agreements with other participants in this example. Two of these chains pass through user/distributor B. Based on a VDE negotiation between user/distributor B and user B, an extended agreement may be reached (if permitted by control information governing both parties) that reflects the conditions under which user B may use one or both sets of control information. In this example, two chains of handling and control may "converge" at user/distributor B, and then pass to user B (and if control information permits, later diverge once again based on distribution and/or redistribution by user B).

In this example, creator C produces one or more sets of control information C_C associated with a VDE content container

created by creator C, as shown in Figure 82. Figure 82 further shows the VDE participants who may receive enabling control information related to creator C's VDE content container. The content in such a container is, in this example, organized into a set of text articles. In this example control information may include one or more component assemblies that describe the articles within such a container (e.g. one or more event methods referencing map tables and/or algorithms that describe the extent of each article). C_C may further include, for example: (a) a requirement that distributors ensure that creator C receive \$1 per article accessed by users and/or user/distributors, which payment allows a user to access such an article for a period of no more than six months (e.g. using a map-type meter method that is aged once per month, time aged decryption keys, expiration dates associated with relevant permissions records, etc.), (b) control information that allows articles from creator C's container to be extracted and embedded into another container for a one time charge per extract/embed of \$10, (c) prohibits extracted/embedded articles from being reextracted, (d) permits distributors to create enabling control information for up to 1000 users or user/distributors per month, (e) requires that information regarding the number of users and user/distributors enabled by a distributor be reported to creator C at least once per week, (f) permits distributors to enable users or user/distributors

to perform up to one move of enabling control information, and
(g) permits up to 2 levels of redistribution by user/distributors.

In this example, distributor B may establish a distribution relationship with creator C. Distributor B in this example may have established a business model that favors the distribution of control information to users and user/distributors that bases payments to distributor B based on the number of accesses performed by such VDE participants. In this example, distributor B may create a modified set $D_B(C_C)$ of enabling control information for distribution to users and/or user/distributors. This set $D_B(C_C)$ may, for example, be based on a negotiation using VDE to establish a fee of \$0.10 per access per user for users and/or user/distributors who receive control information from distributor B. For example, if one or more map-type meter methods have been included in C_C to ensure that adequate information may be gathered from users and/or user/distributors to ensure correct payments to creator C by distributor B based on C_C , such methods may be preserved in the set $D_B(C_C)$, and one or more further meter methods (and any other necessary control structures such as billing and/or budget methods) may be included to record each access such that the set $D_B(C_C)$ will also ensure that distributor B will receive payments based on each access.

The client administrator in this example may receive a set of content control information $D_B(C_C)$ that differs, for example, from control information received by user B from distributor B. For example, the client administrator may use VDE to negotiate with distributor B to establish a set of control information for content from all creators for whom distributor B may provide enabling content control information to the client administrator. For example, the client administrator may receive a set of control information $D_B(C_C)$ that reflects the results of a VDE negotiation between the client administrator and distributor B. The client administrator may include a set of modifications to $D_B(C_C)$ and form a new set $CA(D_B(C_C))$ that includes control information that may only be available to users and user/distributors within the same organization as the client administrator (e.g. coworkers, employees, consultants, etc.) In order to enforce such an arrangement, $CA(D_B(C_C))$ may, for example, include control structures that examine name services information associated with a user or user/distributor during registration, establish a new budget method administered by the client administrator and required for use of the content, etc.

A distributor may provide redistribution rights to a client administrator which allows said administrator to redistribute rights to create permissions records for certain content (redistribute rights to use said content) only within the

administrator's organization and to no other parties. Similarly, such administrator may extend such a "limited" right to redistribute to department and/or other administrator within his organization such that they may redistribute such rights to use content based on one or more restricted lists of individuals and/or classes and/or other groupings of organization personnel as defined by said administrator. This VDE capability to limit redistribution to certain one or more parties and/or classes and/or other groupings of VDE users and/or installations can be applied to content by any VDE content provider, so long as such a control is allowed by senior control information.

User D in this example may receive control information from either the client administrator and/or user/distributor C. User/distributor C may, for example, distribute control information $UD_C(CA(D_B(C_C)))$ to user D that includes a departmental budget method managed by user/distributor C to allow user/distributor C to maintain an additional level of control over the actions of user D. In this case, $UD_C(CA(D_B(C_C)))$ may include multiple levels of organizational controls (e.g. controls originating with the client administrator and further controls originating with user/distributor C) in addition to controls resulting from a commercial distribution channel. In addition or alternatively, the client administrator may refuse to distribute certain classes of control information to user D even if the client

administrator has adequate control information (e.g. control information distributed to user/distributor C that allows redistribution to users such as user D) to help ensure that control information flows through the client administrator's organization in accordance with policies, procedures, and/or other administrative processes.

In this example, user E may receive control information from the client administrator and/or distributor B. For example, user E may have an account with distributor B even though some control information may be received from the client administrator. In this case, user E may be permitted to request and receive control information from distributor B without restriction, or the client administrator may have, as a matter of organizational policy, control information in place associated with user E's electronic appliance that limits the scope of user E's interaction with distributor B. In the latter case, the client administrator may, for example, have limited user E to registering control information with the secure subsystem of user E's electronic appliance that is not available from the client administrator, is from one or more certain classes of distributors and/or creators, and/or has a cost for usage, such as a certain price point (e.g. \$50 per hour of usage). Alternatively or in addition, the client administrator may, for example, limit user E to receiving control information from distributor B in which user

E receives a more favorable price (or other control information criteria) than the price (or other criteria) available in control information from the client administrator.

In this example, creator D may create a VDE content container that is designed primarily for integration with other content (e.g. through use of a VDE extracting/embedding process), for example, content provided by creator B and creator C. Figure 83 shows the VDE participants who may receive enabling control information related a VDE content container produced by creator D. Control information associated with creator D's content (C_D in Figure 83) may include, for example: (a) a requirement that distributors make payment of either \$1.50 per open per user, or \$25 per user for an unlimited number of opens, (b) a discount of 20% for any user that has previously paid for an unlimited number of opens for certain other content created by creator D (e.g. implemented by including one or more billing methods that analyze a secure database of a user's VDE installation to determine if any of such certain other containers are registered, and further determines the character of rights held by a user purchasing rights to this container), (c) a requirement that distributors report the number of users and user/distributors enabled by control information produced in accordance with C_D after such number exceeds 1000, (d) a requirement that distributors limit the number of moves by users

and/or user/distributors to no more than one, (e) a requirement that distributors limit user/distributors to no more than four levels of redistribution, and (f) that distributors may create enabling control information that permits other distributors to create control information as distributors, but may not pass this capability to such enabled distributors, and further requires that audit information associated with use of control information by such enabled distributors shall pass directly to creator D without processing by such enabling distributor and that creator D shall pay such an enabling distributor 10% of any payments received by creator D from such an enabled distributor.

In this example, distributor C may receive VDE content containers from creator B, creator C, and creator D, and associated sets of control information C_B , C_C , and C_D . Distributor C may use the embedding control information and other control information to produce a new container with two or more VDE objects received from creator B, creator C, and creator D. In addition or alternatively, distributor C may create enabling control information for distribution to users and/or user/distributors (or in the case of C_D , for distributors) for such received containers individually. For example, distributor C may create a container including content portions (e.g. embedded containers) from creator B, creator C, and creator D in which each such portion has control information related to its access

and use that records, and allows an auditor to gather, sufficient information for each such creator to securely and reliably receive payments from distributor C based on usage activities related to users and/or user/distributors enabled by distributor C.

Furthermore, distributor C may negotiate using VDE with some or all of such creators to enable a model in which distributor C provides overall control information for the entire container based on a "uniform" fee (e.g. calculated per month, per access, from a combined model, etc.) charged to users and/or user/distributors, while preserving the models of each such creator with respect to payments due to them by distributor C based on C_B , C_C , and/or C_D , and, for example, resulting from each of their differing models for the collection of content usage information and any related (e.g. advertising) information.

In this example, distributor B may receive a VDE content container and associated content control information C_E from creator E as shown in Figure 83. If C_E permits, distributor B may extract a portion of the content in such a container.

Distributor B may then, for example, embed this portion in a container received from distributor C that contains an aggregation of VDE objects created by creator B, creator C, and creator D. Depending on the particular restrictions and/or permissions in the sets of control information received from each creator and distributor C, distributor B may, for example, be able

to embed such an extracted portion into the container received from distributor C as an independent VDE object, or directly into content of "in place" objects from creator B, creator C, and/or creator D. Alternatively, or in addition, distributor B may, if permitted by C_E, choose to distribute such an extracted portion of content as an independent VDE object.

User B may, in this example, receive a VDE content container from distributor C that is comprised of VDE objects created by creator B, creator C, and creator D. In addition, user B may receive a VDE content container from distributor B that contains the same content created by creator B, creator C, and creator D in addition to one or more extracted/embedded portions of content created by creator E. User B may base decisions concerning which of such containers they choose to use (including which embedded containers she may wish to use), and under which circumstances, based on, for example, the character of such extracted/embedded portions (e.g. multimedia presentations illustrating potential areas of interest in the remainder of the content, commentary explaining and/or expositing other elements of content, related works, improved application software delivered as an element of content, etc.); the quality, utility, and/or price (or other attributes of control information) of such portions; and other considerations which distinguish the

containers and/or content control information received, in this example, from distributor B and distributor C.

User B may receive content control information from distributor B for such a VDE content container that permits user B to add and/or modify content contained therein. User B may, for example, desire an ability to annotate content in such a container using a VDE aware word processor or other application(s). If permitted by senior control information, some or all of the content may be available to user B for modification and/or additions. In this case, user B is acting as a VDE creator for added and/or modified content. User B may, for example, provide new control information for such content, or may be required (or desire to) make use of existing control information (or control information included by senior members of a chain of handling for this purpose) to manage such content (based on control information related to such a container and/or contained objects).

In this example, VDE 100 has been used to enable an environment including, for example, content distribution, redistribution, aggregation (extracting and/or embedding), reaggregation, modification, and usage. The environment in this example allows competitive models in which both control information and content may be negotiated for and have different

particulars based on the chain of handling through which control information and/or content has been passed. Furthermore, the environment in this example permits content to be added to, and/or modified by, VDE participants receiving control information that enables such activities.

Example -- Content Distribution Through a Content VDE Chain of Handling

Figure 84 reflects certain aspects of a relatively simple model 3400 of VDE content distribution involving several categories of VDE participants. In this instance, and for simplicity of reference purposes, various portions of content are represented as discrete items in the form of VDE content container objects. One or more of such content portions may also be integrated together in a single object and may (as may the contents of any VDE content container object if allowed by content control information) be extracted in whole or part by a user. In this example, publishers of historical/educational multimedia content have created VDE content containers through the use of content objects available from three content resources:

- a Video Library 3402 product available to Publishers on optical discs and containing video clip VDE objects representing various historical situations,
- an Internet Repository 3404 which stores history information text and picture resources in VDE objects which are available for downloading to Publishers and other users, and

- an Audio Library 3406, also available on optical discs, and containing various pieces of musical performances and vocal performances (for example, historical narrations) which can be used alone or to accompany other educational historical materials.

The information provided in library 3402, repository 3404, and library 3406 may be provided to different publishers 3408(a), 3408(b), ..., 3408(n). Publishers 3408 may, in turn, provide some or all of the information they obtain to end users 3410.

In this example, the Video Library 3402 control information allows publishers to extract objects from the Video Library product container and content control information enabling use of each extracted object during a calendar year if the object has a license cost of \$50 or less, and is shorter than 45 minutes in duration, and 20,000 copies of each of any other extracted objects, and further requires all video objects to be VDE fingerprinted upon decryption. The Audio Library 3404 has established similar controls that match its business model. The Internet Repository 3406 VDE containerizes, including encrypts, selected object content as it streams out of the Repository in response to an online, user request to download an object. The Repository 3406 may fingerprint the identification of the receiving VDE installation into its content prior to encryption

and communication to a publisher, and may further require user identification fingerprinting of their content when decrypted by said Publisher or other content user.

The Publishers 3408 in this example have selected, under terms and conditions VDE negotiated (or otherwise agreed to) with the providing resources, various content pieces which they combine together to form their VDE object container products for their teacher customers. Publisher 3408(A) has combined video objects extracted from the Video Library 3402 (as indicated by circles), text and image objects extracted from the Internet Repository 3404 (indicated by diamonds), and one musical piece and one historical narration extracted from the Audio Library 3406 (as indicated by rectangles). Publisher 3408(B) has extracted a similar array of objects to be combined into his product, and has further added graphical elements (indicated by a hexagon) created by Publisher 3408(B) to enhance the product. Publisher 3408(C) has also created a product by combining objects from the Internet Repository 3404 and the Audio Library 3406. In this example, all publisher products are delivered, on their respective optical discs, in the form of VDE content container objects with embedded objects, to a modern high school for installation on the high school's computer network.

In this particular example, End-Users 3410 are teachers who use their VDE node's secure subsystems to access the VDE installation on their high school server that supports the publishers' products (in an alternative example, the high school may maintain only a server based VDE installation). These teachers license the VDE products from one or more of the publishers and extract desired objects from the VDE product content containers and either download the extracted VDE content in the form of VDE content containers for storage on their classroom computers and/or as appropriate and/or efficient. The teachers may store extracted content in the form of VDE content containers on server mass storage (and/or if desired and available to an end-user, and further according to acceptable pricing and/or other terms and conditions and/or senior content control information, they may store extracted information in "clear" unencrypted form on their nodes' and/or server storage means). This allows the teachers to play, and/or otherwise use, the selected portions of said publishers' products, and as shown in two instances in this example, add further teacher and/or student created content to said objects. End-user 3410(2), for example, has selected a video piece 1 received from Publisher A, who received said object from the Video Library. End-user 3410(3) has also received a video piece 3 from the same Publisher 3408(A) wherein said piece was also available to her from Publisher 3408(B), but perhaps under not as favorable terms and

conditions (such as a support consultation telephone line). In addition, end-user 3410(3) has received an audio historical narration from Publisher 3408(B) which corresponds to the content of historical reference piece 7. End-user 3410(3) has also received a corresponding historical reference piece 7 (a book) from publisher 3408(2) who received said book from the Internet Repository 3404. In this instance, perhaps publisher 3408(2) charged less for said book because end-user 3410(3) has also licensed historical reference piece 7 from him, rather than publisher 3408(1), who also carried the same book. End-user 3410(3), as a teacher, has selected the items she considers most appropriate for her classes and, through use of VDE, has been able to flexibly extract such items from resources available to her (in this instance, extracting objects from various optical products provided by publishers and available on the local high school network server).

Example -- Distribution of Content Control Information Within an Organization

Figure 85 shows two VDE content containers, Container 300(A) and Container 300(B), that have been distributed to a VDE Client Administrator 3450 in a large organization. As shown in the figure, Container 300(A) and Container 300(B), as they arrive at the corporation, carry certain control information specifying available usage rights for the organization. As can be further seen in Figure 85, the client administrator 3450 has distributed certain subsets of these rights to certain department administrators 3452 of her organization, such as Sales and Marketing Administrator 3452(1), Planning Administrator 3452(2), and Research and Development Administrator 3452(k). In each instance, the Client Administrator 3450 has decided which usage options and how much budget should be made available to each department.

Figure 85 is a simplified example and, for example, the Client Administrator 3450 could have added further VDE controls created by herself and/or modified and/or deleted in place controls (if allowed by senior content control information) and/or (if allowed by control information) she could have further divided the available monetary budget (or other budgets) among specific usage activities. In this example, departmental administrators have the same rights to determine the rights of departmental

end-users as the client administrator has in regard to departments. In addition, in this example (but not shown in Figure 85) the client administrator 3450 and/or content provider(s) may also determine certain control information which must directly control (including providing rights related to) end-user content usage and/or the consequences of said usage for all or certain classes of end-users. In the example shown in Figure 85, there are only three levels of VDE participants within the organization:

- a Client Administrator 3450,
- department administrators 3452, and
- end-users 3454.

In other examples, VDE will support many levels of VDE administration (including overlapping groups) within an organization (e.g., division, department, project, network, group, end-users, etc). In addition, administrators in a VDE model may also themselves be VDE content users.

Within an organization, VDE installations may be at each end-user 3454 node, only on servers or other multiple user computers or other electronic appliances, or there may be a mixed environment. Determination as to the mix of VDE server and/or node usage may be based on organization and/or content provider security, performance, cost overhead, or other considerations.

In this example, communications between VDE participants in Figure 85 employs VDE secure communication techniques between VDE secure subsystems supporting PPEs and other VDE secure system components at each VDE installation within the organization.

Example -- Another Content Distribution Example

Creators of VDE protected content may interact with other VDE participants in many different ways. A VDE creator 102 may, for example, distribute content and/or content control information directly to users, distribute content and/or content control information to commercial content repositories, distribute content and/or content control information to corporate content repositories, and/or distribute content and/or content control information to other VDE participants. If a creator 102 does not interact directly with all users of her content, she may transmit distribution permissions to other VDE participants that permit such participants to further distribute content and/or content control information. She may also allow further distribution of VDE content and/or content control information by, for example, not restricting redistribution of control information, or allowing a VDE participant to act as a "conduit" for one or more permissions records that can be passed along to another party, wherein said permissions record provides for including the identification of the first receiving party and/or the second receiving party.

Figure 86 shows one possible arrangement of VDE participants. In this example, creator 102 may employ one or more application software programs and one or more VDE secure subsystems to place unencrypted content into VDE protected

form (i.e., into one or more VDE content containers). In addition, creator 102 may produce one or more distribution permissions 3502 and/or usage permissions 3500 as an aspect of control information associated with such VDE protected content. Such distribution and/or usage permissions 3500, 3502 may be the same (e.g., all distribution permissions may have substantively all the same characteristics), or they may differ based on the category and/or class of participant for whom they are produced, the circumstances under which they are requested and/or transmitted, changing content control models of either creator 102 or a recipient, etc.

In this example, creator 102 transmits (e.g., over a network, via broadcast, and/or through transfer of physical media) VDE protected content to user 112a, user 112b, and/or user 112c. In addition, creator 102 transmits, using VDE secure communications techniques, usage permissions to such users. User 112a, user 112b, and user 112c may use such VDE protected content within the restrictions of control information specified by usage permissions received from creator 102. In this case, creator 102 may, for example, manage all aspects of such users activities related to VDE protected content transmitted to them by creator 102. Alternatively, creator 102 may, for example, include references to control information that must be

available to users that is not provided by creator 102 (e.g., component assemblies managed by another party).

Commercial content repository 200g, in this example, may receive VDE protected (or otherwise securely delivered) content and distribution, permissions and/or other content usage control information from creator 102. Commercial content repository 200g may store content securely such that users may obtain such, when any required conditions are met, content from the repository 200g. The distribution permissions 3502 may, for example, permit commercial content repository 200g to create redistribution permissions and/or usage permissions 3500, 3502 using a VDE protected subsystem within certain restrictions described in content control information received from creator 102 (e.g., not to exceed a certain number of copies, requiring certain payments by commercial content repository 200g to creator 102, requiring recipients of such permissions to meet certain reporting requirements related to content usage information, etc.). Such content control information may be stored at the repository installation and be applied to unencrypted content as it is transmitted from said repository in response to a user request, wherein said content is placed into a VDE container as a step in a secure process of communicating such content to a user. Redistribution permissions may, for example, permit a recipient of such permissions to create a

certain number of usage permissions within certain restrictions (e.g., only to members of the same household, business or other organization, etc.). Repository 200g may, for example, be required by control information received from creator 102 to gather and report content usage information from all VDE participants to whom the repository has distributed permissions.

In this example, power user 112d may receive VDE protected content and redistribution permissions from commercial content repository 200g using the desktop computer 3504. Power user 112d may, for example, then use application software in conjunction with a VDE secure subsystem of such desktop computer 3504 in order to produce usage permissions for the desktop computer 3504, laptop computer 3506 and/or settop appliance 3508 (assuming redistribution permissions received from commercial content repository 200g permit such activities). If permitted by senior control information (for example, from creator 102 as may be modified by the repository 200g), power user 112d may add her own restrictions to such usage permissions (e.g., restricting certain members of power user 112d's household using the settop appliance to certain times of day, amounts of usage, etc. based on their user identification information). Power user 112d may then transmit such VDE protected content and usage permissions to the laptop computer 3506 and the settop appliance 3508 using VDE secure

communications techniques. In this case, power user 112d has redistributed permissions from the desktop computer 3504 to the settop appliance 3508 and the laptop computer 3506, and periodically the settop appliance and the laptop computer may be required to report content usage information to the desktop computer, which in turn may aggregate, and/or otherwise process, and report user usage information to the repository 200g.

User 112e and/or user 112f may receive usage permissions and VDE protected content from commercial content repository 200g. These users may be able to use such content in ways authorized by such usage information. In contrast to power user 112d, these users may not have requested and/or received redistribution permissions from the repository 200g. In this case, these users may still be able to transfer some or all usage rights to another electronic appliance 600, and/or they may be permitted to move some of their rights to another electronic appliance, if such transferring and/or moving is permitted by the usage permissions received from the repository 200g. In this case, such other appliances may be able to report usage information directly to the repository 200g.

In this example, corporate content repository 702 within corporation 700 may receive VDE protected content and

distribution permissions from creator 102. The distribution permissions received by corporate repository 702 may, for example, include restrictions that limit repository 702 to distribution activities within corporation 700.

The repository 702 may, for example, employ an automated system operating in conjunction with a VDE secure subsystem to receive and/or transmit VDE protected content, and/or redistribution and/or usage permissions. In this case, an automated system may, for example, rely on criteria defined by corporate policies, departmental policies, and/or user preferences to determine the character of permissions and/or content delivered to various parties (corporation groups and/or individuals) within corporation 700. Such a system may, for example, automatically produce redistribution permissions for a departmental content repository 704 in response to corporation 700 receiving distribution permissions from creator 102, and/or produce usage permissions for user 112j and/or user 112k.

The departmental repository 704 may automatically produce usage permissions for user 112g, user 112h, and/or user 112i. Such users may access content from the corporate content repository 702, yet receive usage permissions from departmental repository 704. In this case, user 112g, user 112h, and/or user 112i may receive usage permissions from departmental

repository 704 that incorporate departmental restrictions in addition to restrictions imposed by senior control information (in this example, from creator 102, as may be modified by corporate repository 702, as may be further modified by departmental repository 704, that reflect a VDE extended agreement incorporating commercial requirements of creator 102 and corporation 700 in addition to corporate and/or departmental policies and agreements with corporate personnel of corporation 700).

Example—“Virtual Silicon Container”

As discussed above, VDE in one example provides a "virtual silicon container" ("virtual black box") in that several different instances of SPU 500 may securely communicate together to provide an overall secure hardware environment that "virtually" exists at multiple locations and multiple electronic appliances 600. Figure 87 shows one model 3600 of a virtual silicon container. This virtual container model 3600 includes a content creator 102, a content distributor 106, one or more content redistributors 106a, one or more client administrators 700, one or more client users 3602, and one or more clearinghouses 116. Each of these various VDE participants has an electronic appliance 600 including a protected processing environment 655 that may comprise, at least in part, a silicon-based semiconductor hardware element secure processing unit

500. The various SPUs 500 each encapsulate a part of the virtual distribution environment, and thus, together form the virtual silicon container 3600.

Example -- Testing/Examinations

A scheduled SAT examination for high school seniors is prepared by the Educational Testing Service. The examination is placed in a VDE container for scheduled release on November 15, 1994 at 1:00 PM Eastern Standard time. The SAT prepares one copy of the container for each school or other location which will conduct the examination. The school or other location ("test site") will be provided with a distributed examination container securely containing the VDE identification for the "administration" electronic appliance and/or test administrator at the test site (such as, a testing organization) and a budget enabling, for example, the creation of 200 test VDE content containers. Each container created at the test site may have a permissions record containing secure identification information for each electronic appliance 600, on the test site's network, that will be used by a test taker, as well as, for example, an identification for the student who will take the test. The student identification could, for example, be in the form of a secure PIN password which is entered by the student prior to taking the test (a test monitor or administrator might verify the student

identification by entering in a PIN password). Of course, identification might take the form of automated voice recognition, handwriting recognition (signature recognition), fingerprint information, eye recognition, or similar one or more recognition forms which may be used either to confirm the identity of the test taker (and/or test monitor/administrator) and/or may be stored with the test results in a VDE container or the like or in a location pointed to by certain container information. This identification may be stored in encrypted or unencrypted form. If stored in encrypted or otherwise protected form, certain summary information, such as error correction information, may be stored with the identification information to authenticate the associated test as corresponding to the identification.

As the student takes the test using the computer terminal, the answers selected may be immediately securely stored (but may be changed by the student during the test session). Upon the completion of the test, the student's answers, along with a reference to the test, are securely stored in a VDE reporting object which is passed along to the network to the test administrator and the administration electronic appliance 600. All test objects for all students could then be placed in a VDE object 300 for communication to the Educational Testing Service, along with whatever other relevant information (which may also be secured by VDE 100), including summary information giving

average and mean scores, and other information that might be desirable to summarize and/or act as an authentication of the test objects sent. For example, certain information might be sent separately from each student summary object containing information which helps validate the object as an "authentic" test object.

Applying VDE to testing scenarios would largely eliminate cheating resulting from access to tests prior to testing (normally the tests are stolen from a teacher or test administrator). At ETS, individuals who have access to tests could be limited to only a portion of the test to eliminate the risk of the theft of a "whole" test. Employing VDE would also ensure against processing errors or other manipulation of test answers, since absolutely authentic test results can be archived for a reasonable period of time.

Overall, employing VDE 100 for electronic testing will enable the benefits of electronic testing to be provided without the substantial risks associated with electronic storing, communicating, and processing of test materials and testing results. Electronic testing will provide enormous efficiency improvements, significantly lowering the cost of conducting and processing tests by eliminating printing, shipping, handling, and human processing of tests. At the same time, electronic testing

will allow users to receive a copy (encrypted or unencrypted) of their test results when they leave the test sessions. This will help protect the tested individual against lost of, or improperly processed, test results. Electronic testing employing VDE 100 may also ensure that timing related variables of testing (for example precise starting, duration, and stopping times) can be reliably managed. And, of course, proper use of VDE 100 for the testing process can prevent improper access to test contents prior to testing and ensure that test taking is properly audited and authenticated, that is which person took which test, at which time, on which electronic appliance, at which location. Retesting due to lost, stolen, improperly timed, or other variables can be avoided or eliminated.

VDE assisted testing may, of course, be employed for many different applications including secure identification of individuals for security/authentication purposes, for employment (e.g. applying for jobs) applications, and for a full range of evaluation testing. For example, an airline pilot, or a truck, train, or bus driver might take a test immediately prior to departure or during travel, with the test evaluating alertness to test for fatigue, drug use, etc. A certain test may have a different order and/or combination of test activities each time, or each group of times, the test is taken. The test or a master test might be stored in a VDE container (the order of, and which, test

questions might be determined by a process executed securely within an PPE 650). The test responses may be encrypted as they occur and either locally stored for aggregated (or other test result) transmission or dynamically transmitted (for example, to a central test administration computer). If the test taker "flunks" the test, perhaps he or she is then prevented from operating the vehicle, either by a local PPE 650 issuing control instructions to that effect on some portion of the vehicle's electronic control system or a local PPE failing to decrypt or otherwise provide certain key information required for vehicle operation.

Example -- Appliance Rental

Through use of the present invention, electronic appliances can be "leased" or otherwise provided to customers who, rather than purchasing a given appliance for unlimited usage, may acquire the appliance (such as a VCR, television, microwave oven, etc.) and be charged according to one or more aspects of use. For example, the charge for a microwave might be for each time it is used to prepare an item and/or for the duration of time used. A telephone jack could be attached, either consistently or periodically, to an inexpensive modem operatively attached or within the microwave (the modem might alternatively be located at a location which services a plurality of items and/or functions -- such as burglar alarm, light and/or heat control). Alternatively,

such appliances may make use of a network formed by the power cables in a building to transmit and receive signals.

At a periodic interval, usage information (in summary form and/or detailed) could be automatically sent to a remote information utility that collects information on appliance usage (the utility might service a certain brand, a certain type of appliance, and/or a collection of brands and/or types). The usage information would be sent in VDE form (e.g. as a VDE object 300). The information utility might then distribute information to financial clearinghouse(s) if it did not itself perform the billing function, or the information "belonging" to each appliance manufacturer and/or lessor (retailer) might be sent to them or to their agents. In this way a new industry would be enabled of leased usage of appliances where the leases might be analogous to car leasing.

With VDE installed, appliances could also be managed by secure identification (PIN, voice or signature recognition, etc.). This might be required each time a unit is used, or on some periodic basis. Failure to use the secure identification or use it on a timely basis could disable an appliance if a PPE 650 issued one or more instructions (or failed to decrypt or otherwise provide certain information critical to appliance operation) that prevented use of a portion or all of the appliance's functions.

This feature would greatly reduce the desirability of stealing an electronic appliance. A further, allied use of VDE is the "registration" of a VDE secure subsystem in a given appliance with a VDE secure subsystem at some control location in a home or business. This control location might also be responsible for VDE remote communications and/or centralized administration (including, for example, restricting your children from viewing R rated movies either on television or videocassettes through the recognition of data indicating that a given movie, song, channel, game, etc. was R rated and allowing a parent to restrict viewing or listening). Such a control location may, for example, also gather information on consumption of water, gas, electricity, telephone usage, etc. (either through use of PPEs 650 integrated in control means for measuring and/or controlling such consumption, or through one or more signals generated by non-VDE systems and delivered to a VDE secure subsystem, for example, for processing, usage control (e.g. usage limiting), and/or billing), transmit such information to one or more utilities, pay for such consumption using VDE secured electronic currency and/or credit, etc.

In addition, one or more budgets for usage could be managed by VDE which would prevent improper, excessive use of a certain, leased appliance, that might, for example lead to failure of the appliance, such as making far more copies using a

photocopier than specified by the duty cycle. Such improper use could result in a message, for example on a display panel or television screen, or in the form of a communication from a central clearinghouse, that the user should upgrade to a more robust model.

While the invention has been described in connection with what is presently considered to be the most practical and preferred embodiment, it is to be understood that the invention is not to be limited to the disclosed embodiment, but on the contrary, is intended to cover various modifications and equivalent arrangements included within the spirit and scope of the appended claims.

WE CLAIM:

1. A method for secure content delivery including:
 - a) encapsulating digital information within one or more digital containers;
 - b) encrypting at least one portion of said digital information;
 - c) associating at least partially secure control information for managing interaction with said encrypted digital information and/or the digital container;
 - d) delivering one or more of said one or more digital containers to a digital information user;
 - e) employing a protected processing environment for securely controlling decryption of at least a portion of said digital information.

2. A system for secure content delivery including:

encrypting means for encrypting at least one portion of digital information;

container processing means for encapsulating digital information within one or more digital containers and for associating at least partially secure control information for managing interaction with said encrypted digital information;

delivery means for delivering one or more of said one or more digital containers to a digital information user; and at least one protected processing environment for securely controlling decryption of at least a portion of said digital information.

3. A method for secure digital information delivery characterized by the steps of: (a) encrypting at least a portion of said digital information through the use of a first at least one VDE node, (b) creating and encrypting, through the use of said first at least one VDE node, control information to control use of at least a portion of said digital information by plural, users, (c) securely providing said control information to said plural users, and (d) employing at least one VDE node different from said first at least one VDE node to process at least portions of said control information and to control use of said encrypted digital information by said users.

4. A system for secure digital information delivery characterized by:

a first at least one VDE node for encrypting at least a portion of said digital information,

means for creating and encrypting, through the use of said first at least one VDE node, control information to control use of at least a portion of said digital information by plural, users,

means for securely providing said control information to said plural users, and

at least one VDE node different from said first at least one VDE node for processing at least portions of said control information and to control use of said encrypted digital information by said users.

5. A method for secure content delivery wherein at least partially encrypted content is encapsulated within at least one digital container and the digital container is delivered to a digital information user, the method characterized by the steps of:

associating, with the encapsulated content and/or the digital container, at least partially secure control information for managing interaction with the container and/or the content; and

employing a protected processing environment for securely controlling decryption of at least a portion of the encrypted content based at least in part on the control information.

6. A system for secure content delivery wherein at least partially encrypted content is encapsulated within at least one digital container and the digital container is delivered to a digital information user, the system characterized by:

a data structure that associates, with the encapsulated content and/or the digital container, at least partially secure

control information for managing interaction with the information; and

a protected processing environment for securely controlling decryption of at least a portion of the encrypted content based at least in part on the control information.

7. A method for secure digital information delivery characterized by the steps of: (a) encrypting at least a portion of said digital information, (b) associating protected control information to at least a portion of said digital information, and c) providing at least a portion of said encrypted digital information to a first user and at least in part controlling use of at least a portion of said encrypted digital information through the use of at least a portion of said protected control information, wherein said first user further provides at least one of (a) a copy of said at least a portion of said encrypted digital information , or (b) said encrypted digital information, to a second user, and wherein said second user associates further control information with said encrypted digital information for use in controlling use of said encrypted digital information by a third user.

8. A system for secure digital information delivery characterized by:

means for encrypting at least a portion of said digital information,

means for associating protected control information to at least a portion of said digital information,

means for providing at least a portion of said encrypted digital information to a first user

means for at least in part controlling use of at least a portion of said encrypted digital information through the use of at least a portion of said protected control information,

means for allowing the first user to provide at least one of (a) a copy of said at least a portion of said encrypted digital information, or (b) said encrypted digital information, to a second user, and

means for allowing said second user to associate further control information with said encrypted digital information for use in controlling use of said encrypted digital information by a third user.

9. A method for secure digital transaction management including:

- a) encrypting digital information at a first location;
- b) enabling a first party to securely associate at least one control with said information for use in ensuring at least one consequence of use of said information;
- c) enabling one or more additional parties to securely associate at least one further control with said

information for use in ensuring at least one consequence of use of said information;

- d) distributing at least a portion of said information to a party other than the first and additional parties at a location different from the locations of the first and additional locations; and
- f) decrypting at least a portion of said information at said third location, and ensuring said consequences of use of said information.

10. A system for secure digital transaction management including interconnected structures for performing the following functions:

- a) encrypting digital information;
- b) enabling a first party to securely associate at least one control with said information for use in ensuring at least one consequence of use of said information;
- c) enabling one or more additional parties to securely associate at least one further control with said information for use in ensuring at least one additional consequence of use of said information;
- d) distributing at least a portion of said information to a further party; and
- e) decrypting at least a portion of said information; and
- f) securely ensuring said consequences.

11. A system for secure digital transaction management wherein digital information is encrypted by a first party at a first location and distributed, characterized by:

a first protected processing environment for enabling the first party to securely associate at least a first control with said information,

a further protected processing environment for enabling the further party to securely associate at least a further control with said information, and

a still further protected processing environment for decrypting at least a portion of said information while controlling at least one consequence of use of the information based at least in part on the first and further controls.

12. A method for secure digital transaction management wherein digital information is encrypted by a first party at a first location and distributed, characterized by the following steps:

enabling the first party to securely associate at least a first control with said information,

enabling a further party to securely associate at least a further control with said information, and

transmitting the first and further controls; and

decrypting at least a portion of said information while controlling at least one consequence at least in part on the transmitted controls.

13. A method for securely automating distributed electronic processes including:

- a) providing secure, interoperable, general purpose rights management processing means to multiple, parties;
- b) establishing secure process management controls for automatically, at least partially remotely, and securely supporting requirements related to electronic events;
- c) securely distributing process management controls to party sites;
- d) securely maintaining at least a portion of said process management controls under the control of party processing means at said party sites;
- e) automatically managing electronic processes at said party sites to enforce interests related to said electronic content.

14. A system for securely automating distributed electronic processes including:

interoperable rights management processing means disposed at multiple parties' sites;

control establishing means for establishing secure process management controls; for remotely, automatically, and securely supporting requirements related to electronic events; and for

securely distributing process management controls to party sites;

security means for securely maintaining at least a portion of said process management controls under the control of processing means at said party sites; and

managing means for automatically managing electronic processes at plural party sites to enforce interests related to said electronic events.

15. A method for automating distributed electronic processes using interoperable processors at multiple sites, characterized by the following steps:

securely distributing, to the processors, process management controls for automatically, and securely supporting requirements related to electronic events;

securely maintaining at least a portion of said process management controls under the control of the processors; and

automatically managing, in a distributed manner with the processors, electronic processes at the multiple sites to enforce interests related to electronic events.

16. A system for automating distributed electronic processes using interoperable processors at multiple sites, characterized by the following:

distributing means connected to the processors for securely distributing, to the processors, process management controls for remotely, automatically, and securely supporting requirements related to electronic events;

process control means for securely maintaining at least a portion of said process management controls under the control of the processors; and

management means for automatically managing, in a distributed manner with the processors, electronic processes at the multiple sites to enforce the interests related to the electronic events.

17. A method of securely enforcing a rights seniority system characterized by the steps of:

allowing a first user to create at least one control over electronic content; and

allowing a second user to contribute at least one further control over electronic content and/or alter the control in place, the second control being subject to the first control.

18. A system for securely enforcing a rights seniority system characterized by:

a first secure environment for allowing a first user to contribute at least one control over electronic content; and

a second secure environment for allowing a second user to contribute at least one further control over electronic content and/or alter the control in place, the second control being subject to the first control.

19. A method of securely enforcing a rights seniority system characterized by the step of allowing a first user to create at least one electronic control that at least in part dictates the rights a second user has to create further electronic controls over the use of and/or access to electronic content.

20. A system for securely enforcing a rights seniority system characterized by at least one means for allowing a first user to create at least one electronic control that at least in part dictates the rights a second user has to create further electronic controls over the use of and/or access to electronic content.

21. A method for employing protected processing environments including:

- a) distributing interoperable protected processing environments to plural parties;
- b) providing a first interoperable protected processing environment for use by a first party to enable said party to (a) encrypt digital information, and (b)

- create control information for managing at least one aspect of use of said digital information;
- c) encrypting said digital information in response to one or more instructions from said first party;
 - d) making said digital information available to a second party;
 - e) through the use of a second interoperable protected processing environment, satisfying requirements enforced by said control information and allowing said second party to use at least a portion of said digital information;
 - f) through the use of said second interoperable protected processing environment securely reporting information reflecting at least one aspect of said second party use of said digital information.

22. A system for employing protected processing environments including:

interoperable protected processing environments distributed to plural parties, including a first interoperable protected processing environment for use by a first party to enable said party to (a) encrypt digital information, and (b) create control information for managing at least one aspect of use of said digital information, and further including a second interoperable protected processing environment;

means for encrypting said digital information in response to one or more instructions from said first party, and for making said digital information available to a second party;

means for a second interoperable protected processing environment to satisfy requirements enforced by said control information and to allow said second party to use at least a portion of said digital information; and to securely report information reflecting at least one aspect of said second party's use of said digital information.

23. A method for employing protected processing environments distributed to plural parties characterized by the following steps:

using a first protected processing environment to encrypt digital information, and control information specifying requirements for managing at least one aspect of use of said digital information;

using a second protected processing environment interoperable with the first protected processing environment to enforce the requirement specified by said control information and conditionally allowing use of at least a portion of said digital information; and

using the second protected processing environment to report information reflecting at least one aspect of use of said digital information.

24. A system for employing protected processing environments distributed to plural parties characterized by:

- a first protected processing environment to encrypt digital information, and for handling control information specifying requirements for managing at least one aspect of use of said digital information;
- a second protected processing environment interoperable with the first protected processing environment for enforcing at least one requirement specified by said control information and conditionally allowing use of at least a portion of said digital information; and for reporting information reflecting at least one aspect of use of said digital information.

25. A secure network architecture comprising multiple cooperating interconnected nodes having protected processing environments, at least a portion of said nodes being able to intercommunicate, characterized in that VDE-protected information can be moved from a source node to a destination node and processed at least in part by the destination node.

26. In a secure network architecture comprising multiple cooperating interconnected nodes having protected processing environments, the nodes being able to intercommunicate, a method comprising the step of moving VDE-protected

information from a source node to a destination node and processed at least in part by the destination node.

27. A secure local area network topology comprising multiple cooperating interconnected nodes, characterized in that at least some of the nodes comprise network workstations with software defining protected processing environments, and at least one of the nodes comprises a secure database server that provides information in protected form for processing by the network workstation protected processing environments.

28. In a secure local area network topology comprising multiple cooperating interconnected nodes, a method characterized by the steps of:

executing, at least in part with network workstations, software defining protected processing environments, and providing, with a secure database server, information for processing by the network workstation protected processing environments.

29. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that at least one of the plural nodes provides a protected processing environment that performs

a server function for a client comprising at least a portion of the protected processing environment of at least one other node.

30. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a method characterized by providing, with at least one of the plural nodes, a protected processing environment; and performing, with the protected processing environment, a server function for a client comprising at least a portion of the protected processing environment of at least one other node.

31. A method for securely managing electronic negotiations related to electronic commerce value chain activities including:

- a) employing a protected processing environment by a first party to securely specify rules and/or controls for managing an electronic commerce process;
- b) securely making said specified rules and/or controls available to a second party;
- c) employing a protected processing environment different from said first protected processing environment to further securely specify rules and/or controls for managing at least one commerce process related to the common commercial interests of said first party and said second party;

- d) employing said protected processing environment to securely electronically negotiate at least one aggregate rules and/or controls set representing the electronic interests of both said first party and said second party;
- e) employing a protected processing environment to manage said electronic commerce process consistent with at least a portion of said aggregate rules and/or controls set.

32. A system for securely managing electronic negotiations related to electronic commerce value chain activities including:

a first party's protected processing environment for securely specifying rules and/or controls for managing an electronic commerce process, and for securely making said specified rules and/or controls available to a second party;

a second party's protected processing environment different from said first party's protected processing environment to further securely specify rules and/or controls including means for managing at least one commerce process related to the common commercial interests of said first party and said second party;

at least one of the first party's and the second party's protected processing environment for securely electronically negotiating at least one aggregate rules and/or controls set

representing the electronic interests of both said first party and said second party; and

at least one of the first party's and the second party's protected processing environment including means for managing said electronic commerce process consistent with said at least a portion of said aggregate rules and/or controls set.

33. A method for securely managing electronic negotiations related to electronic commerce value chain activities through use of first and second protected processing environment characterized by:

using the first environment, securely specifying rules and/or controls for managing an electronic commerce process;

using the second environment, further securely specifying rules and/or controls for managing at least one commerce process related to the commercial interests of a first and a second party;

employing at least one of the first and second protected processing environments to securely electronically negotiate at least one aggregate rules and/or controls set representing the electronic interests of the first party and said second party; and

employing at least one of the first and second protected processing environment to manage said electronic commerce process consistent with at least a portion of said aggregate rules and controls set.

34. A system for securely managing electronic negotiations related to electronic commerce value chain activities through use of first and second protected processing environment characterized by:

the first environment including means for securely specifying rules for managing an electronic commerce process;

the second environment including means for further securely specify rules for managing at least one commerce process related to the commercial interests of first and second parties;

at least one of the first and second protected processing environments including means for securely electronically negotiating at least one aggregate rules set at least partially representing the electronic interests of said first party and said second party; and

at least one of the first and second protected processing environment including means for managing said electronic commerce process consistent with said at least a portion of said aggregate rules set.

35. A method for managing a distributed electronic commerce environment including:

- a) establishing a secure, certificate authority for authenticating a user identity for an electronic

- commerce participant wherein said identity includes one or more user class parameters;
- b) certifying said user identity through the use of one or more certificates enabled by said certificate authority;
 - c) controlling the use of distributed electronic information based at least in part on class parameter information included in such certified identity.

36. A system for securely managing a distributed electronic commerce environment including:

means for establishing a user identify for an electronic commerce participant wherein said identity includes one or more user class parameters;

a certificate authority for authenticating such user identity by certifying said user identity through the use of one or more certificates enabled by said certificate authority; and

means for controlling the use of distributed electronic information based at least in part on class parameter information included in such certified identity.

37. A method for securely managing a distributed electronic commerce environment to allow interaction with an electronic commerce participant having a user identity that is certified by a certificate authority, characterized by:

establishing a user identity;
certifying the user identity and the user class parameter;
and
associating, with the user identity, at least one user class parameter, wherein said certified class parameter, at least in part, is used to control use of distributed electronic information.

38. A system for managing a distributed electronic commerce environment to allow interaction with an electronic commerce participant having a certified user identity, characterized by:

means for associating at least one user class parameter with an established user identity;

means for ascertaining the authenticity of the user identity and/or the user class parameter; and

means for controlling use of distributed electronic information based at least in part on said status.

39. A system as in claim 38 wherein the class parameter represents the user's age, and the controlling means includes means for controlling the use of distributed electronic information based on the user's age.

40. A method of securely establishing user identity through use of certificates, the method characterized by:

presenting an electronic token reflecting at least one user class characteristic;

determining whether an electronic certificate authenticates the user class characteristic reflected by the token; and

using the token as a basis for granting rights.

41. A system for identifying a user through use of certificates, the system characterized by:

means presenting an electronic token reflecting at least one user class characteristic;

means for obtaining an electronic certificate;

means for determining whether the electronic certificate authenticates the user class characteristic reflected by the token;

and

means for using the certified, authenticated token as a basis for granting rights.

42. A system for securely managing a distributed electronic commerce environment including:

means for identifying an electronic commerce participant by specifying at least one user category;

means for authenticating such user identity; and

means for controlling the use of distributed electronic information based at least in part on the user category.

43. A method for securely managing a distributed electronic commerce environment to allow interaction with an electronic commerce participant, characterized by:

establishing a user identity and an associated user class parameter; and

using the class parameter to, at least in part, control use of distributed electronic information.

44. A system for managing a distributed electronic commerce environment to allow interaction with an electronic commerce participant, characterized by:

means for associating at least one user class parameter with a user identity;

means for authenticating the user identity and/or the user class parameter; and

means for controlling use of distributed electronic information based at least in part on said status.

45. A system as in claim 44 wherein the class parameter represents the user's age, and the controlling means includes means for controlling the use of distributed electronic information based on the user's age.

46. A method of securely establishing user identity, the method characterized by:

presenting an electronic token reflecting at least one user class characteristic;

determining the user class characteristic reflected by the token is authentic; and

using the token as at least a partial basis for granting rights.

47. A system for securely establishing user identity characterized by:

means presenting an electronic token reflecting at least one user class characteristic;

authenticating the user class characteristic reflected by the token; and

means for using the authenticated token as a basis for granting rights.

48. A method of authenticating a user identity, the method characterized by:

receiving a certificate request and associated user identity; and

issuing an electronic certificate for use in authenticating at least one user class characteristic associated with the user identity for granting rights based on the user class characteristic.

49. A system for authenticating user identity, characterized by:

- means for receiving a certificate request and associated user identity; and
- means for issuing an electronic certificate for use in authenticating at least one user class characteristic associated with the user identity for granting rights based on the user class characteristic.

50. A method of securely establishing user identity, the method characterized by:

- receiving a certificate request; and
- issuing an electronic certificate specifying at least one user class characteristic.

51. A system for securely establishing user identity through use of certificates, characterized by:

- means for receiving a certificate request and associated user identity; and
- means for issuing an electronic certificate specifying at least one user class characteristic.

52. A method or system of managing rights characterized in that a cryptographically signed token is used to certify membership in a class, the token is authenticated, and the class membership represented by the token is used as a basis for granting and/or withholding rights and/or permissions.

53. A method or system of managing rights characterized in that a cryptographically signed token is used to certify membership in a class, the status of such token is ascertained, and the class membership represented by the token is used as a basis for allowing a user presenting the token to create electronic rules.

54. A method or system of managing rights characterized in that a cryptographically signed token is used to certify membership in a class, the token is validated, and the class membership represented by the token is used as a basis for allowing a user presenting the token to exercise rights under electronic rules.

55. A method for enabling a distributed electronic commerce electronic agreement system including:

- a) enabling distributed, interoperable secure client protected processing environment nodes;

- b) establishing at least one system wide secure communications key;
- c) employing public key encryption for communications between plural client nodes;
- d) supporting the delivery of electronic control information by individual clients wherein said control information at least in part specifies their respective electronic commerce agreement rights;
- e) supporting at least one protected processing environment for determining the respective and/or collective rights of said clients by establishing one or more electronic agreements based at least in part on said secure delivery of electronic control information;
- f) employing a secure software container data control structure for ensuring persistent maintenance of the electronic rights of the clients;
- g) using secure software containers which provide for data structures that support rules and/or controls corresponding to electronic commerce model agreement enforcement.

56. A distributed electronic agreement system including:
plural distributed, interoperable secure client protected processing environment nodes for supporting delivery of electronic control information by individual clients wherein said

control information at least in part specifies said client's respective electronic commerce model agreement rights, and for employing public key encryption and authentication for communications between said plural client nodes;

means coupled to said nodes for establishing at least one system wide secure communications key; and

at least one protected processing environment for:

- (a) determining the respective and/or collective rights of electronic commerce model clients by establishing one or more electronic agreements based at least in part on said secure delivery of electronic control information;
- (b) employing a secure software container data control structure for ensuring persistent maintenance of the electronic rights of commerce model clients; and
- (c) using secure software containers which provide for data structures that support controls corresponding to electronic commerce model agreement enforcement.

57. A method for enabling a distributed electronic commerce electronic agreement system including distributed, interoperable secure client protected processing environment nodes employing at least one system wide secure communications key, employing public key encryption and authentication for

communications between plural client nodes, and employing an certification authority for establishing client identity, the method characterized by:

supporting the , secure delivery of electronic commerce model agreement rights control information;

determining the respective and/or collective rights of electronic commerce model clients by establishing one or more electronic agreements based at least in part on said secure delivery of the electronic control information;

employing a secure software container data control structure for ensuring remote, persistent maintenance of the electronic rights of commerce model clients; and

using secure software containers which provide for data structures supporting rules and controls corresponding to electronic commerce model agreement enforcement.

58. A distributed electronic commerce electronic agreement system including:

distributed, interoperable secure client protected processing environment nodes employing at least one system wide secure communications key, employing public key encryption and authentication for communications between plural client nodes, employing an certification authority for establishing client identity, and supporting the, secure delivery of electronic commerce model agreement rights control information;

means disposed in at least one node for determining the respective and/or collective rights of electronic commerce model clients by establishing one or more electronic agreements based at least in part on said secure delivery of the electronic control information; and

means disposed in at least one node for employing a secure software container data control structure for ensuring remote, persistent maintenance of the electronic rights of commerce model clients, and for using secure software containers which provide for data structures supporting rules and controls corresponding to electronic commerce model agreement enforcement.

59. A method of securely handling electronic currency characterized by the following steps:

packaging electronic currency within a software container, and

delivering the software container as payment for goods or services.

60. A system for securely handling electronic currency characterized by:

means for packaging electronic currency within a software container, and

means for delivering the software container as payment for goods or services.

61. A method or system for managing rights within an organization characterized in that electronic containers are distributed within the organization, the electronic containers having controls associated therewith, the controls enforcing, at least in part, an organizational hierarchy relating to the use of the containers and/or the contents thereof.

62. A method of organizational rights management characterized by the steps of:
distributing an electronic container within an organization and
restricting usage, access and/or further distribution of the electronic container or the contents thereof within or outside of the organization based on electronic controls associated with the electronic container.

63. A system for organizational rights management characterized by:
means for distributing an electronic container and
means for restricting usage, access and/or further distribution of the electronic container or the contents thereof

within or outside of the organization based on electronic controls associated with the electronic container.

64. A method of organizational rights management characterized by the steps of:

distributing electronic containers within an organization,
and

using the electronic containers, at least in part, to administer content usage by persons within the organization.

65. A system for organizational rights management characterized by:

means for distributing electronic containers within an organization, and

means for using the electronic containers, at least in part, to administer content usage by persons within the organization.

66. A method of organizational rights management characterized by the steps of:

distributing electronic containers within an organization,
and

using the electronic containers, at least in part, to administer use of money within the organization.

67. A system for organizational rights management characterized by electronic containers distributed within an

organization for, at least in part, administering use of money within the organization.

68. A method of organizational rights management characterized by the steps of:
distributing protected processing environments within an organization, and
using the environments to, at least in part, to administer content usage by persons within the organization.

69. A system for organizational rights management characterized by protected processing environments distributed within an organization, for, at least in part, administering content usage within the organization.

70. A method of organizational rights management characterized by the steps of:
distributing protected processing environments within an organization, and
using the processing environments to, at least in part, to administer use of money by persons within the organization.

71. A system for organizational rights management characterized by plural protected processing environments

distributed within an organization for, at least in part,
administering use of money within the organization.

72. A rights management appliance including:
a user input device,
a user display device,
at least one processor, and
at least one element defining a protected processing
environment,
characterized in that the protected processing environment
stores and uses permissions, methods, keys, programs and/or
other information to electronically manage rights.

73. In a rights management appliance including:
a user input device,
a user display device,
at least one processor, and
at least one element defining a protected processing
environment,
a method of operating the appliance characterized by the
step of storing and using permissions, methods, keys, programs
and/or other information to electronically manage rights.

74. A rights management appliance including at least one
processor element at least in part defining a protected processing

environment, characterized in that the protected processing environment stores and uses permissions, methods, keys, programs and/or other information to electronically manage rights.

75. In a rights management appliance including at least one processor element at least in part defining a protected processing environment, a method comprising storing and using permissions, methods, keys, programs and/or other information to electronically manage rights.

76. A method of electronically storing information in a repository and distributing it on request, characterized in that the information is protected by associating electronic controls with the information, the electronic controls serving to enforce rights in the information.

77. A system for electronically storing information in a repository and distributing it on request, characterized by means for protecting information by associating electronic controls with the information, and further including means for using the electronic controls to enforce rights in the information.

78. A self-protecting electronic container comprising:
an electronic container structure for containing digital information, and
an electronic protection mechanism that protects or destroys the digital information in the event of tampering.

79. A method for a self-protecting electronic container comprising an electronic container structure for containing digital information, the method characterized by detecting an attempt at tampering and protecting or destroying the digital information in the said attempt.

80. A method of creating a self-protecting container system comprising:
providing at least one property,
providing at least one attribute,
providing at least one cryptographic key,
providing at least one organizational structure relating the key to the property and/or attribute, and
encapsulating the property, the attribute, the cryptographic key and the organizational structure, either explicitly or by reference, into an electronic container structure.

81. A self-protecting container system comprising:
at least one property,

at least one attribute,
at least one cryptographic key, and
at least one organizational structure relating the key to the
property and/or attribute.

82. A distributed electronic rights management system
comprising plural nodes having protected processing
environments, characterized in that each node can perform self-
administering processes in response to electronic components.

83. A self-administering electronic component comprising:
at least one method for performing at least a portion of a
transaction,
at least one method for generating audit information, and
at least one method for securely receiving and interpreting
administrative information.

84. A self-administering electronic component performing
the following methods:
at least one method for performing at least a portion of a
transaction,
at least one method for generating audit information, and
at least one method for securely receiving and interpreting
administrative information.

85. A self-describing electronic component defining at least one parameter and/or function, characterized in that the component includes at least one secure, descriptive portion used to create a human readable interface describing the parameter and/or function.

86. A method for processing a self-describing electronic component defining at least one parameter and/or function, characterized by the step of creating, at least in part with the component, a human readable interface describing the parameter and/or function based at least in part on at least one secure, descriptive portion of the component.

87. A method of performing an electronic transaction comprising:

receiving plural components,
electronically detecting the occurrence of an event,
determining, based on the event, a subset of the plural received components to process the event, and
performing, in response to the event, at least one electronic process based on the component subset.

88. A system for performing an electronic transaction comprising:

means for receiving plural components,

means for electronically detecting the occurrence of an event,

means for determining, based on the event, a subset of the plural received components to process the event, and

means for performing, in response to the event, at least one electronic process based on the component subset.

89. A distributed transaction processing method characterized by the following steps:

receiving a first electronic component at a first location,

receiving a second electronic component at a second location,

electronically detecting occurrence of an event at the first location,

processing, in response to the event detection, a first portion of an electronic transaction at the first location based at least in part on the first electronic component,

securely transmitting at least one signal from the first location to the second location, and

processing at least a second portion of the electronic transaction at the second location based at least in part on the second electronic component.

90. A method as in claim 89 further characterized by:
sending at least one signal from the second location to the first location, and
performing at least a third portion of the electronic transaction at the first location based at least in part on receipt of the signal from the second location.

91. A distributed transaction processing system characterized by:
means at a first location for receiving a first electronic component, for electronically detecting occurrence of an event, for processing, in response to the event detection, a first portion of an electronic transaction at the first location based at least in part on the first electronic component, and for securely transmitting at least one signal from the first location to a second location; and
means at the second location for receiving a second electronic component, and for processing at least a second portion of the electronic transaction based at least in part on the second electronic component.

92. A system as in claim 91 further characterized by:
means at the second location for sending at least one signal from the second location to the first location, and

means at the first location for performing at least a third portion of the electronic transaction at the first location based at least in part on receipt of the signal from the second location.

93. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that each node can perform electronic processes in response to receipt and assembly of electronic components, and the node authenticates each of the electronic components before assembling them.

94. A distributed electronic rights management method comprising:

performing, with at least one protected processing environment, electronic processes in response to receipt and assembly of electronic components, and

authenticating, within the protected processing environment, each of the electronic components before assembling them.

95. A method as in claim 94 wherein the authenticating step includes the step of obtaining a corresponding certificate from a certifying authority.

96. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that each node can perform electronic processes in response to receipt and assembly of electronic components, and the node authenticates each of the electronic components by obtaining a corresponding certificate from a certifying authority.

97. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a certifying authority that issues certificates allowing each node to authenticate electronic components before assembling them to perform and/or control electronic rights management processes.

98. In a distributed electronic rights management system comprising plural nodes each having a protected processing environment, a method characterized by the step of issuing certificates allowing each node to authenticate electronic components before assembling them to perform and/or control electronic rights management processes.

99. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that said nodes enforce usage

and/or access controls and is capable of electronically obtaining compensation from a user and/or other processing of usage information for subsequent transfer to rights holders.

100. In a distributed electronic rights management system comprising plural nodes having a protected processing environment, a method characterized by the step of enforcing usage and/or access controls and electronically obtaining compensation from a user and/or other processing of usage information for subsequent transfer to rights holders.

101. A distributed electronic rights management system comprising plural nodes each having a protected processing environment, characterized in that each node enforces usage and/or access controls based on receipt of information from multiple other nodes.

102. A distributed electronic rights management method characterized by the step of enforcing, with a protected processing environment, usage and/or access controls based on receipt of information from multiple other nodes.

103. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that said nodes are capable of at

least temporarily extending electronic credit to an associated user for use in compensating rights holders.

104. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a method of operating the environment characterized by the step of at least temporarily extending electronic credit to an associated user for use in compensating rights holders.

105. A distributed electronic rights management system comprising plural nodes each having a protected processing environment, characterized in that said nodes are capable of requesting and obtaining a user-specific electronic credit assurance from a clearinghouse before granting the user rights to access and/or use electronically protected information.

106. In a distributed electronic rights management system comprising plural nodes each having a protected processing environment, a method characterized by the step of requesting and obtaining a user-specific electronic credit assurance from a clearinghouse before granting the user rights to access and/or use electronically protected information.

107. A distributed electronic rights management system comprising plural nodes each having a protected processing environment, characterized in that each node is capable of performing and/or requesting an electronic debit or credit transaction as a condition to granting the user rights to access and/or use electronically protected information.

108. In a distributed electronic rights management system comprising plural nodes each having a protected processing environment, a method characterized by the step of performing and/or requesting an electronic debit or credit transaction as a condition to granting the user rights to access and/or use electronically protected information.

109. A distributed electronic rights management system comprising plural nodes each having a protected processing environment, characterized in that each node can maintain an audit trail of user activities for reporting to a centralized location, the centralized location analyzing the user activities based on the audit trail.

110. In a distributed electronic rights management system comprising plural nodes each having a protected processing environment, a method characterized by the steps of:

maintaining, a plural locations, audit trails of user activities for reporting to a centralized location, and analyzing, at the centralized location, the user activities based on the audit trail.

111. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that said node can monitor user activities and trigger the occurrence of unrelated events based on the user activities and/or the electronic controls that associate the user activities with the unrelated events.

112. A system as in claim 111 wherein the unrelated event is activation of an application program.

113. A system as in claim 111 wherein the unrelated event is use of a secure container.

114. A system as in claim 111 wherein the unrelated event is use of the protected processing environment.

115. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a method characterized by the step of monitoring user activities at said nodes, and triggering the occurrence of

unrelated events based on the user activities and electronic controls that associate the user activities with the unrelated events.

116. A method as in claim 115 wherein the unrelated event is at least one of:

activation of an application program,
use of a secure container, and
use of the protected processing environment.

117. A method of compromising a distributed electronic rights management system comprising plural nodes having protected processing environments, characterized by the following steps:

exposing a certification private key to allow a person to pass a challenge/response protocol,
defeating at least one of (a) an initialization challenge/response security, and/or (b) exposing external communication keys,
creating a processing environment based at least in part on the above-mentioned steps, and
participating in distributed rights management using the processing environment.

118. A processing environment for compromising a distributed electronic rights management system comprising plural nodes having protected processing environments, characterized by the following:

means including an exposed certification private key to pass a challenge/response protocol,

means for defeating at least one of (a) an initialization challenge/response security, and/or (b) exposing external communication keys, and

means for participating in distributed rights management.

119. A method of compromising a distributed electronic rights management system comprising plural nodes having protected processing environments, characterized by the step of compromising the permissions record of an electronic container and using the compromised permissions record to access and/or use electronic information.

120. A system for compromising a distributed electronic rights management system comprising plural nodes having protected processing environments, characterized by means for using a compromised permissions record of an electronic container for accessing and/or using electronic information.

121. A method of tampering with a protected processing environment characterized by the steps of:
discovering at least one system-wide key, and
using the key to obtain access to content and/or administrative information without authorization.

122. An arrangement including means for using at least one compromised system-wide key to decrypt and compromise content and/or administrative information of a protected processing environment without authorization.

123. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that said nodes can electronically fingerprint content before releasing it in unprotected form.

124. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a method characterized by performing, in at least one of the nodes, the step of electronically fingerprinting content before releasing it in unprotected form.

125. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that said nodes can embed,

within the electronic content, an electronic fingerprint containing specified information identifying a content rights holder and/or an indication of origin before including the content in an electronic container or allowing access to such content.

126. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a method characterized by the step of embedding, within electronic content, an electronic fingerprint containing specified information, including information identifying a content rights holder and/or an indication of origin before including the content in an electronic container or allowing access to such content.

127. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that the system includes one or more usage clearinghouses that receive usage information from one or more of the plural nodes.

128. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a method characterized by the step of receiving, with a usage clearinghouse, usage information from one or more of said plural nodes.

129. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that the system includes one or more financial clearinghouses that receive financial information relating to the use of or access to content from one or more of nodes.

130. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a method characterized by the step of receiving, with one or more financial clearinghouses, financial information from one or more of the plural nodes.

131. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that the system includes one or more analysis clearinghouses that receive information from one or more of the plural nodes and analyzes the received information.

132. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a method characterized by the step of receiving, with one or more analysis clearinghouses, information from one

or more of the plural nodes and analyzing the received information.

133. A method of processing information pertaining to the use of or access to electronic content wherein such information is received from one or more nodes having protected processing environments.

134. A method of providing credit for interaction with content to a protected processing environment node.

135. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that the system includes one or more clearinghouses that transmits rights and/or permissioning information to one or more of the plural nodes.

136. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a method characterized by the step of transmitting rights and/or permissioning information from a clearinghouse to one or more of the plural nodes.

137. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that the system includes one or more clearinghouses that periodically transmit cryptographic material to one or more of said nodes, the cryptographic material renewing and/or replacing expiring cryptographic material.

138. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a method characterized by the step of periodically transmitting cryptographic material from one or more clearinghouses to one more of said nodes, the cryptographic material renewing and/or replacing expiring cryptographic material.

139. A secure electronic container characterized in that the container contains electronic controls for controlling the use of and/or access to electronic content that is external to the container.

140. A method comprising:
accessing electronic controls within a secure electronic container; and

using the controls for at least in part controlling the use of and/or access to electronic content that is external to the container.

141. A secure electronic container characterized in that the container contains electronic controls for controlling, at least in part, the use of and/or access to distributed electronic content.

142. A method comprising:
accessing electronic controls within a secure electronic container; and
using the controls for controlling, at least in part, the use of and/or access to distributed electronic content.

143. A secure electronic container characterized in that the container contains electronic controls that cause electronic content to expire on a time-dependent basis.

144. A method for processing a secure electronic container including the step of causing, at least in part based on electronic controls within the container, electronic content to expire on a time-dependent basis.

145. A method of metering use of and/or access to electronic information characterized by the step of maintaining a bitmap meter data structure including data partitions that subdivide the metering information by time and/or subject matter.

146. A system for metering use of and/or access to electronic information characterized by means for maintaining a bitmap meter data structure including data partitions that subdivide the metering information by time and/or subject matter.

147. A distributed electronic rights management system comprising plural nodes having protected processing environments, characterized in that the system permits at least some of the nodes to securely describe permitted uses of electronic content and securely enforces said description.

148. In a distributed electronic rights management system comprising plural nodes having protected processing environments, a method characterized by the steps of permitting at least some of the nodes to securely describe permitted uses of electronic content, and securely enforcing said description.

149. A document management system comprising one or more electronic appliances containing one or more secure processing units and one or more secure databases operatively connected to at least one of said secure processing units, said system further including protected usage control information wherein (a) at least a portion of said control information is securely stored within one or more of said secure databases, and (b) at least a portion of said control information governs the production of usage information, at least a portion of which usage information is reported to one or more parties.

150. In a document management system comprising one or more electronic appliances containing one or more secure processing units and one or more secure databases operatively connected to at least one of said secure processing units, a method for processing protected usage control information including the steps of securely storing at least a portion of said control information within one or more of said secure databases, and (b) based at least in part on said control information, governing the production of usage information and the reporting of at least a portion of said usage information to one or more parties.

151. A document management system comprising plural electronic appliances containing protected processing

environments and one or more secure databases operatively connected to at least one of said protected processing environments, said system further including protected usage control information, wherein (a) at least a portion of said control information is securely stored within one or more of said secure databases, and (b) at least a portion of said control information governs the production of usage information and the reporting of at least a portion of said usage information to one or more parties.

152. In a document management system comprising plural electronic appliances containing protected processing environments and one or more secure databases operatively connected to at least one of said protected processing environments, a method of handling usage control information including the steps of (a) securely storing at least a portion of said control information within one or more of said secure databases, and (b) governing, based on at least a portion of said control information, the production of usage information and the reporting of at least a portion of said usage information to one or more parties.

153. An electronic contract system comprising electronic appliances containing one or more secure processing units and one or more secure databases operatively connected to at least

one of the secure processing units, said system furthering including means for enabling plural parties to enter into an electronic arrangement, at least one of said databases containing secure control information for managing at least a portion of a plural party electronic arrangement.

154. In an electronic contract system comprising plural electronic appliances containing one or more secure processing units and one or more secure databases operatively connected to at least one of the secure processing units, a method characterized by the steps of enabling plural parties to enter into to an electronic arrangement, and using secure control information contained by at least one of said databases for managing at least a portion of a plural party electronic arrangement.

155. An electronic appliance arrangement containing at least one secure processing unit and at least one secure database operatively connected to at least one of said secure processing unit(s), said arrangement including means to monitor usage of at least one aspect of appliance usage and control said usage based at least in part upon protected appliance usage control information.

156. In an electronic appliance arrangement containing at least one secure processing unit and at least one secure database operatively connected to at least one of said secure processing unit(s), a method characterized by the steps of monitoring usage of at least one aspect of appliance usage and controlling said usage based at least in part upon protected appliance usage control information.

157. An electronic appliance arrangement containing a protected processing environment and at least one secure database operatively connected to said protected processing environment, said arrangement including means to monitor usage of at least one aspect of an amount of appliance usage and control said usage based at least in part upon protected appliance usage control information processed at least in part through use of said protected processing environment.

158. In an electronic appliance arrangement containing a protected processing environment and at least one secure database operatively connected to said protected processing environment, a method characterized by the steps of monitoring usage of at least one aspect of appliance usage and controlling said usage based at least in part upon protected appliance usage control information processed at least in part through use of said protected processing environment.

159. An electronic appliance arrangement containing one or more CPUs wherein at least one of the CPUs incorporates an integrated secure processing unit, said arrangement storing protected appliance usage control information designed to be securely processed by said integrated secure processing unit.

160. In an electronic appliance arrangement containing one or more CPUs wherein at least one of the CPUs incorporates an integrated secure processing unit, a method including the step of storing and securely processing protected modular component appliance usage control information with said integrated secure processing unit.

161. An electronic appliance arrangement containing at least one first secure processing unit and one or more video controllers where at least one of the video controllers incorporates at least one second secure processing unit, said arrangement storing protected video function control information designed to be securely processed by said incorporated secure processing unit(s).

162. In an electronic appliance arrangement containing at least one first secure processing unit and one or more video controllers where at least one of the video controllers incorporates at least one second secure processing unit, the

method characterized by the step of storing protected video function control information designed to be securely processed by said incorporated secure processing unit(s).

163. An electronic appliance arrangement containing one or more video controllers where at least one of the video controllers incorporates at least one secure processing unit, said arrangement storing protected video function control information designed to be securely processed by said incorporated secure processing unit(s), wherein at least a portion of said video function control information is stored within a secure database operatively connected to at least one of said at least one secure processing units.

164. In an electronic appliance arrangement containing one or more video controllers where at least one of the video controllers incorporates at least one secure processing unit, a method including the steps of storing protected video function control information designed to be securely processed by said incorporated secure processing unit(s), within a database operatively connected to at least one of said at least one secure processing units.

165. An electronic appliance arrangement containing one or more video controllers and at least one secure processing unit,

said arrangement storing component, modular protected video function control information designed to be securely processed by said secure processing unit(s), wherein at least a portion of said video function control information is stored within a secure database operatively connected to at least one of said at least one secure processing unit(s).

166. An electronic appliance arrangement containing one or more video controllers and at least one secure processing unit, a method including the step of storing component, modular protected video function control information designed to be securely processed by said secure processing unit(s), within a secure database operatively connected to at least one of said at least one secure processing unit(s).

167. An electronic appliance arrangement containing at least one secure processing unit and one or more network communications means where at least one of the network communications means incorporates at least one further secure processing unit, said arrangement storing protected networking control information designed to be processed by said incorporated secure processing unit(s).

168. In an electronic appliance arrangement containing at least one secure processing unit and one or more network

communications means, a method characterized by the steps of incorporating, within at least one of the network communications means, at least one further secure processing unit, storing networking control information at least in part within said incorporated secure processing unit(s), and securely processing said protected networking control information with said secure processing unit(s).

169. An electronic appliance arrangement containing one or more modems where at least one of the modems incorporates at least one secure processing unit, said arrangement storing modular, component protected modem control information designed to be securely processed by said incorporated secure processing unit(s).

170. In an electronic appliance arrangement containing one or more modems where at least one of the modems incorporates at least one secure processing unit, a method characterized by the step of storing and securely processing modular, component protected modem control information with said incorporated secure processing unit(s).

171. An electronic appliance arrangement containing at least one secure processing unit and one or more modems where at least one of the modems includes at least one further secure

processing unit, said arrangement storing protected modem control information designed to be securely processed by said included secure processing unit(s).

172. In an electronic appliance arrangement containing at least one secure processing unit and one or more modems where at least one of the modems includes at least one further secure processing unit, a method including the step of storing and securely processing protected modem control information within said included secure processing unit(s).

173. An electronic appliance arrangement containing at least one secure processing unit and one or more CD-ROM devices where at least one of the CD-ROM devices incorporates at least one further secure processing unit, said arrangement storing protected CD-ROM control information designed to be securely processed by said incorporated secure processing unit(s).

174. In an electronic appliance arrangement containing at least one secure processing unit and one or more CD-ROM devices where at least one of the CD-ROM devices incorporates at least one further secure processing unit, a method characterized by the step of storing and securely processing protected CD-ROM

control information within said incorporated secure processing unit(s).

175. An electronic appliance arrangement containing one or more network communications means where at least one of the network communications means incorporates at least one secure processing unit, said arrangement storing modular, component, protected networking control information designed to be securely processed by said incorporated secure processing unit(s).

176. In an electronic appliance arrangement containing one or more network communications means where at least one of the network communications means incorporates at least one secure processing unit, a method characterized by the step of storing and securely processing protected networking control information with said incorporated secure processing unit(s).

177. A set-top controller arrangement containing a protected processing environment and a database operatively connected to said protected processing environment, said arrangement further containing control information for controlling usage of said controller based upon processing of at least a portion of said control information within said protected processing environment, wherein at least a portion of said control information is stored within said database.

178. In a set-top controller arrangement containing a protected processing environment and a database operatively connected to said protected processing environment, a method characterized by the step of: (a) using control information within the set-top controller arrangement for controlling usage of said controller based upon processing of at least a portion of said control information within said protected processing environment, and storing at least a portion of said control information within said database.

179. An electronic game arrangement containing a protected processing environment for controlling the use of electronic games, said arrangement including game usage control information, database means operatively connected to said protected processing environment for, at least in part, storing usage control information for regulating at least some aspect of use of at least a portion of at least one of said games, and traveling objects containing protected electronic game content.

180. In an electronic game arrangement containing a protected processing environment for controlling the use of electronic games, a method including the steps of:

(a) including game usage control information within a database means operatively connected to said protected processing environment; and

(b) regulating, at least in part with the stored usage control information, at least some aspect of use of at least a portion of at least one of said games.

181. A method as in claim 178 further including the step of regulating the use of traveling objects containing protected electronic game content.

182. An electronic game arrangement containing interoperable protected processing environments for controlling the use of interactive games, said arrangement including protected game usage control information, and database means operatively connected to said protected processing environments for, at least in part, storing game usage control information.

183. In an electronic game arrangement containing protected processing environments, a method comprising:

(a) storing, within a secure database means operatively connected to said protected processing environments protected game usage control information; and

(b) controlling the use of interactive games based at least in part on the storing game usage control information.

184. An electronic game arrangement containing interoperable protected processing environments for controlling

the use of games, said arrangement including component, modular, protected game usage control information, wherein at least a portion of said protected control information was provided independently by plural parties securing their respective rights in at least one electronic value chain.

185. In an electronic game arrangement containing interoperable protected processing environments for controlling the use of games, a method including the steps of:

(a) providing at least a portion of component, modular, protected game usage control information independently by plural parties; and

(b) using the control information at least in part to securing respective rights of said plural parties in at least one electronic value chain.

186. An electronic multimedia arrangement containing protected processing environments for controlling the use of multimedia, said arrangement including component, modular multimedia usage control information and database means operatively connected to said protected processing environments for, at least in part, storing multimedia usage control information.

187. In an electronic multimedia arrangement containing protected processing environments for controlling the use of multimedia, a method including the steps of storing multimedia usage control information within a database means operatively connected to said protected processing environments, and using the stored control information to control multimedia.

188. An electronic multimedia arrangement containing a protected processing environment for controlling the use of multimedia, said arrangement including multimedia usage control information, database means operatively connected to said protected processing environment for, at least in part, storing multimedia usage control information, and protected traveling objects containing distributed multimedia electronic content.

189. In an electronic multimedia arrangement containing a protected processing environment, a method characterized by the steps of storing multimedia usage control information within a database means operatively connected to said protected processing environment, and controlling, based at least in part on the stored information, protected traveling objects containing distributed multimedia electronic content.

190. An electronic multimedia arrangement containing interoperable protected processing environments for controlling the use of multimedia, said arrangement including component, modular, protected multimedia usage control information, wherein at least a portion of said protected control information was provided independently by plural parties securing their respective rights in at least one electronic value chain.

191. A system as in claim 188 further including a secure processing unit.

192. In an electronic multimedia arrangement containing protected processing environments, a method comprising providing at least a portion of component, modular, protected multimedia usage control information independently by plural parties securing their respective rights in at least one electronic value chain, and using the usage control information to control the use of multimedia.

193. A method as in claim 190 wherein the using step is performed at least in part within a secure processing unit.

194. An integrated circuit supporting multiple encryption algorithms comprising at least one microprocessor, memory, input/output means, at least one circuit for encrypting and/or

decrypting information and one or more software programs for use with at least one of the microprocessors to perform encryption and/or decryption functions.

195. In a secure integrated circuit supporting multiple encryption algorithms comprising at least one microprocessor, memory, input/output means, and providing a protected processing environment, a method characterized by executing at least a portion of one or more software programs with the microprocessor to perform encryption and/or decryption functions within the integrated circuit.

196. An integrated circuit comprising at least one microprocessor, memory, at least one real time clock, at least one random number generator, at least one circuit for encrypting and/or decrypting information and independently delivered and/or independently deliverable certified software.

197. An integrated circuit comprising at least one microprocessor, memory, input/output means, a tamper resistant barrier and at least a portion of a Rights Operating System.

198. An integrated circuit comprising at least one microprocessor, memory, input/output means, at least one real

time clock, a tamper resistant barrier and means for recording interruption of power to at least one of the real time clocks.

199. A method of distributing information characterized by the steps of compressing information, encrypting the compressed information at the first location, distributing the encrypted information to one or more second locations, using a tamper resistant integrated circuit to first decrypt and then decompress the information.

200. A system for distributing information characterized by:

means for compressing information,

means for encrypting the compressed information at the first location,

means for distributing the encrypted information to one or more second locations, and

means for using a tamper resistant integrated circuit to first decrypt and then decompress the information.

201. A method of securely managing distributed events characterized by the steps of providing secure event processing environments to one or more users, enabling a first user to specify control information for event management through the use of a first secure event processing environment, and managing

the processing of such an event through the use of a second secure event processing environment.

202. A system for securely managing distributed events characterized by:

a first secure event processing environment for enabling a first user to specify control information for event management, and

a second secure event processing environment interoperable with the first event processing environment for managing the processing of such an event.

203. A method for enabling electronic commerce chain of handling and control characterized by the step of a first and a second party independently specifying protected, modular component control information describing requirements related to the operation of an electronic commerce value chain.

204. A system for enabling electronic commerce chain of handling and control characterized by means for permitting a first and a second party to independently specify protected, modular component control information describing requirements related to the operation of an electronic commerce value chain of handling and control, and means for securely enforcing the requirements described by the control information.

205. A method for enabling electronic commerce characterized by the step of a first and a second party independently stipulating control information managing the use of digital information, wherein said first and said second party independently maintain persistent rights enforced by said control information as said digital information moves through a chain of handling and control.

206. A system for enabling electronic commerce including:
means for allowing a first party to stipulate control information managing the use of digital information,
means for allowing a second party to stipulate control information managing the use of the digital information, and
chain of handling and control means for maintaining persistent rights enforced by said control information as said digital information moves from one location and/or process to another.

207. A method for secure maintenance of electronic rights comprising a first step of plural parties in a value chain independently and securely stipulating control information regarding their electronic rights, wherein said control information is used to enforce conditions related to the use of electronic information distributed in software containers.

208. A system for secure maintenance of electronic rights comprising:

means permitting plural parties in a value chain to independently and securely stipulates control information regarding their electronic rights, and

means for using said control information to enforce conditions related to the use of electronic information distributed in software containers.

209. A method for securely controlling the use of protected electronic content including the step of supporting modular separate control information arrangements for managing at least one event related to use of said content such that a user may select between separate control information arrangements for managing such at least one event.

210. A system for securely controlling the use of protected electronic content including modular separate control information arrangements for managing at least one event related to use of said content such that a user may select between separate control information arrangements for managing such at least one event.

211. A method employing separate, modular control structures for managing the use of encrypted digital information

characterized by the step of enabling commercial value chain participants to support plural relationships between two or more of: (1) content event triggering, (2) auditing, and (3) budgeting, control variables.

212. A system for employing separate, modular control structures for managing the use of encrypted digital information characterized by means for enabling commercial value chain participants to support plural relationships between two or more of: (1) content event triggering, (2) auditing, and (3) budgeting, control variables.

213. A method of chain of handling and control enabling a party not directly participating in an electronic value chain to contribute secure control information to enforce at least one control requirement, said method characterized by a first step of a first value chain participant stipulating control information associated with digital information and a second step wherein said not directly participating party independently and securely contributes secure control information for inclusion in an aggregate control information set including said associated control information, said aggregate control information at least in part managing conditions related to the use of at least a portion of said digital information by a second value chain participant.

214. A chain of handling and control system for enabling a party not directly participating in an electronic value chain to contribute secure control information to enforce at least one control requirement, said system characterized by:

means for allowing a first value chain participant to stipulate control information associated with digital information,

means for allowing the not directly participating party to independently and securely contribute secure control information for inclusion in an aggregate control information set including said associated control information,

and means responsive to said aggregate control information for at least in part managing conditions related to the use of at least a portion of said digital information by a second value chain participant.

215. A method of electronic commerce control information management for delegating the administration of certain rights held by a value chain party to a second value chain party characterized by the step of said first party stipulating secure control information describing at least a portion of their rights related to one or more chain of handling and control electronic events wherein said first party provides further control information authorizing said second party to administer some or all of said rights as an agent for said first party.

216. A system for electronic commerce control information management for delegating the administration of certain rights held by a value chain party to a second value chain party characterized by:

means for allowing said first party to stipulate secure control information describing at least a portion of their rights related to one or more chain of handling and control electronic events; and

means for allowing said first party to provide further control information authorizing said second party to administer some or all of said rights as an agent for said first party.

217. A method of governing taxation of commercial events resulting from electronic chain of handling and control characterized by a first step of distributing secure digital information to a user and specifying secure control information controlling at least one condition for use of said digital information and a second step of a government agency securely, independently contributing secure control information for automatically governing tax payments for said commercial events.

218. A system for governing taxation of commercial events resulting from electronic chain of handling and control characterized by:

means for distributing secure digital information to a user;
means for specifying secure control information controlling
at least one condition for use of said digital information; and
means for allowing a government agency to securely,
independently contribute secure control information for
automatically governing tax payments for said commercial
events.

219. A method of governing privacy rights related to
electronic events characterized by a first step of a first party
protecting digital information containing information descriptive
of preventing a second party from at least one unauthorized use
and a second step of specifying certain control information
related to use of at least a portion of said protected digital
information, wherein said control information enforces at least
one right of said second party related to privacy and/or permitted
use(s) of personal and/or proprietary information included in said
protected digital information.

220. A system for governing privacy rights related to
electronic events characterized by:

means for permitting a first party to protect digital
information containing information descriptive of preventing a
second party from at least one unauthorized use;

means for specifying certain control information related to use of at least a portion of said protected digital information; and

means for using the control information to enforce at least one right of said second party related to privacy and/or permitted use(s) of personal and/or proprietary information included in said protected digital information.

221. A method of governing privacy rights related to electronic events characterized by a first step of a first party protecting digital information from at least one unauthorized use and stipulating certain control information for establishing conditions for use of said protected information and a second step of a user of said digital information stipulating further control information regulating the reporting of information regarding said user's use of at least a portion of said digital information.

222. A system for governing privacy rights related to electronic events characterized by:

means for allowing a first party to protect digital information from at least one unauthorized use and for stipulating certain control information for establishing conditions for use of said protected information; and

means for allowing a user of said digital information to stipulate further control information regulating the reporting of

information regarding said user's use of at least a portion of said digital information.

223. A secure method for regulating electronic conduct and commerce characterized by a step of distributing interoperable protected processing environments and circulating amongst plural recipients of said protected processing environments software containers containing digital content and related content control information prepared for use by at least a portion of said protected processing environments, wherein said method includes the further step of regulating the use at least some of said digital content based, at least in part, on the secure processing of at least a portion of said control information through the use of at least one protected processing environment.

224. A secure system for regulating electronic conduct and commerce characterized by:

distributed interoperable protected processing environments,

means for circulating, amongst said protected processing environments, software containers containing digital content and related content control information prepared for use by at least a portion of said protected processing environments, and

means within at least some of the protected processing environments for regulating the use at least some of said digital

content based, at least in part, on the secure processing of at least a portion of said control information.

225. A method of electronic commerce networking for enabling a secure electronic retail environment characterized by the step of supplying user certified control information, smart cards, secure processing units, and retailing terminal arrangements networked together using VDE communication techniques and secure software containers.

226. An electronic commerce networking system for enabling a secure electronic retail environment characterized by:
means for networking together smart cards, secure processing units, and retailing terminal arrangements; and
means for making the smart cards, secure processing units, and retailing terminal arrangements interoperable with one another and with VDE communication techniques and secure software containers.

227. A method of enabling electronic commerce appliances for securely administering user rights in commerce activities characterized by the step of providing to users at least a portion of a VDE node contained within a physical device, said device being configured to be compatible with mating connectors in host

systems for supporting secure, interoperable transaction activity between plural parties.

228. A system for securely administering user rights in commerce activities comprising a physical device including at least a portion of a portable VDE node, said device being configured to be compatible with mating connectors in host systems for supporting secure, interoperable transaction activity between plural parties.

229. A method for enabling a programmable, electronic commerce environment characterized by the step of providing to multiple parties secure commerce nodes that securely process separate, modular component billing management methods, budgeting management methods, metering management methods, and related auditing management methods and further characterized by the step of supporting triggering of metering, auditing, billing, and budgeting methods in response to electronic commerce event activities.

230. A programmable, electronic commerce environment characterized by secure commerce nodes each including:

means for securely processing separate, modular component billing management methods, budgeting management

methods, metering management methods, and related auditing management methods, and

means for supporting triggering of metering, auditing, billing, and budgeting methods in response to electronic commerce event activities.

231. An electronic commerce system including modular, standardized control components comprising electronic commerce event control instructions stipulated by commerce participants, and plural electronic appliances containing one or more secure processing units which process at least a portion of such commerce event control instructions, said system further containing one or more databases, operatively connected to at least one of the secure processing units, for at least in part securely storing at least a portion of such control instructions for use by said at least one secure processing unit.

232. In an electronic commerce system including modular, standardized control components comprising electronic commerce event control instructions stipulated by commerce participants, and plural electronic appliances containing one or more secure processing units which process at least a portion of such commerce event control instructions, a method characterized by the step of providing one or more secure databases, operatively connected to at least one of the secure processing units, and at

least in part securely storing, within the secure databases, at least a portion of such control instructions for use by said at least one secure processing unit.

233. A content distribution system comprising plural electronic appliances containing one or more interoperable secure processing units operatively connected to one or more databases for use with at least one of said secure processing units, said one or more databases containing (a) one or more decryption keys for use in decrypting distributed, encrypted digital information, and (b) encrypted audit information, said audit information reflecting at least one aspect of use of said distributed digital information

234. A content distribution method comprising:
distributing plural electronic appliances containing one or more interoperable secure processing units
operatively connecting the appliances to one or more databases,
storing within said one or more databases one or more decryption keys,
using the decryption keys for decrypting distributed, encrypted digital information, and
storing within the one or more databases encrypted audit information, said audit information reflecting at least one aspect of use of said distributed digital information.

235. An electronic currency system comprising plural, electronic appliances containing (a) protected processing environments, (b) encrypted electronic currency and related secure control information configured so as to be useable by at least one of said protected processing environments, and (c) usage reporting means for securely communicating electronic currency usage related information from a first interoperable protected processing environment to a second interoperable protected processing environment.

236. An electronic currency method comprising:
distributing plural, electronic appliances containing (a) protected processing environments, (b) encrypted electronic currency and related secure control information configured so as to be useable by at least one of said protected processing environments, and
securely communicating electronic currency usage related information from a first interoperable protected processing environment to a second interoperable protected processing environment.

237. A method for electronic financial activities characterized by the steps of:

communicating digital containers containing financial information from a first interoperable secure node to a second interoperable secure node, communicating modular, standard control information to said second secure node to, at least in part, set the conditions for use of at least a portion of said financial information, reporting information related to said use to said first interoperable secure node.

238. A system for electronic financial activities characterized by:

means for communicating digital containers containing financial information from a first interoperable secure node to a second interoperable secure node,

means for communicating modular, standard control information to said second secure node,

means at the second node for, at least in part, setting the conditions for use of at least a portion of said financial information, and

means for reporting information related to said use from the second secure node to said first interoperable secure node.

239. A method for electronic currency management including:

communicating encrypted electronic currency from a first, interoperable secure user node to a second interoperable user node using at least one secure container, and

providing secure control information for use with said at least one secure container, said secure control information, at least in part, maintaining conditionally anonymous currency usage information.

240. A system for electronic currency management including:

means for communicating encrypted electronic currency from a first, interoperable secure user node to a second interoperable user node using at least one secure container, and

means for providing secure control information for use with said at least one secure container, said secure control information, at least in part, maintaining conditionally anonymous currency usage information.

241. A method for electronic financial activities management characterized by the steps of:

securely communicating from a first secure node to a second secure node financial information standardized control information for controlling the use of financial information used in a financial value chain,

securely communicating from said first secure node to a third secure node said financial information standardized control information for controlling the use of financial information used in a financial value chain,

securely communicating encrypted financial information from said second secure node to said third secure node, including communicating secure control information, processing said financial information at said third node at least in part through the use of secure control information supplied by said first and said second secure nodes, wherein said standardized control information is at least in part stored in a secure database contained within said third secure node.

242. A system for electronic financial activities management characterized by the steps of:

means coupled to a first and a second secure node for securely communicating from said first secure node to said second secure node financial information standardized control information for controlling the use of financial information used in a financial value chain,

means coupled between the first secure node and a third secure node for securely communicating from said first secure node to said third secure node said financial information standardized control information for controlling the use of financial information used in a financial value chain,

means coupled between the second and third nodes for securely communicating encrypted financial information from said second secure node to said third secure node, including communicating secure control information, and

means at the third node for processing said financial information at said third node at least in part through the use of secure control information supplied by said first and said second secure nodes, and

a secure database at the third node for at least in part storing said standardized control information.

243. A method of information management characterized by the steps of creating at least one smart object at a first location, protecting at least a portion of said smart object including protecting at least one rule and/or control assigned to said smart object, distributing said at least one smart object to at least one second location, securely processing at least a portion of the contents of said at least one smart object at said at least one second location in accordance with at least a portion of at least one said rule and/or control assigned to said smart object.

244. An information management system characterized by:
means for creating at least one smart object at a first location,

means for protecting at least a portion of said smart object including means for protecting at least one rule and/or control assigned to said smart object,

means for distributing said at least one smart object to at least one second location, and

means for securely processing at least a portion of the contents of said at least one smart object at said at least one second location in accordance with at least a portion of at least one said rule and/or control assigned to said smart object.

245. An object processing system comprising at least one secure object containing at least in part protected executable content and at least one at least in part protected rule and/or control associated with operations related to the execution of such content, and at least one secure execution environment for processing the executable content in accordance with at least a portion of at least one of said at least one associated rule and/or control.

246. An object processing method comprising:

providing at least one secure object containing at least in part protected executable content and at least one at least in part protected rule and/or control associated with operations related to the execution of such content,

processing, within at least one secure execution environment, the executable content in accordance with at least a portion of at least one of said at least one associated rule and/or control.

247. A rights distributed database environment including (a) means allowing one or more central authorities to establish control information for use of encrypted digital information, (b) interoperable database management systems at plural user sites for securely storing control information and audit information, (c) secure communication means for securely communicating control information and audit information between user sites, and (d) centralized database means for compiling and analyzing usage information from plural user sites.

248. Within a rights distributed database environment, a method characterized by the following steps:

establishing control information for use of encrypted digital information,

securely storing, within interoperable database management systems at plural user sites, control information and audit information,

securely communicating control information and audit information between user sites, and

compiling and analyzing usage information from plural user sites.

249. A method of distributed database searching characterized by the steps of creating at least one secure object containing search criteria, transmitting at least one such secure object to one or more second locations to perform database searches in accordance with at least one rule and/or control, processing at least one database search based at least in part on the search criteria within a secure object in accordance with at least a portion of at least one of the said at least one associated rule and/or control, storing database search results in the same and/or one or more new secure objects, and transmitting the secure object containing search results to the first location.

250. A method as in claim 247 further characterized by the additional step of associating at least one additional rule and/or control with the search results for establishing at least one condition related to the use of at least one portion of said search results.

251. A system for distributed database searching characterized by:

means for creating at least one secure object containing search criteria,

means for transmitting at least one such secure object to one or more second locations to perform database searches in accordance with at least one rule and/or control,

means for processing at least one database search based at least in part on the search criteria within a secure object in accordance with at least a portion of at least one of the said at least one associated rule and/or control,

means for storing database search results in the same and/or one or more new secure objects, and

means for transmitting the secure object containing search results to the first location.

252. A system as in claim 249 further characterized by means for associating at least one additional rule and/or control with the search results for establishing at least one condition related to the use of at least one portion of said search results.

253. A rights management system comprising protected information, at least two protected processing arrangements, and a rights management language that allows the expression of permitted operations and the consequences of performing such operations on at least a portion of the information processed at least in part by at least one of the protected processing arrangements.

254. A rights management method comprising:
providing protected information for processing by at least two protected processing arrangements, and
expressing, in a rights management language, permitted operations and the consequences of performing such operations on at least a portion of the information processed at least in part by at least one of the protected processing arrangements.

255. A method of protecting digital information characterized by the steps of encrypting at least a portion of the information, using a rights management language to describe the conditions related to use of the information, distributing at least a portion of such information and at least a portion of such rights language expressed conditions to one or more recipients, using an electronic appliance arrangement including at least one protected processing arrangement to securely govern at least a portion of the use of such information.

256. A system for protecting digital information characterized by:
means for encrypting at least a portion of the information,
means for using a rights management language to describe the conditions related to use of the information,

means for distributing at least a portion of such information and at least a portion of such rights language expressed conditions to one or more recipients, and

an electronic appliance arrangement including at least one protected processing arrangement for securely governing at least a portion of the use of such information.

257. A distributed digital information management system comprising software components, a rights management language for expressing processing relationships between two or more of the software components, protected processing means for at least a portion of the software components and at least a portion of the rights management expressions, means for protecting content, means for creating software objects that relate protected content to rights management expressions, and means for delivering protected content, rights management expressions, and such software objects from a providing location to a user's location.

258. A distributed digital information management method comprising:

expressing, in a rights management language, processing relationships between two or more of the software components, processing, within at least one protected environment, at least a portion of the software components and at least a portion of the rights management expressions,

protecting content,
creating software objects that relate protected content to
rights management expressions, and
delivering protected content, rights management
expressions, and such software objects from a providing location
to a user's location.

259. An authentication system comprising at least two
electronic appliances, at least two digital certificates reflecting
identity information encrypted using different certifying private
keys where such certificates are stored in a first electronic
appliance, communications means for transmitting and receiving
signals between electronic appliances, means for determining
compromised and/or expired certifying private keys operatively
connected to a second electronic appliance, means for the second
electronic appliance to request transmission of one of the digital
certificates from the first electronic appliance based at least in
part on such determination, and means operatively connected to
such second electronic appliance for decrypting such certificate
and determining such certificate's validity and/or the validity of
identity information.

260. In a system comprising at least two electronic
appliances, an authenticating method comprising:

issuing at least two digital certificates reflecting identification information, including the step of encrypting the two certificates using different certifying private keys, storing the certificates in a first electronic appliance, transmitting and receiving signals between electronic appliances, determining compromised and/or expired certifying private keys operatively connected to a second electronic appliance, requesting, with the second electronic appliance, transmission of one of the digital certificates from the first electronic appliance based at least in part on such determination, decrypting such certificate with the second electronic appliance, and determining such certificate's validity and/or the validity of identity information.

261. An authentication system comprising at least two electronic appliances, at least two digital certificates reflecting identify information encrypted using different certifying private keys where such certificates are stored in a first electronic appliance, communications means for transmitting and receiving signals between electronic appliances, means for a second electronic appliance to request transmission of one of the digital certificates from the first electronic appliance wherein the selection of which certificate is requested is based at least in part

on a random or pseudo-random number, means operatively connected to such second electronic appliance for decrypting such certificate and determining such certificate's validity and/or the validity of identity information.

262. In a system comprising at least two electronic appliances, an authenticating method comprising:

issuing at least two digital certificates reflecting identify information, including the step of encrypting the two digital certificates using different certifying private keys,

storing such certificates in a first electronic appliance, transmitting and receiving signals between electronic appliances,

requesting, with a second electronic appliance, transmission of one of the digital certificates from the first electronic appliance, including the step of selecting a certificate based at least in part on a random or pseudo-random number,

decrypting such certificate with the second electronic appliance; and

determining such certificate's validity and/or the validity of identity information.

263. A method of secure electronic mail characterized by the steps of creating at least one electronic message using an interoperable protected processing environment, encrypting at

least a portion of said at least one message, securely associating one or more sets of control information with one or more messages to set at least one condition for the use of said at least one message, communicating the protected electronic messages to one or more recipients having protected processing environments, securely communicating at least one set of the same or differing control information to each recipient, enabling recipients of both control information and protected messages to use message information at least in part in accordance with the conditions specified by the control information.

264. A system for secure electronic mail including multiple protected processing environments, the system characterized by:

- a first protected processing environment for creating at least one electronic message, the first environment including means for encrypting at least a portion of said at least one message, means for securely associating one or more sets of control information with one or more messages to set at least one condition for the use of said at least one message, and means for communicating the protected electronic messages to one or more recipients having interoperable protected processing environments,
- means for securely communicating at least one set of the same or differing control information to each recipient, and

means for enabling recipients of both control information and protected messages to use message information at least in part in accordance with the conditions specified by the control information.

265. A method of information management characterized by the steps of protecting content from unauthorized use, securely associating enabling control information with at least a portion of such protected content wherein such enabling control information incorporates information describing how the enabling control information may be redistributed, delivering at least a portion of the protected content to a first user, delivering such enabling control information to such first user, receiving a request to redistribute such enabling control information from such first user, using the description of how enabling control information may be redistributed to create new enabling control information where such new enabling control information may be the same or different than the enabling control information received by such first user, delivering the new enabling control information and/or protected information to a second user.

266. An information management system characterized by:

means for protecting content from unauthorized use,

means for securely associating enabling control information with at least a portion of such protected content, including means for incorporating enabling control information describing how the enabling control information may be redistributed,

means for delivering at least a portion of the protected content to a first user,

means for delivering such enabling control information to such first user,

means for receiving a request to redistribute such enabling control information from such first user,

means for using the description of how enabling control information may be redistributed to create new enabling control information where such new enabling control information may be the same or different than the enabling control information received by such first user, and

means for delivering the new enabling control information and/or protected information to a second user.

267. A method of controlling redistribution of distributed digital information including the steps of encrypting digital information, distributing said encrypted digital information from a first party to a second party, establishing control information regarding the redistribution of at least a portion of said encrypted digital information from said second party to at least one third

party, regulating the redistribution of said at least a portion of said encrypted digital information through the use of a protected processing environment processing said control information.

268. A system for controlling redistribution of distributed digital information including:

means for encrypting digital information,

means for distributing said encrypted digital information from a first party to at least one second party,

means for establishing control information regarding the redistribution of at least a portion of said encrypted digital information from said second party to at least one third party,
and

a protected processing environment for processing said control information and for regulating the redistribution of said at least a portion of said encrypted digital information.

269. A method of controlling a robot characterized by the steps of creating instructions for one or more robots, creating a secure container incorporating such instructions, associating control information with such secure container, incorporating at least one secure processing unit into such one or more robots, and performing at least a portion of such instructions in accordance with at least a portion of such control information.

270. A method as in claim 267 further characterized in that such control information includes information describing the conditions under which such instructions may be used and the nature of audit reports required when such instructions are performed.

271. A robot control system characterized by:
means for creating instructions for one or more robots,
means for creating a secure container incorporating such instructions,
means for associating control information with such secure container,
means for incorporating at least one secure processing unit into such one or more robots, and
means for performing at least a portion of such instructions in accordance with at least a portion of such control information.

272. A system as in claim 269 further characterized by means for creating such control information, including means for describing the conditions under which such instructions may be used and the nature of audit reports required when such instructions are performed.

273. A method of detecting fraud in electronic commerce characterized by the steps of creating at least one secure

container, associating control information with such one or more containers including control information requiring that audit information be collected and transmitted to an auditing party, delivering such one or more containers and such control information to at least one user, recording information identifying each container and each such user, receiving audit information, creating a profile of usage based at least in part on such received audit information and/or such control information, detecting cases where certain audit information differs at least in part from such profile of usage.

274. A system for detecting fraud in electronic commerce characterized by
- means for creating at least one secure container,
 - means for associating control information with such one or more containers including control information requiring that audit information be collected and transmitted to an auditing party,
 - means for delivering such one or more containers and such control information to at least one user,
 - means for recording information identifying each container and each such user,
 - means for receiving audit information,

means for creating a profile of usage based at least in part on such received audit information and/or such control information, and

means for detecting cases where certain audit information differs at least in part from such profile of usage.

275. A method of detecting fraud in electronic commerce characterized by the steps of distributing at least in part protected digital information to customers, distributing one or more rights to use at least a portion of such digital information across an electronic network, allowing a customer to use at least a part of said at least in part protected digital information through the use of a protected processing environment and at least one of said one or more distributed rights, detecting unusual usage activity related to use of said digital information.

276. A system for detecting fraud in electronic commerce characterized by

means for distributing at least in part protected digital information to customers,

means for distributing one or more rights to use at least a portion of such digital information across an electronic network,

a protected processing environment for allowing a customer to use at least a part of said at least in part protected

digital information through at least one of said one or more distributed rights, and

means for detecting unusual usage activity related to use of said digital information.

277. A programmable component arrangement comprising a tamper resistant processing environment including a microprocessor, memory, a task manager, memory manager and external interface controller, means for loading arbitrary components at least in part into the memory, means for initiating one or more tasks associated with processing such components, means for certifying the validity, integrity and/or trustedness of such components, means for creating arbitrary components, means for associating arbitrary events with such created components, means for certifying the validity, integrity and/or trustedness of such created components, and means for securely delivering such created components.

278. In a programmable component arrangement comprising a tamper resistant processing environment including a microprocessor, memory, a task manager, memory manager and an external interface controller, a processing method characterized by the following steps:

creating arbitrary components,

associating arbitrary events with such created components,

loading the arbitrary components at least in part into the memory,
initiating one or more tasks associated with processing such loaded components,
certifying the validity, integrity and/or trustedness of such created components, and
securely delivering such created components.

279. A distributed, protected, programmable component arrangement comprising at least two tamper resistant processing environments including a microprocessor, memory, a task manager, memory manager and external interface controller, means for loading arbitrary components at least in part into the memory, means for initiating one or more tasks associated with processing such components, and means for certifying the validity, integrity and/or trustedness of such components, said arrangement further comprising means for creating arbitrary components, means for associating arbitrary events with such created components, means for certifying the validity, integrity and/or trustedness of such created components, means for securely delivering such created components between at least two of said at least two tamper resistant processing environments.

280. In a distributed, protected, programmable component arrangement comprising at least two tamper resistant processing

environments including a microprocessor, memory, a task manager, memory manager and external interface controller, a method comprising

- creating arbitrary components,
- certifying the validity, integrity and/or trustedness of such components,
- loading arbitrary components at least in part into the memory,
- initiating one or more tasks associated with processing such components,
- associating arbitrary events with such created components,

and

- securely delivering such created components between at least two of said at least two tamper resistant processing environments.

281. An electronic appliance comprising at least one CPU, memory, at least one system bus, at least one protected processing environment, and at least one of a Rights Operating System or Rights Operating System layer associated with a host operating system.

282. An operating system comprising at least one task manager, at least one memory manager, at least one input/output manager, at least one protected processing environment, means

for detecting events, means for associating events with rights control functions, means for performing rights control functions at least in part within such one or more protected processing environments.

283. In an operating system comprising at least one task manager, at least one memory manager, at least one input/output manager, at least one protected processing environment, an operating method comprising:

detecting events,
associating events with rights control functions, and
performing rights control functions at least in part within such one or more protected processing environments.

284. A method of business automation characterized by the steps of creating one or more secure containers including accounting and/or other administrative information, associating control information with such one or more secure containers including a description of (a) the one or more parties to whom the container may and/or must be delivered and/or (b) the operations that one or more parties may and/or must perform with respect to such accounting and/or other administrative information, delivering one or more of such containers to one or more parties, and enabling the description and/or enforcement of at least a portion of such control information prior, during and/or

subsequent to use of such accounting and/or other administrative information by one or more parties.

285. A method as in claim 282 where such control information further includes at least one requirement that audit information be collected and delivered to one or more auditing parties, and further includes the step of delivering at least a portion of such audit information to one or more parties.

286. A method as in claim 283 where at least a portion of such audit information is automatically processed by at least one of such auditing parties, and further includes the step of transmitting further accounting, administrative and/or audit information to one or more parties that may be the same and/or differ from the one or more parties from whom audit information was received based at least in part on the receipt and/or content of such received audit information.

287. A method as in claim 282 where at least two of such parties are associated with different businesses and/or other organizations and such control information includes information that at least in part describes an accounting, administrative, reporting and/or other audit relationship between such businesses and/or other organizations.

288. A method as in claim 282, 283, 284, or 285 where some or all of such accounting and/or other administrative information is included in such control information.

289. A business automation system characterized by:

- means for creating one or more secure containers including accounting and/or other administrative information,
- means for associating, with such one or more secure containers, control information including a description of (a) the one or more parties to whom the container may and/or must be delivered and/or (b) the operations that one or more parties may and/or must perform with respect to such accounting and/or other administrative information,
- means for delivering one or more of such containers to one or more parties, and
- means for enabling the description and/or enforcement of at least a portion of such control information prior, during and/or subsequent to use of such accounting and/or other administrative information by one or more parties.

290. A system as in claim 287 where the associating means further includes means for associating at least one requirement that audit information be collected and delivered to one or more auditing parties, and the delivering means includes

means for delivering at least a portion of such audit information to one or more parties.

291. A system as in claim 288 further including means for automatically processing at least a portion of such audit information, and the system further includes means for transmitting further accounting, administrative and/or audit information to one or more parties that may be the same and/or differ from the one or more parties from whom audit information was received based at least in part on the receipt and/or content of such received audit information.

292. A system as in claim 287 where at least two of such parties are associated with different businesses and/or other organizations and the associating means includes means for generating control information including information that at least in part describes an accounting, administrative, reporting and/or other audit relationship between such businesses and/or other organizations.

293. A system as in claim 286, 287, 288, or 290 where some or all of such accounting and/or other administrative information is included in such control information.

294. A method of distributing content characterized by the steps of creating one or more first secure containers, associating control information with such first containers including information describing the conditions under which some or all of the content of such first containers may be extracted, delivering at least a portion of such first containers and such control information to one or more parties, detecting a request by one or more of such parties to extract some or all of the content of such first containers, determining if such request is permitted in whole or in part by such control information, to the extent permitted by such control information creating one or more second secure containers in accordance with such request and such control information, associating control information with such one or more second secure containers based at least in part on control information associated with such first containers.

295. A system for distributing content characterized by:
means for creating one or more first secure containers,
means for associating control information with such first containers including information describing the conditions under which some or all of the content of such first containers may be extracted,
means for delivering at least a portion of such first containers and such control information to one or more parties,

means for detecting a request by one or more of such parties to extract some or all of the content of such first containers,

means for determining if such request is permitted in whole or in part by such control information, to the extent permitted by such control information creating one or more second secure containers in accordance with such request and such control information, and

means for associating control information with such one or more second secure containers based at least in part on control information associated with such first containers.

296. A method of distributing content characterized by the steps of creating one or more first secure containers, associating control information with such first secure containers including information describing the conditions under which such first secure containers (a) may in whole or in part be embedded into and/or securely associated with one or more second secure containers and/or (b) may allow one or more secure containers to be in whole or in part embedded into and/or securely associated with such first secure containers, delivering at least a portion of such first secure containers and such control information to one or more parties, detecting a request by one or more of such parties or by additional parties to (a) in whole or in part embed into and/or securely associate with such first containers one or

more second containers and/or (b) in whole or in part embed into and/or securely associate with a secure container such first secure containers, determining if such request is permitted by control information, to the extent permitted by control information performing one or more embedding and/or secure association operations, to the extent required by control information and/or requested by one or more of such parties, modifying and/or creating new control information at least in part as a consequence of such one or more embedding and/or secure association operations.

297. A system for distributing content characterized by means for creating one or more first secure containers, means for associating control information with such first secure containers including information describing the conditions under which such first secure containers (a) may in whole or in part be embedded into and/or securely associated with one or more second secure containers and/or (b) may allow one or more secure containers to be in whole or in part embedded into and/or securely associated with such first secure containers, means for delivering at least a portion of such first secure containers and such control information to one or more parties, means for detecting a request by one or more of such parties to (a) in whole or in part embed into and/or securely associate with such first containers one or more second

containers and/or (b) in whole or in part embed into and/or securely associate with a secure container such first secure containers, and

means for determining if such request is permitted by control information, to the extent permitted by control information performing one or more embedding and/or secure association operations, to the extent required by control information and/or requested by one or more of such parties, modifying and/or creating new control information at least in part as a consequence of such one or more embedding and/or secure association operations.

298. A method of distributing information characterized by the steps of protecting information from unauthorized use, associating control information with such protected information, delivering at least a portion of such protected information to one or more parties using plural pathways, delivering at least a portion of such control information to one or more parties using the same or different plural pathways, enabling at least one of such parties to make at least some use of such protected information delivered using a first pathway in accordance with control information at least a portion of which is delivered using a second pathway.

299. A method as in claim 296 in which at least one of such pathways of delivering protected information and/or control information is described by such control information.

300. A system for distributing information characterized by:

means for protecting information from unauthorized use,

means for associating control information with such protected information,

means for delivering at least a portion of such protected information to one or more parties using plural pathways,

means for delivering at least a portion of such control information to one or more parties using the same or different plural pathways,

means for enabling at least one of such parties to make at least some use of such protected information delivered using a first pathway in accordance with control information at least a portion of which is delivered using a second pathway.

301. A system as in claim 298 wherein the delivering means includes means for delivering, over at least one of such pathways, protected information and/or control information described by such control information.

302. A method of distributing information characterized by the steps of protecting information from unauthorized use, associating control information with such protected information including information requiring the collection of audit information, enabling one or more parties to receive and/or process audit information, delivering at least a portion of such protected information and such control information to one or more parties, enabling at least some use of such protected information in accordance with at least a portion of such control information that requires the collection of audit information, delivering such audit information to one or more of such enabled auditing parties different from such delivering party or parties.

303. A method as in claim 300 in which at least one of such auditing parties is specified in such control information.

304. A system for distributing information characterized by
means for protecting information from unauthorized use,
means for associating control information with such protected information including information requiring the collection of audit information,
means for enabling one or more parties to receive and/or process audit information,

means for delivering at least a portion of such protected information and such control information to one or more parties, means for enabling at least some use of such protected information in accordance with at least a portion of such control information that requires the collection of audit information, and means for delivering such audit information to one or more of such enabled auditing parties different from such delivering party or parties.

305. A system as in claim 302 in which at least one of such auditing parties is specified in such control information.

306. A secure component-based operating process including:

- (a) retrieving at least one component;
- (b) retrieving a record that specifies a component assembly;
- (c) checking said component and/or said record for validity;
- (d) using said component to form said component assembly in accordance with said record; and
- (e) performing a process based at least in part on said component assembly.

307. A process as in claim 304 wherein said step (c) further comprises executing said component assembly.

308. A process as in claim 304 wherein said component comprises executable code.

309. A process as in claim 304 wherein said component comprises a load module.

310. A process as in claim 304 wherein:

said record comprises:

(i) directions for assembling said component assembly;

and

(ii) information that at least in part specifies a control;

and

said process further comprises controlling said step (d) and/or said step (e) based at least in part on said control.

311. A process as in claim 304 wherein said component has a security wrapper, and said controlling step comprises selectively opening said security wrapper based at least in part on said control.

312. A process as in claim 304 wherein:

said permissions record includes at least one decryption key; and

said controlling step includes controlling use of said decryption key.

313. A process as in claim 304 including performing at least two of said steps (a) and (e) within a protected processing environment.

314. A process as in claim 304 including performing at least two of said steps (a) and (e) at least in part within tamper-resistant hardware.

315. A method as in claim 304 wherein said performing step (e) includes metering usage.

316. A method as in claim 304 wherein said performing step (e) includes auditing usage.

317. A method as in claim 304 wherein said performing step (e) includes budgeting usage.

318. A secure component operating system process including:

receiving a component;

receiving directions specifying use of said component to form a component assembly;

authenticating said received component and/or said directions;

forming, using said component, said component assembly based at least in part on said received directions; and using said component assembly to perform at least one operation.

319. A method comprising performing the following steps within a secure operating system environment:

providing code;

providing directions specifying assembly of said code into an executable program;

checking said received code and/or said assembly directors for validity; and

in response to occurrence of an event, assembling said code in accordance with said received assembly directions to form an assembly for execution.

320. A method for managing at least one resource with a secure operating environment, said method comprising:

securely receiving a first control from a first entity external to said operating environment;

securely receiving a second control from a second entity external to said operating environment, said second entity being different from said first entity;

securely processing, using at least one resource, a data item associated with said first and second controls; and

securely applying said first and second controls to manage said resource for use with said data item.

321. A method for securely managing at least one operation on a data item performed at least in part by an electronic arrangement, said method comprising:

(a) securely delivering a first procedure to said electronic arrangement;

(b) securely delivering, to said electronic arrangement, a second procedure separable or separate from said first procedure;

(c) performing at least one operation on said data item, including using said first and second procedures in combination to at least in part securely manage said operation; and

(d) securely conditioning at least one aspect of use of said data item based on said delivering steps (a) and (b) having occurred.

322. A method as in claim 319 including performing said delivering step (b) at a time different from the time said delivering step (a) is performed.

323. A method as in claim 319 wherein said step (a) includes delivering said first procedure from a first source, and said step (b) includes delivering said second procedure from a second source different from said first source.

324. A method as in claim 319 further including ensuring the integrity of said first and second procedures.

325. A method as in claim 319 further including validating each of said first and second procedures.

326. A method as in claim 319 further including authenticating each of said first and second procedures.

327. A method as in claim 319 wherein said using step (c) includes executing at least one of said first and second procedures within a tamper-resistant environment.

328. A method as in claim 319 wherein said step (c) includes the step of controlling said data item with at least one of said first and second procedures.

329. A method as in claim 319 further including establishing a relationship between at least one of said first and second procedures and said data item.

330. A method as in claim 319 further including establishing correspondence between said data item and at least one of said first and second procedures.

331. A method as in claim 319 wherein said delivering step (b) comprises delivering at least one load module encrypted at least in part.

332. A method as in claim 329 wherein said delivering step (a) comprises delivering at least one further load module encrypted at least in part.

333. A method as in claim 319 wherein said delivering step (b) comprises delivering at least one content container carrying at least in part secure control information.

334. A method as in claim 319 wherein said delivering step (b) comprises delivering a control method and at least one further method.

335. A method as in claim 319 wherein said delivering step (a) includes:

- encrypting at least a portion of said first procedure,
- communicating said at least in part encrypted first procedure to said electronic arrangement,
- decrypting at least a portion of said first procedure at least in part using said electronic arrangement, and
- validating said first procedure with said electronic arrangement.

336. A method as in claim 319 wherein said delivering step (b) includes delivering at least one of said first and second procedures within an administrative object.

337. A method as in claim 319 wherein said delivering step (b) includes codelivering said second procedure in at least in part encrypted form with said data item.

338. A method as in claim 319 wherein said performing step includes metering usage.

339. A method as in claim 319 wherein said performing step includes auditing usage.

340. A method as in claim 319 wherein said performing step includes budgeting usage.

341. A method for securely managing at least one operation performed at least in part by a secure electronic appliance, comprising:

(a) selecting an item that is protected with respect to at least one operation;

(b) securely independently delivering plural separate procedures to said electronic appliance;

(c) using said plural separate procedures in combination to at least in part securely manage said operation with respect to said selected item; and

(d) conditioning successful completion of said operation on said delivering step (b) having occurred.

342. A method for processing based on deliverables comprising:

securely delivering a first piece of code defining a first part of a process;

separately, securely delivering a second piece of code defining a second part of said process;

ensuring the integrity of the first and second delivered pieces of code; and

performing said process based at least in part on said first and second delivered code pieces.

343. A method as in claim 340 wherein a first piece of code for said process at least in part controls decrypting content.

344. A method as in claim 340 wherein said ensuring step includes validating said first and second pieces of code.

345. A method as in claim 340 wherein said ensuring step includes validating said first and second pieces of code relative to one another.

346. A method as in claim 340 wherein said performing step includes metering usage.

347. A method as in claim 340 wherein said performing step includes auditing activities.

348. A method as in claim 340 wherein said performing step includes budgeting usage.

349. A method as in claim 340 wherein said performing step includes electronically processing content based on electronic controls.

350. A method of securely controlling at least one protected operation with respect to a data item comprising:

- (a) supplying at least a first control from a first party;
- (b) supplying at least a second control from a second party different from said first party;
- (c) securely combining said first and second controls to form a set of controls;

(d) securely associating said control set with said data item; and

(e) securely controlling at least one protected operation with respect to said data item based on said control set.

351. A method as in claim 348 wherein said data item is protected.

352. A method as in claim 348 wherein at least one of said plural controls includes a control relating to metering at least one aspect of use of said protected data item.

353. A method as in claim 348 wherein at least one of said plural controls include a control relating to budgeting at least one aspect of use of said protected data item.

354. A secure method for combining data items into a composite data item comprising:

(a) securely providing a first data item having at least a first control associated therewith;

(b) securely providing a second data item having at least a second control associated therewith;

(c) forming a composite of said first and second data items;

(d) securely combining said first and second controls into a composite control set; and

(e) performing at least one operation on said composite of said first and second data items based at least in part on said composite control set.

355. A method as in claim 352 wherein said combining step includes preserving each of said first and second controls in said composite set.

356. A method as in claim 352 wherein said performing step comprises governing the operation on said composite of said first and second data items in accordance with said first control and said second control .

357. A method as in claim 352 wherein said providing step includes ensuring the integrity of said association between said first controls and said first data item is maintained during at least one of transmission, storage and processing of said first data item.

358. A method as in claim 352 wherein said providing step comprises delivering said first data item separately from said first control .

359. A method as in claim 352 wherein said providing step comprises codelivering said first data item and said first control .

360. A secure method for controlling a protected operation comprising:

(a) delivering at least a first control and a second control;
and

(b) controlling at least one protected operation based at least in part on a combination of said first and second controls, including at least one of the following steps:

resolving at least one conflict between said first and second controls based on a predefined order;

providing an interaction with a user to form said combination; and

dynamically negotiating between said first and second controls.

361. A method as in claim 358 wherein said controlling step (b) includes controlling decryption of electronic content.

362. A method as in claim 358 further including:

receiving protected electronic content from a party; and

authenticating the identity of said party prior to using said received protected electronic content.

363. A secure method comprising:
selecting protected data;
extracting said protected data from an object;
identifying at least one control to manage at least one aspect of use of said extracted data;
placing said extracted data into a further object; and
associating said at least one control with said further object.

364. A method as in claim 361 further including limiting at least one aspect of use of said further object based on said at least one control.

365. A secure method of modifying a protected object comprising:
(a) providing a protected object; and
(b) embedding at least one additional element into said protected object without unprotecting said object.

366. A method as in claim 60 further including:
associating at least one control with said object; and
limiting usage of said element in accordance with said control.

367. A method as in claim 363 further including a permissions record within said object.

368. A method as in claim 364 further including at least in part encrypting said object.

369. A method for managing at least one resource with a secure operating environment, said method comprising:

securely receiving a first load module from a first entity external to said operating environment;

securely receiving a second load module from a second entity external to said operating environment, said second entity being different from said first entity;

securely processing, using at least one resource, a data item associated with said first and second load modules; and

securely applying said first and second load modules to manage said resource for use with said data item.

370. A method for negotiating electronic contracts, comprising:

receiving a first control set from a remote site;

providing a second control set;

performing, within a protected processing environment, an electronic negotiation between said first control set and said

second control set, including providing interaction between said first and second control sets; and

producing a negotiated control set resulting from said interaction between said first and second control sets.

371. A system for supporting electronic commerce including:

means for creating a first secure control set at a first location;

means for creating a second secure control set at a second location;

means for securely communicating said first secure control set from said first location to said second location; and

means at said second location for securely integrating said first and second control sets to produce at least a third control set comprising plural elements together comprising an electronic value chain extended agreement.

372. A system for supporting electronic commerce including:

means for creating a first secure control set at a first location;

means for creating a second secure control set at a second location;

means for securely communicating said first secure control set from said first location to said second location; and

negotiation means at said second location for negotiating an electronic contract through secure execution of at least a portion of said first and second secure control sets.

373. A system as in claim 370 further including means for controlling use by a user of protected information content based on at least a portion of said first and/or second control sets.

374. A system as in claim 370 further including means for charging for at least a part of said content use.

375. A secure component-based operating system including:

component retrieving means for retrieving at least one component;

record retrieving means for retrieving a record that specifies a component assembly;

checking means, operatively coupled to said component retrieving means and said record retrieving means, for checking said component and/or said record for validity;

using means, coupled to said checking means, for using said component to form said component assembly in accordance with said record; and

performing means, coupled to said using means, for performing a process based at least in part on said component assembly.

376. A secure component-based operating system including:

a database manager that retrieves, from a secure database, at least one component and at least one record that specifies a component assembly;

an authenticating manager that checks said component and/or said record for validity;

a channel manager that uses said component to form said component assembly in accordance with said record; and

an execution manager that performs a process based at least in part on said component assembly.

377. A secure component operating system including:

means for receiving a component;

means for receiving directions specifying use of said component to form a component assembly;

means, coupled to said receiving means, for authenticating said received component and/or said directions;

means, coupled to said authenticating means, for forming, using said component, said component assembly based at least in part on said received directions; and

means, coupled to said forming means, for using said component assembly to perform at least one operation.

378. A secure component operating environment including:

a storage device that stores a component and directions specifying use of said component to form a component assembly;

an authenticating manager that authenticates said component and/or said directions;

a channel manager that forms, using said component, said component assembly based at least in part on said directions; and

a channel that executes said component assembly to perform at least one operation.

379. A secure operating system environment comprising:

a storage device that stores code and directions specifying assembly of said code into an executable program;

a validating device that checks said received code and/or said assembly directors for validity; and

an event-driven channel that, in response to occurrence of an event, assembles said code in accordance with said assembly directions to form an assembly for execution.

380. A secure operating environment system for managing at least one resource comprising:

a communications arrangement that securely receives a first control from a first entity external to said operating environment, and securely receives a second control from a second entity external to said operating environment, said second entity being different from said first entity; and

a protected processing environment, coupled to said communications arrangement, that:

(a) securely processes, using at least one resource, a data item associated with said first and second controls, and

(b) securely applies said first and second controls to manage said resource for use of said data item.

381. A system for negotiating electronic contracts, comprising:

a storage arrangement that stores a first control set received from a remote site, and stores a second control set;

a protected processing environment, coupled to said storage arrangement, that:

(a) performs an electronic negotiation between said first control set and said second control set,

(b) provides interaction between said first and second control sets, and

(c) produces a negotiated control set resulting from said interaction between said first and second control sets.

382. A system as in claim 379 further including means for electronically enforcing said negotiated control set.

383. A system as in claim 379 further including means for generating an electronic contract based on said negotiated control set.

384. A method for supporting electronic commerce including:
creating a first secure control set at a first location;
creating a second secure control set;
electronically negotiating, at said location different from said first location, an electronic contract, including the step of securely executing at least a portion of said first and second control sets.

385. An electronic appliance comprising:
a processor; and
at least one memory device connected to said processor;
wherein said processor includes:
retrieving means for retrieving at least one component, and at least one record that specifies a component assembly, from said memory device,
checking means coupled to said retrieving means for checking said component and/or said record for validity, and

using means coupled to said retrieving means for using said component to form said component assembly in accordance with said record.

386. An electronic appliance comprising:
at least one processor;
at least one memory device connected to said processor;
and
at least one input/output connection operatively coupled to said processor,
wherein said processor at least in part executes a rights operating system to provide a secure operating environment within said electronic appliance.

387. An electronic appliance as in claim 384 wherein said processor includes means for providing a channel, said channel assembling independently deliverable components into a component assembly and executing said component assembly.

388. An electronic appliance as in claim 384 further including a secondary storage device coupled to said processor, said secondary storage device storing a secure database, said processor including means for decrypting information obtained from said secure database and for encrypting information to be written to said secure database.

389. An electronic appliance as in claim 384 wherein said processor and said memory device are disposed in a secure, tamper-resistance encapsulation.

390. An electronic appliance as in claim 384 wherein said processor includes a hardware encryptor/decryptor.

391. An electronic appliance as in claim 384 wherein said processor includes a real time clock.

392. An electronic appliance as in claim 384 wherein said processor includes a random number generator.

393. An electronic appliance as in claim 384 wherein said memory device stores audit information.

394. A method for auditing the use of at least one resource with a secure operating environment, said method comprising:
securely receiving a first control from a first entity external to said operating environment;
securely receiving a second control from a second entity external to said operating environment, said second entity being different from said first entity;
using at least one resource;

securely sending to said first entity in accordance with said first control, first audit information concerning use of said resource; and

securely sending to said second entity in accordance with said second control, second audit information concerning use of said resource, said second audit information being at least in part different from said first audit information.

395. A method for auditing the use of at least one resource with a secure operating environment, said method comprising:

securely receiving first and second control alternatives from an entity external to said operating environment;

selecting one of said first and second control alternatives;

using at least one resource;

if said first control alternative is selected by said selecting step, securely sending to said entity in accordance with said first control alternative, first audit information concerning use of said resource; and

if said second control alternative is selected by said selecting step, securely sending to said second entity in accordance with said second control alternative, second audit information concerning use of said resource, said second audit information being at least in part different from said first audit information.

396. A method and/or system for enabling a sale of protected digital information that has been previously distributed to users, the method or system being characterized by a secure element that selectively controls access to the protected digital information based on electronic controls associated with the information.

397. A distributed, secure electronic point of sale system or method characterized by a secure processing element for selectively releasing goods and/or services in exchange for compensation.

398. In a distributed digital network, an advertising method characterized by the steps of tracking usage of digital information that has associated with it one or more controls with respect to access to and/or usage of said information; and targeting advertising messages based at least in part on said tracking.

399. A distributed electronic advertising system characterized in that the system uses a distributed network of interoperable protected processing environments to at least in part deliver advertising to users.

400. A distributed, secure, virtual black box comprised of nodes located at VDE content container creators, other content providers, client users, and recipients of secure VDE content usage information) site, the nodes of said virtual black box including a secure subsystem having at least one secure hardware element such as a semiconductor element or other hardware module for securely executing VDE control processes, said secure subsystems being distributed at nodes along a pathway of information storage, distribution, payment, usage, and/or auditing.

401. A protected processing system or method providing multiple currencies and/or payment arrangements for the secure processing and releasing of protected digital information.

402. A distributed secure method or system characterized in that a user's age is used as a criteria for electronically, securely releasing information and/or resources to the user.

403. A method of renting an electronic appliance defining a secure processing environment.

404. A virtual distribution environment providing any one or more of the following features and/or elements and/or combinations thereof:

a configurable protected, distributed event management system; and/or

a trusted, distributed transaction and storage management arrangement; and/or

plural pathways for providing information, for control information, and/or for reporting; and/or

multiple payment methods; and/or

multiple currencies; and/or

EDI; and/or

Electronic banking; and/or

electronic document management; and/or

electronic secure communication; and/or

e-mail; and/or

distributed asynchronous reporting; and/or

combination asynchronous and online management; and/or

privacy control by users; and/or

testing; and/or

using age as a class; and/or

appliance control (renting, etc.); and/or

telecommunications infrastructure; and/or

games management; and/or

extraction of content from an electronic container; and/or

embedding of content into an electronic container; and/or

multiple certificate to allow for breach of a key; and/or

virtual black box; and/or

independence of control information from content; and/or
multiple, separate, simultaneous control sets for one digital
information property; and/or
updating control information for already distributed digital
information; and/or
organization information management; and/or
coupled external and organization internal chain of
handling and control; and/or
a content usage consequence management system
(reporting, payment, etc., multiple directions); and/or
a content usage reporting system providing differing audit
information and/or reduction going to multiple parties holding
rights in content; and/or
an automated remote secure object creation system; and/or
infrastructure background analysis to identify improper
use; and/or
seniority of control information system; and/or
secure distribution and enforcement of rules and controls
separately from the content they apply to; and/or
redistribution management by controlling the rights and/or
number of copies and or pieces etc. that may be redistributed;
and/or
an electronic commerce taxation system; and/or
an electronic shopping system; and/or
an electronic catalog system; and/or

a system handling electronic banking, electronic shopping,
and electronic content usage management; and/or
an electronic commerce multimedia system; and/or
a distributed, secure, electronic point of sale system; and/or
advertising; and/or
electronics rights management; and/or
a distributed electronic commerce system; and/or
a distributed transaction system or environment; and/or
a distributed event management system; and/or
a distributed right systems.

405. A Virtual Distribution Environment substantially as
shown in Figure 1.

406. An "Information Utility" substantially as shown in
Figure 1A.

407. A chain of handling and control substantially as
shown in Figure 1.

408. Persistent rules and control information substantially
as shown in Figure 2A.

409. A method of providing different control information
substantially as shown in Figure 1.

410. Rules and/or control information substantially as shown in Figure 4.
411. An object substantially as shown in Figures 5A and 5B.
412. A Secure Processing Unit substantially as shown in Figure 6.
413. An electronic appliance substantially as shown in Figure 7.
414. An electronic appliance substantially as shown in Figure 8.
415. A Secure Processing Unit substantially as shown in Figure 9.
416. A "Rights Operating System" ("ROS") architecture substantially as shown in Figure 10.
417. Functional relationship(s) between applications and the Rights Operating System substantially as shown in Figures 11A-11C.

418. Components and component assemblies substantially as shown in Figures 11D-11J.

419. A Rights Operating System substantially as shown in FIGURE 12.

420. A method of objection creation substantially as shown in Figure 12A.

421. A "protected processing environment" software architecture substantially as shown in Figure 13.

422. A method of supporting a channel substantially as shown in Figure 15.

423. A channel header and channel detail record substantially as shown in Figure 15 A.

424. A method of creating a channel substantially as shown in Figure 15B.

425. A secure data base substantially as shown in Figure 16.

426. A logical object substantially as shown in Figure 17.

427. A stationary object substantially as shown in
FIGURE 18.

428. A travelling object substantially as shown in FIGURE
19.

429. A content object substantially as shown in FIGURE
20.

430. An administrative object substantially as shown in
Figure 21.

431. A method core substantially as shown in Figure 22.

432. A load module substantially as shown in FIGURE
23.

433. A User Data Element (UDE) and/or Method Data
Element (MDE) substantially as shown in FIGURE 24.

434. Map meters substantially as shown in FIGURES
25A-25C.

435. A permissions record (PERC) substantially as shown
in FIGURE 26.

436. A permissions record (PERC) substantially as shown in FIGURES 26A and 26B.

437. A shipping table substantially as shown in FIGURE 27.

438. A receiving table substantially as shown in FIGURE 28.

439. An administrative event log substantially as shown in FIGURE 29.

440. A method of interrelating and using an object registration table, a subject table and a user rights table substantially as shown in Figure 30.

441. A method of using a site record table and a group record table to track portions of a secure database substantially as shown in FIGURE 34.

442. A process for updating a secure database substantially as shown in FIGURE 35.

443. A process of inserting new elements into a secure database substantially as shown in FIGURE 36.

444. A process of accessing elements in a secure database substantially as shown in FIGURE 37.

445. A process of protecting a secure database element substantially as shown in FIGURE 38.

446. A process of backing up a secure database substantially as shown in FIGURE 39.

447. A process of recovering a secure database substantially as shown in FIGURE 40.

448. A process of enabling performing reciprocal methods to provide a chain of handling and control substantially as shown in FIGURES 41A-41D.

449. A "reciprocal" BUDGET method substantially as shown in FIGURES 42A-42D.

450. A reciprocal audit method substantially as shown in FIGURES 44A-44C.

451. A method for controlling release of content or other method substantially as shown in any of FIGURES 45-48.

452. An event method substantially as shown in
FIGURES 53A-53B.

453. A billing method substantially as shown in FIGURE
53C.

454. An extract method substantially as shown in
FIGURE 57A.

455. An embed method substantially as shown in FIGURE
57A.

456. An obscure method substantially as shown in
FIGURE 58A.

457. A fingerprint method substantially as shown in
FIGURE 58B.

458. A fingerprint method substantially as shown in
FIGURE 58C.

459. A meter method substantially as shown in FIGURE
6.

460. A key "convolution" process substantially as shown in FIGURE 62.

461. A process of generating different keys using a key convolution process to determine a "true" key substantially as shown in FIGURE 63.

462. A process of initializing protected processing environment keys substantially as shown in FIGURES 64 and/or 65.

463. A process for decrypting information contained within stationary objects substantially as shown in FIGURE 66.

464. A process for decrypting information contained within traveling objects substantially as shown in FIGURE 67.

465. A process for initializing a protected processing environment substantially as shown in FIGURE 68.

466. A process of downloading firmware into a protected processing environment substantially as shown in FIGURE 69.

467. Multiple VDE electronic appliances connected together with a network or other communications means substantially as shown in FIGURE 70.

468. A portable VDE electronic appliance substantially as shown in FIGURE 71.

469. "Pop-up" displays that may be generated by the user notification and exception interface substantially as shown in Figures 72A-72D.

470. A smart object substantially as shown in FIGURE 73.

471. A method of processing smart objects substantially as shown in FIGURE 74.

472. Electronic negotiation substantially as shown in any of FIGURES 75A-75D.

473. An electronic agreement substantially as shown in FIGURES 75E-75F.

474. Electronic negotiation processes substantially as shown in any of FIGURES 76A-76B.

475. A chain of handling and control substantially as shown in FIGURE 77.

476. A VDE "repository" substantially as shown in FIGURE 78.

477. A process of using a chain of handling and control to evolve and transform VDE managed content and control information substantially as shown in any or all of FIGURES 79-83.

478. A chain of handling and control involving several categories of VDE participants substantially as shown in FIGURE 84.

479. A chain of distribution and handling within an organization substantially as shown in FIGURE 85.

480. A chain of handling and control substantially as shown in Figures 86 and/or 86A.

481. A virtual silicon container model substantially as shown in Figure 87.

482. A method of business automation characterized by the steps of (a) creating one or more secure containers including encrypted accounting and/or other administrative information content, (b) associating control information with one or more of such one or more secure containers including a description of (i) the one or more parties whom may use one or more of the one or more containers, and (ii) the operations that will be performed for one or more parties with respect to such accounting and/or other administrative information, (c) electronically delivering one or more of such one or more containers such to one or more parties, and (d) enabling through the use of a protected processing environment the enforcement of at least a portion of such control information.

483. A business automation system characterized by:
means for providing at least one secure container including administrative information content having control information associated therewith, and
a protected processing environment for enforcing, at least in part, the control information.

484. A business automation system comprising (a) distributed, interoperable protected processing environment installations, (b) secure containers for distribution of digital

information, (c) control information supporting the automation of chain of handling and control functions.

485. A method of business automation characterized by the steps of providing interoperable protected processing environment nodes to plural parties, communicating first encrypted digital information from a first party to a second party, communicating second encrypted digital information including at least a portion of said first communicated digital information and/or information related to the use of said first digital information, to a third party different from said first or second parties, wherein use of said second encrypted digital information is regulated, at least in part, by an interoperable protected processing environment available to said third party.

486. A business automation system characterized by:
plural protected processing environment nodes,
means for communicating digital information between the nodes, and

wherein at least one of the nodes includes means for regulating the use of said communicated digital information.

487. A method for chain of handling and control characterized by the steps of (a) a first party placing protected digital information into a first software container and stipulating

rules and controls governing use of at least a portion of said digital information, (b) providing said software container to a second party, wherein said second party places said software container into a further software container and stipulates rules and controls for at least in part managing use of at least a portion of said digital information and/or said first software container by a third party.

488. A chain of handling and control system characterized by:

means for placing digital information into a first software container and for stipulating rules and/or controls governing use of at least a portion of said digital information, and

means for placing said software container into a further software container and for stipulating further rules and/or controls for at least in part managing use of at least a portion of said digital information and/or said first software container.

489. A system for chain of handling and control including (a) a first container containing at least in part protected digital information, (b) at least in part protected control information stipulated by a first party establishing conditions for use of at least a portion of said digital content, (c) a second container different from said first container, said second container containing said first container, (d) control information stipulated

independently by a second party for at least in part setting conditions for managing use of the contents of said second container.

490. A system for electronic advertising including: (a) means to provide digital information to users for their use, (b) means to provide advertising content to said users in combination with said digital information, (c) means to audit use of said digital information, (d) means to securely acquire usage information regarding use of advertising content, (e) means to securely report information based upon said advertising content usage information, (f) compensating at least one content provider at least in part based upon use of said advertising content.

491. A method for electronic advertising characterized by the steps of (a) placing digital information into a container, (b) associating advertising information with at least a portion of said digital information, (c) securely providing said container to a container user, (d) monitoring user viewing of advertising information, and (d) receiving payment from an advertiser, wherein said payment is related to user viewing of said advertising information.

492. A system for electronic advertising involving (a) means to containerize digital information including both content

and advertising information, (b) means to monitor viewing of at least a portion of said advertising information, (c) means to charge for user viewing of at least a portion of said advertising information, (d) means to securely communicate information based upon said viewing in a secure container, and (e) control information related to said containerized digital information for managing the communication of said information based upon said viewing.

493. A method for electronic advertising characterized by the steps of (a) containerizing digital information including both content and advertising information, (b) monitoring user viewing of at least a portion of said advertising information, (c) charging for user viewing of at least a portion of said advertising information, (d) securely communicating information based upon said viewing in a secure container, and (e) at least in part managing, through the use of control information related to said advertising information, the communication of information based upon said viewing.

494. A method of clearing transaction information characterized by the steps of (a) securely distributing digital information to a first user of an interoperable protected processing environment, (b) securely distributing further digital information to a user of an interoperable protected processing

environment different from said at first user (c) receiving information related to usage of said digital information, (d) receiving information related to usage of said further digital information, and (e) processing information received according to steps (c) and (d) to perform at least one of (I) an administrative, or (II) an analysis, function.

495. A system for clearing transaction information including (a) a first container containing at least in part protected digital information and associated control information, (b) a second secure container containing further at least in part protected digital information and associated control information, (c) means to distribute said first and second containers to users, (d) communication means for communicating information at least in part derived from user usage of said first container digital information, (e) communication means for communicating information at least in part derived from user usage of said second container digital information, (f) processing means at a clearinghouse site for receiving the information communicated through steps (d) and (e), wherein said processing means perform administrative and/or analysis processing of at least a portion of said communicated information.

496. A method for clearinghouse analysis characterized by the steps of: (a) enabling plural independent clearinghouses for

administering and/or analyzing usage of distributed, at least in part protected, digital information, (b) providing interoperable protected processing environments to plural, independent users, and (c) enabling a user to select a clearinghouse for use with an interoperable protected processing environment

497. A system for clearinghouse analysis including (a) plural independent clearinghouses for administering and/or analyzing usage of distributed, at least in part protected, digital information, (b) at least one interoperable protected processing environments at each of plural user locations, (c) selecting means for enabling a user to select one of said plural independent clearinghouse to perform payment and/or analysis functions related to the use of at least a portion of said at least in part protected, digital information.

498. A method of electronic advertising characterized by the steps of

creating one or more electronic advertisements, creating one or more secure containers including at least a portion of such advertisements,

associating control information with such advertisements including control information describing at least one of: (a) reporting at least some advertisement usage information to one or more content providers, advertisers and/or agents, (b)

providing one or more credits to a user based on such user's viewing and/or other usage of such advertisements, (c) reporting advertisement usage information to one or more market analysts, (d) providing a user with ordering information for and/or means for ordering one or more products and/or services, and/or (e) providing one or more credits to a content provider based on one or more users' viewing and/or other usage of such advertisements,

providing such containers and such control information to one or more users,

enabling such users to use such containers at least in part in accordance with such control information.

499. A system for electronic advertising including (a) means to provide digital information to users for their use, (b) means to provide advertising content to said users in combination with said digital information, (c) means to audit use of said digital information, (d) means to acquire usage information regarding use of advertising content, (e) means to securely report information based upon said advertising content usage information, and (f) compensating at least one content provider at least in part based upon use of such advertising content.

500. A system for chain of handling and control including (a) a first container containing at least in part protected digital information, (b) at least in part protected control information stipulated by a first party establishing condition for use of at least a portion of said digital content, (c) a second container different from said first container, said second container containing said first container, and (d) control information stipulated independently by a second party for at least in part setting conditions for managing use of the contents of said second container.

501. A method of operating a clearinghouse characterized by the steps of receiving usage information related at least in part to use of secure containers from plural parties, determining payments due to one or more parties based at least in part on such usage information, performing and/or causing to be performed transactions resulting in payments to such parties based at least in part on such determinations.

502. An electronic clearinghouse comprising:
means for receiving usage information related at least in part to use of secure containers from plural parties,
means for determining payments due to one or more parties based at least in part on such usage information,

means for performing and/or causing to be performed transactions resulting in payments to such parties based at least in part on such determinations.

503. A method of operating a clearinghouse characterized by the steps of receiving usage information related at least in part to use of secure containers from plural parties, determining reports of usage for one or more parties based at least in part on such usage information, creating and/or causing to be created reports of usage based at least in part on such determination, delivering at least one of such reports to at least one of such parties.

504. A method of operating a clearinghouse characterized by the steps of receiving permissions and/or other control information from one or more content providers including information that enables delivery of at least one right in at least one secure container to other parties, receiving requests from plural parties for one or more rights in one or more secure containers, delivering permissions and/or other control information to such parties based at least in part on such requests.

505. A method of operating a clearinghouse characterized by the steps of receiving information from one or more parties

establishing a party's identity information, creating one or more electronic representations of at least a portion of such identity information for use in enabling and/or withholding at least one right in at least one secure container, performing an operation to certify such electronic representations, delivering such electronic representations to such party.

506. A method of operating a clearinghouse characterized by the steps of receiving a request for credit from a party for use with secure containers, determining an amount of credit based at least in part on such request, creating control information related to such an amount, delivering such control information to such user, receiving usage information related to use of such credit, performing and/or causing to be performed at least one transaction associated with collecting payment from such user.

507. A method for contributing secure control information with respect to an electronic value chain wherein control information is contributed by a party not directly participating in said value chain, comprising steps of: aggregating said contributed control information with control information associated with digital information stipulated by one or more parties in an electronic value chain, said aggregate control information at least in part managing conditions related to the use of at least a portion of said digital information.

508. A method for entering the payment of taxes associated with commercial events wherein secure control information for automatically governing tax payments for said commercial events is contributed by a party comprising steps of: aggregating said secure control information with control information that has been contributed by a separate party and controlling at least one condition for use of digital information.

509. A method for general purpose reusable electronic commerce arrangement characterized by the steps of:

(a) providing component structures, modular methods that can be configured together to comprise event controlled

(b) providing integrateable protected processing environments to plural independent users;

(c) employing secure communications means for communicating digital control information between integrateable protected processing environments; and

(d) enabling database managers operably connected to said processing environments for storing at least a portion of said provided component modular methods.

510. A system for general purpose, reusable electronic commerce including:

(a) component modular methods configured together to comprise event control structures;

(b) at least one interoperable processing environment at each of plural independent user locations;

(c) secure communications means for communicating digital control information between interoperable protected processing environments; and

(d) secured database managers operably connected to said protected processing environments for storing at least a portion of said component modular methods.

511. A general purpose electronic commerce credit system including:

(a) a secure interoperable protected processing environment;

(b) general purpose credit control information for providing credit for user usage of at least in part protected digital information; and

(c) at least in part protected digital information related control information for providing necessary information for employing credit through the use, at least in part, of said general purpose credit control information.

512. A method for enabling a general purpose electronic commerce credit system including:

(a) providing secure interoperable protected processing environments;

(b) supplying general purpose credit control information for providing credit for user usage of at least in part protected digital information; and

(c) providing, at least in part, protected digital information related control information for providing necessary information for employing credit through the use, at least in part, of said general purpose credit control information.

513. A document management system comprising one or more electronic appliances containing one or more SPUs and one or more secure databases operatively connected to at least one of the SPUs.

514. An electronic contract system comprising one or more electronic appliances containing one or more SPUs and one or more secure databases operatively connected to at least one of the SPUs.

515. An electronic appliance containing at least one SPU and at least one secure database operatively connected to at least one of the SPU(s).

516. An electronic appliance containing one or more CPUs where at least one of the CPUs is integrated with at least one SPU.

517. An electronic appliance containing one or more video controllers where at least one of the video controllers is integrated with at least one SPU.

518. An electronic appliance containing one or more network communications means where at least one of the network communications means is integrated with at least one SPU.

519. An electronic appliance containing one or more modems where at least one of the modems is integrated with at least one SPU.

520. An electronic appliance containing one or more CD-ROM devices where at least one of the CD-ROM devices is integrated with at least one SPU.

521. An electronic appliance containing one or more set-top controllers where at least one of the set-top controllers is integrated with at least one SPU.

522. An electronic appliance containing one or more game systems where at least one of the game systems is integrated with at least one SPU.

523. An integrated circuit supporting multiple encryption algorithms comprising at least one microprocessor, memory, input/output means, at least one circuit for encrypting and/or decrypting information and one or more software programs for use with at least one of the microprocessors to perform encryption and/or decryption functions.

524. An integrated circuit comprising at least one microprocessor, memory, at least one real time clock, at least one random number generator, at least one circuit for encrypting and/or decrypting information and independently delivered and/or independently deliverable certified software.

525. An integrated circuit comprising at least one microprocessor, memory, input/output means, a tamper resistant barrier and at least a portion of a Rights Operating System.

526. An integrated circuit comprising at least one microprocessor, memory, input/output means, at least one real time clock, a tamper resistant barrier and means for recording interruption of power to at least one of the real time clocks.

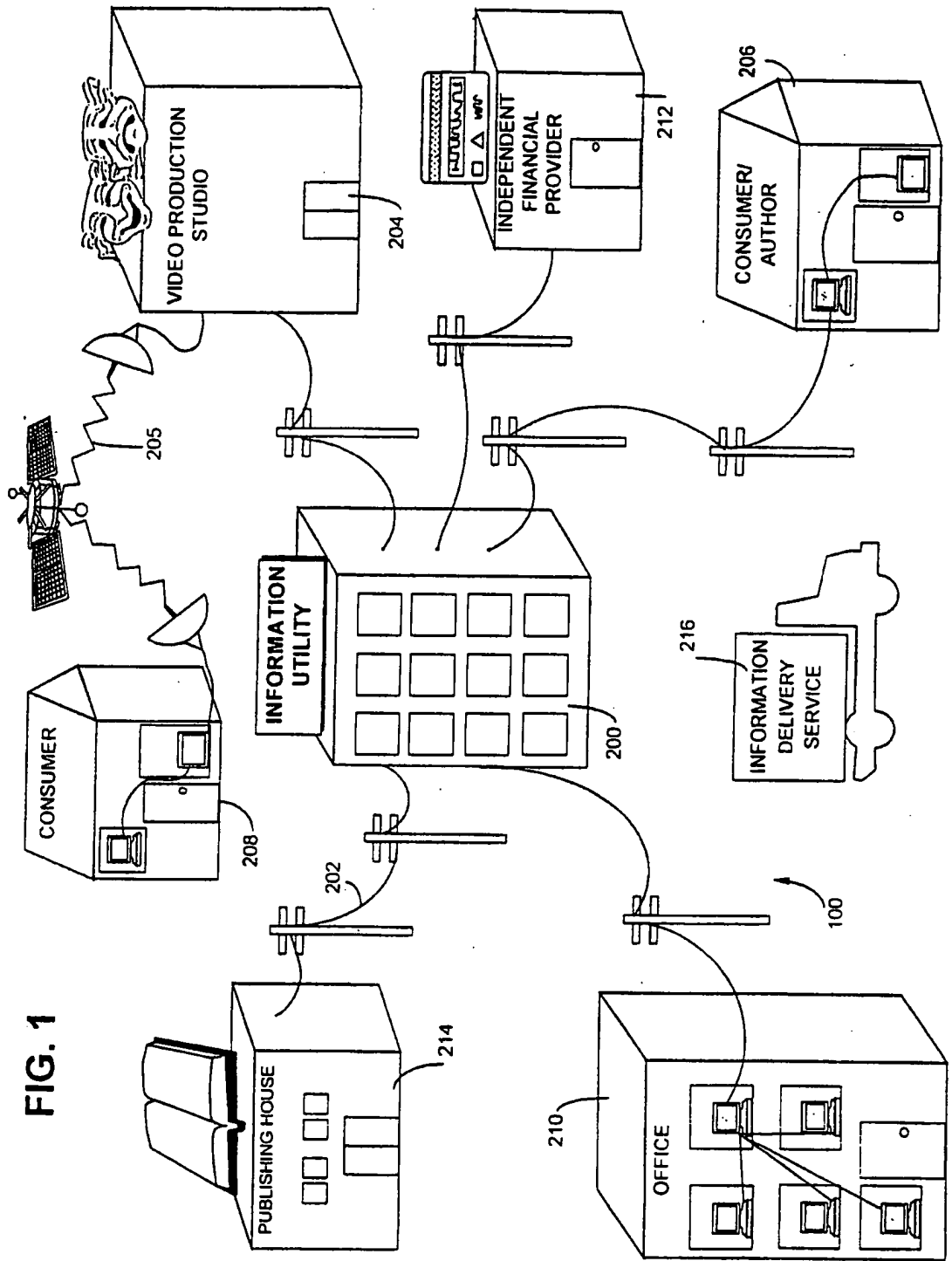


FIG. 1

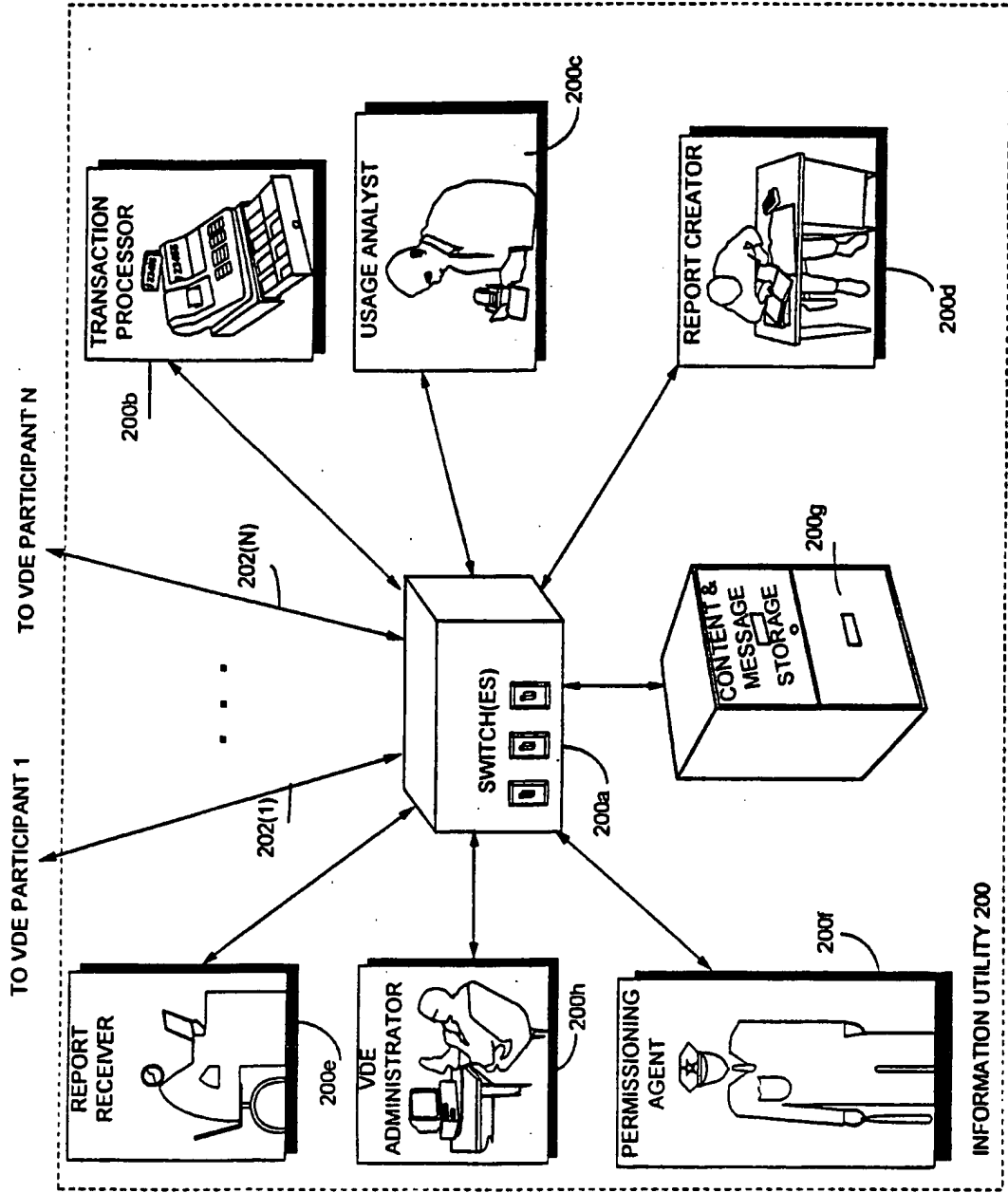


FIG. 1A

FIG. 2

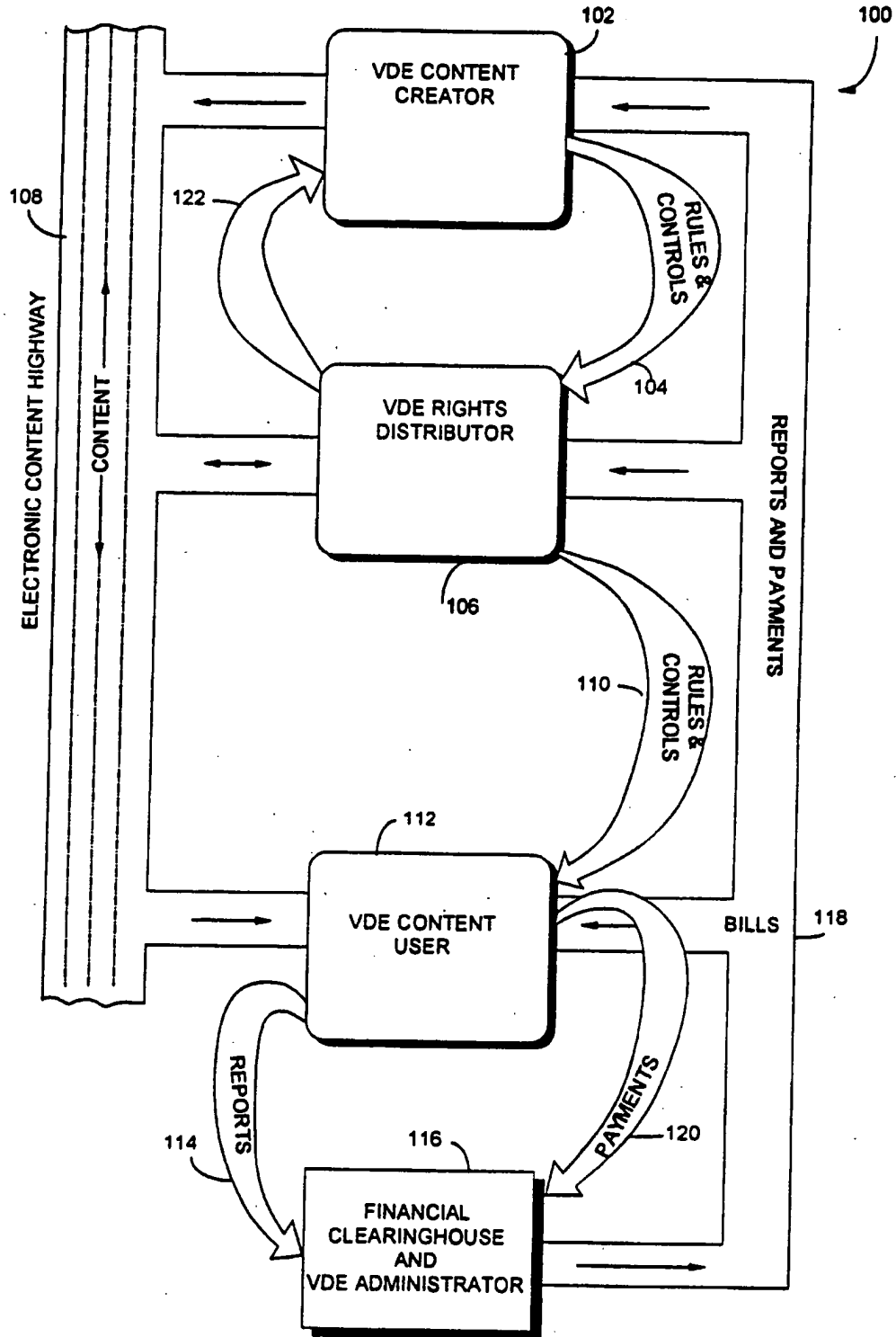
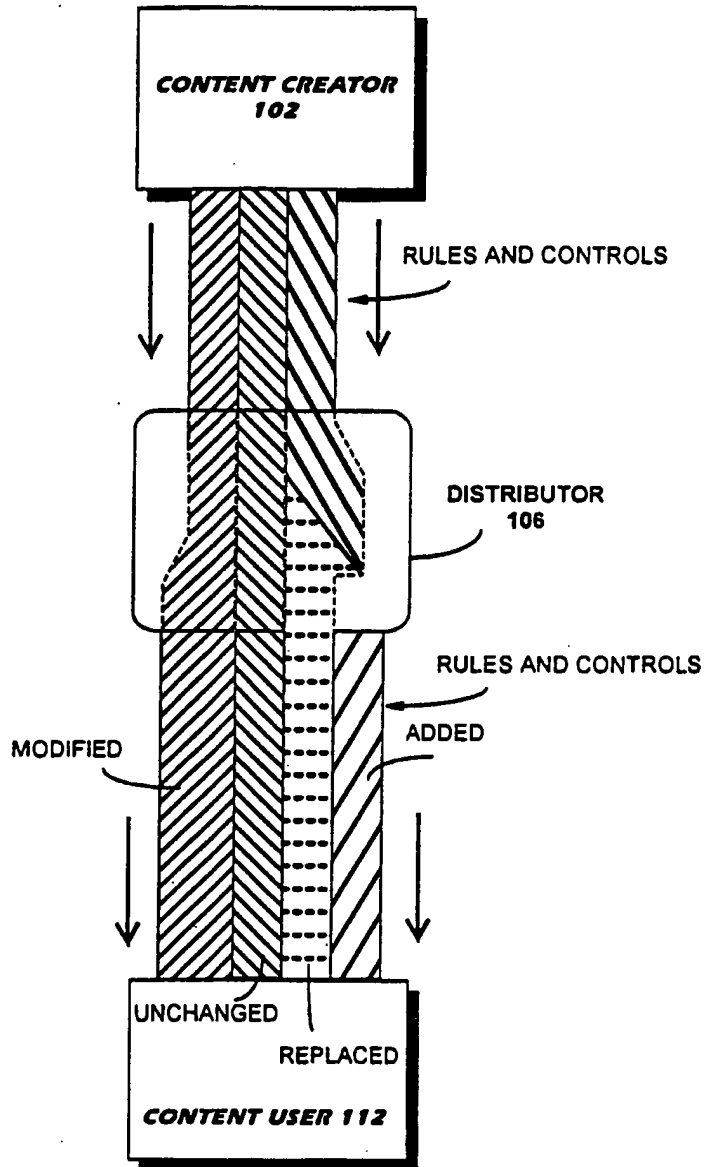
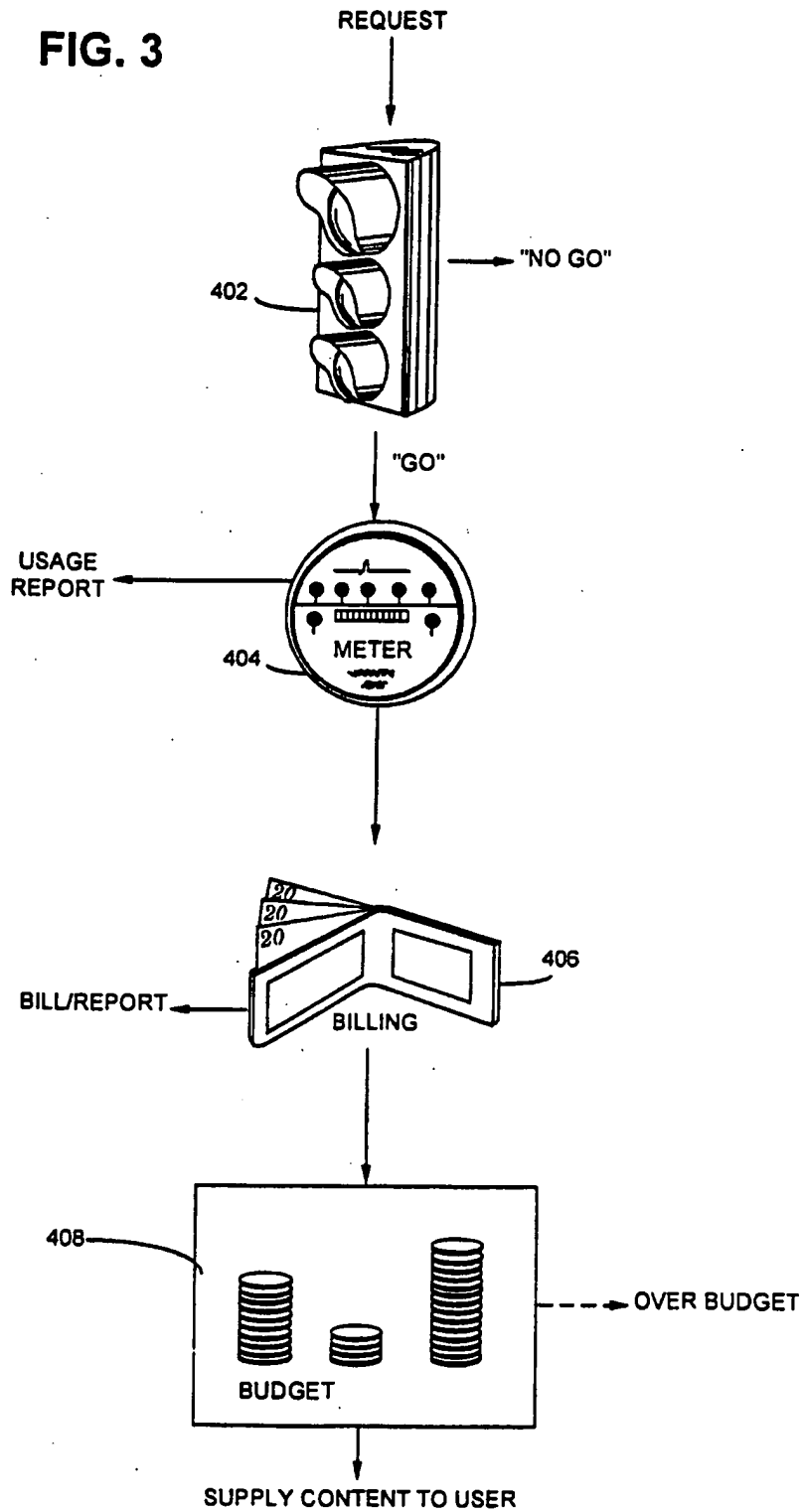


FIG. 2A



5/146

FIG. 3



SUBSTITUTE SHEET (RULE 26)

6/146

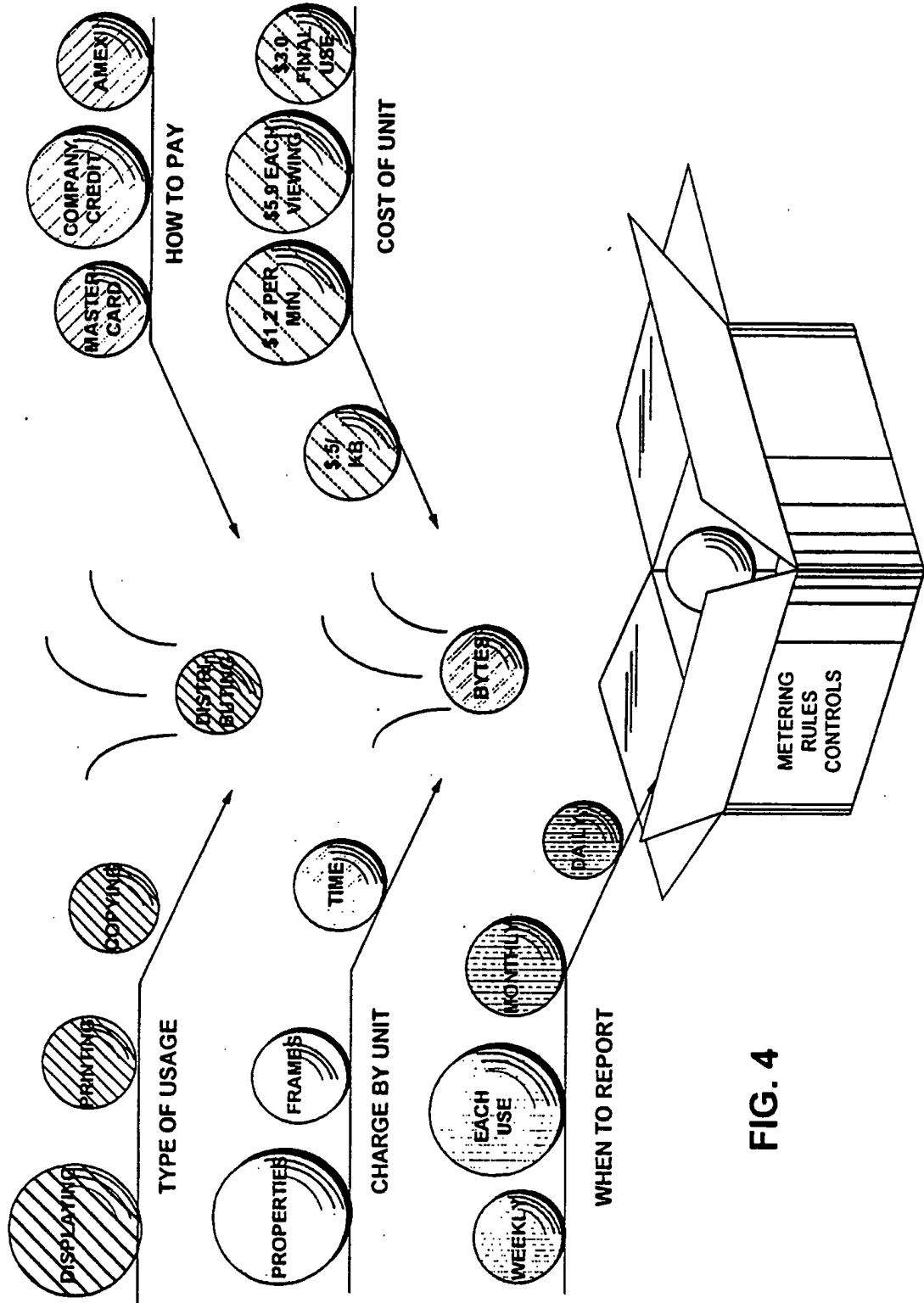
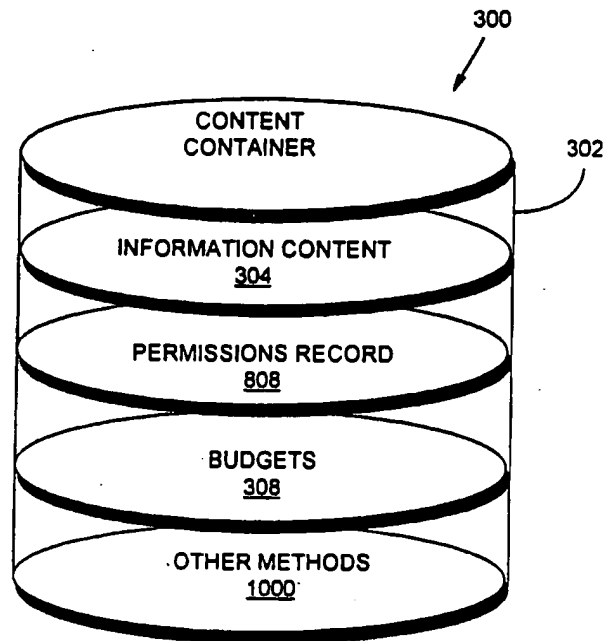


FIG. 4

7/146

FIG. 5A



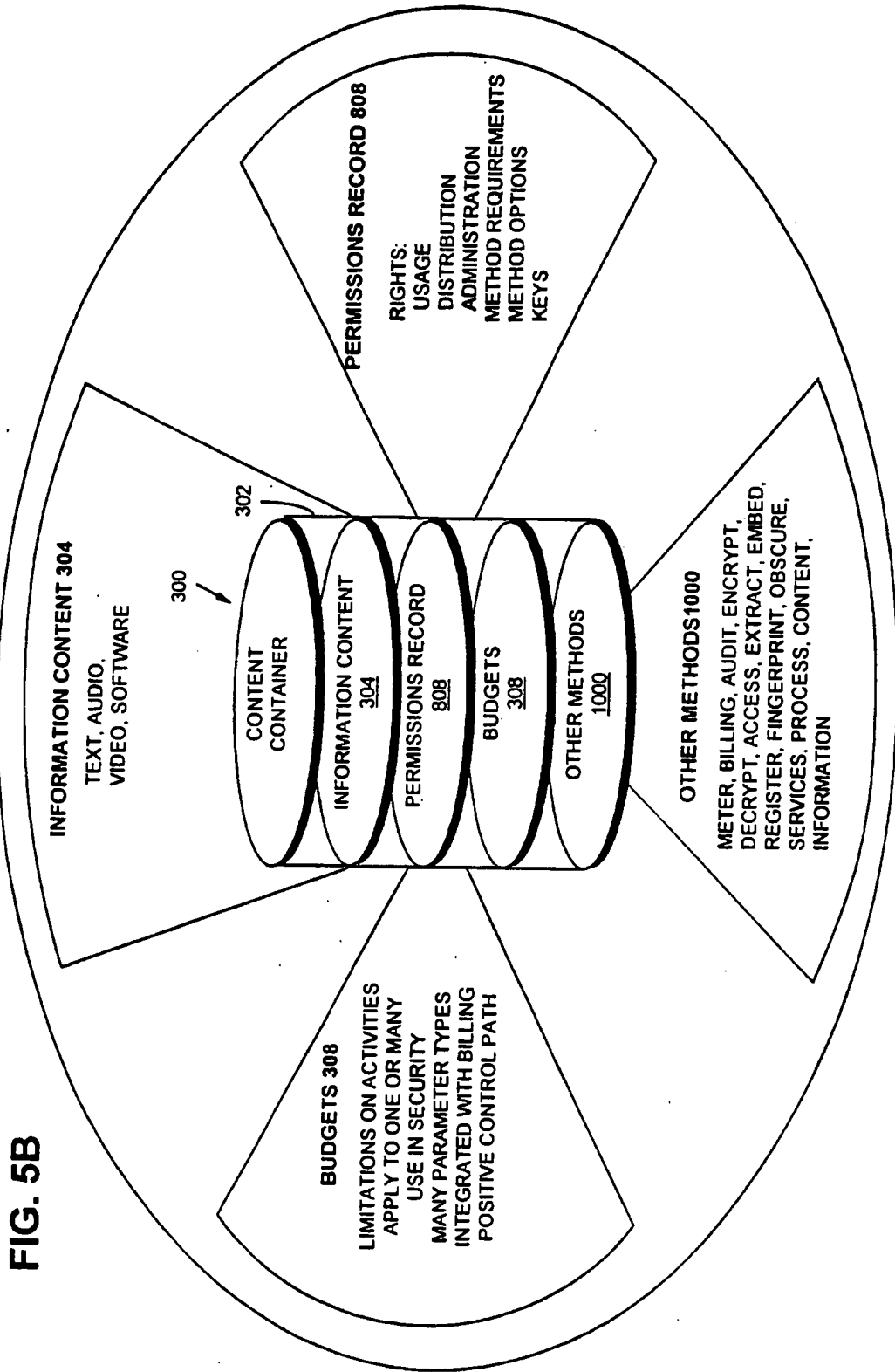
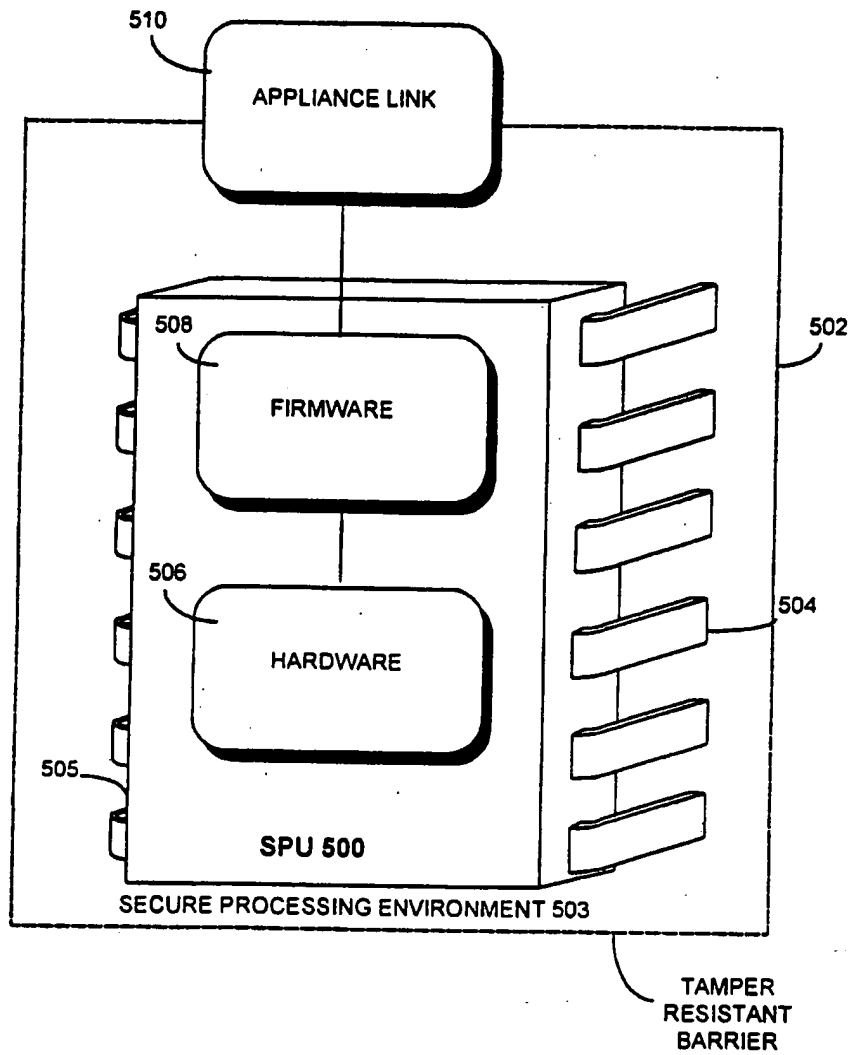


FIG. 5B

9/146

FIG. 6



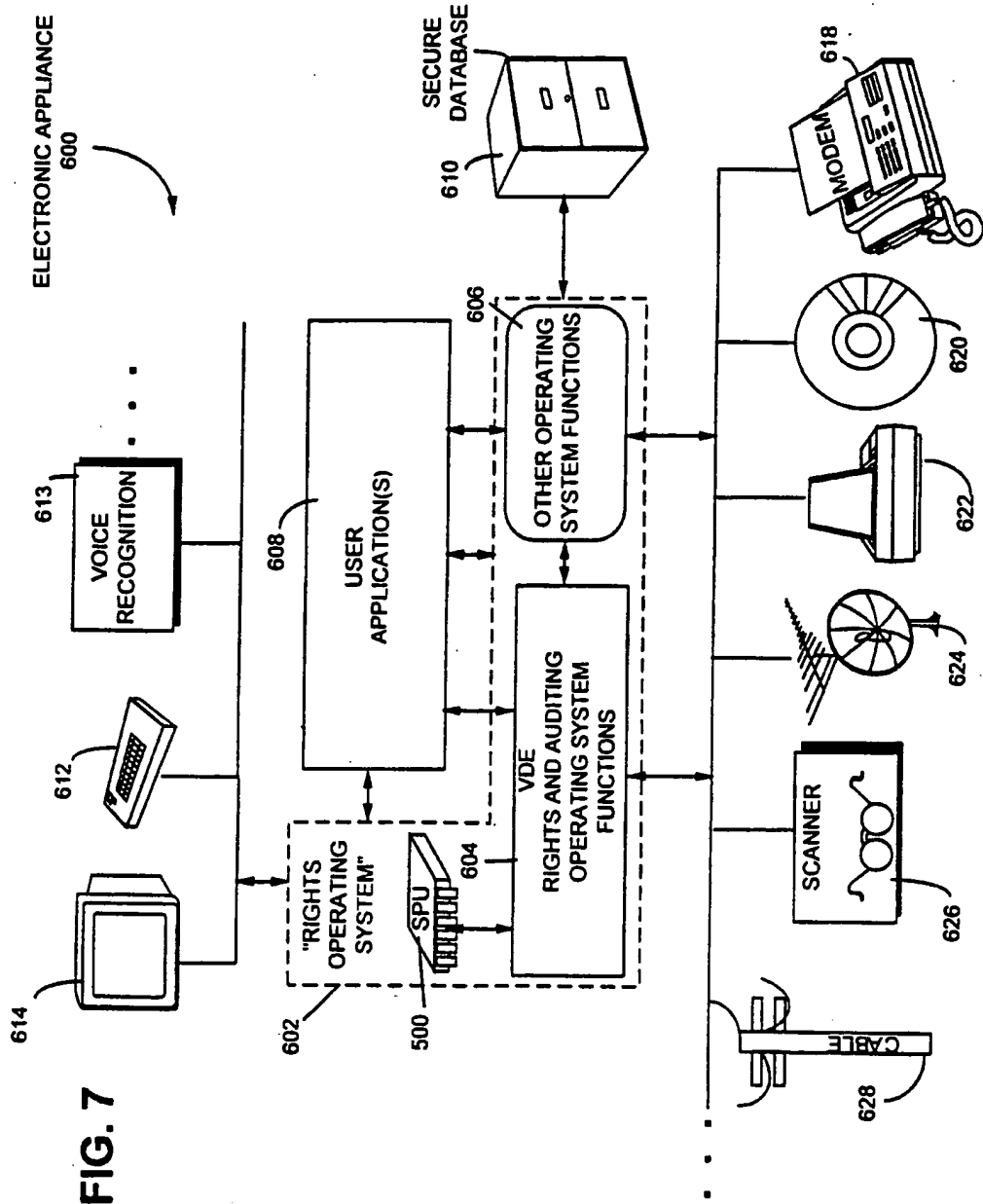
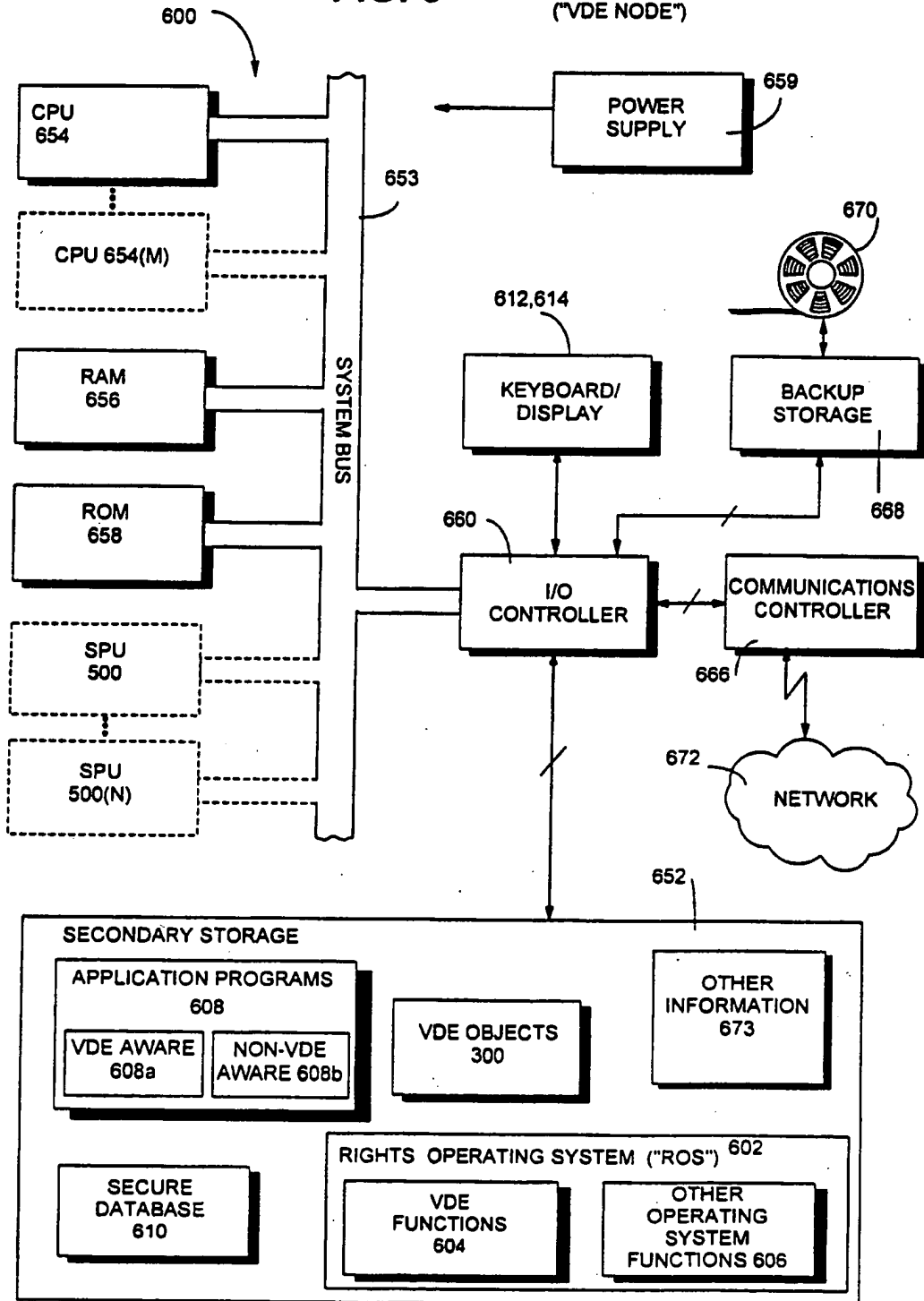


FIG. 7

FIG. 8 ELECTRONIC APPLIANCE 600 ("VDE NODE")



12/146

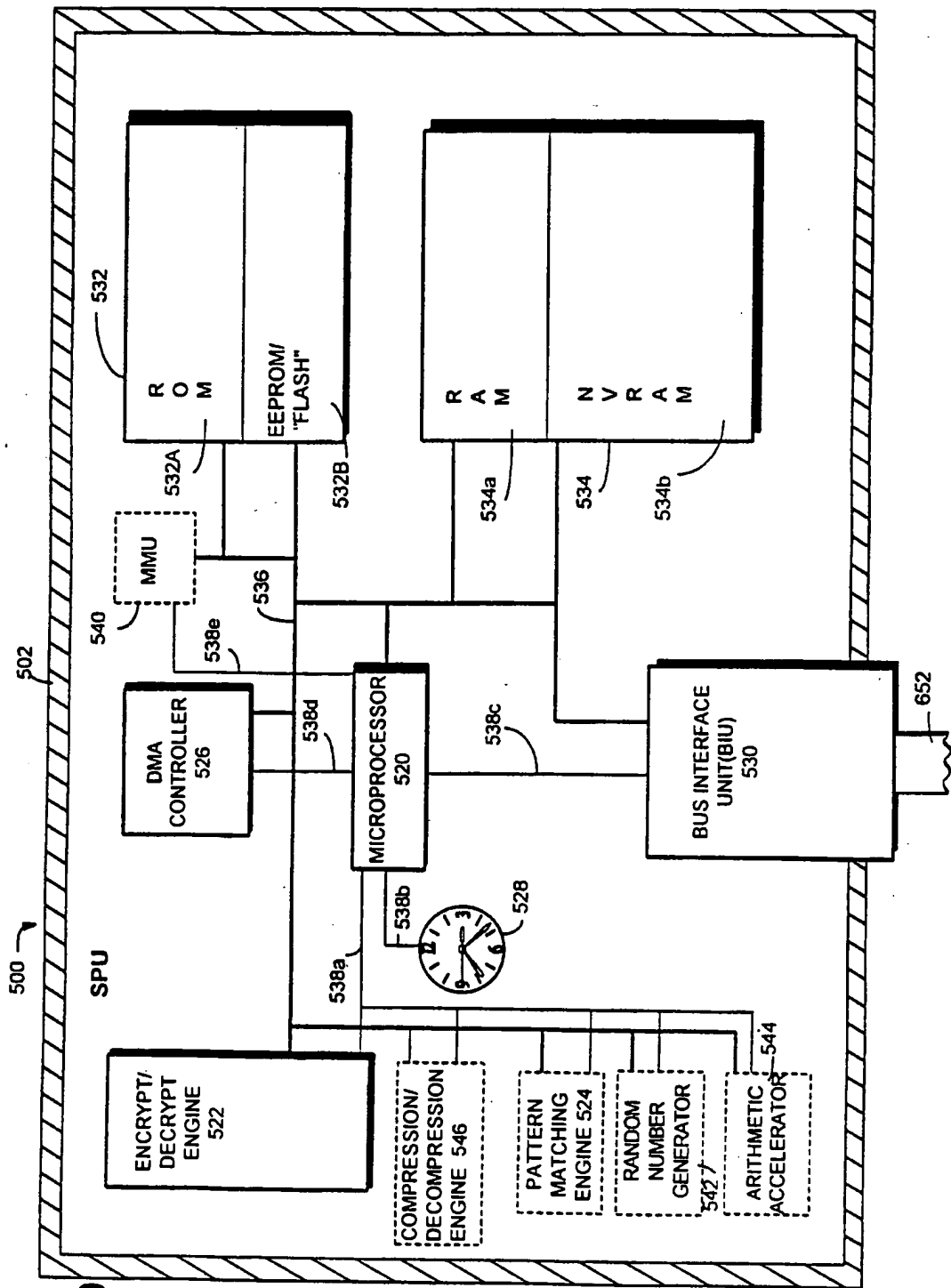


FIG. 9

SUBSTITUTE SHEET (RULE 26)

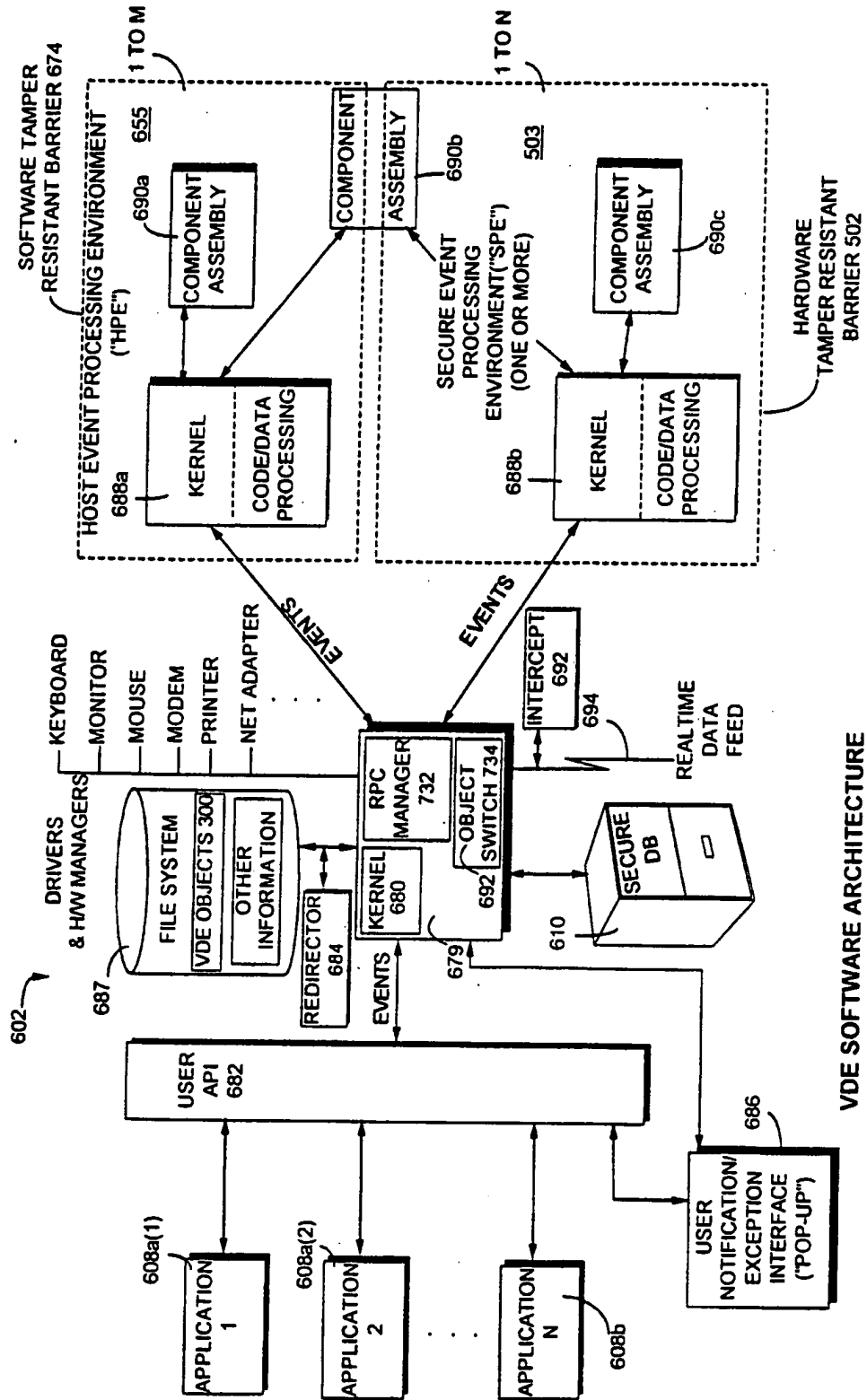
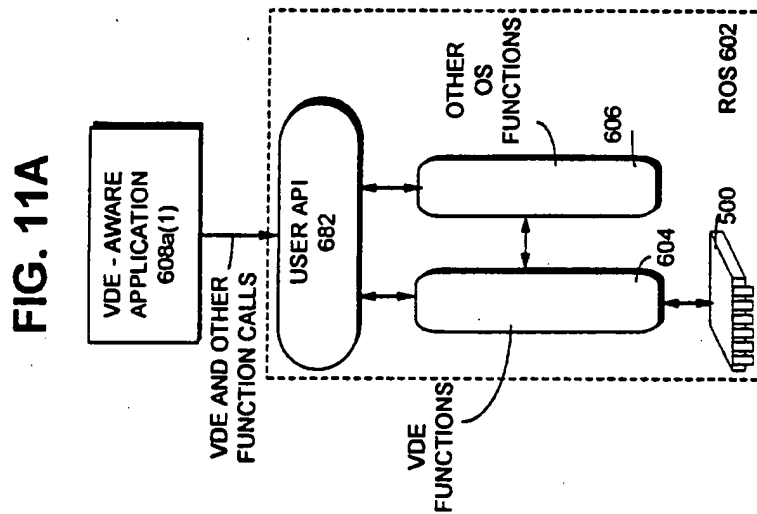
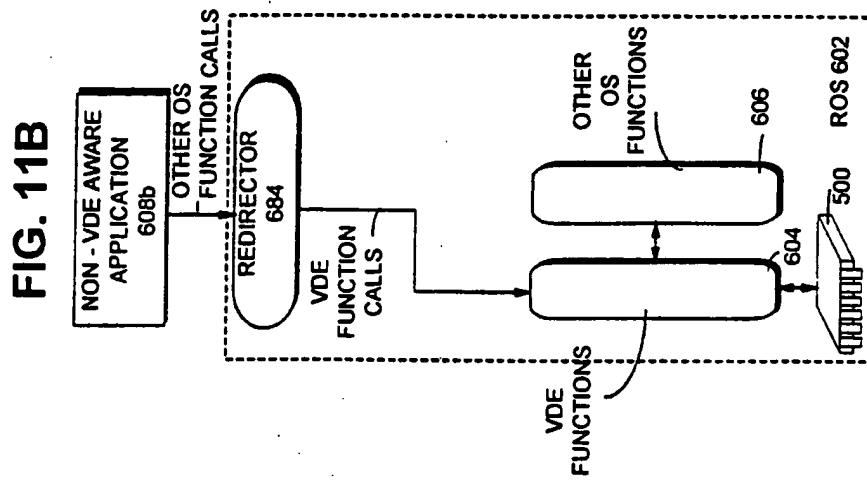
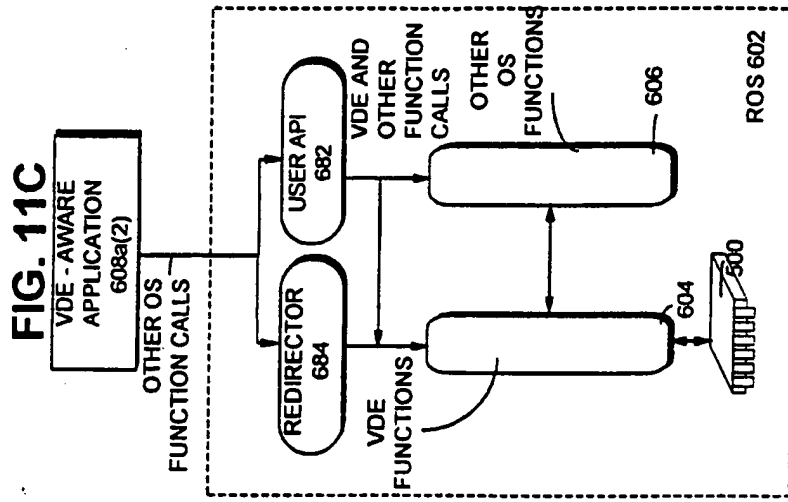


FIG. 10



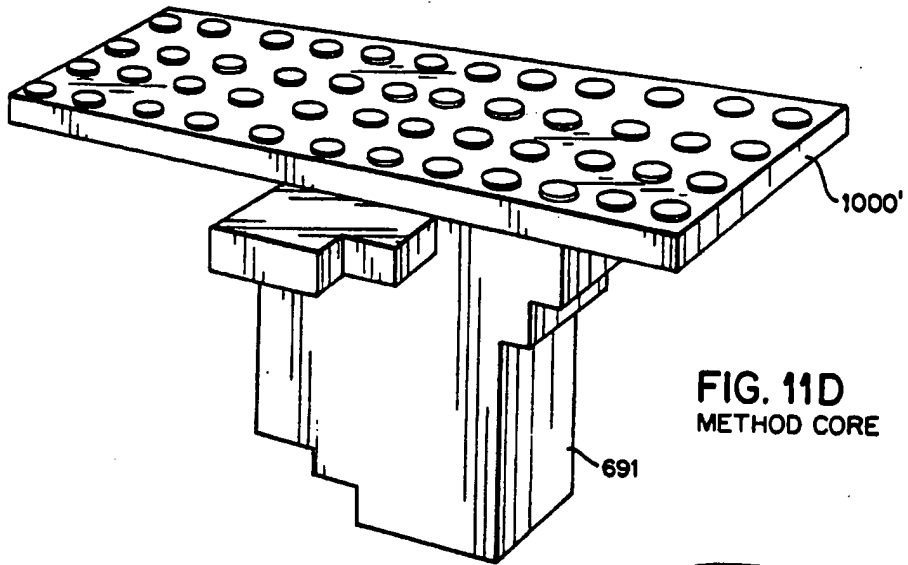


FIG. 11D
METHOD CORE

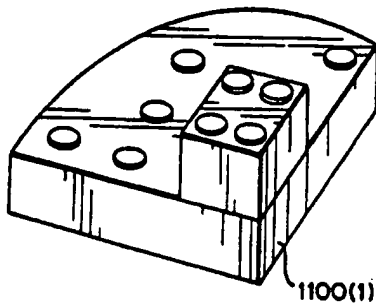


FIG. 11E
LOAD MODULE
WITH DTD

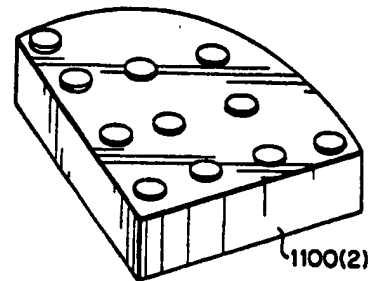


FIG. 11F
LOAD MODULE

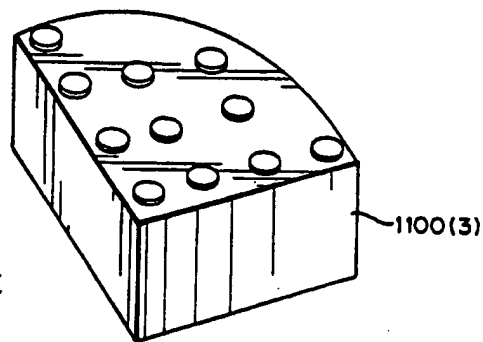


FIG. 11G
LOAD MODULE

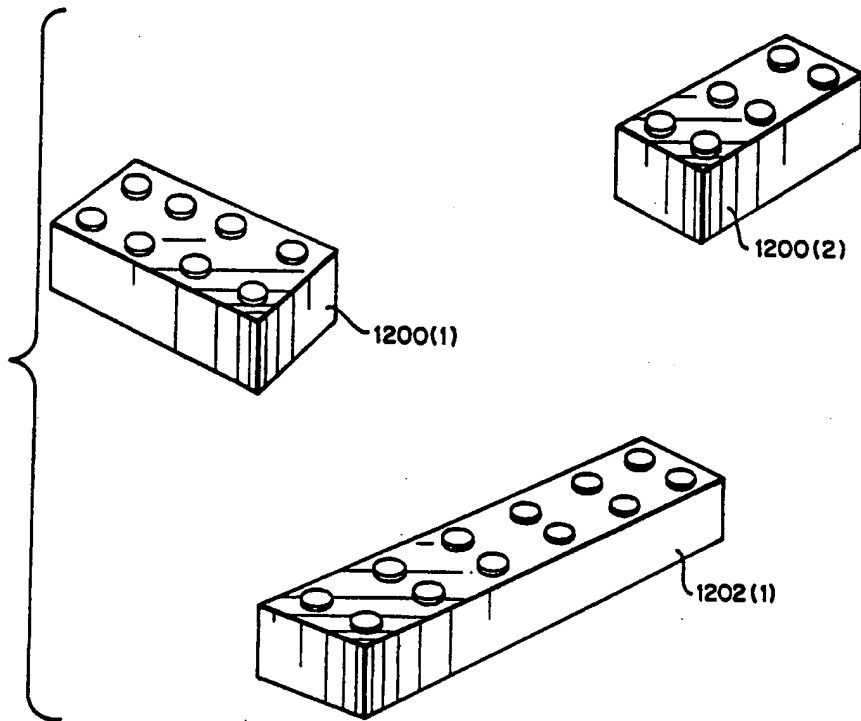
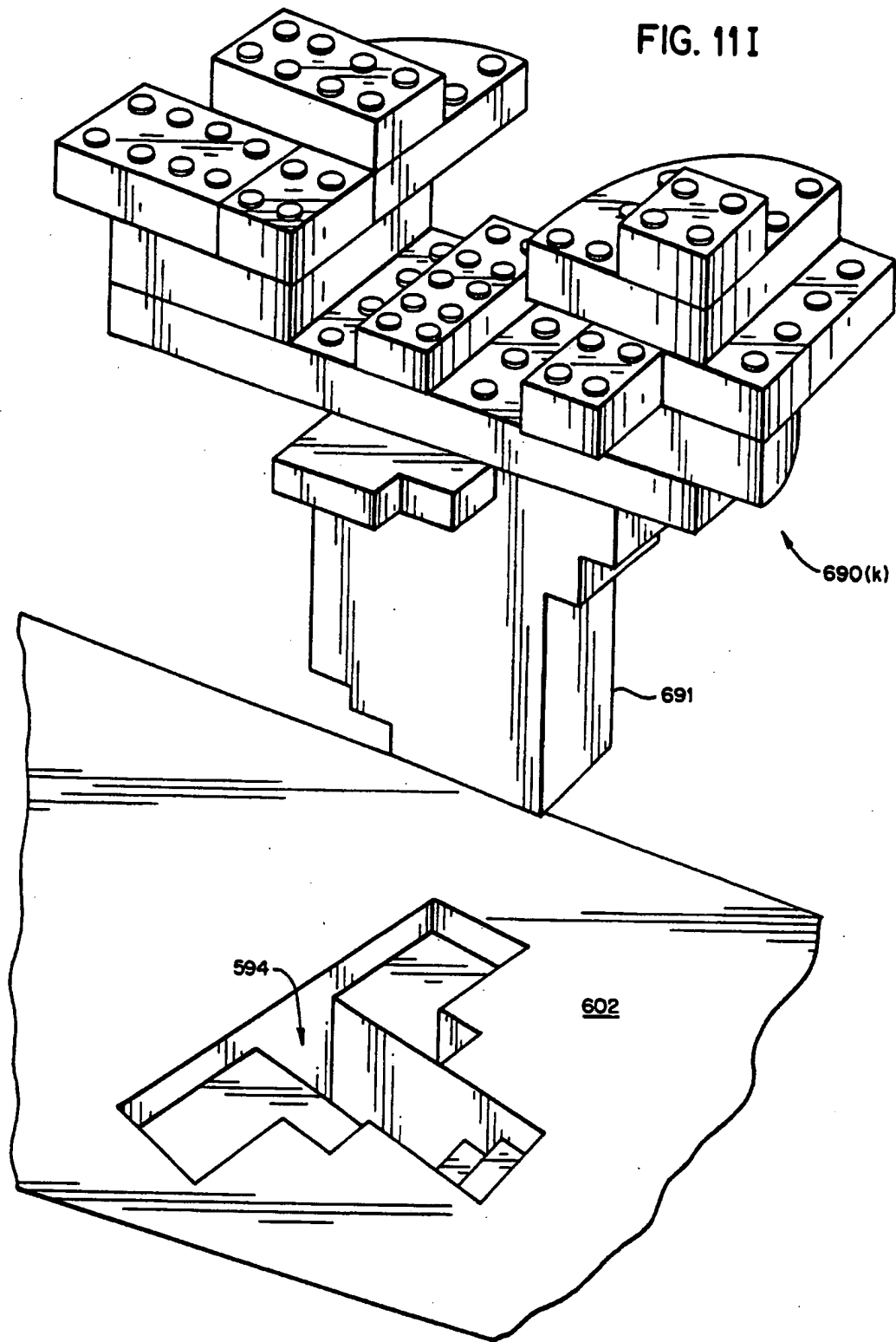


FIG. 11H
DATA STRUCTURES

17/146

FIG. 11I



SUBSTITUTE SHEET (RULE 26)

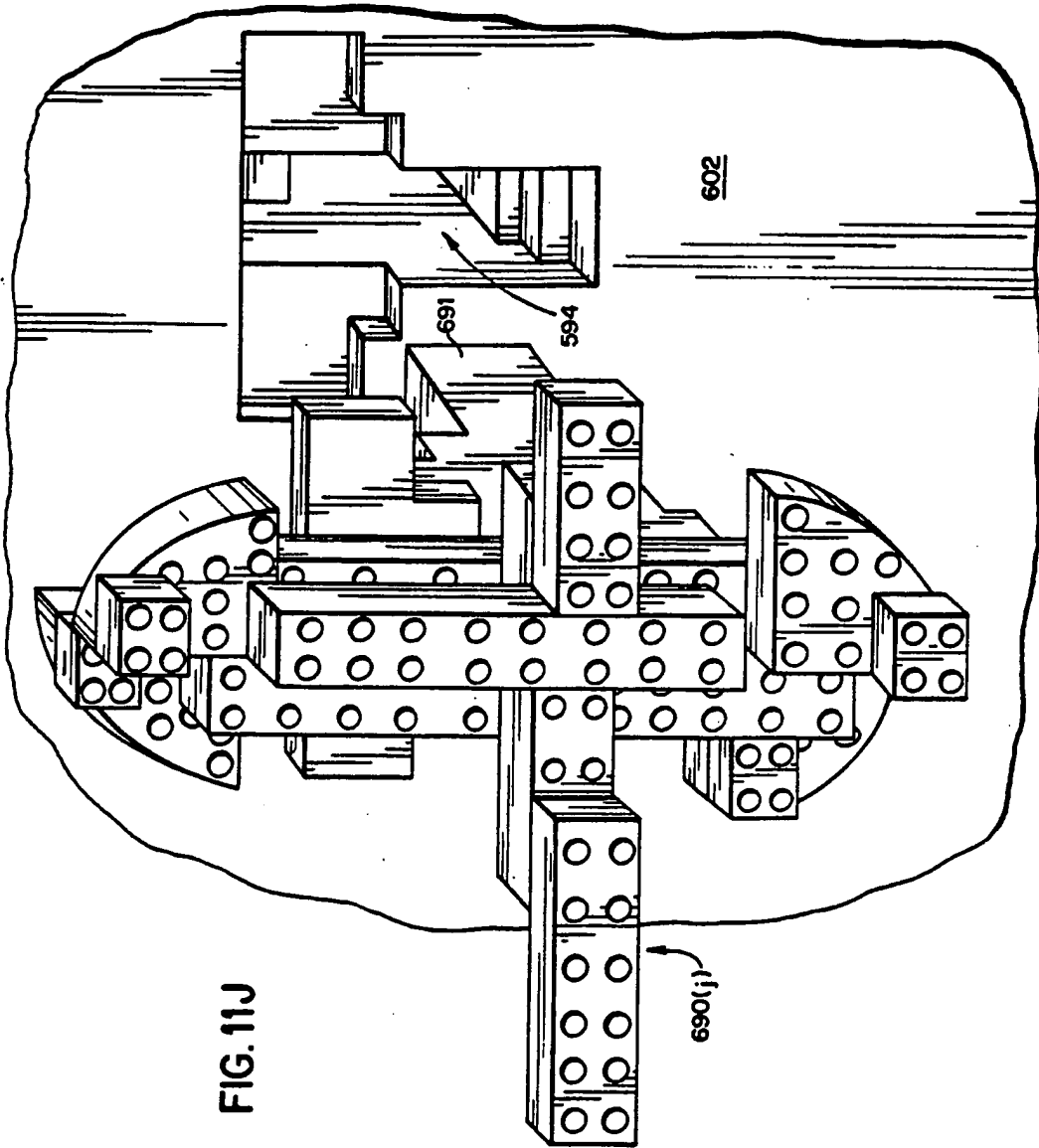


FIG. 11J

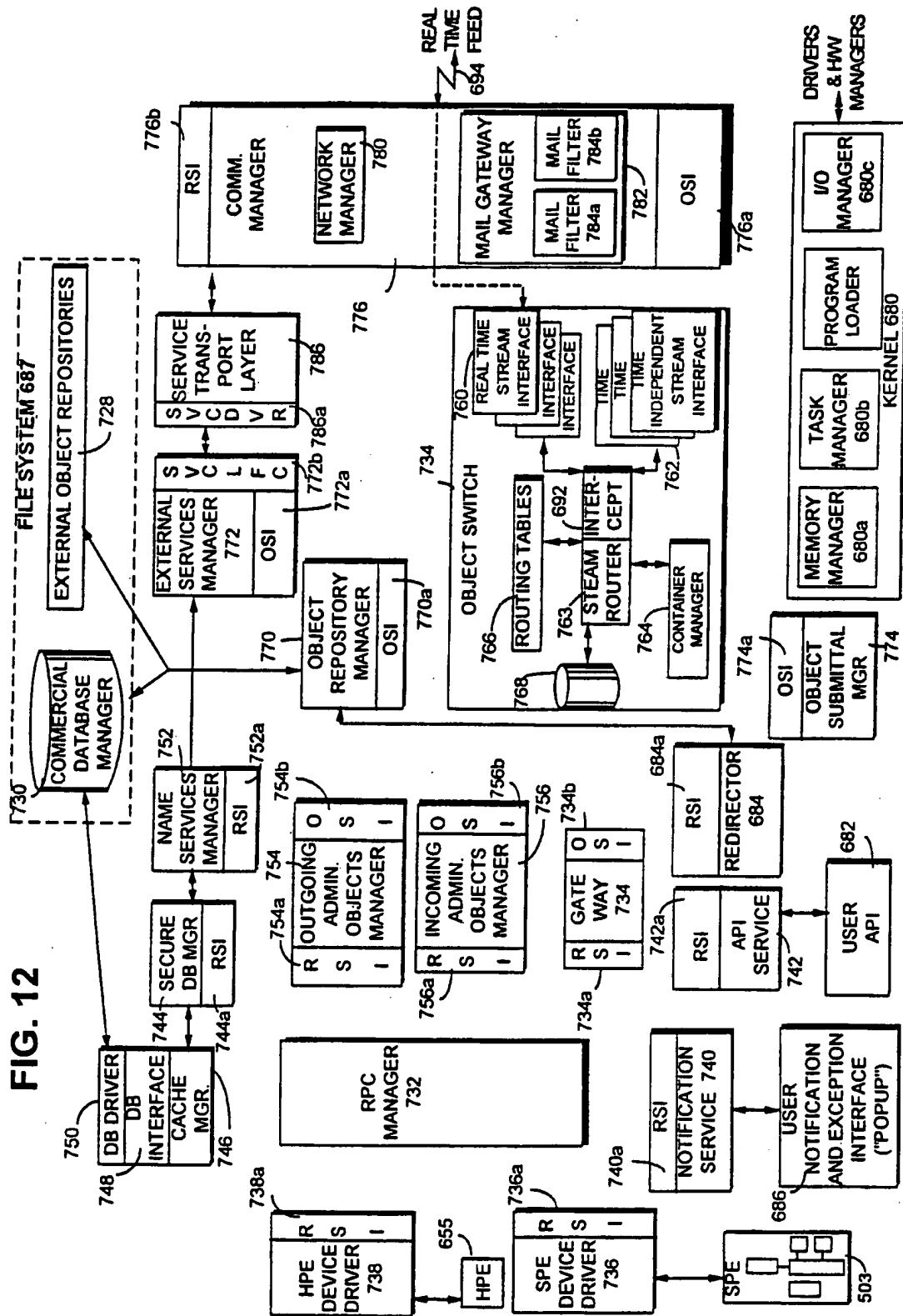


FIG. 12

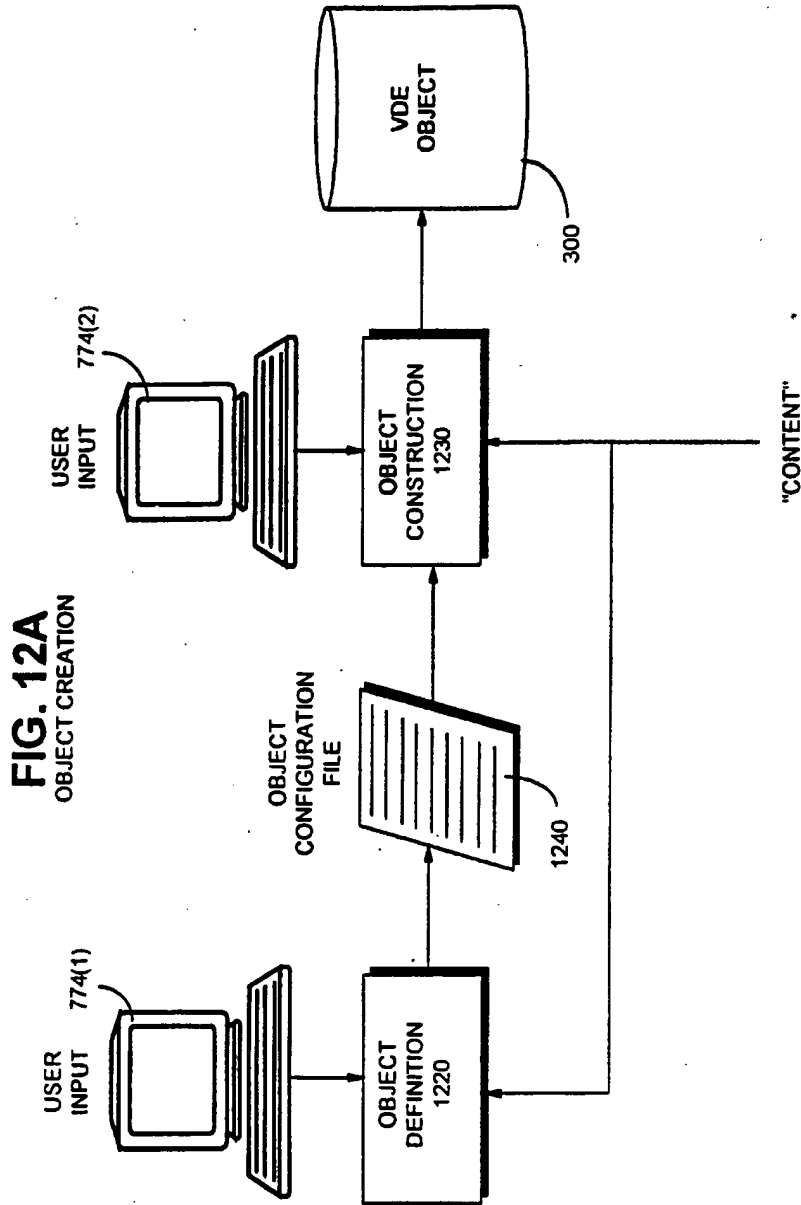
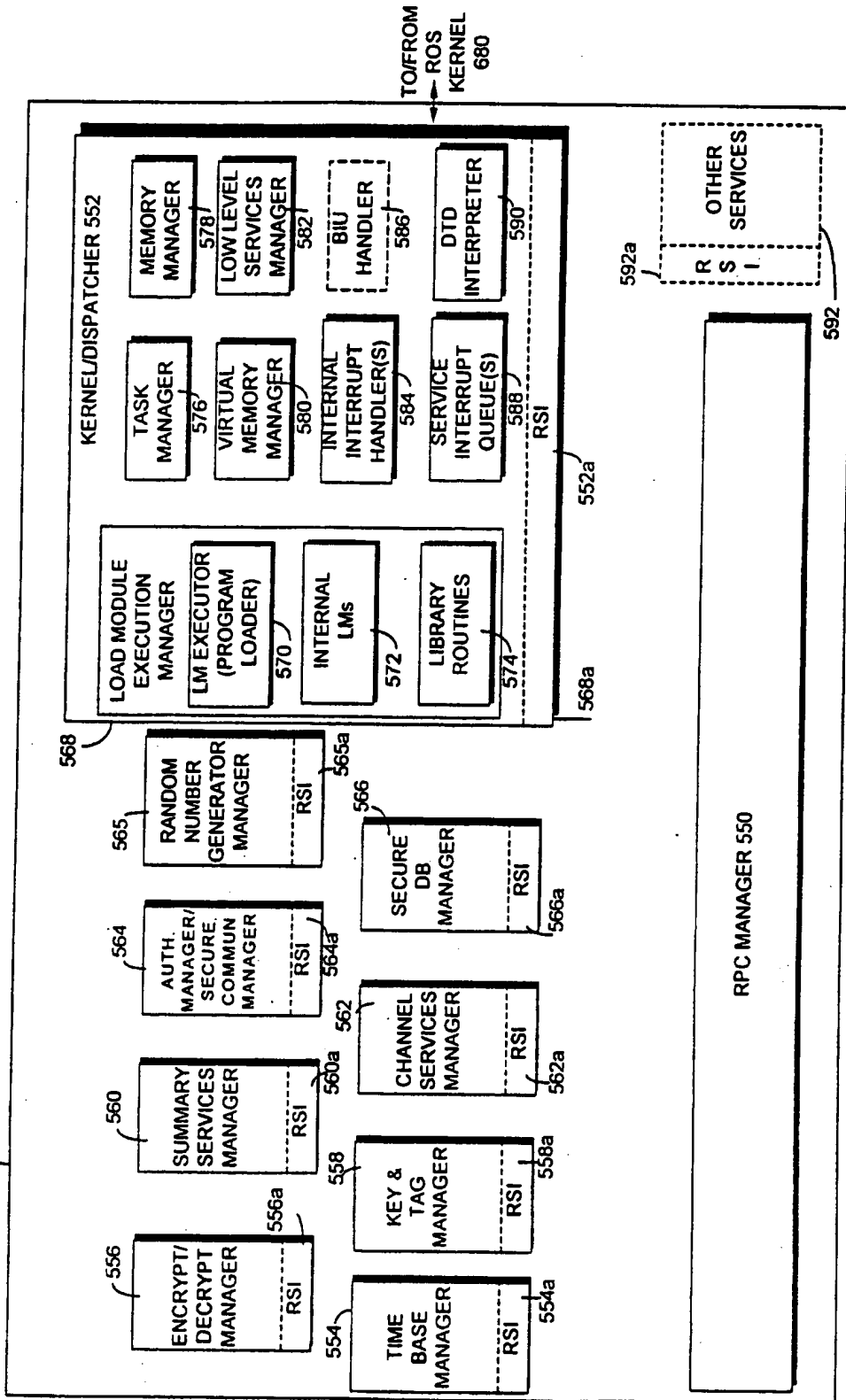


FIG. 12A
OBJECT CREATION

FIG. 13

PROTECTED PROCESSING ENVIRONMENT 650

503, 655

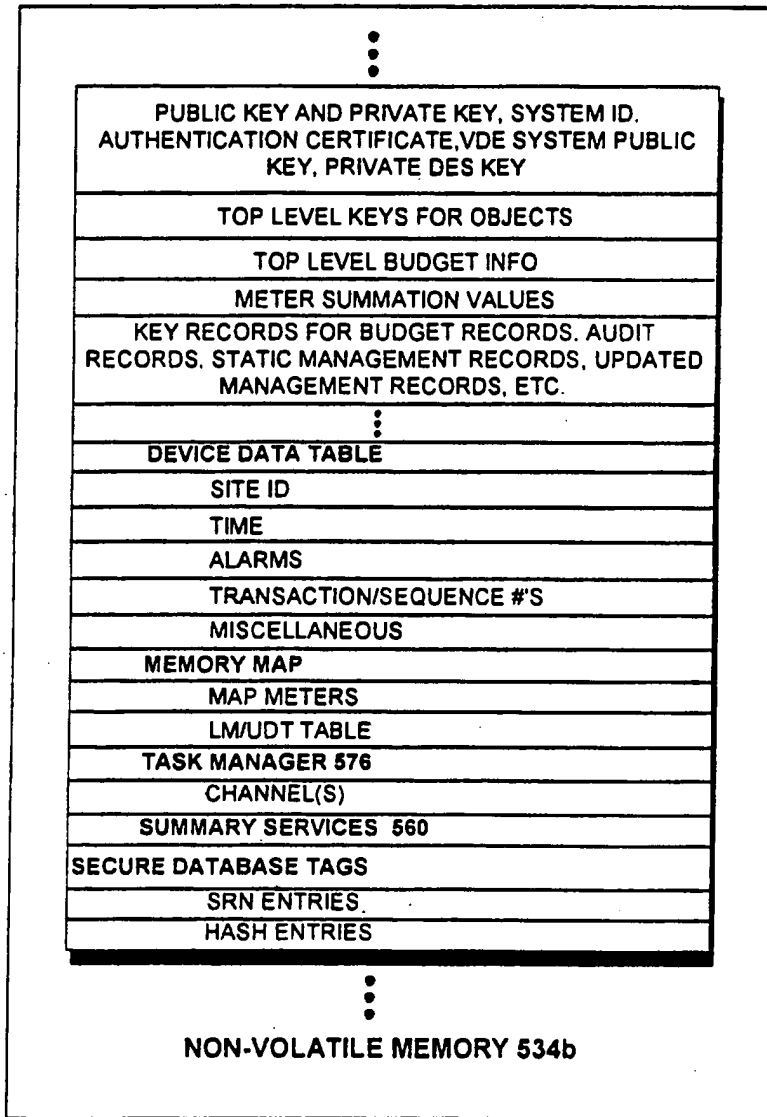


DEVICE FIRM WIRE LOW LEVEL SERVICES 582	TIME BASE MANAGER 554
INITIALIZATION	ENCRYPTION/DECRYPTION MANAGER 556
POST	PK
DOWNLOAD CHALLENGE/RESPONSE AND AUTHENTICATION	BULK
RECOVERY	KEY AND TAG MANAGER 558
EEPROM/FLASH MEMORY MANAGER	KEY STORAGE IN EEPROM
KERNEL/DISPATCHER 552	KEY LOCATOR
INITIALIZATION	KEY GENERATOR
TASK MANAGER 576 (SLEEP/AWAKE/CONTEXT SWAP)	CONVOLUTION ALGORITHM
INTERRUPT HANDLER 584 (TIMER/BIU/POWER FAIL/WATCHDOG TIMER/ENCRYPTION COMPLETED)	SUMMARY SERVICES MANAGER 560
BIU HANDLER 586	EVENT SUMMARIES
MEMORY MANAGER 578	BUDGET SUMMARIES
INITIALIZATION (SETTING MMU TABLES)	DISTRIBUTER SUMMARY SERVICES
ALLOCATE	CHANNEL SERVICES MANAGER 562
DEALLOCATE	CHANNEL HEADERS
VIRTUAL MEMORY MANAGER 580	CHANNEL DETAILS
SWAP BLOCK PAGING	LOAD MODULE EXECUTION SERVICES 568
EXTERNAL MODULE PAGING	AUTHENTICATION MANAGER/SECURE COMMUNICATION MANAGER 564
MEMORY COMPRESS	DATABASE MANAGER 566
RPC AND TABLES 550	MANAGEMENT FILE SUPPORT
INITIALIZATION	TRANSACTION AND SEQUENCE NUMBER SUPPORT
MESSAGING CODE /SERVICES MANAGER	SRN/ HASH
SEND/RECEIVE	DTD INTERPRETER 590
STATUS	LIBRARY ROUTINES 574
RPC DISPATCH TABLE	100 CALLS (STRING SEARCH ETC.)
RPC SERVICE TABLE	MISC. ITEMS THAT ARE PROBABLY LIBRARY ROUTINES
⋮	TAG CHECKING, MD5, CRC'S
	INTERNAL LM'S 572 FOR BASIC METHODS
	METER LOAD MODULE(S)
	BILLING LOAD MODULE(S)
	BUDGET LOAD MODULE(S)
	AUDIT LOAD MODULE(S)
	READ OBJECT LOAD MODULE(S)
	WRITE OBJECT LOAD MODULE(S)
	OPEN OBJECT LOAD MODULE(S)
	CLOSE OBJECT LOAD MODULE(S)
	⋮
	SPU ROM/EEPROM/FLASH 532

FIG. 14A

23/146

FIG. 14B



24/146

FIG. 14C

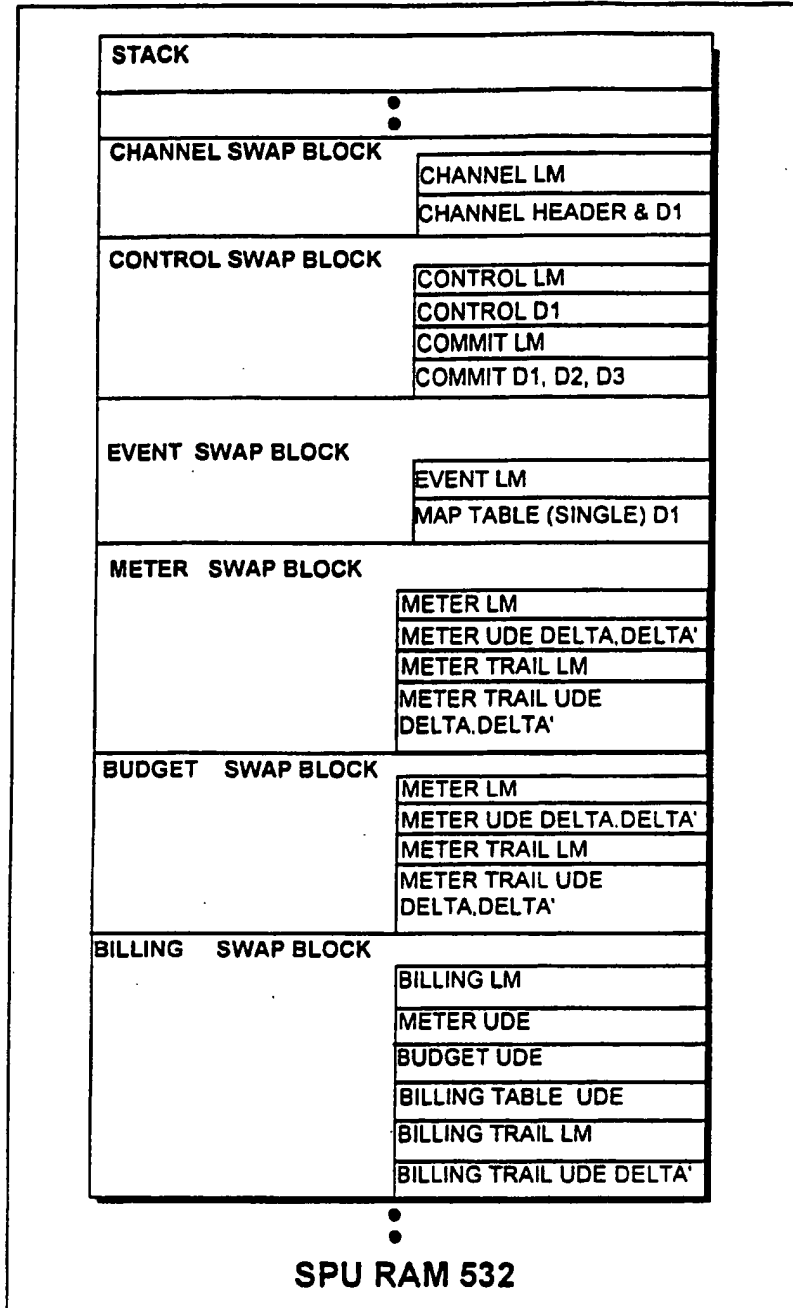
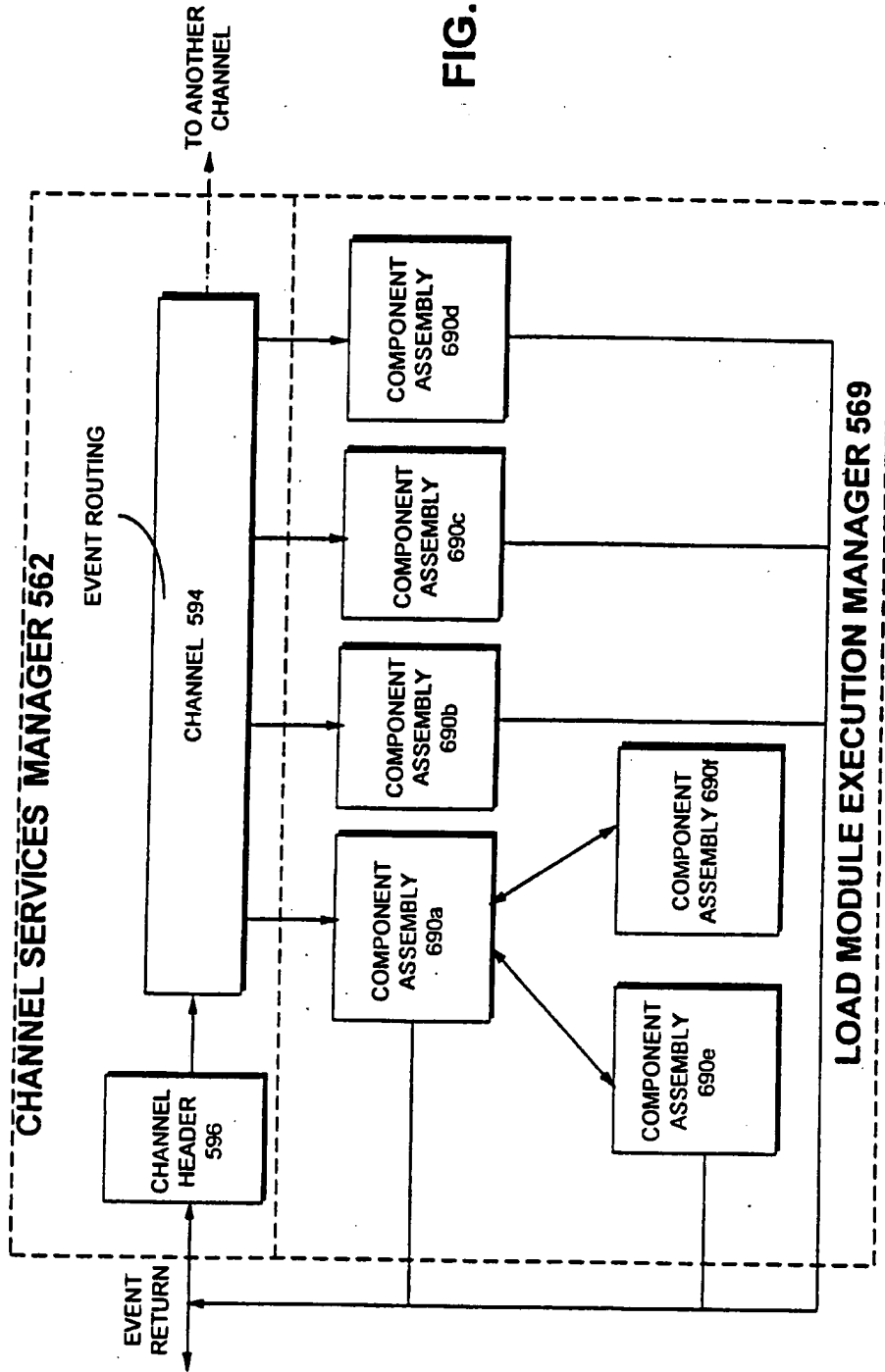
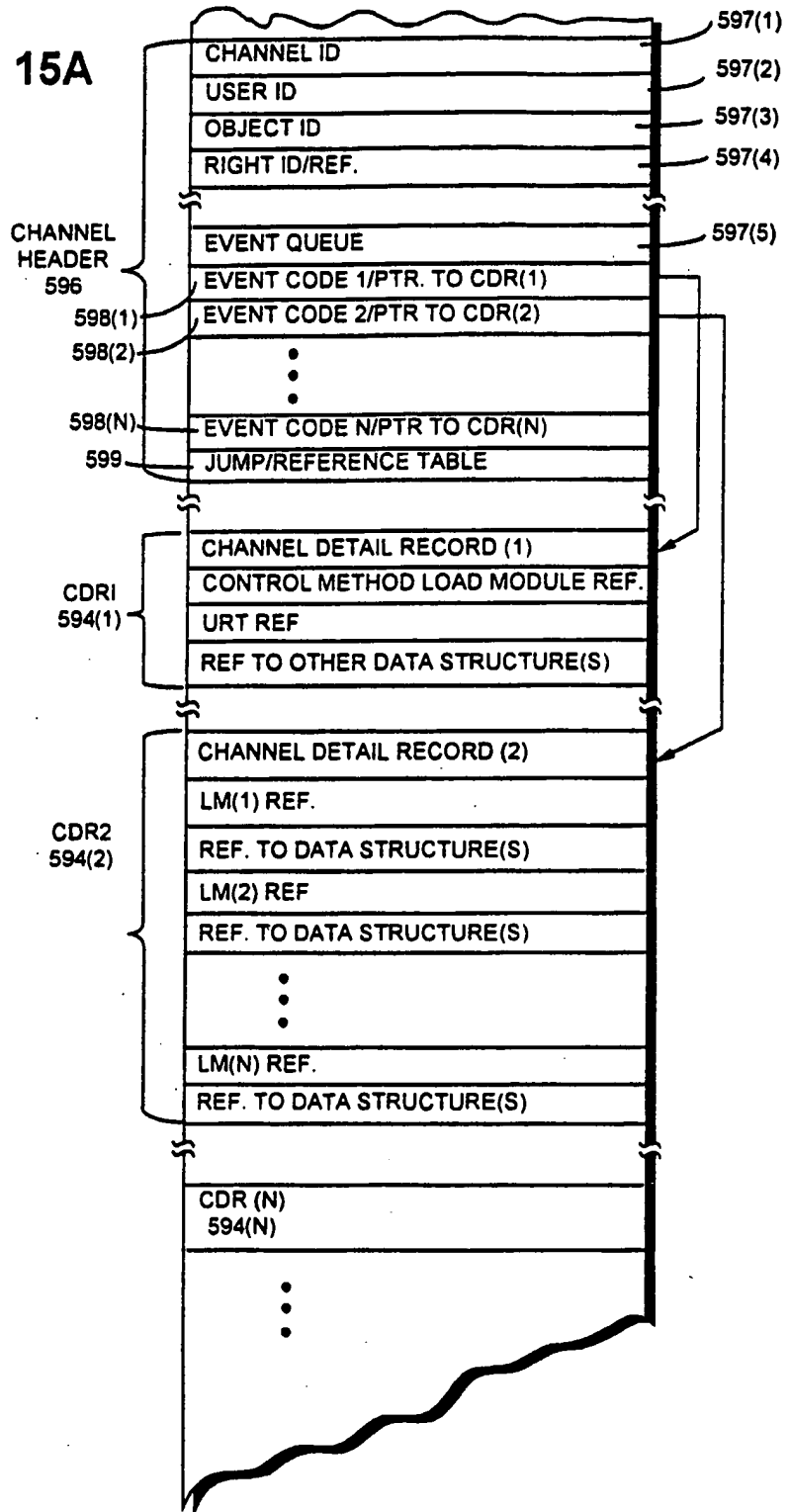


FIG. 15



26/146

FIG. 15A



27/146

FIG. 15B

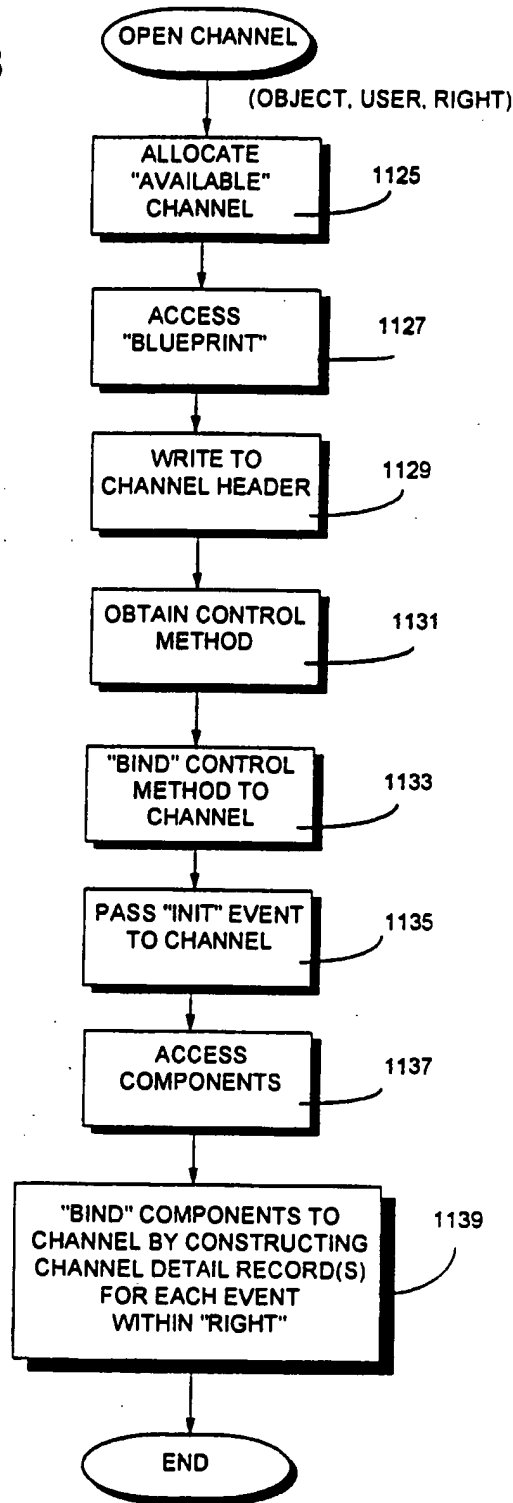
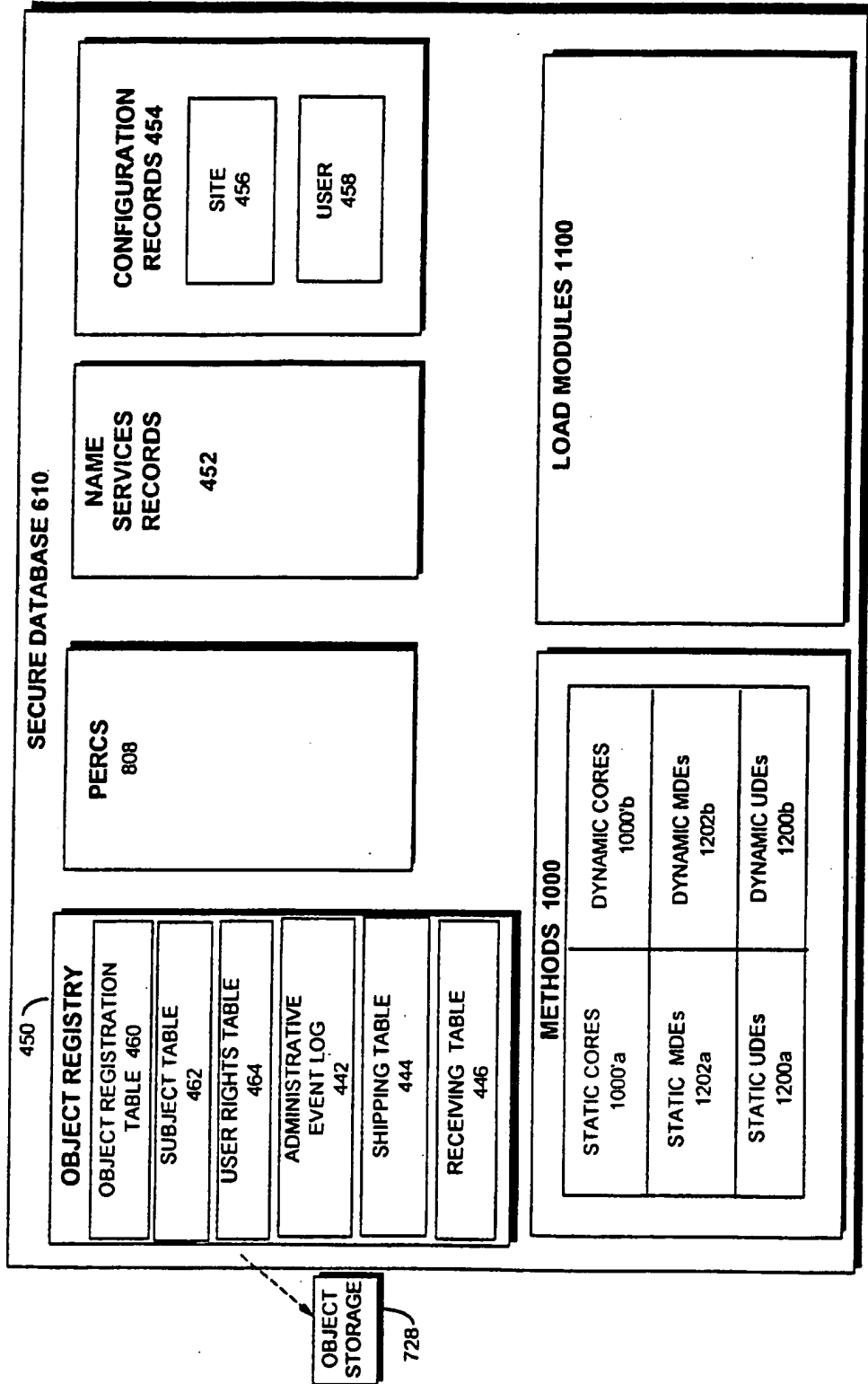
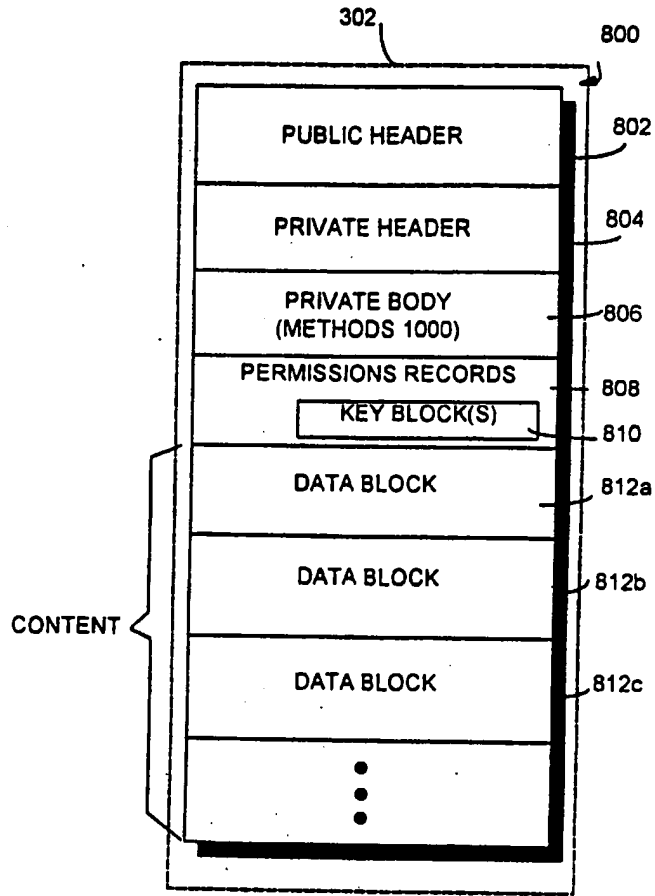


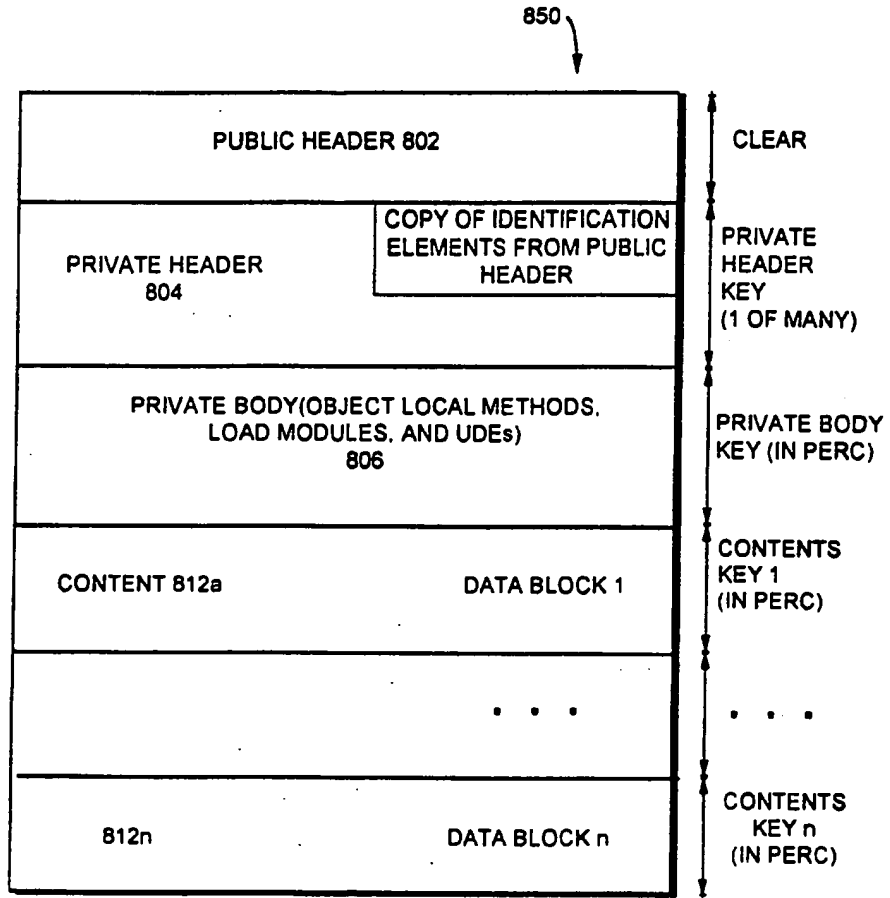
FIG. 16





LOGICAL OBJECT

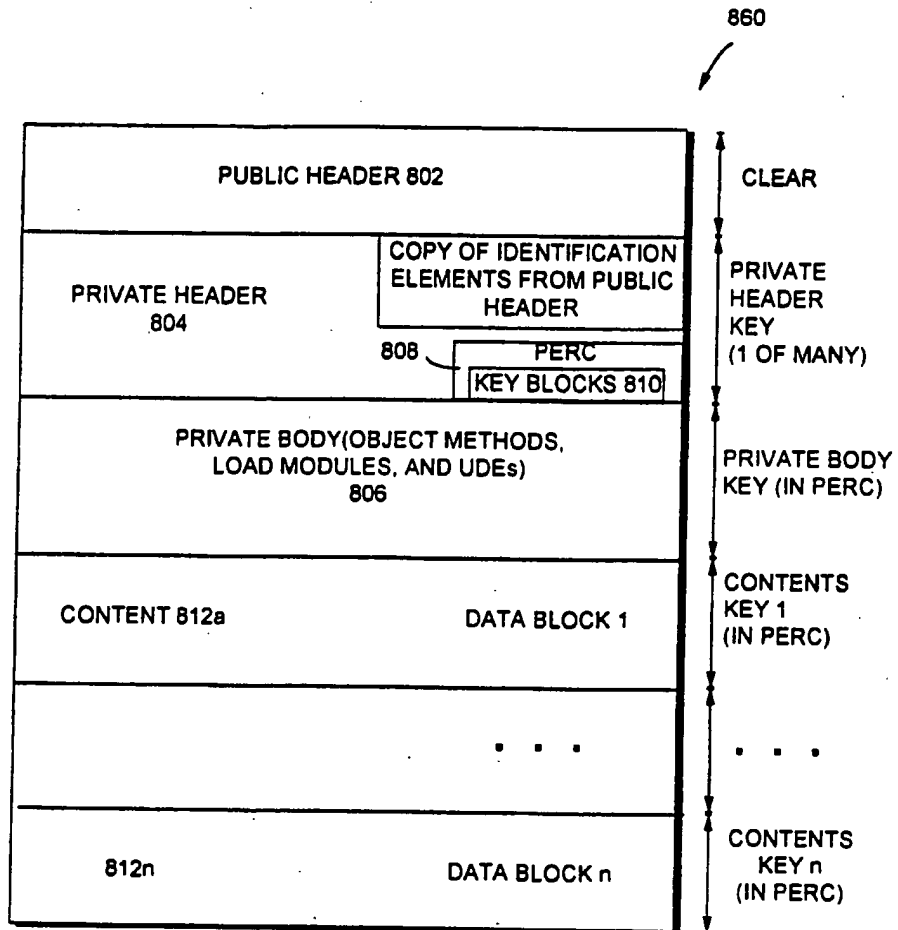
FIG. 17



STATIONARY OBJECT

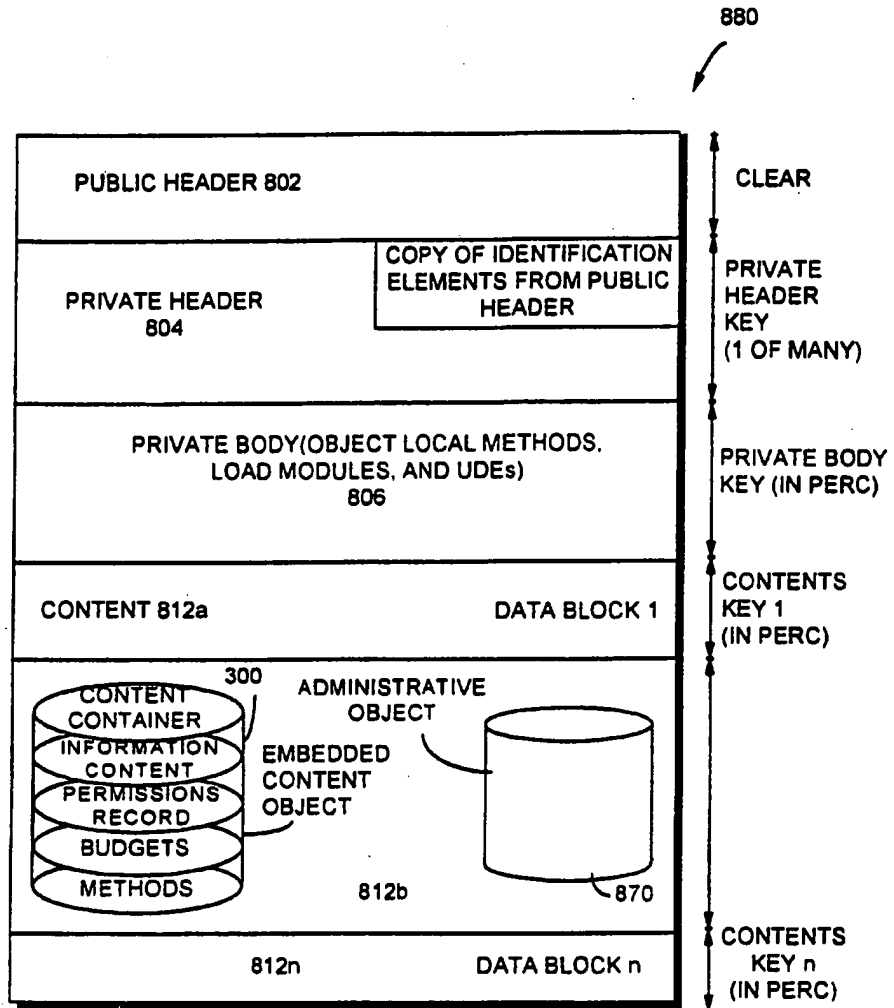
FIG. 18

SUBSTITUTE SHEET (RULE 26)



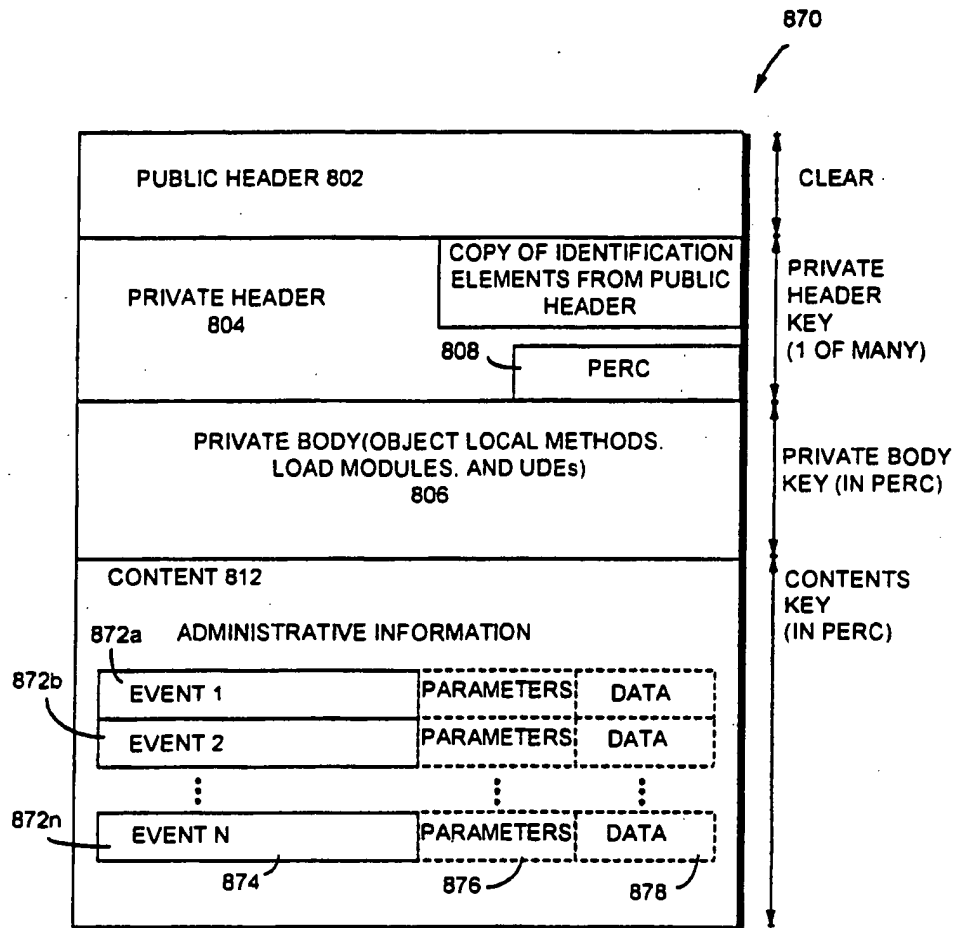
TRAVELING OBJECT

FIG. 19



CONTENT OBJECT

FIG. 20



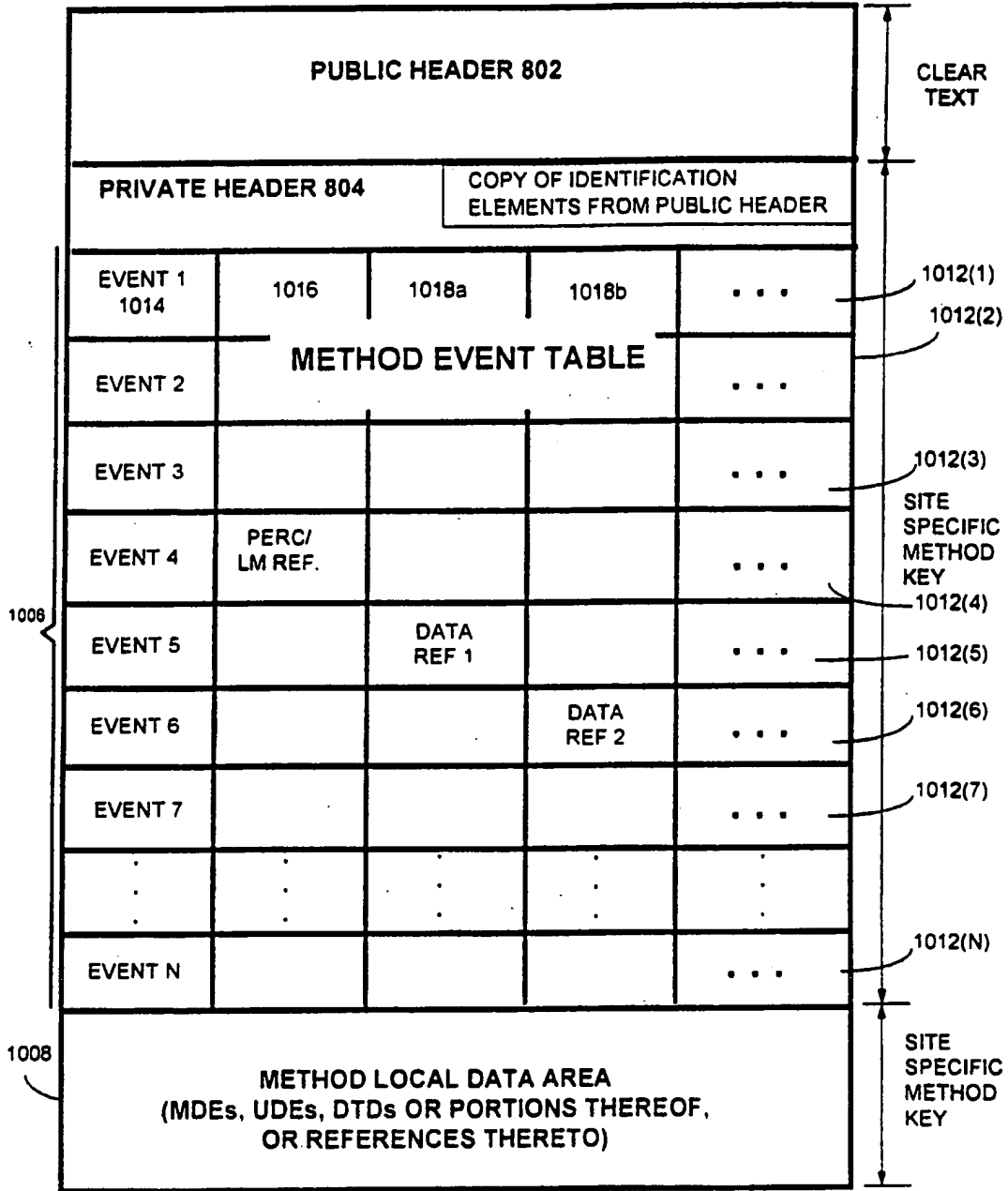
ADMINISTRATIVE OBJECT

FIG. 21

34/146

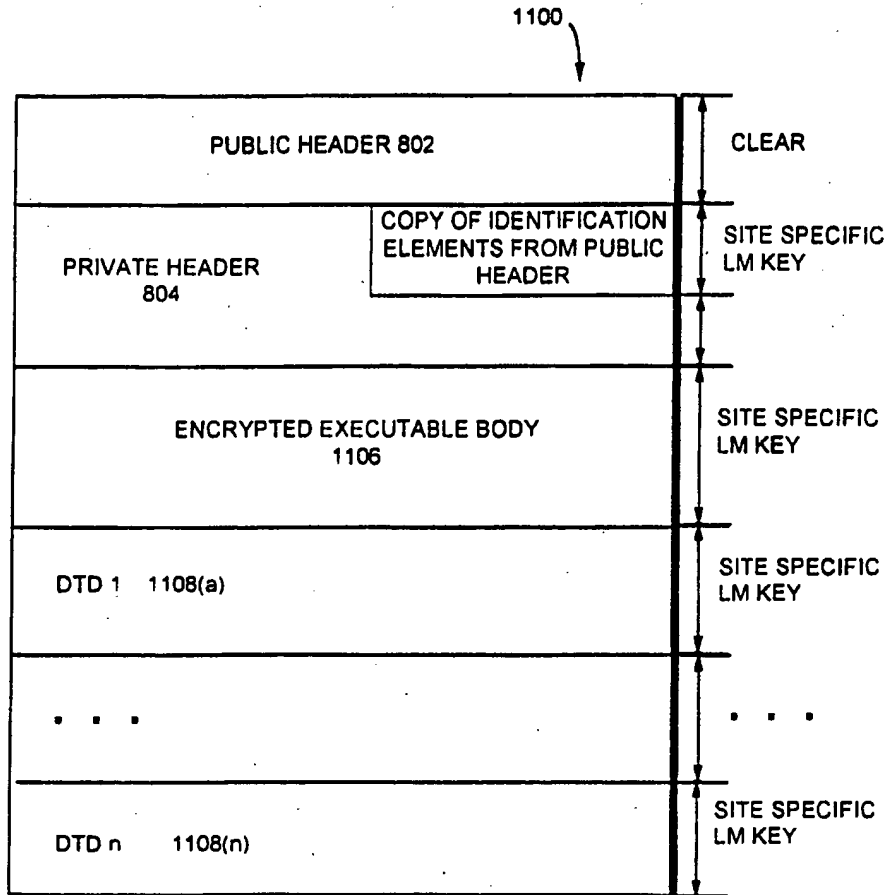
FIG. 22

1000'



METHOD "CORE"

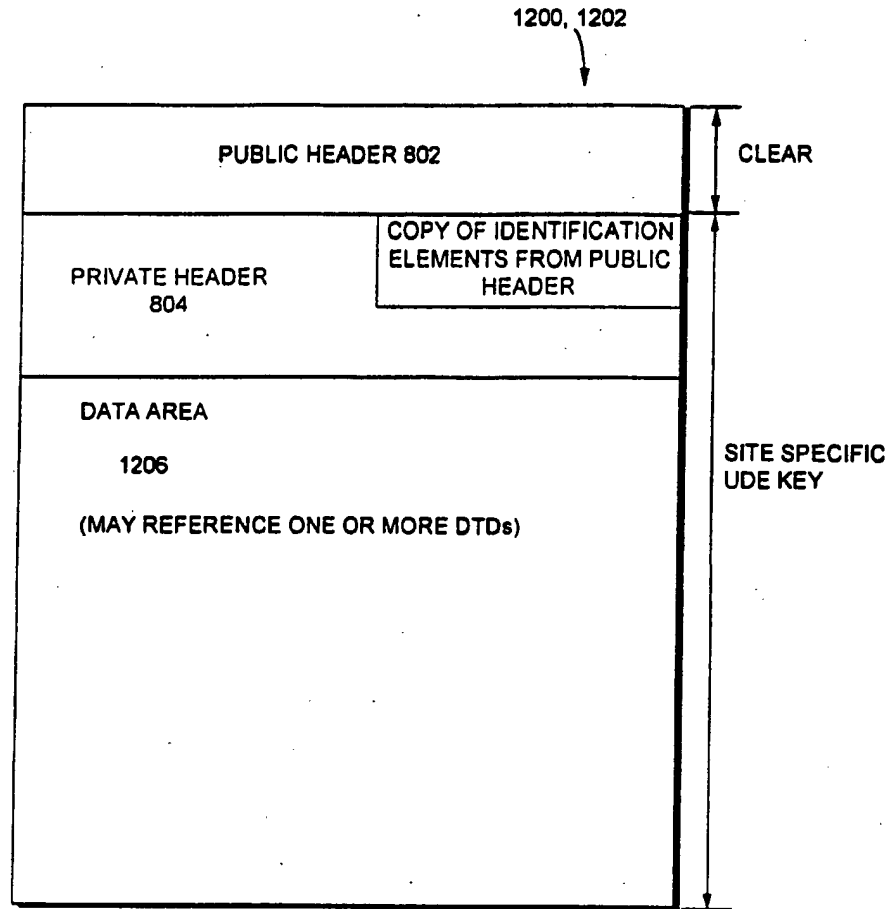
FIG. 23



LOAD MODULE

36/146

FIG. 24



UDE (MDE)

SUBSTITUTE SHEET (RULE 26)

37/146

FIG. 25A

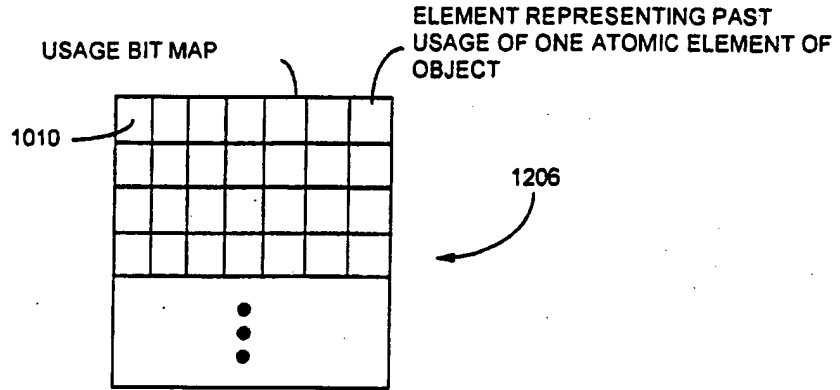


FIG. 25B

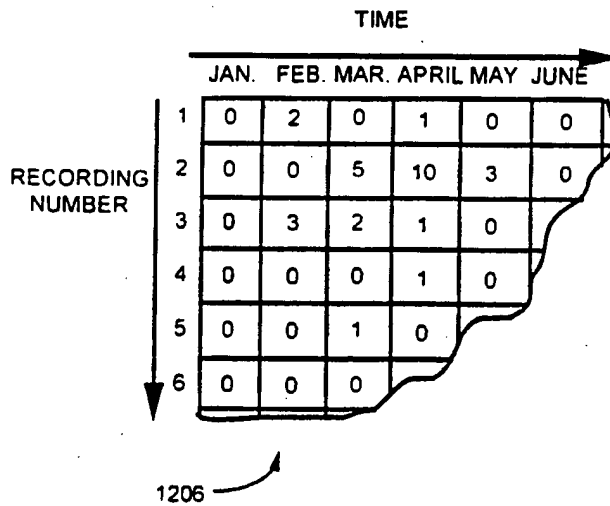


FIG. 25C

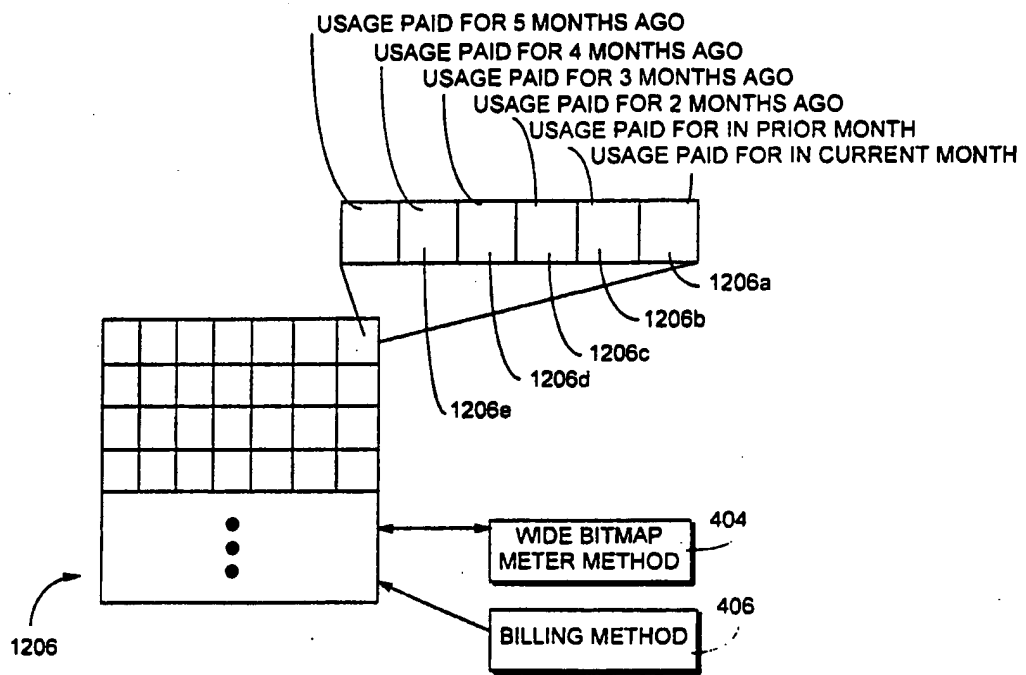
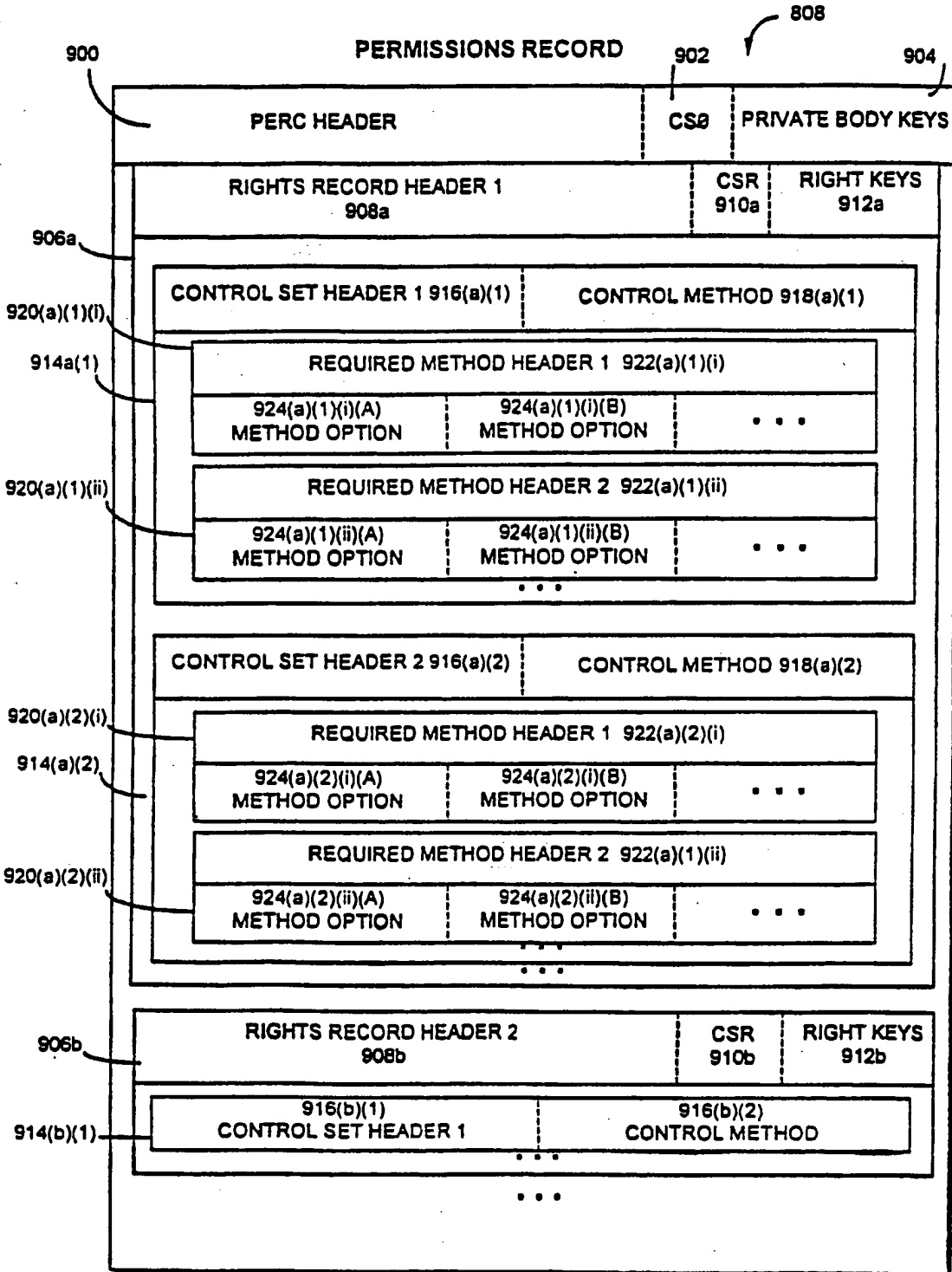
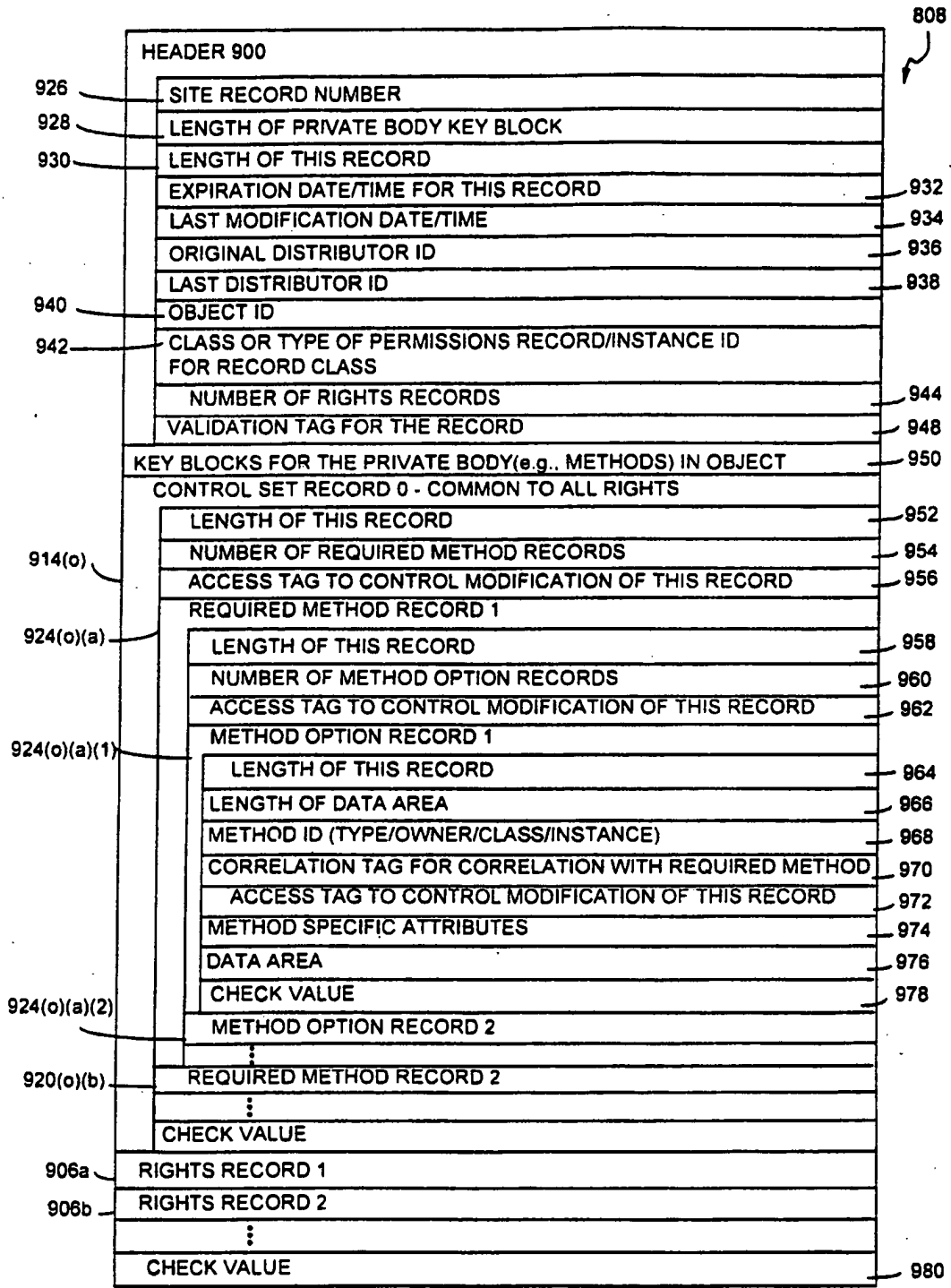


FIG. 26



40/146

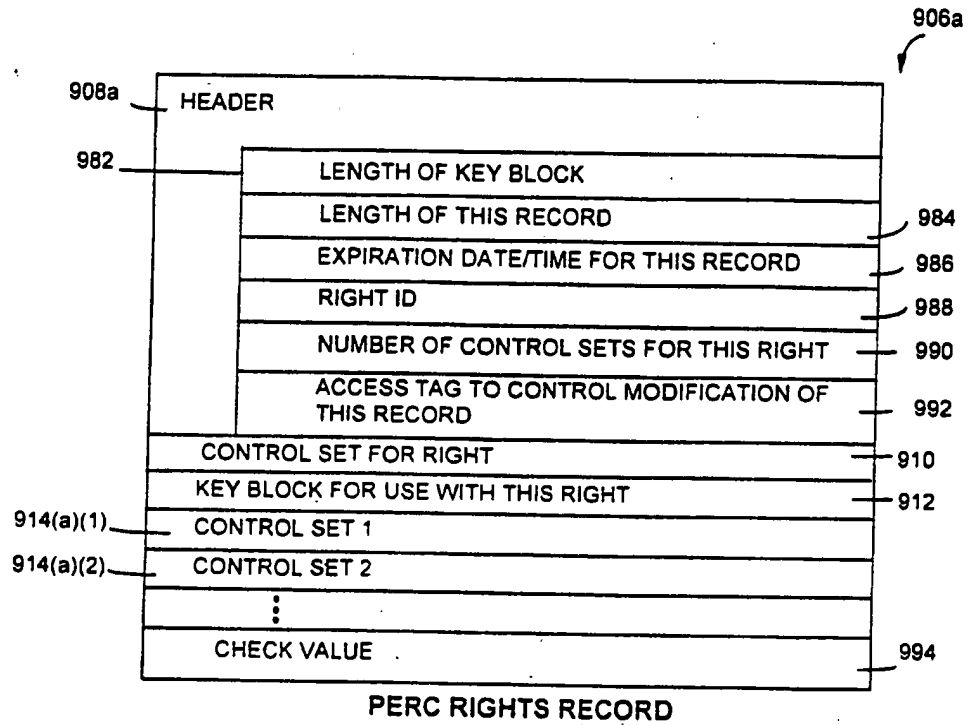
FIG. 26A



PERC

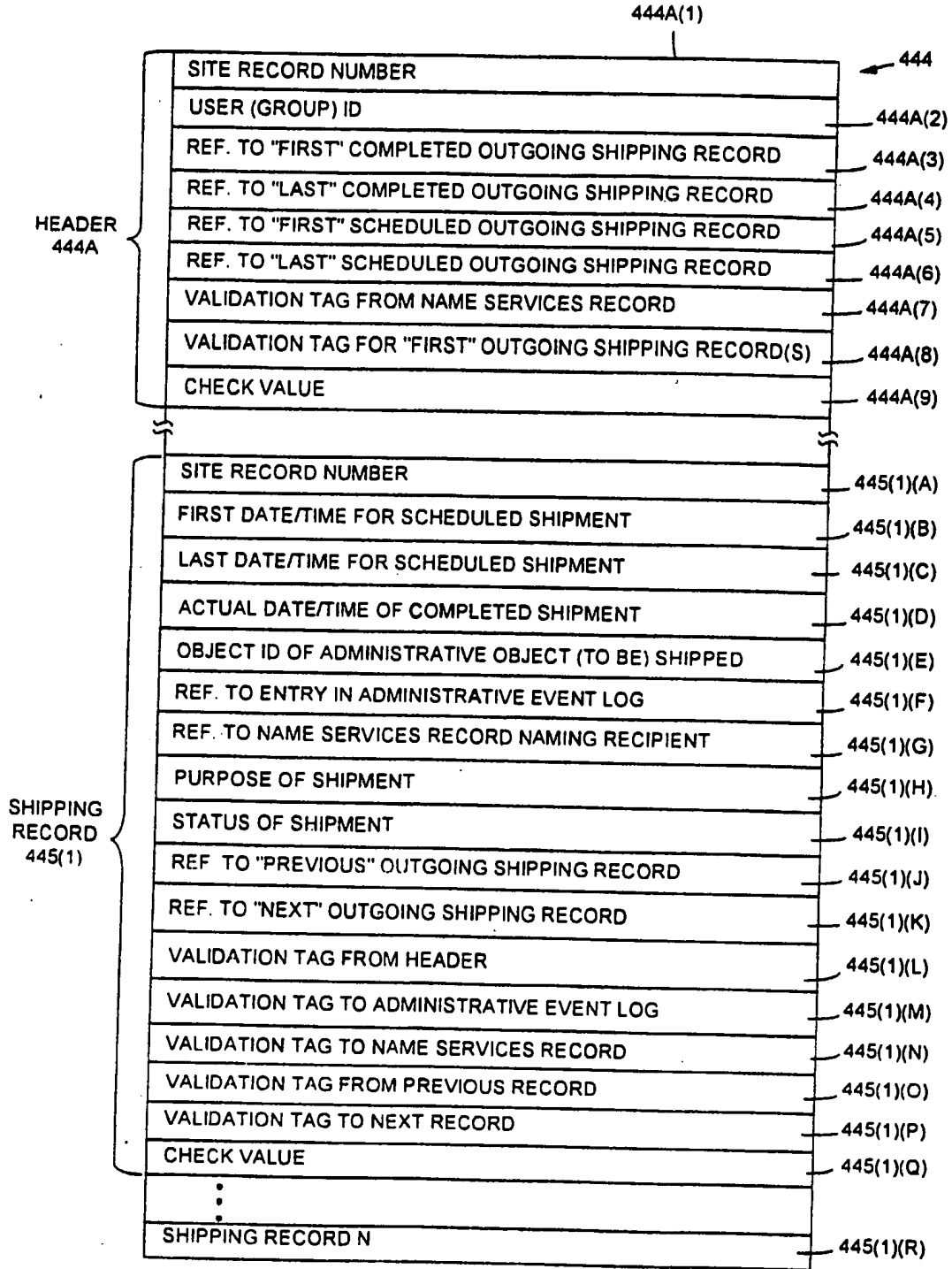
SUBSTITUTE SHEET (RULE 26)

FIG. 26B



42/146

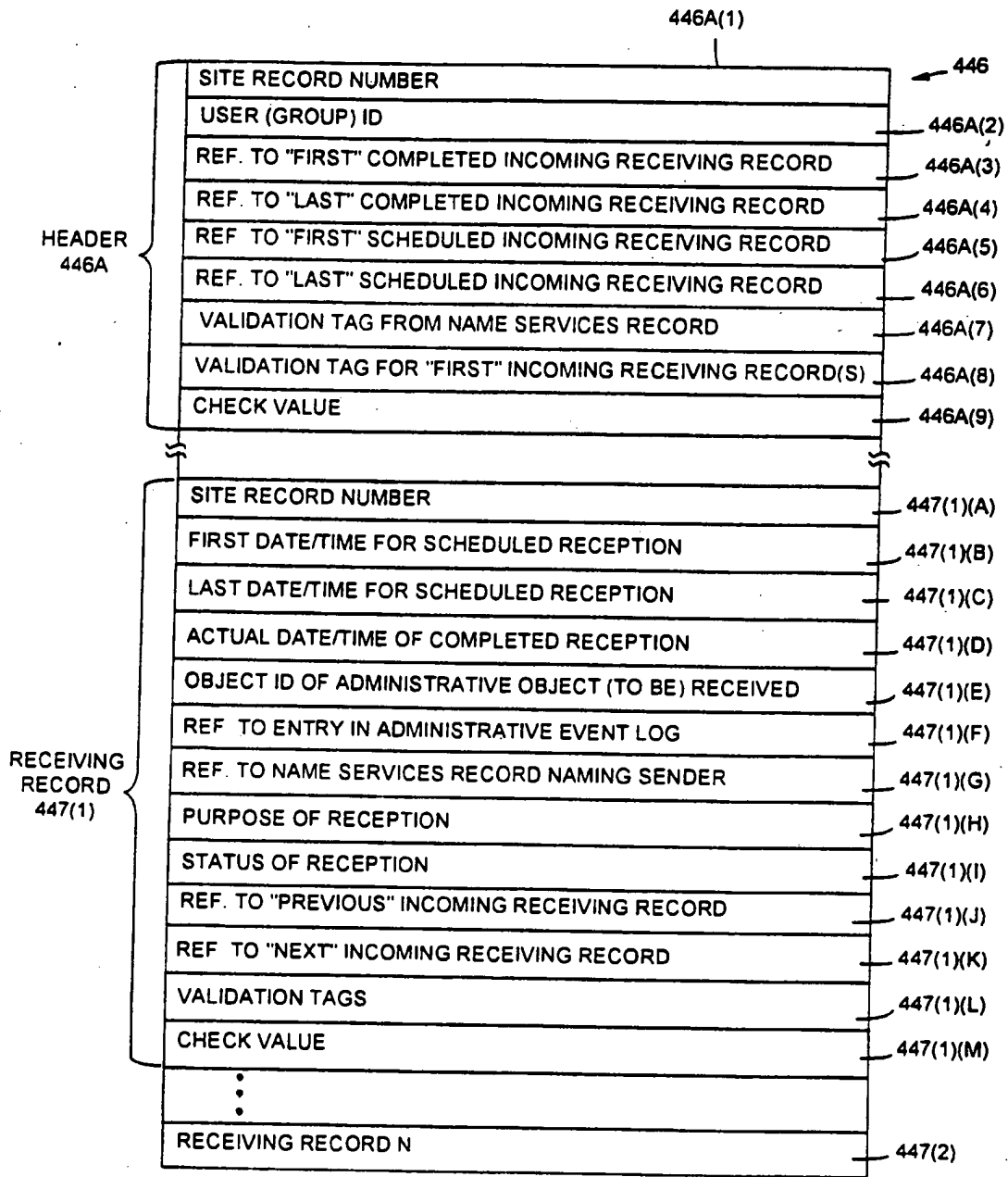
FIG. 27
SHIPPING TABLE



SUBSTITUTE SHEET (RULE 26)

43/146

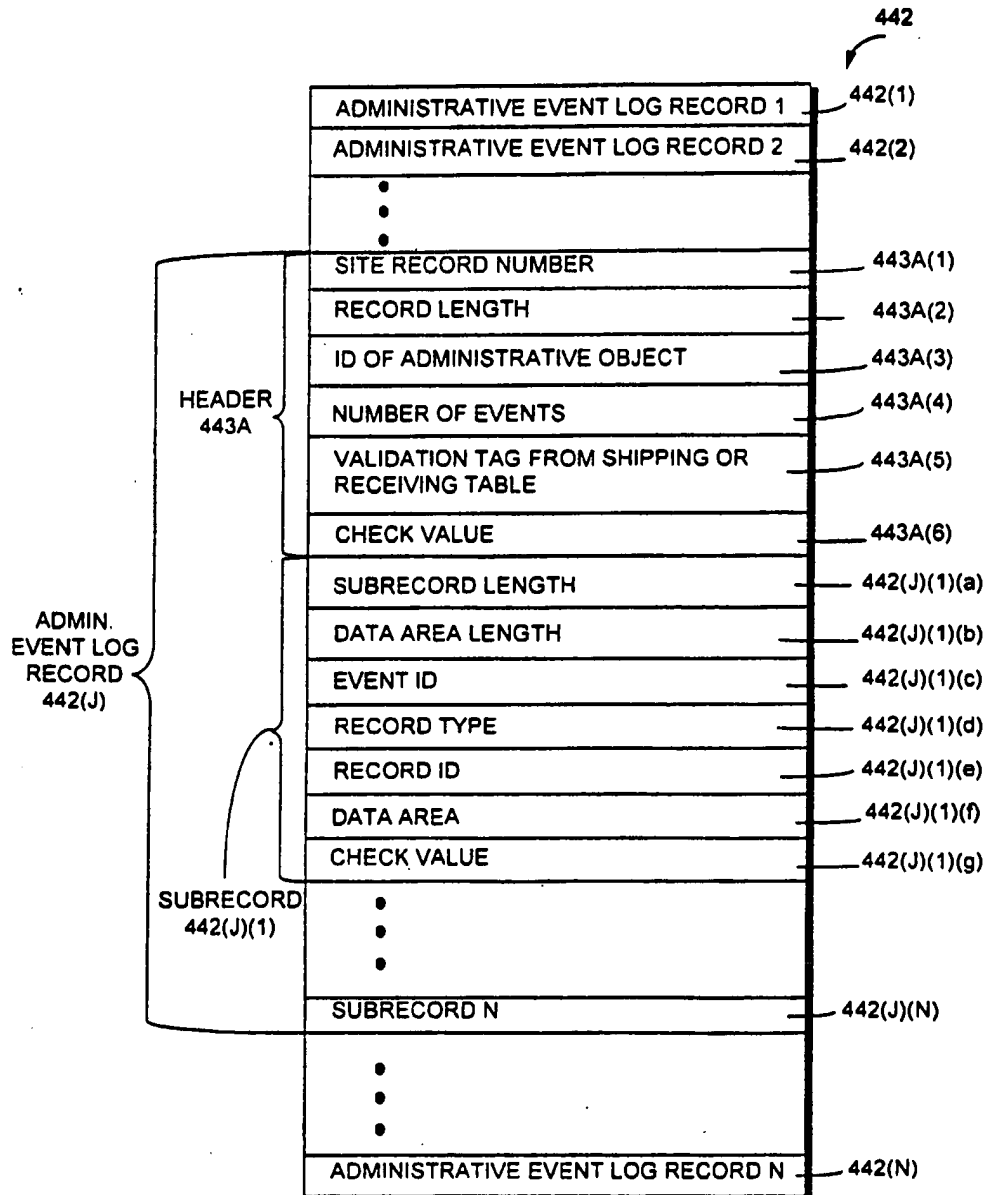
FIG. 28
RECEIVING TABLE



SUBSTITUTE SHEET (RULE 26)

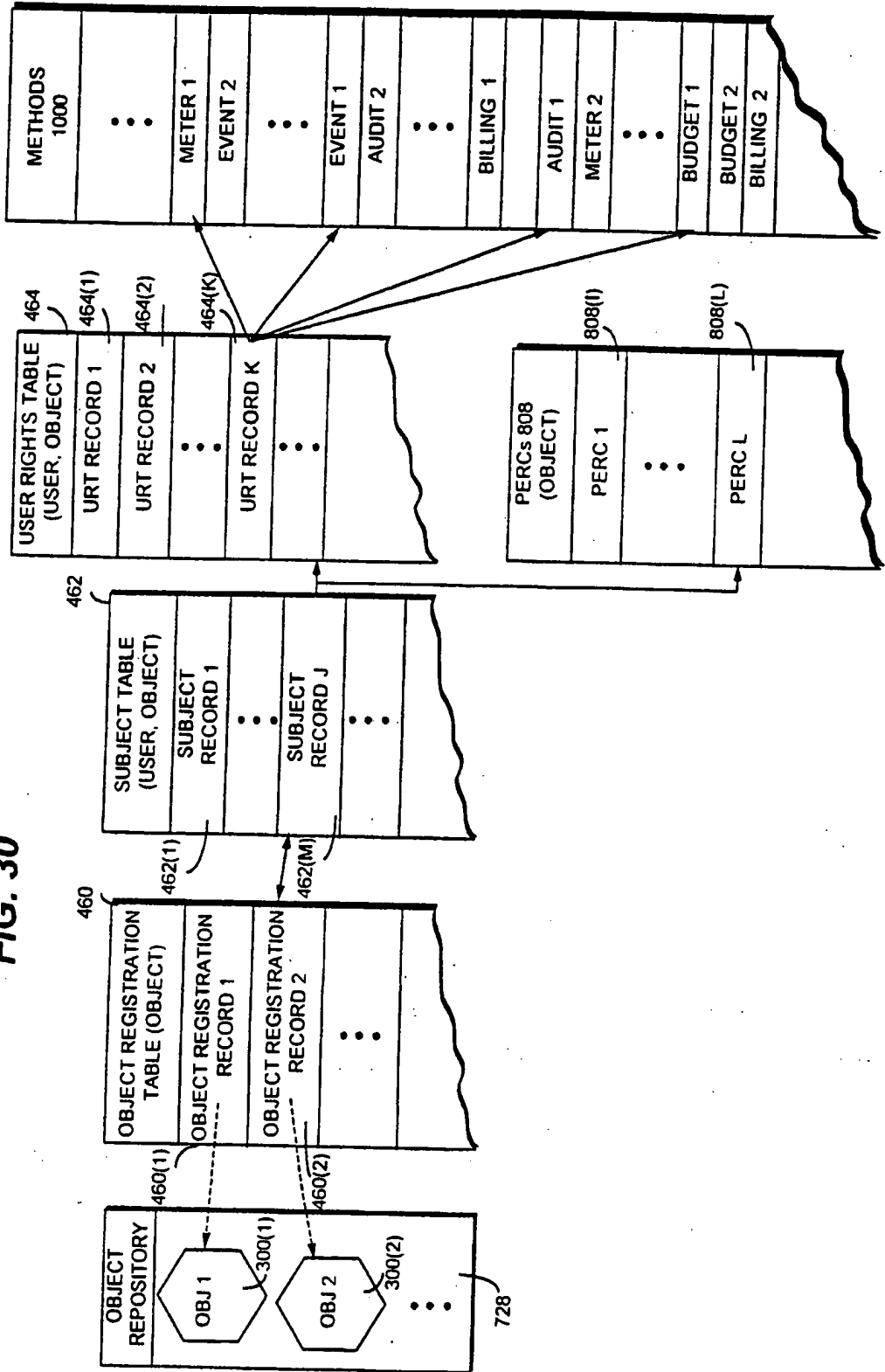
44/146

FIG. 29
ADMINISTRATIVE EVENT LOG



SUBSTITUTE SHEET (RULE 26)

FIG. 30



46/146

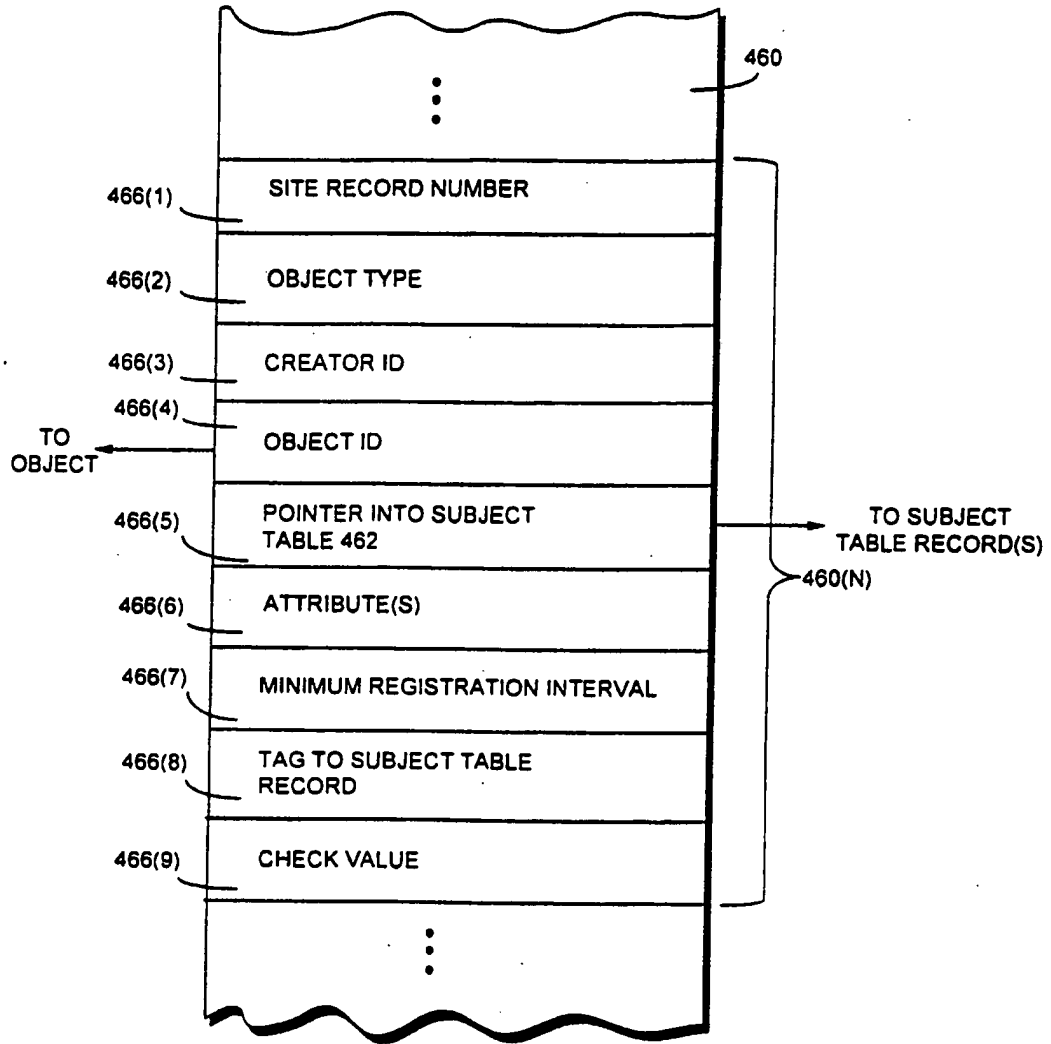


FIG. 31
OBJECT REGISTRATION TABLE

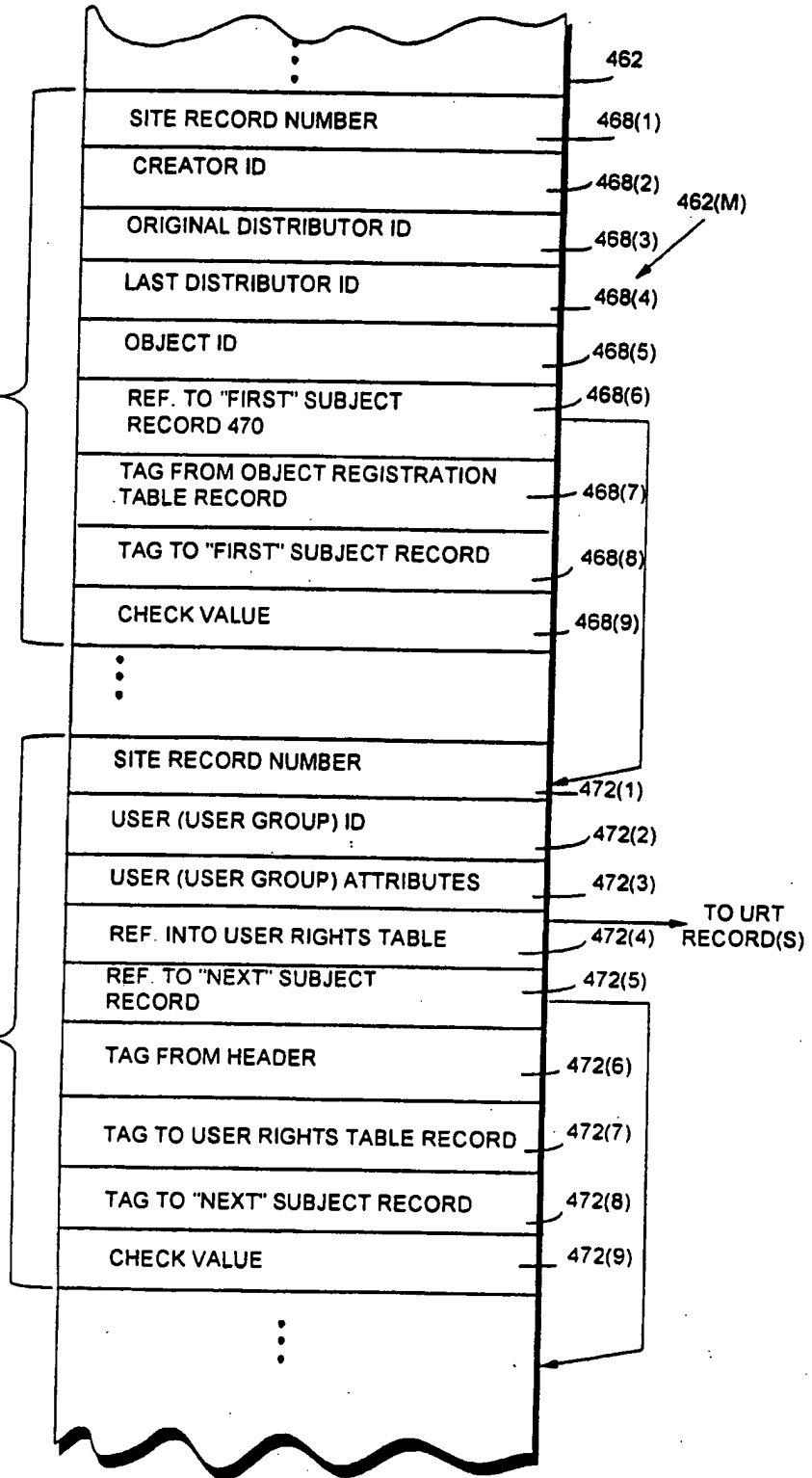
47/146

FIG. 32

SUBJECT TABLE

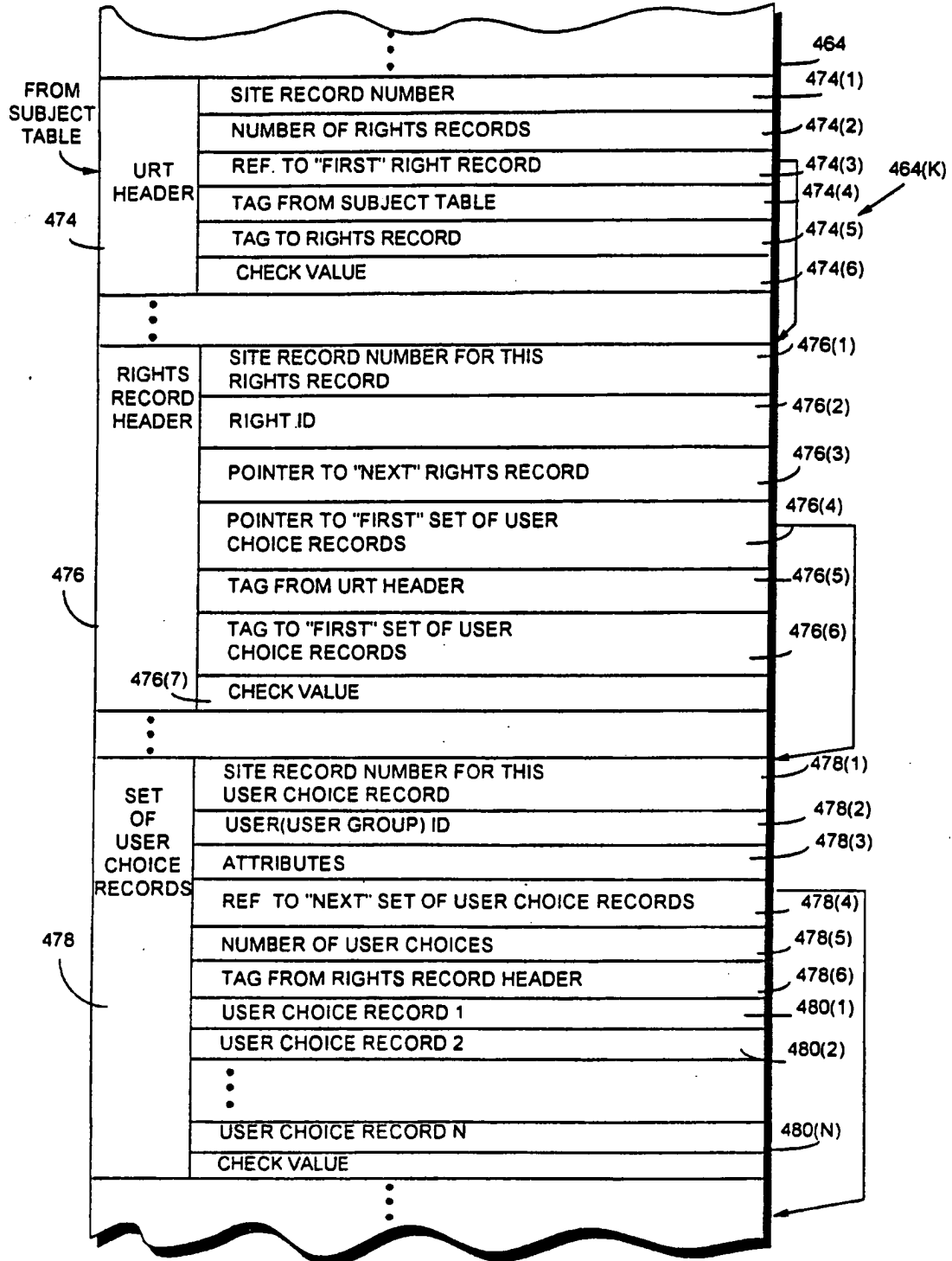
"HEADER"
468

SUBJECT RECORD
470(1)



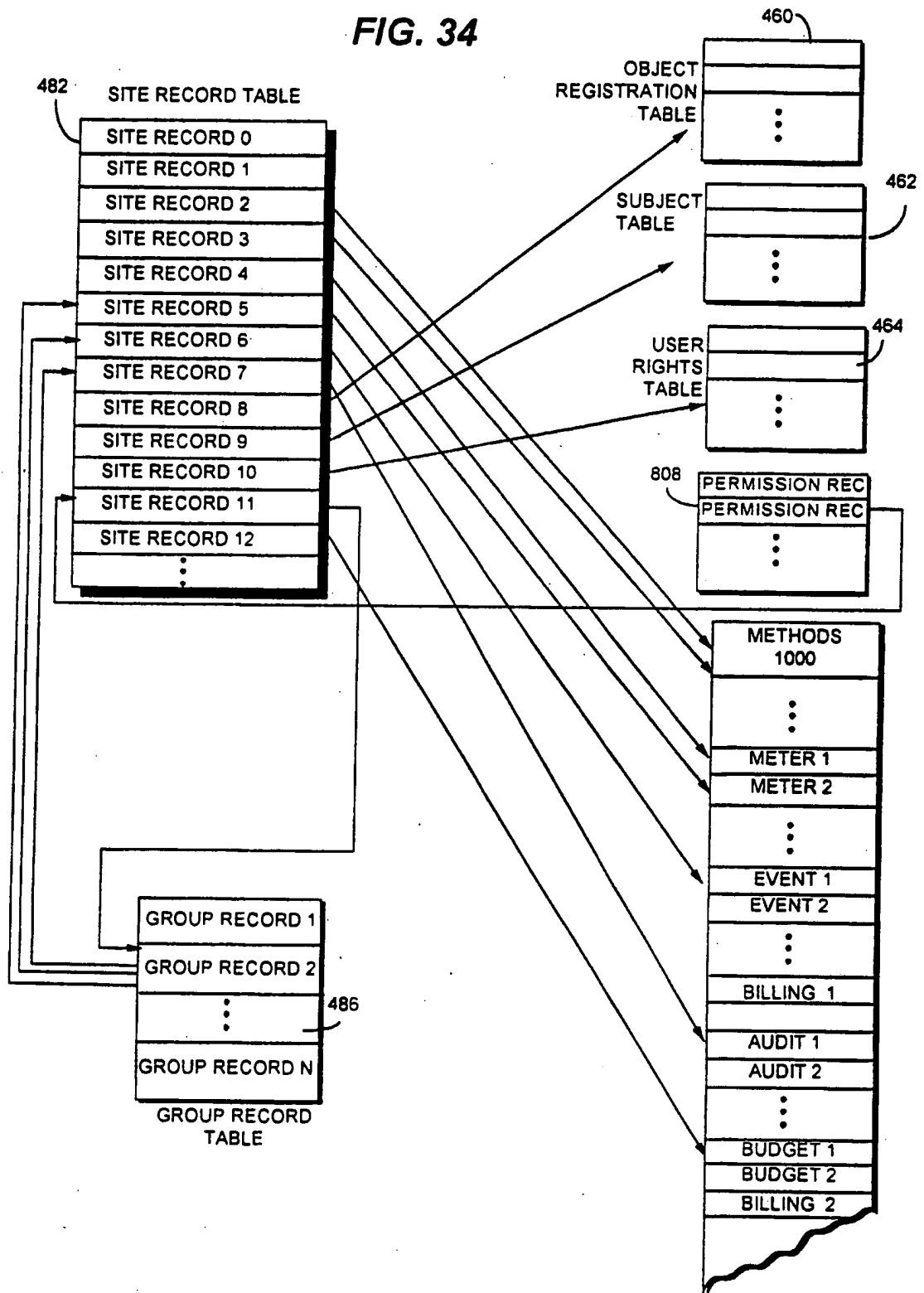
48/146

FIG. 33 USER RIGHTS TABLE



49/146

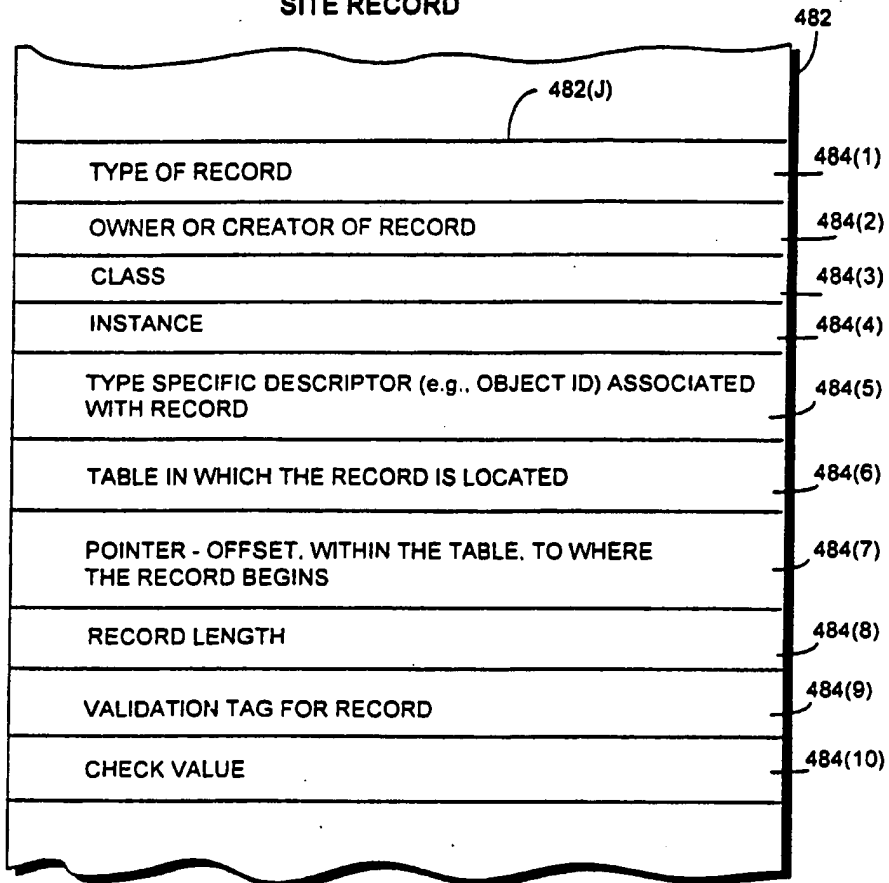
FIG. 34



50/146

FIG. 34A

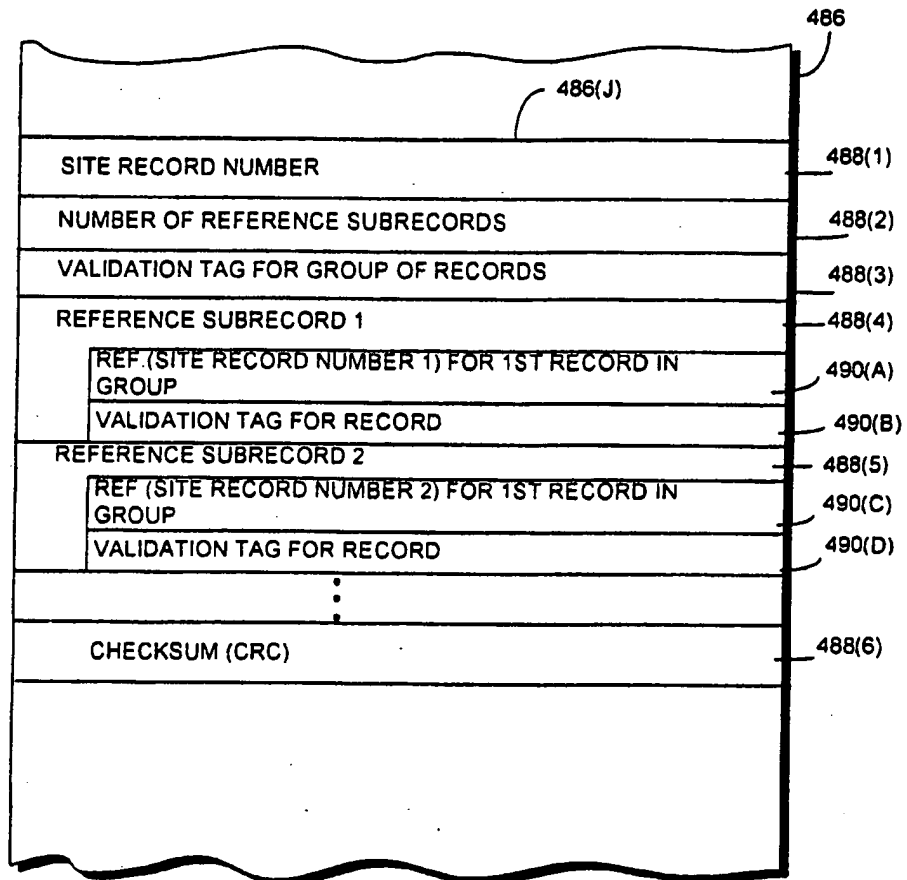
SITE RECORD



51/146

FIG. 34B

GROUP RECORD



52/146

FIG. 35

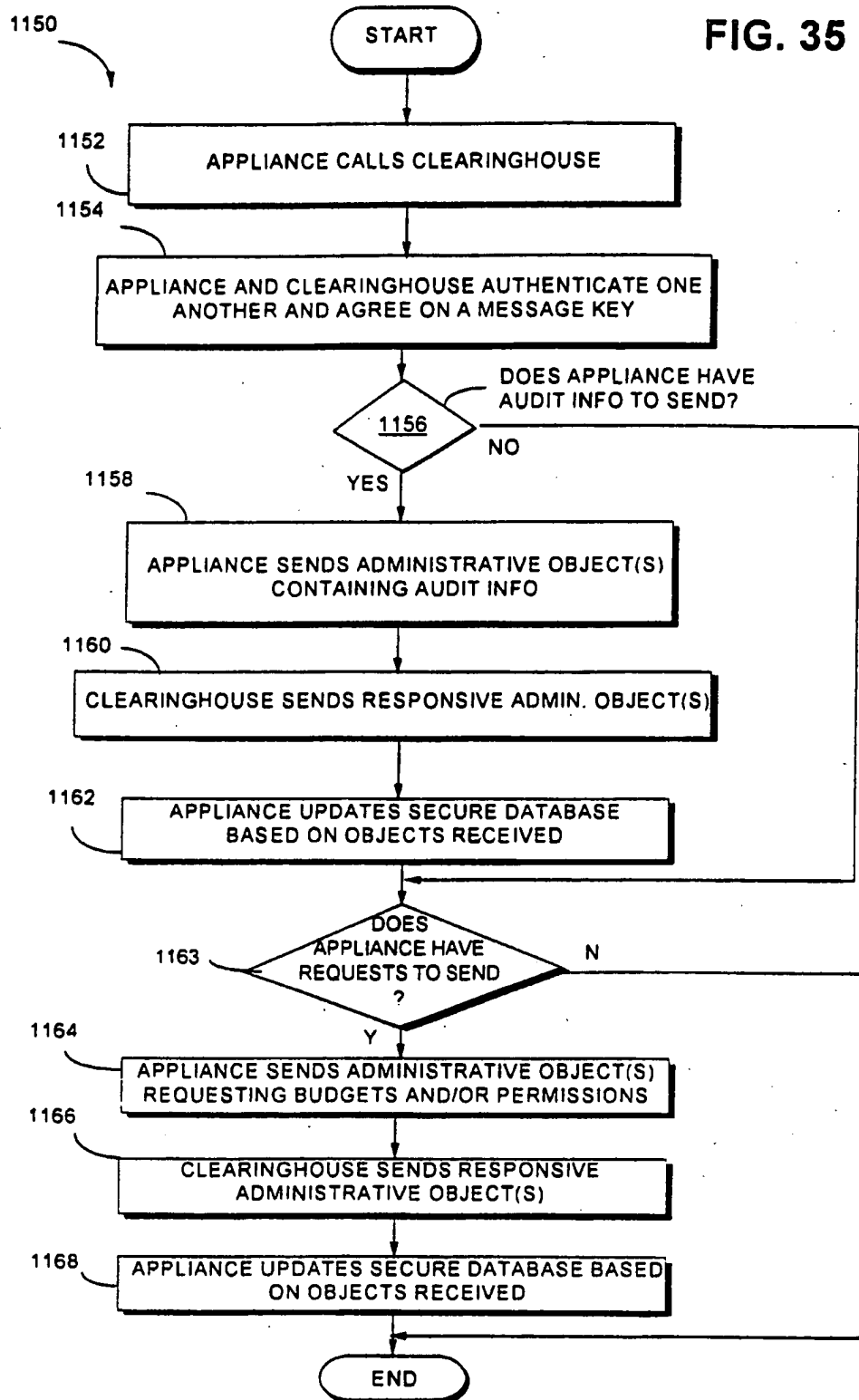


FIG. 36

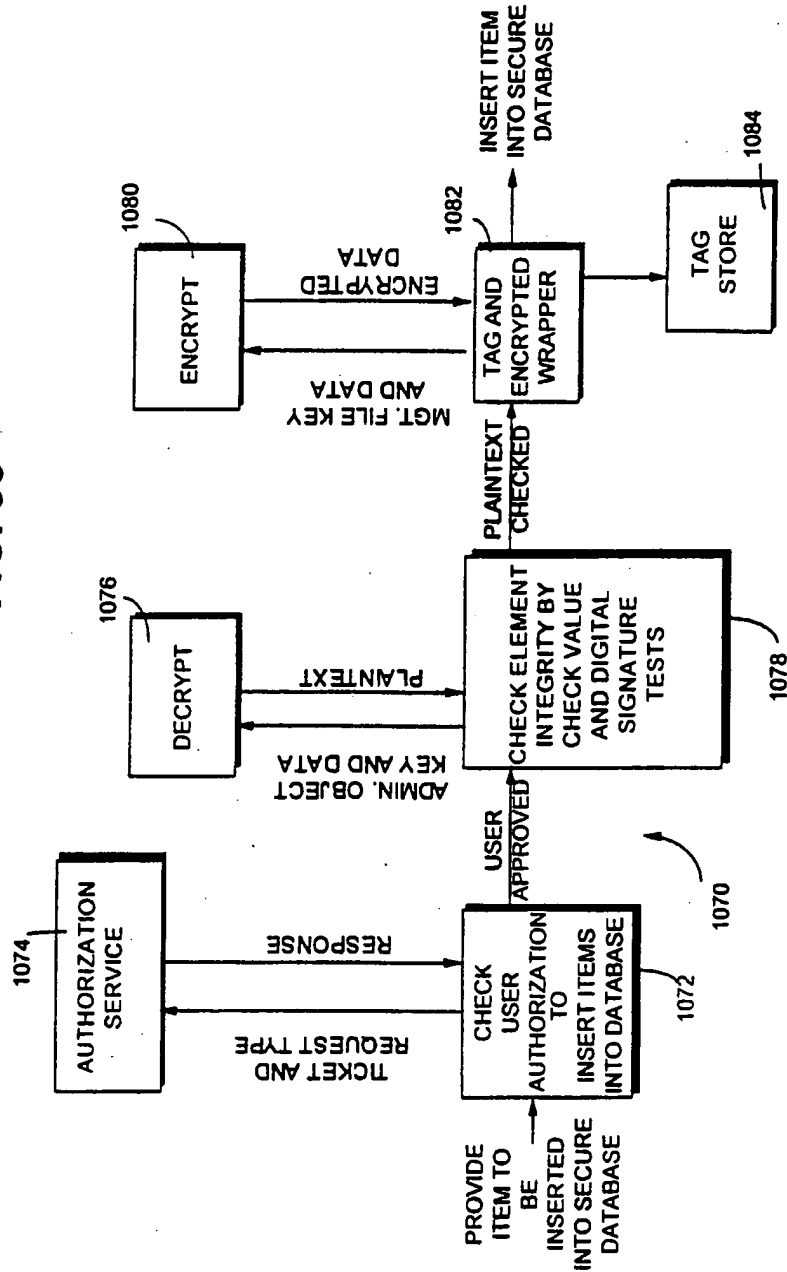
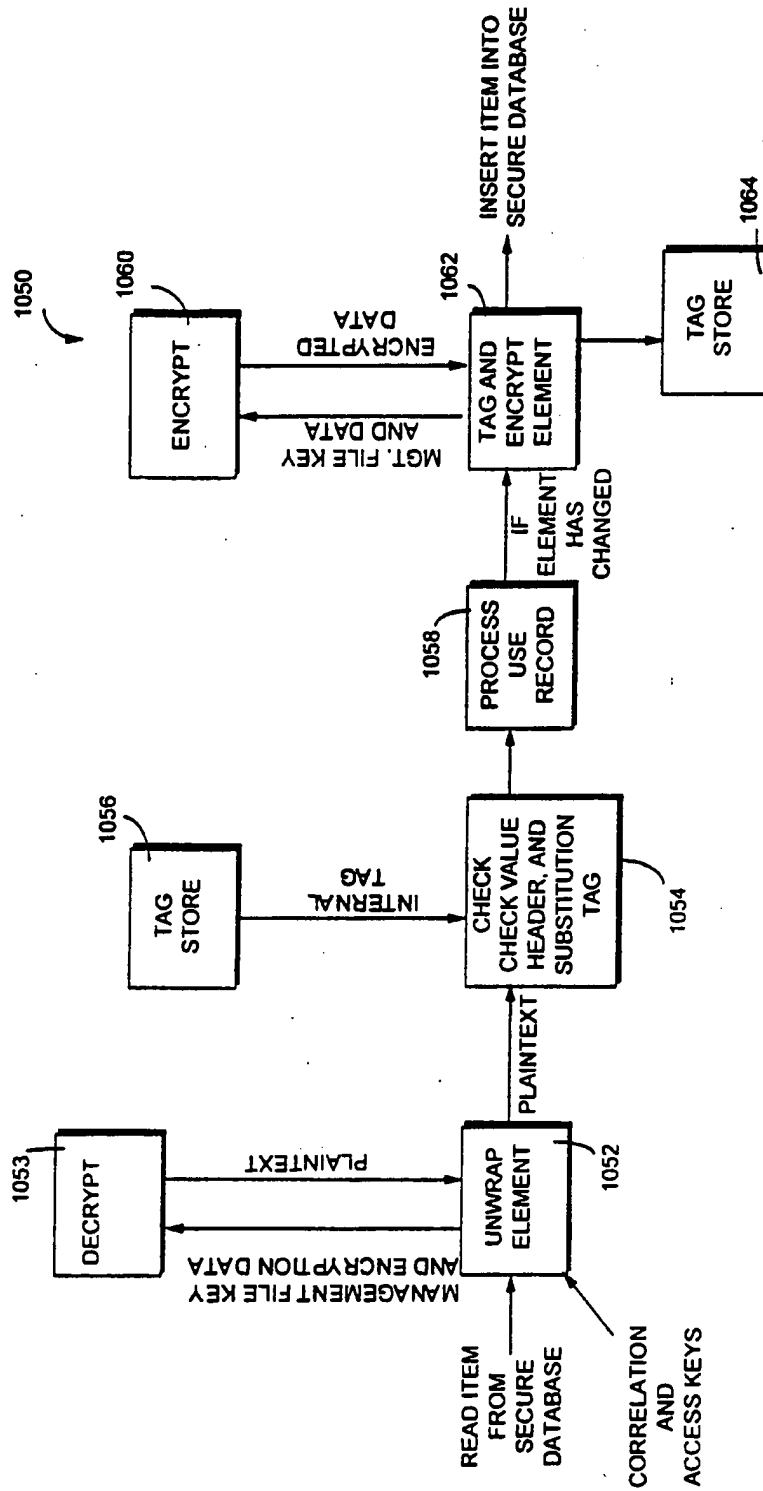
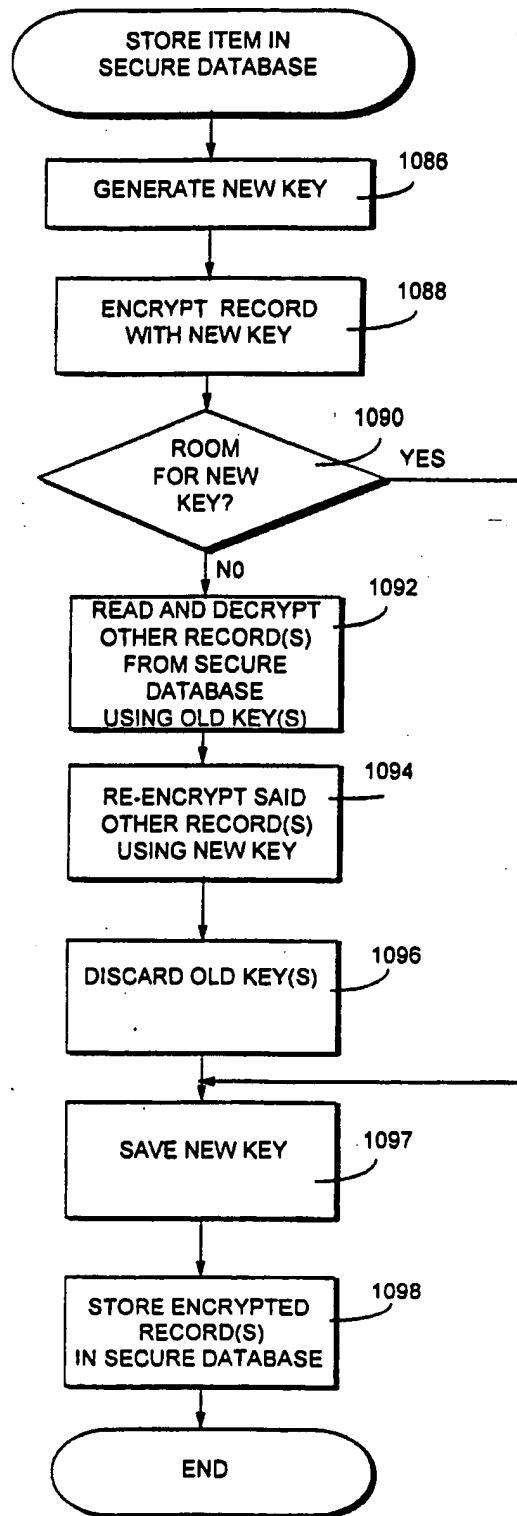


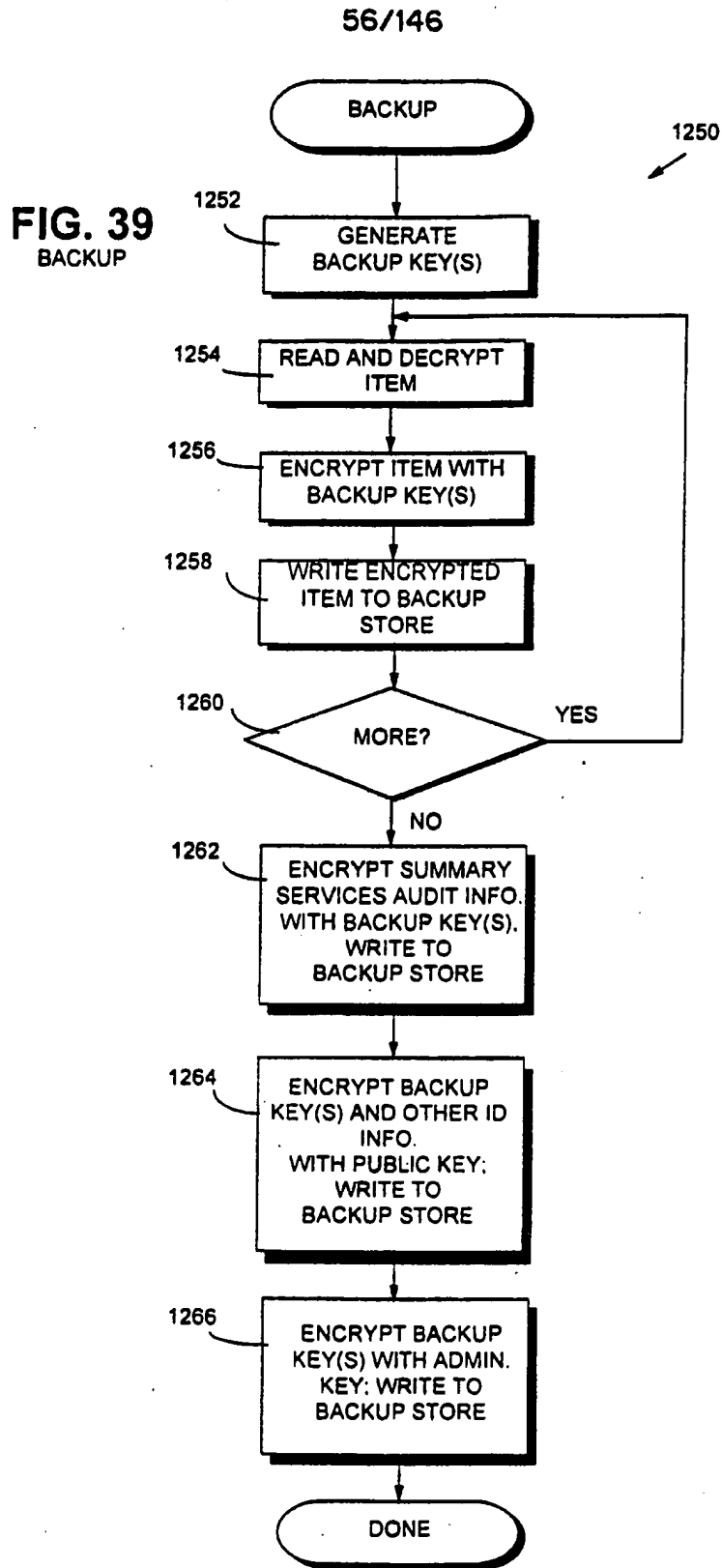
FIG. 37



55/146

FIG. 38

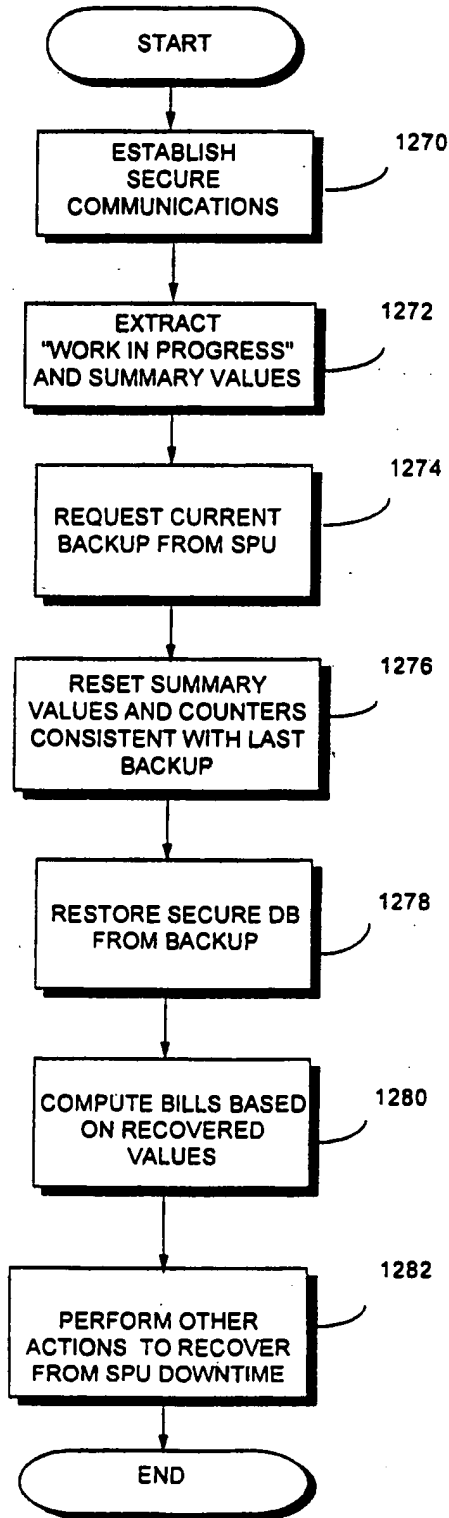




57/146

FIG. 40
RECOVER SECURE DATABASE

1268



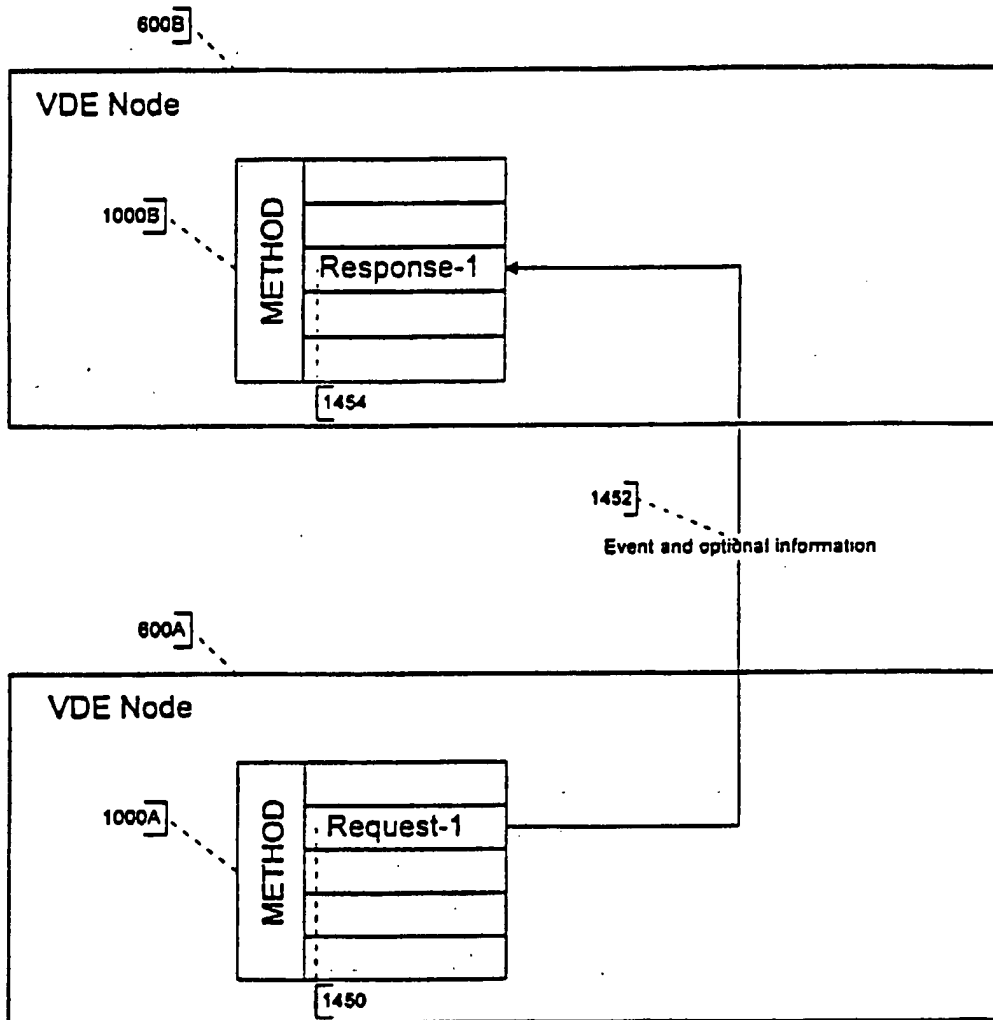


Figure 41a

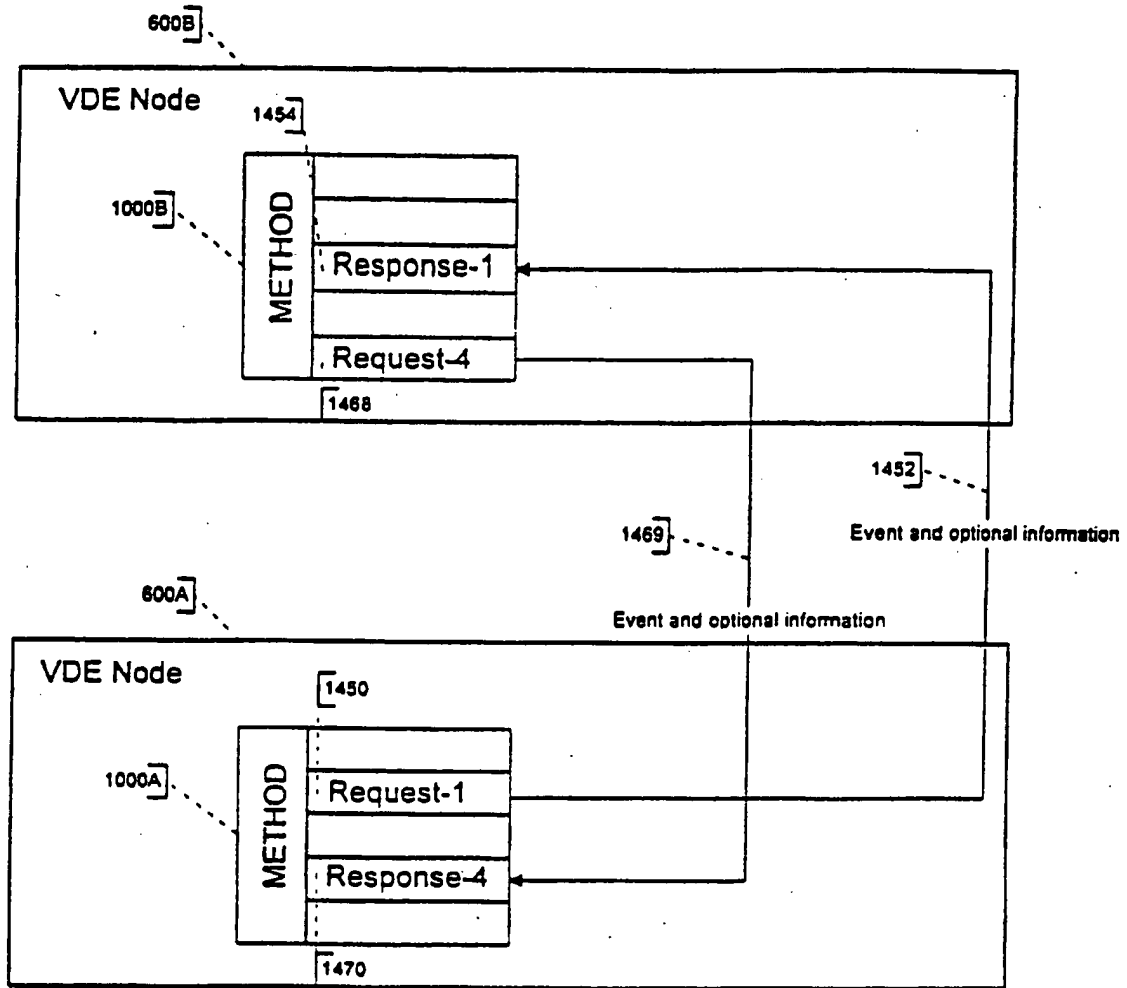


Figure 41b

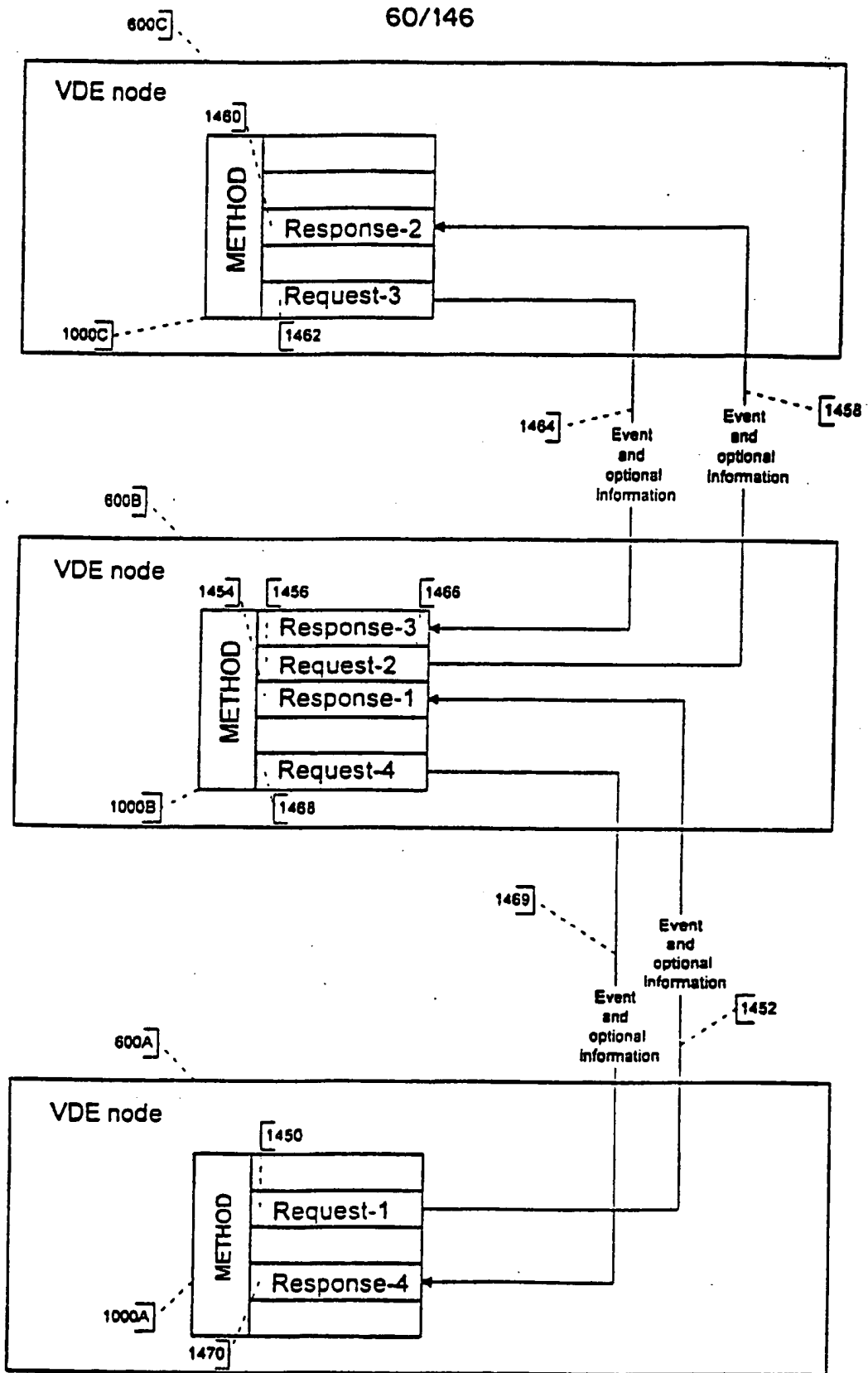


Figure 41c

61/146

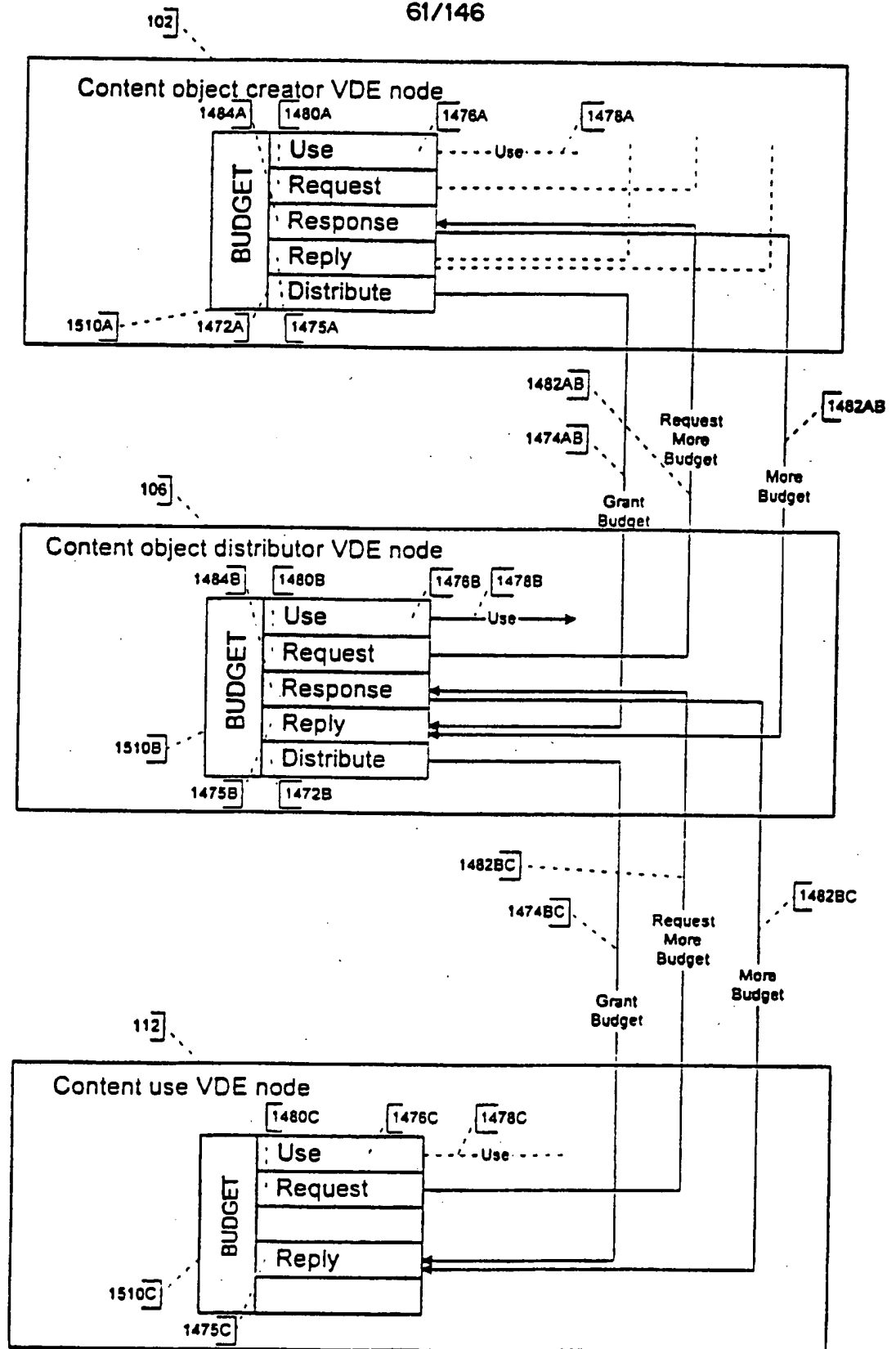


Figure 41d

SUBSTITUTE SHEET (RULE 26)

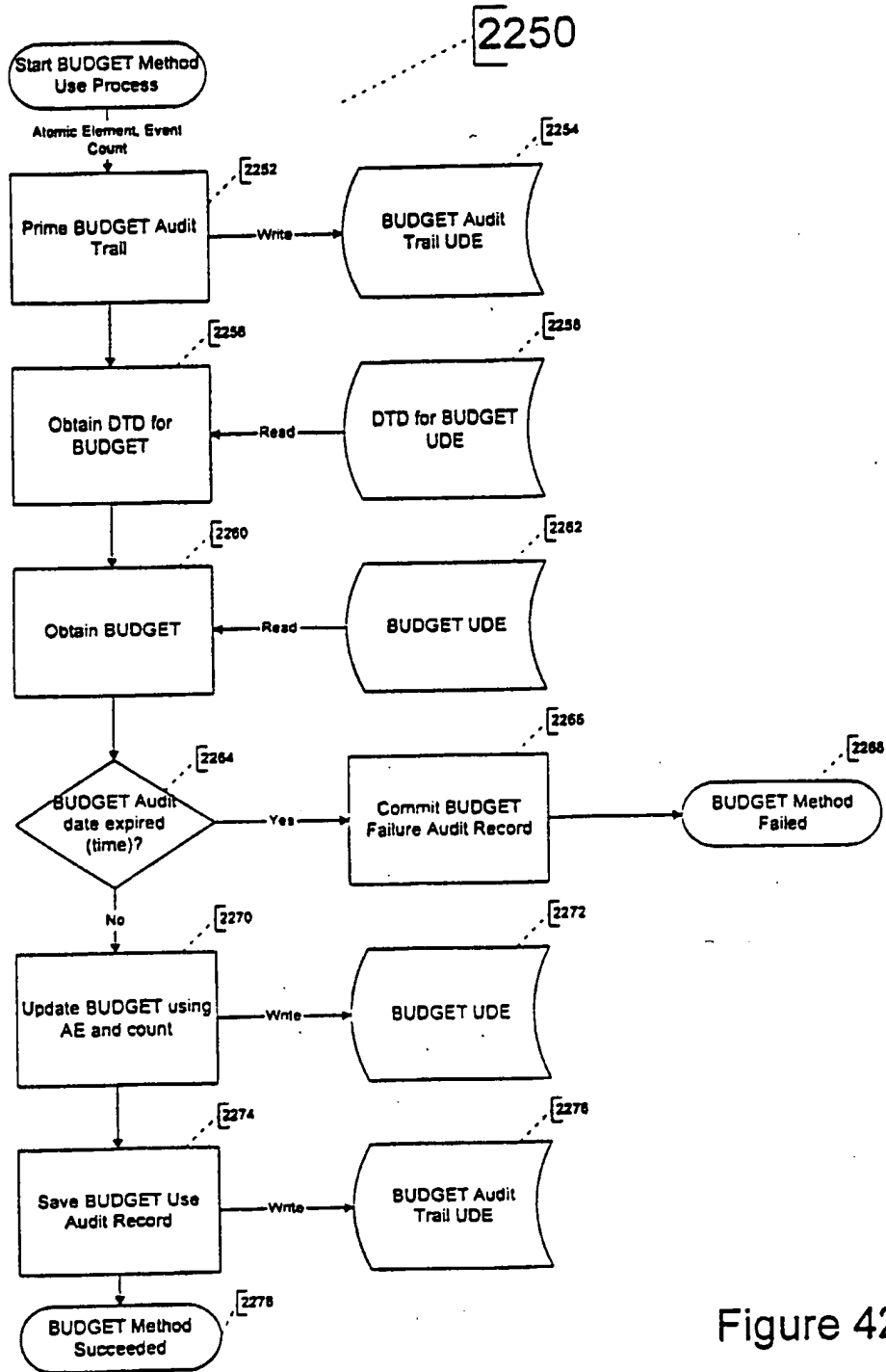


Figure 42a

63/146

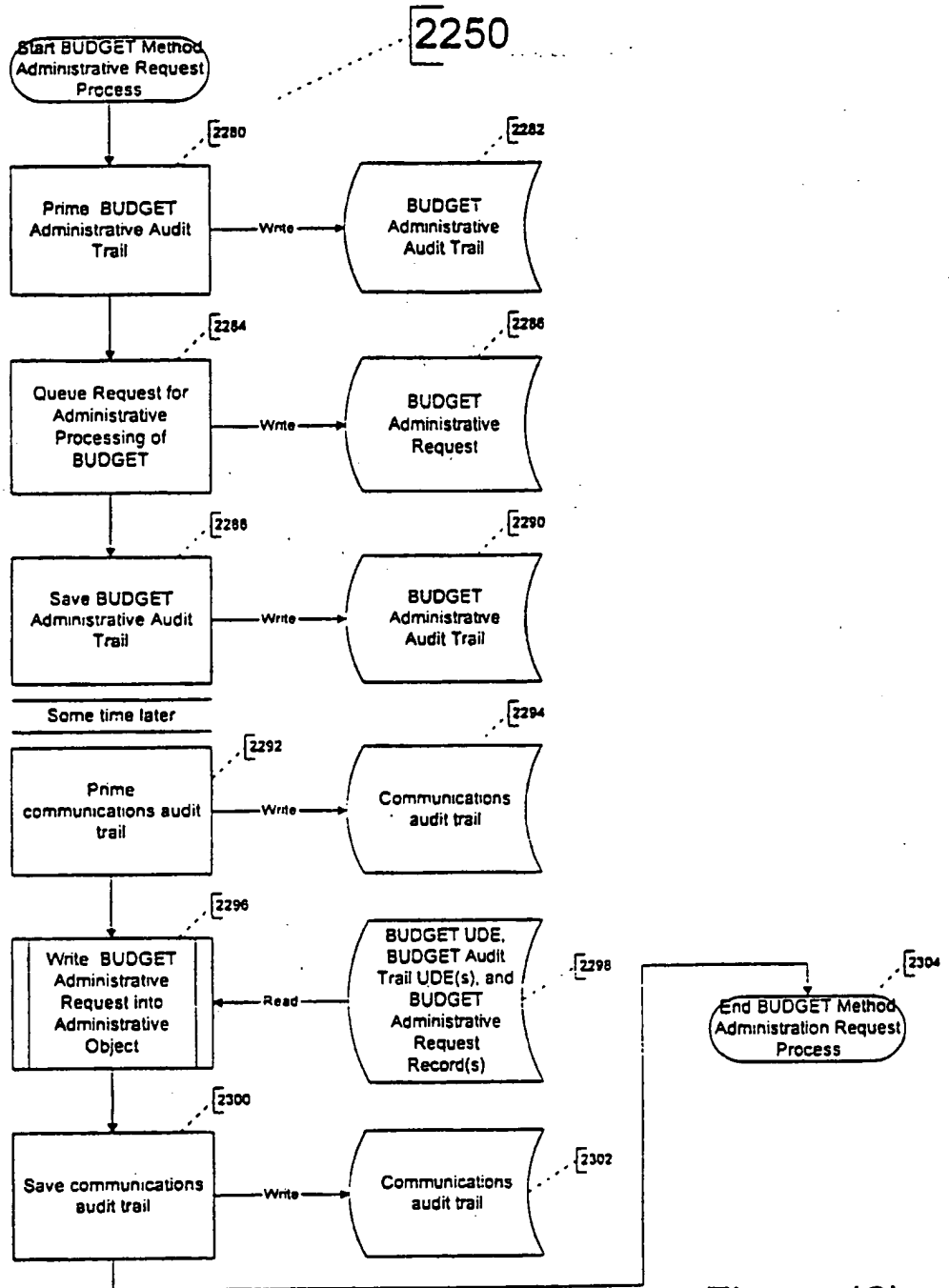


Figure 42b

64/146

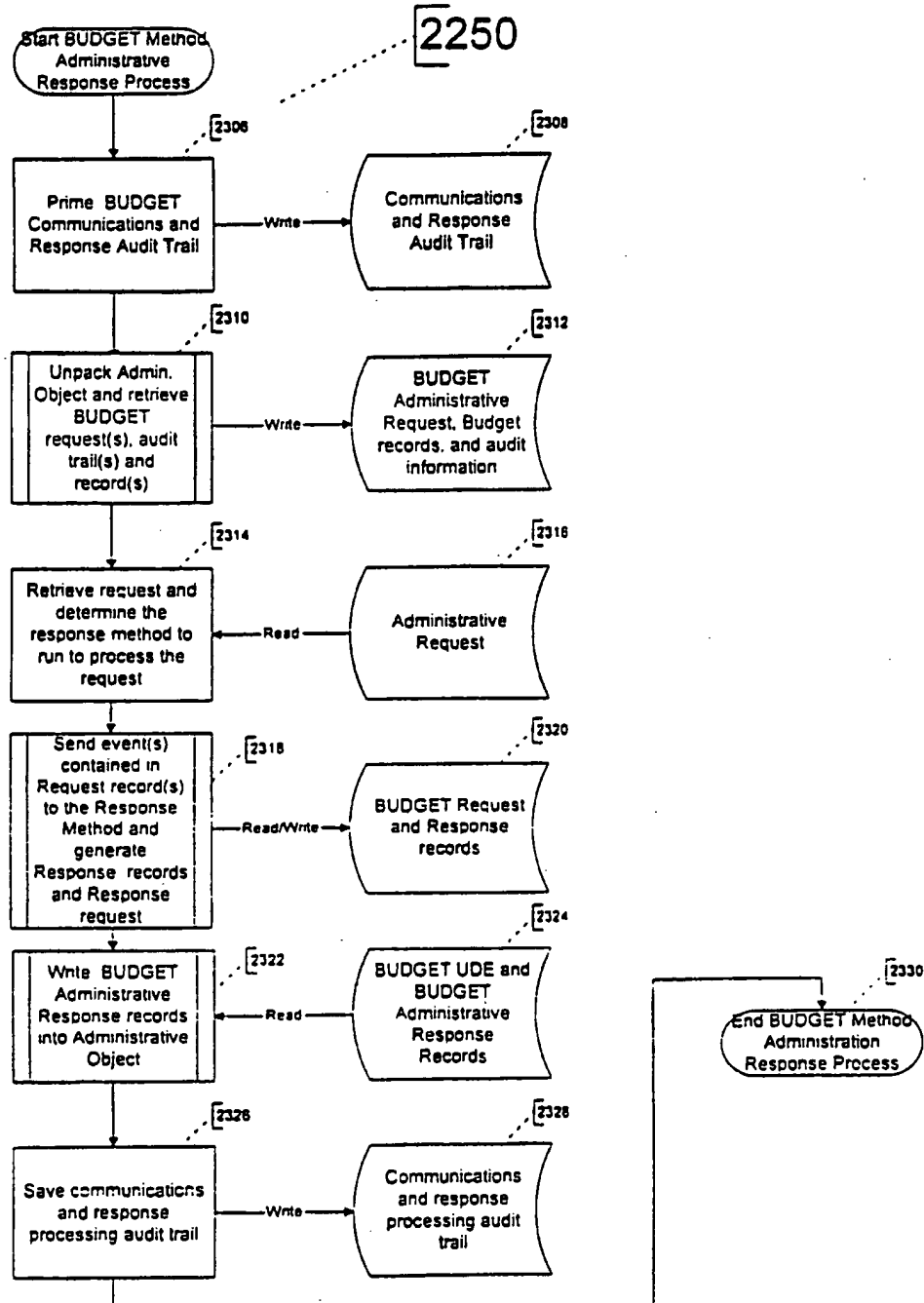


Figure 42c

65/146

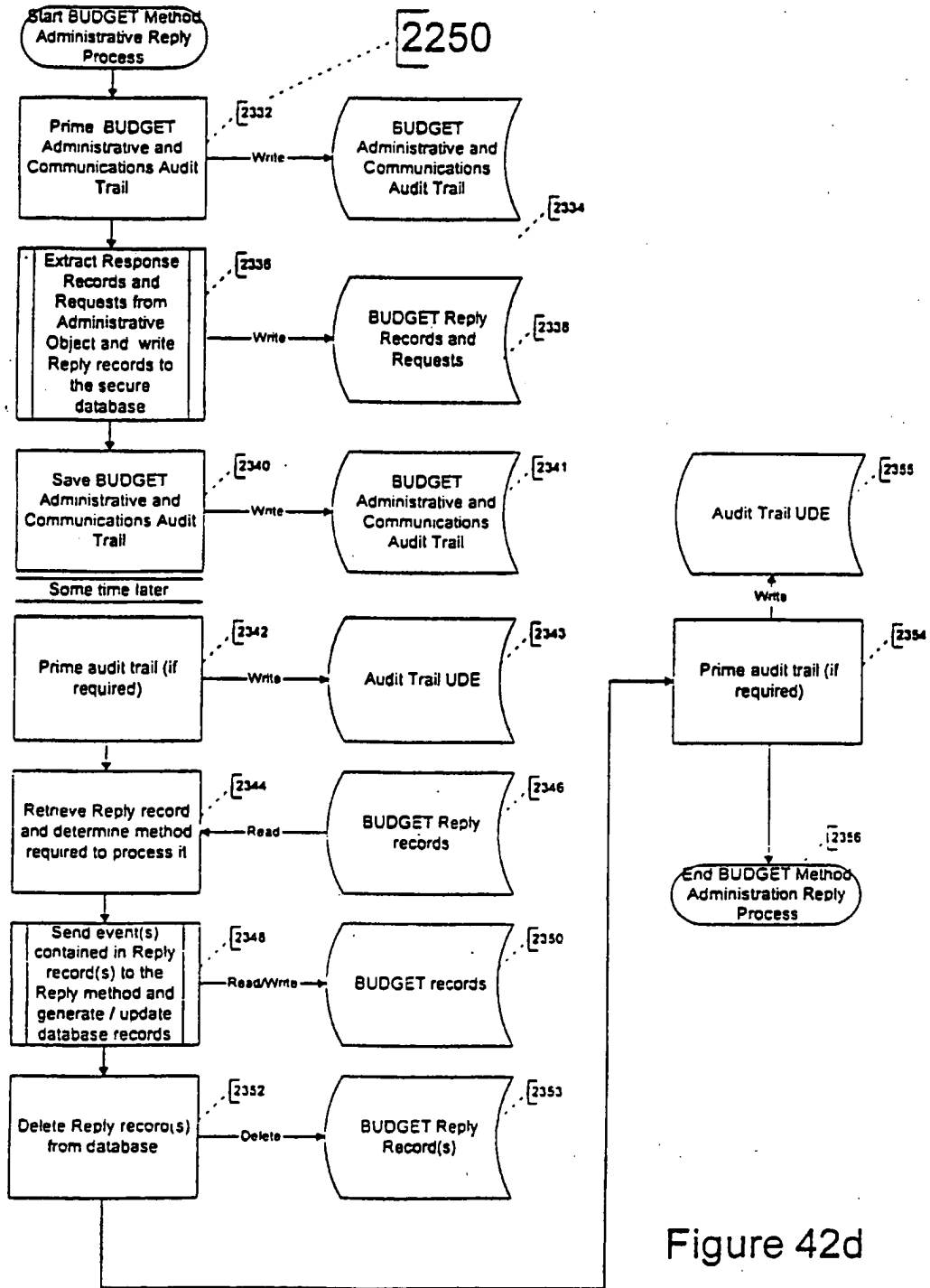


Figure 42d

66/146

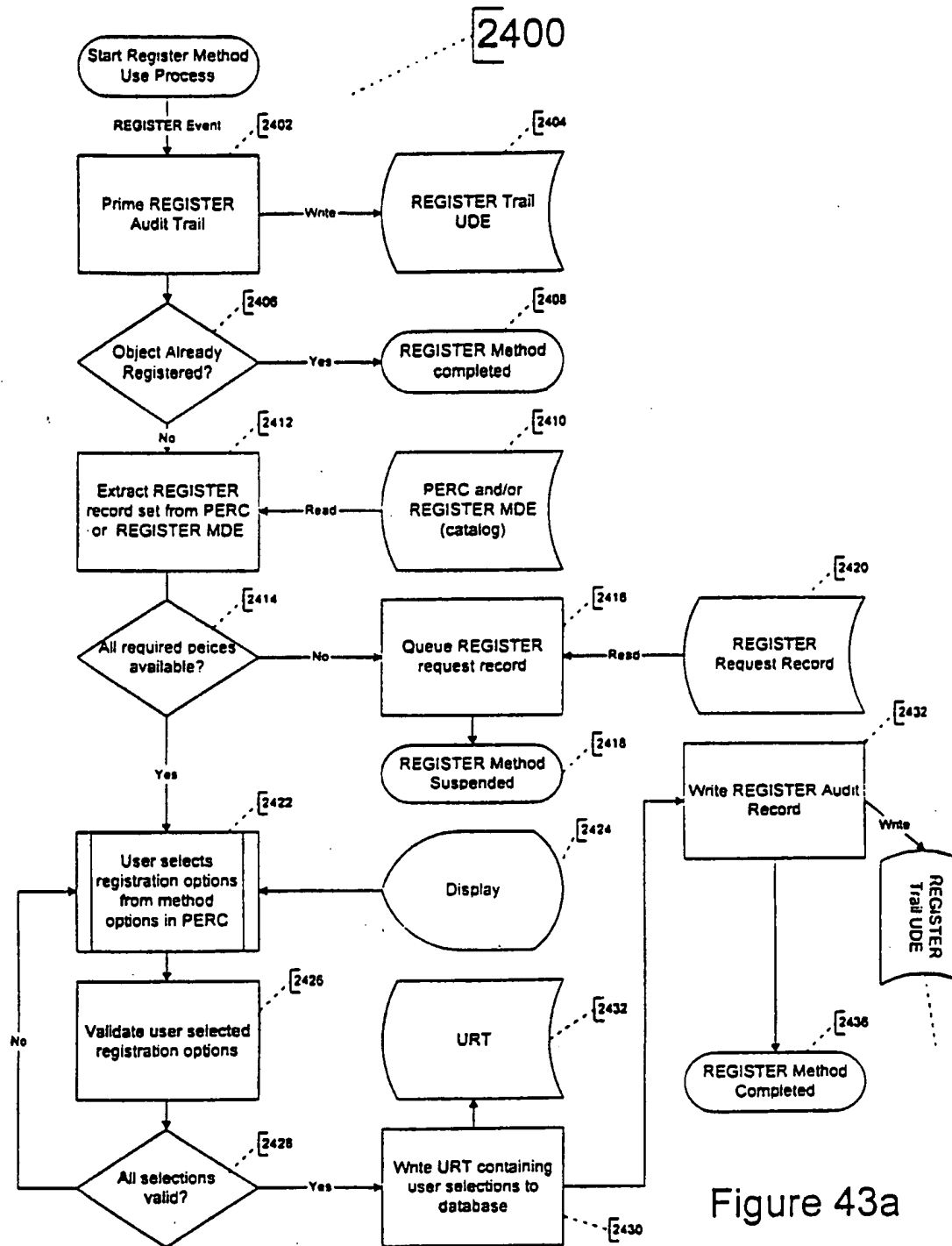


Figure 43a

67/146

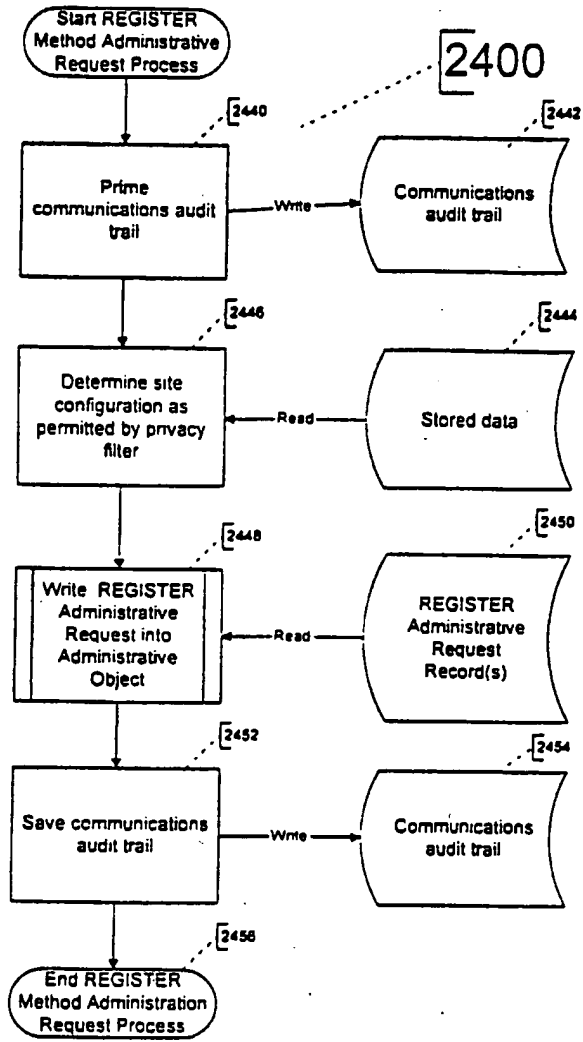


Figure 43b

68/146

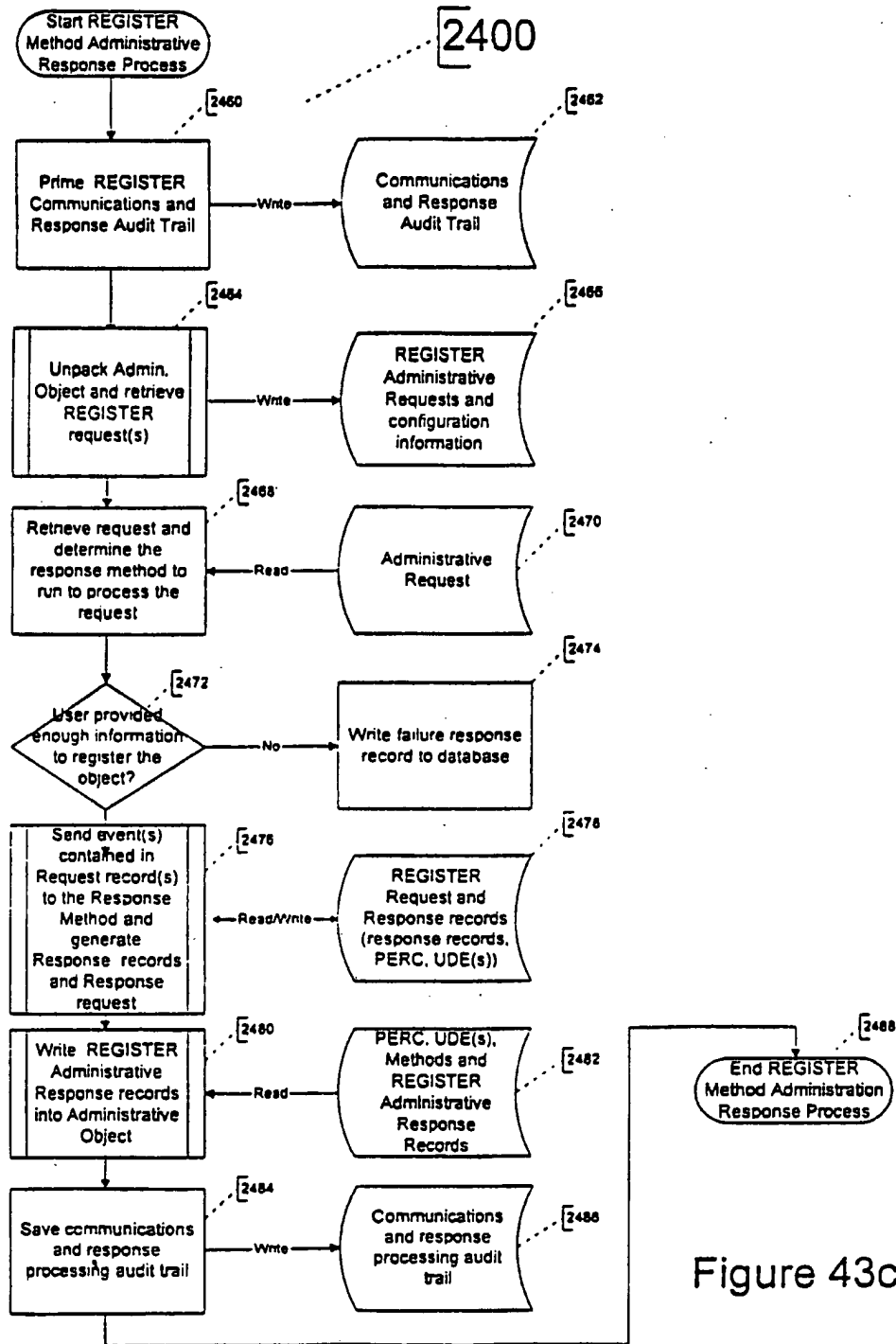


Figure 43c

69/146

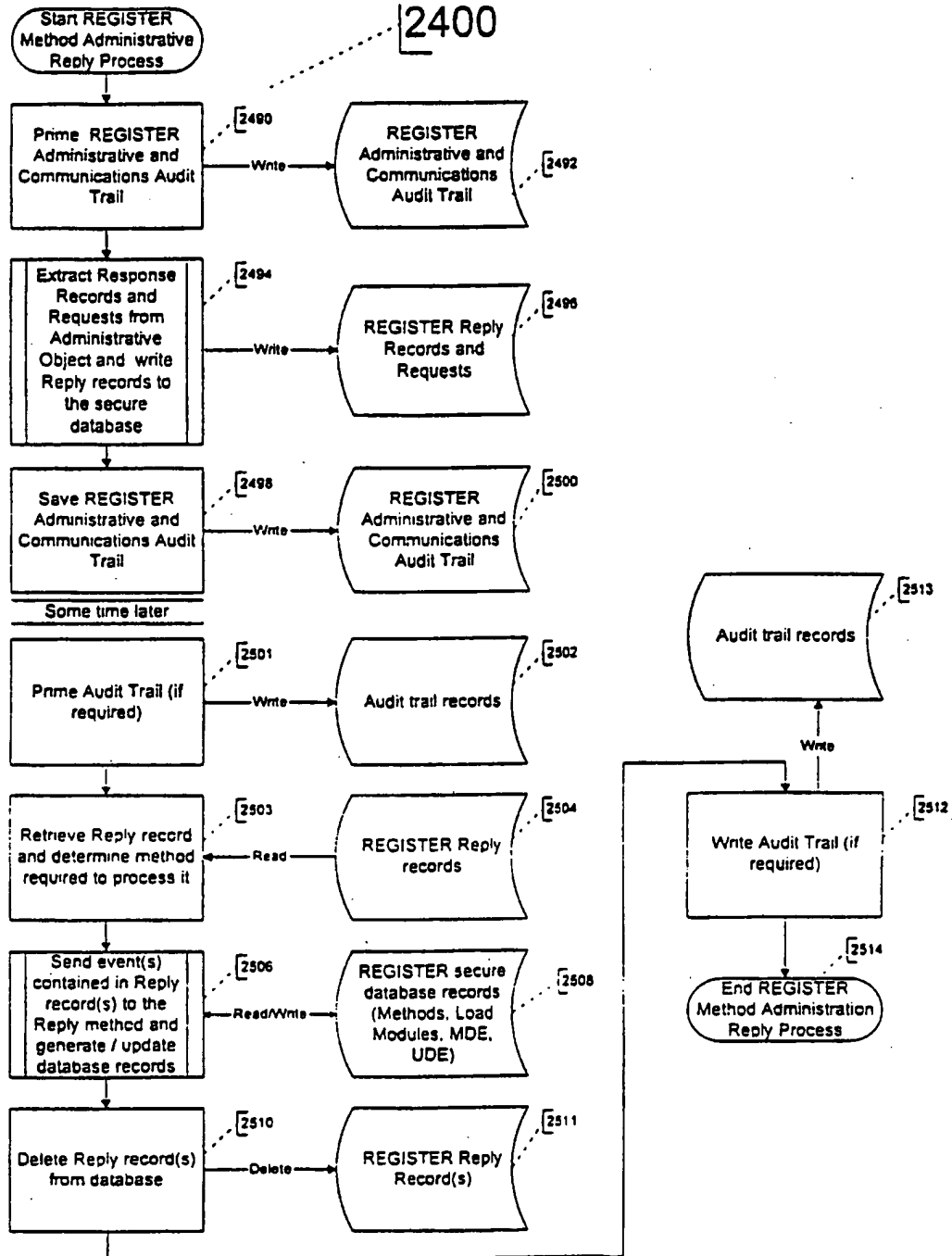


Figure 43d

70/146

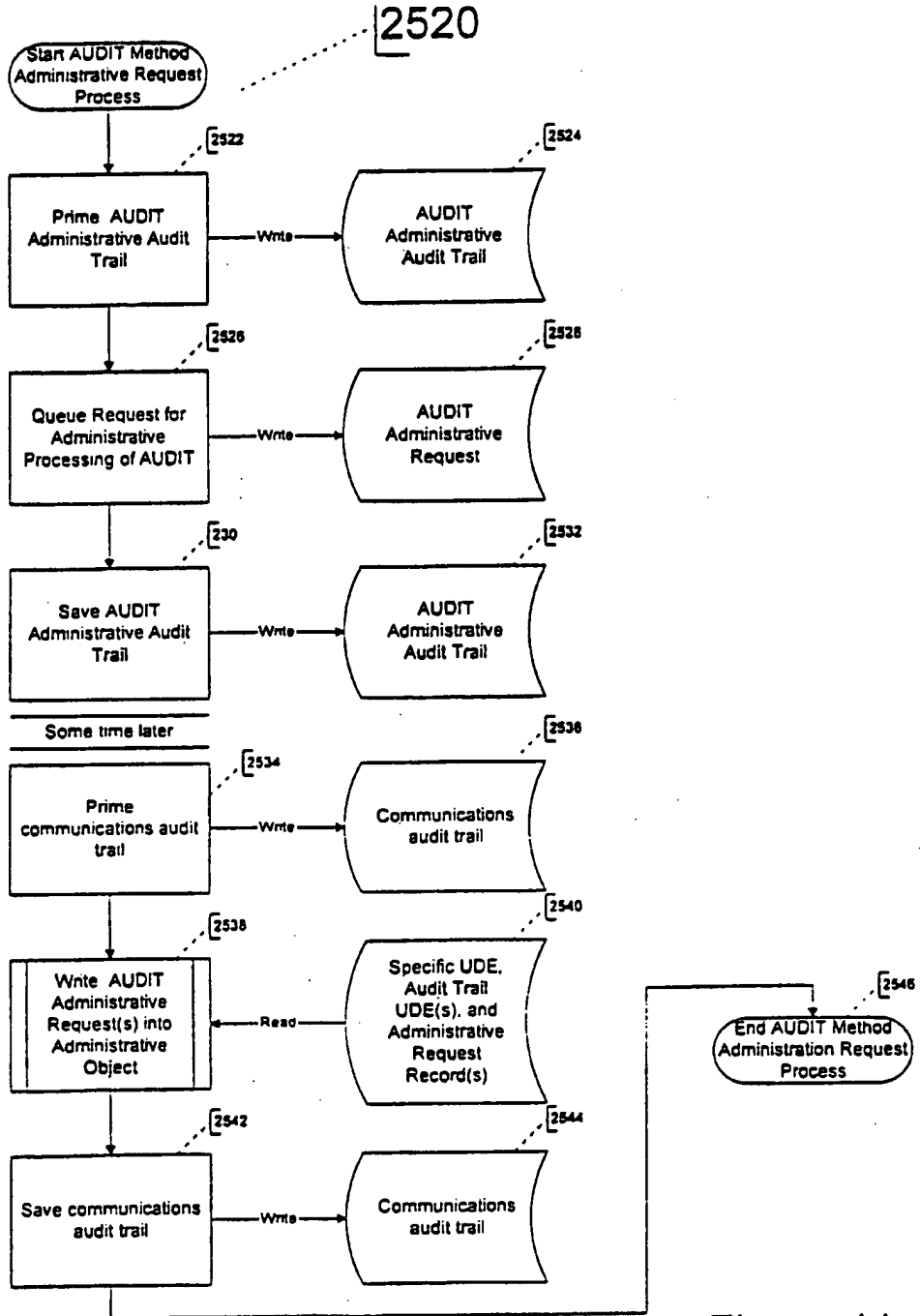


Figure 44a

71/146

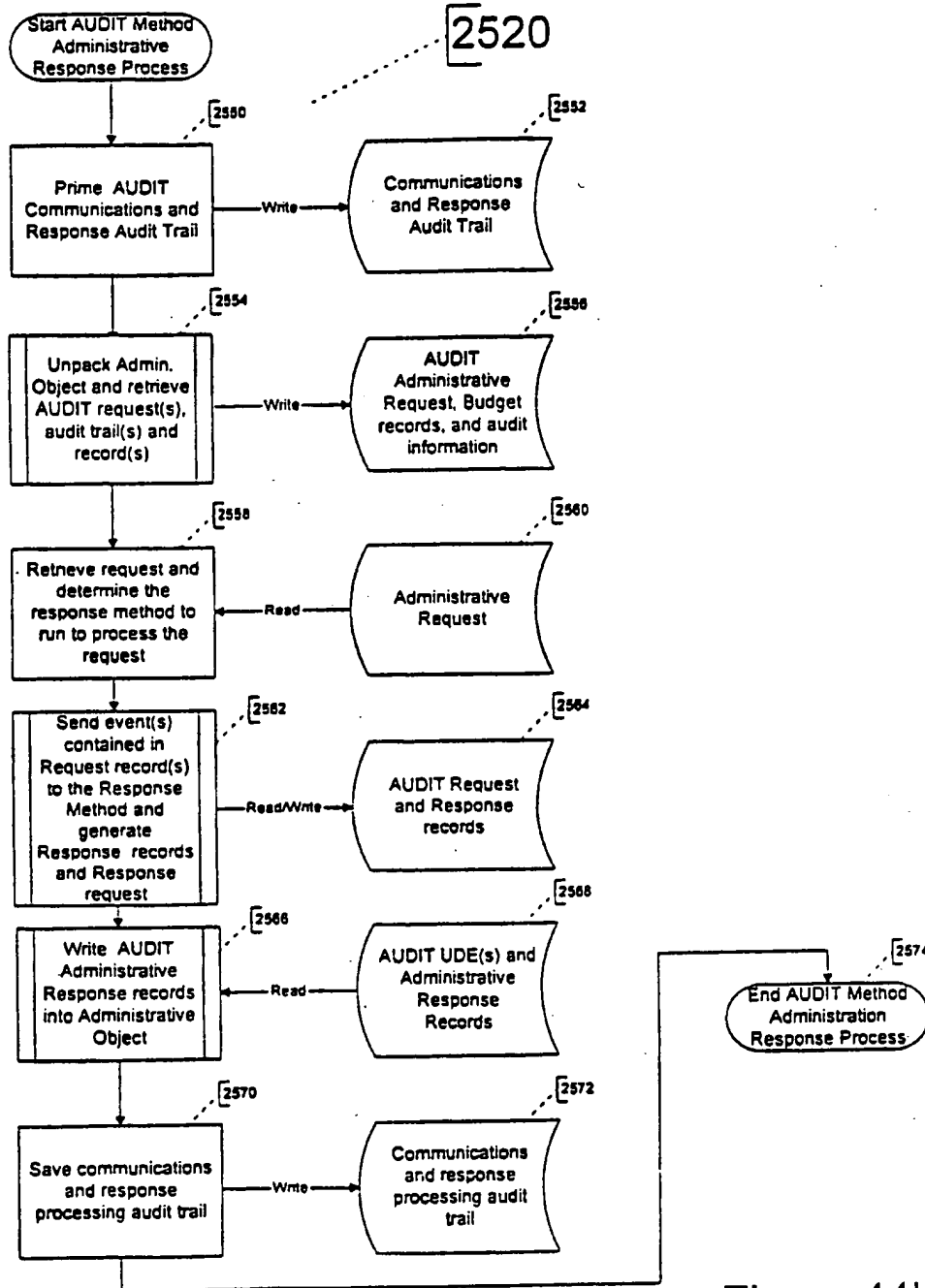


Figure 44b

72/146

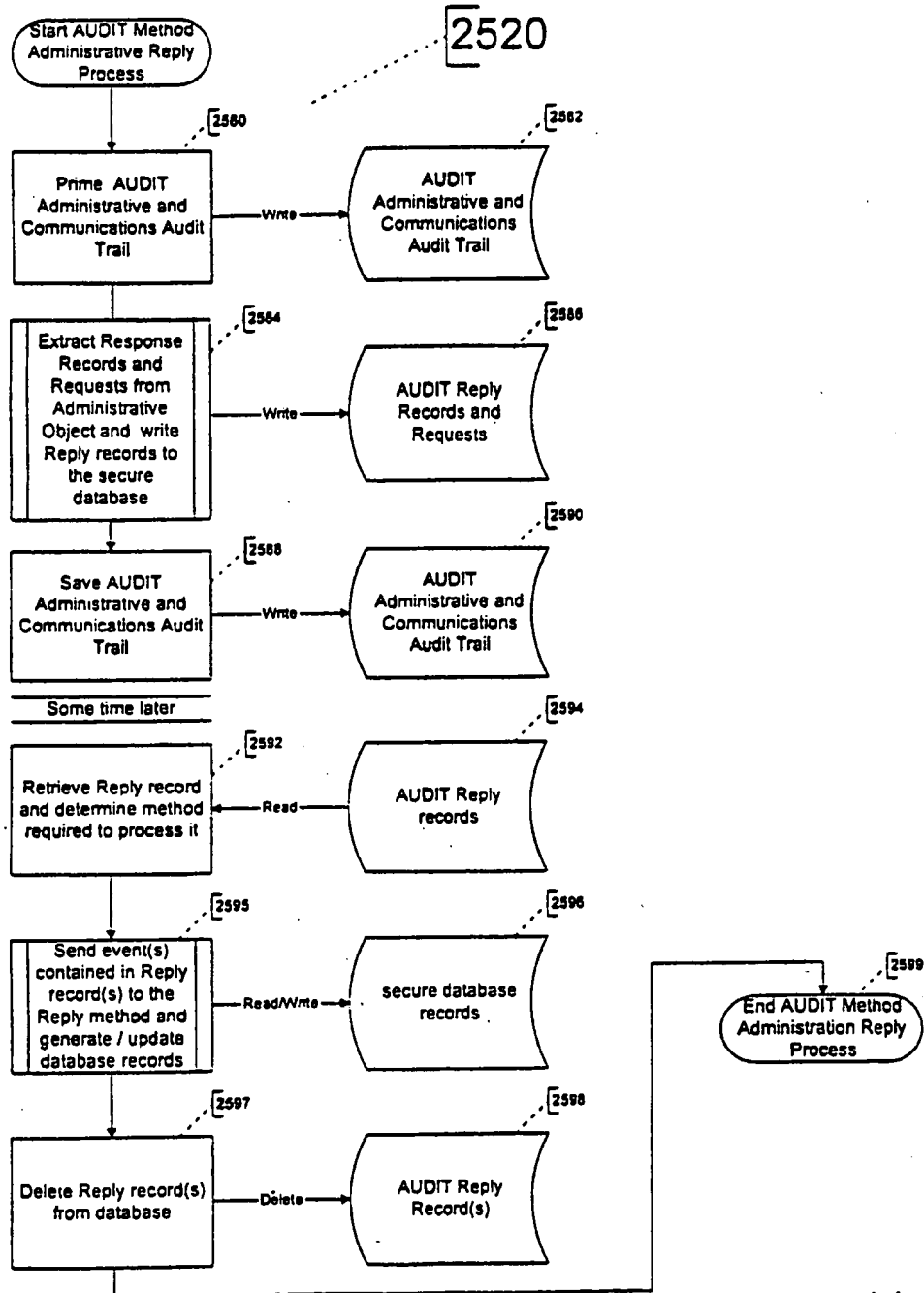
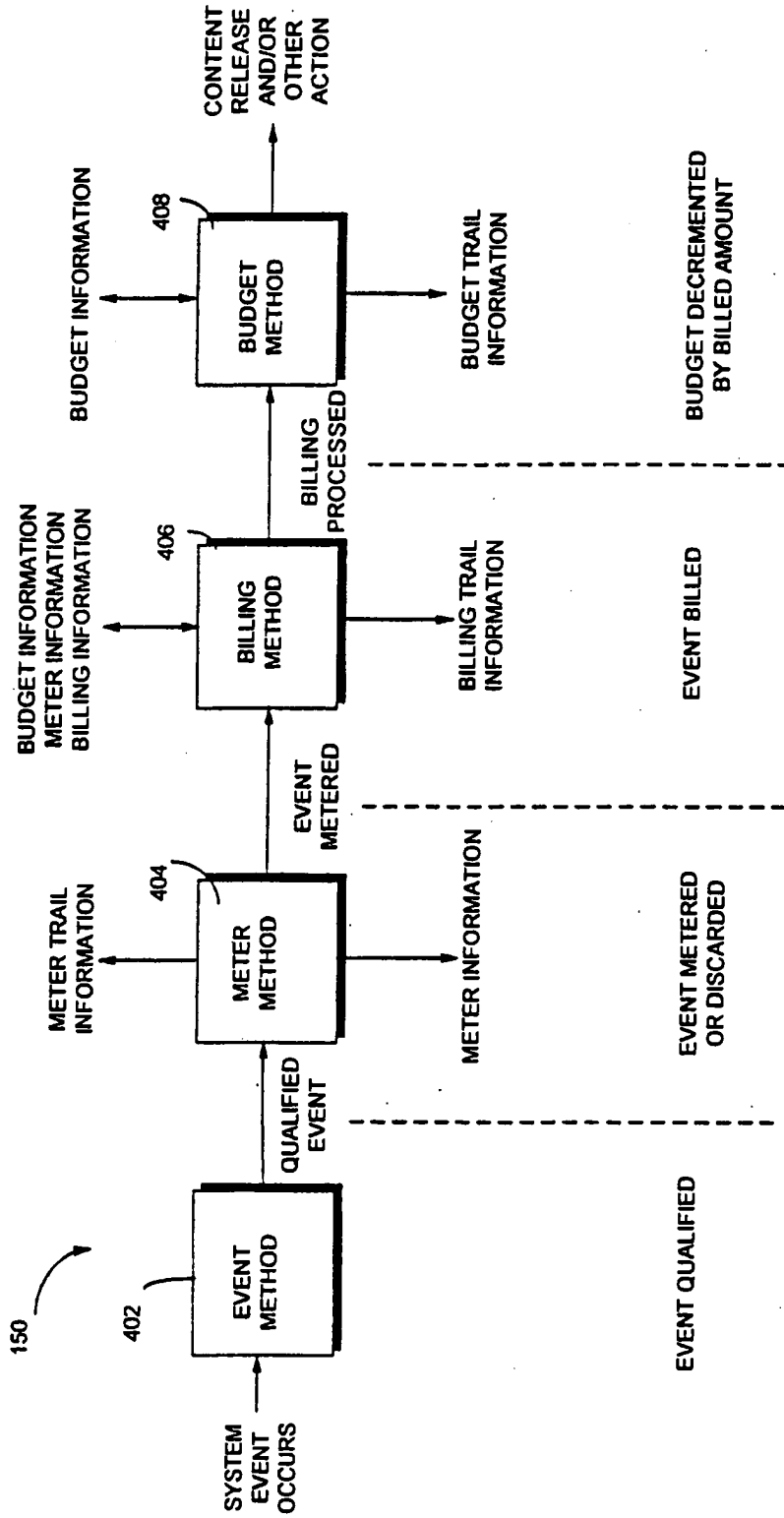


Figure 44c

FIG. 45



74/146

FIG. 46

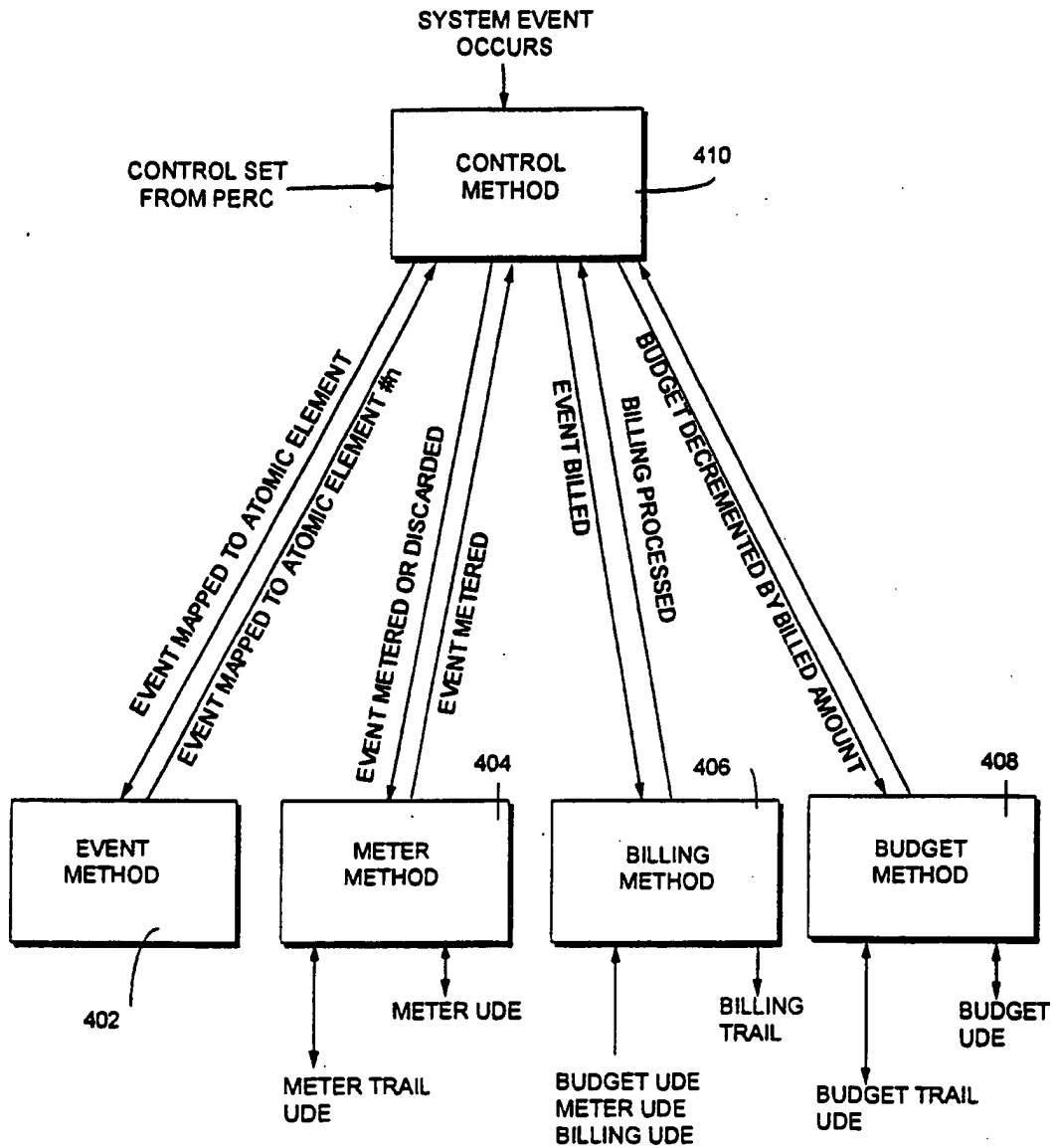
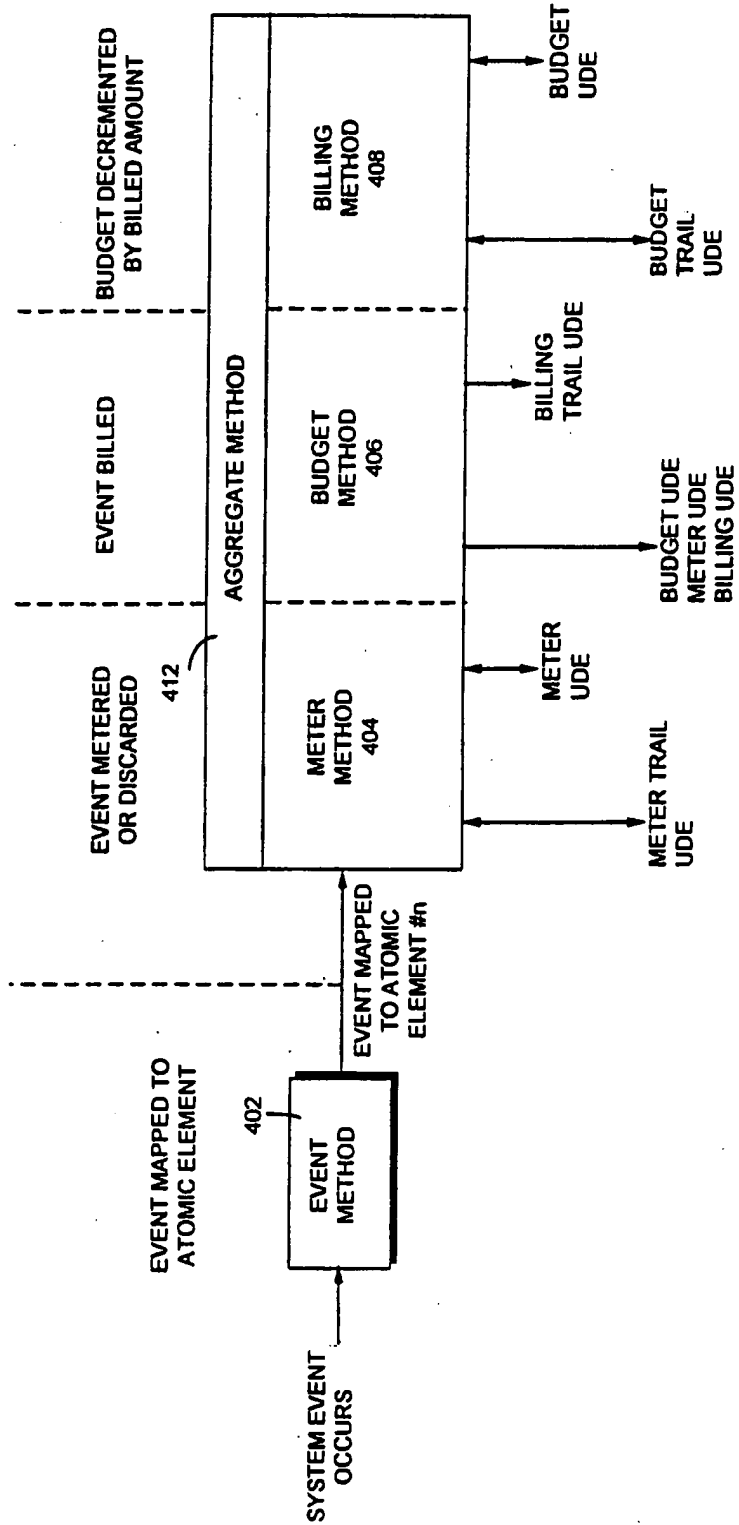


FIG. 47



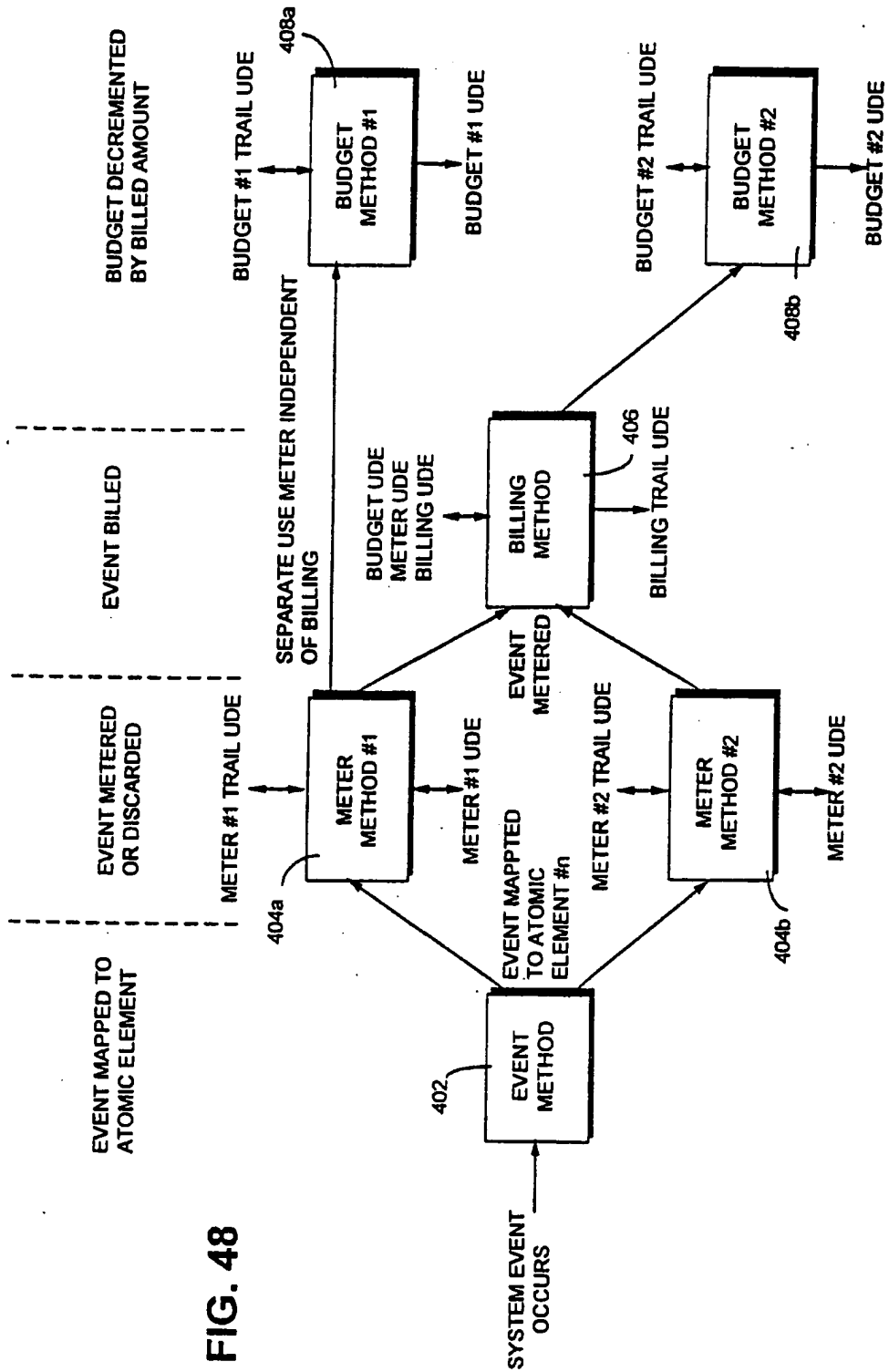


FIG. 48

77/146

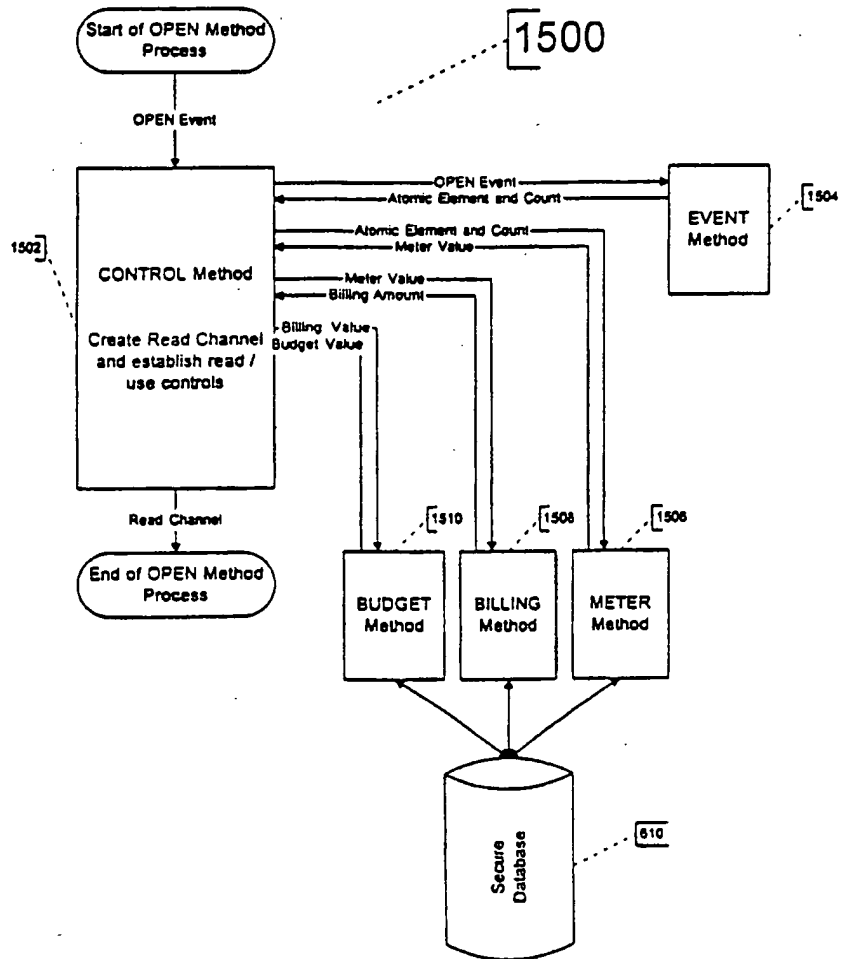


Figure 49

78/146

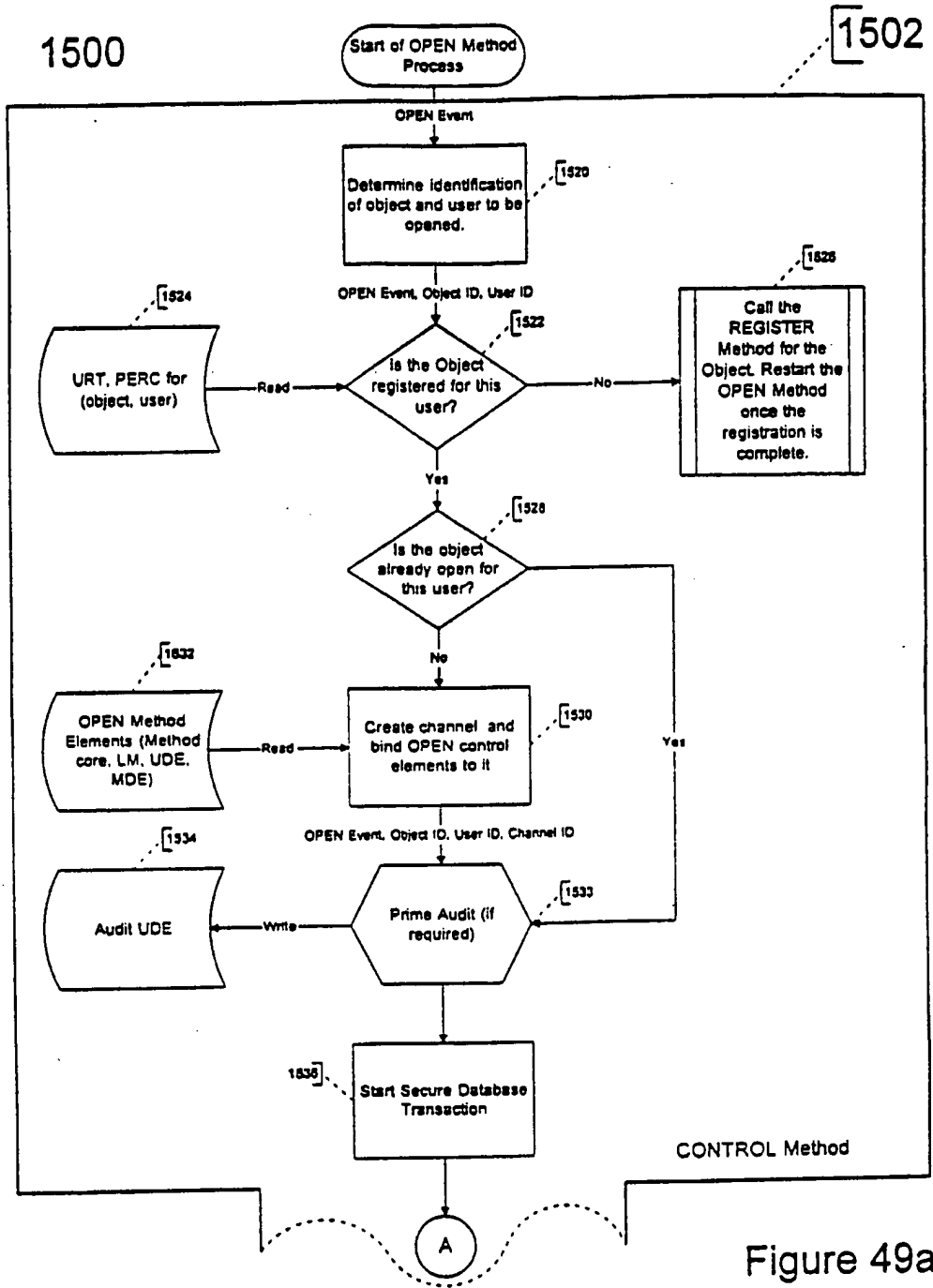


Figure 49a

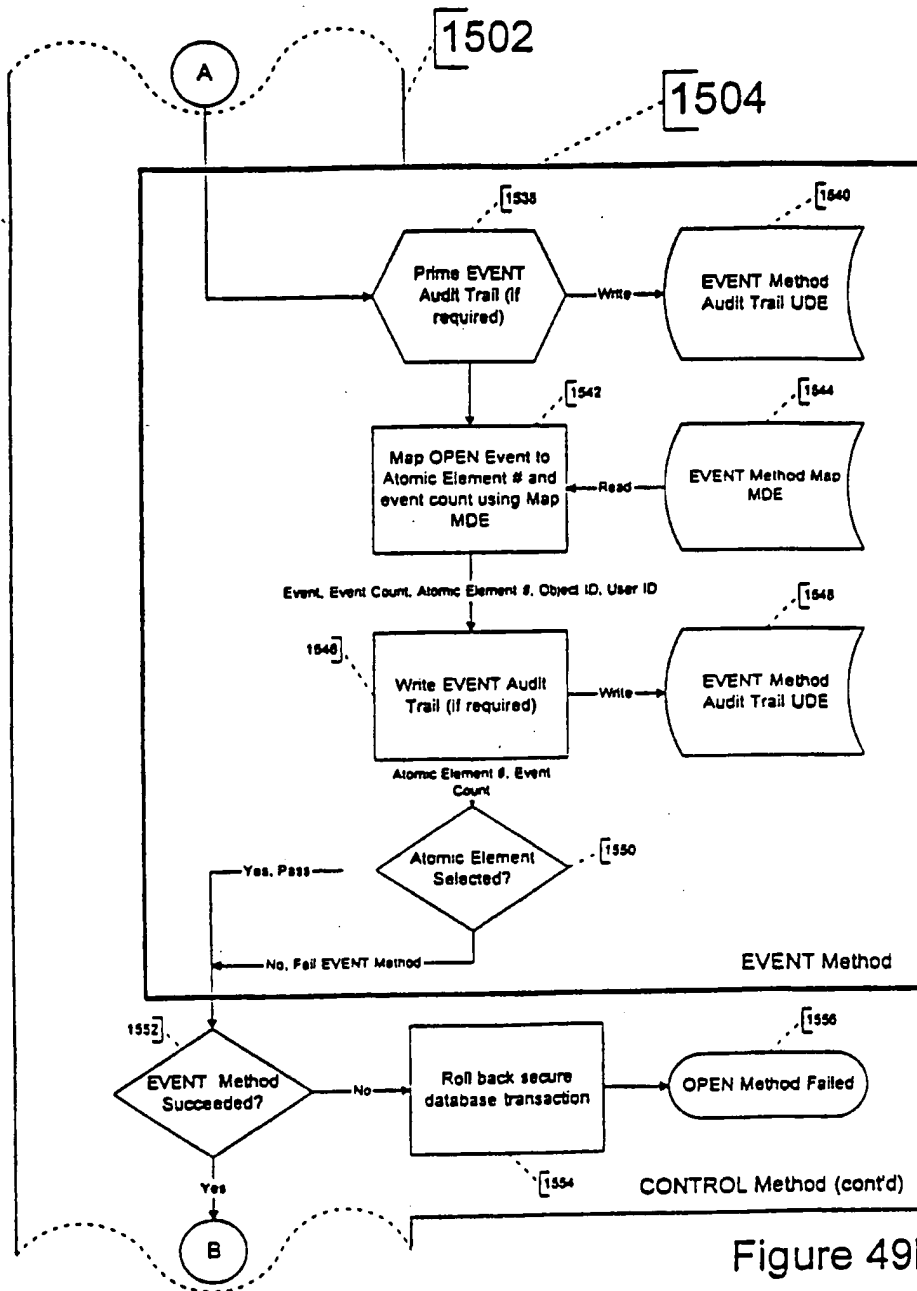


Figure 49b

80/146

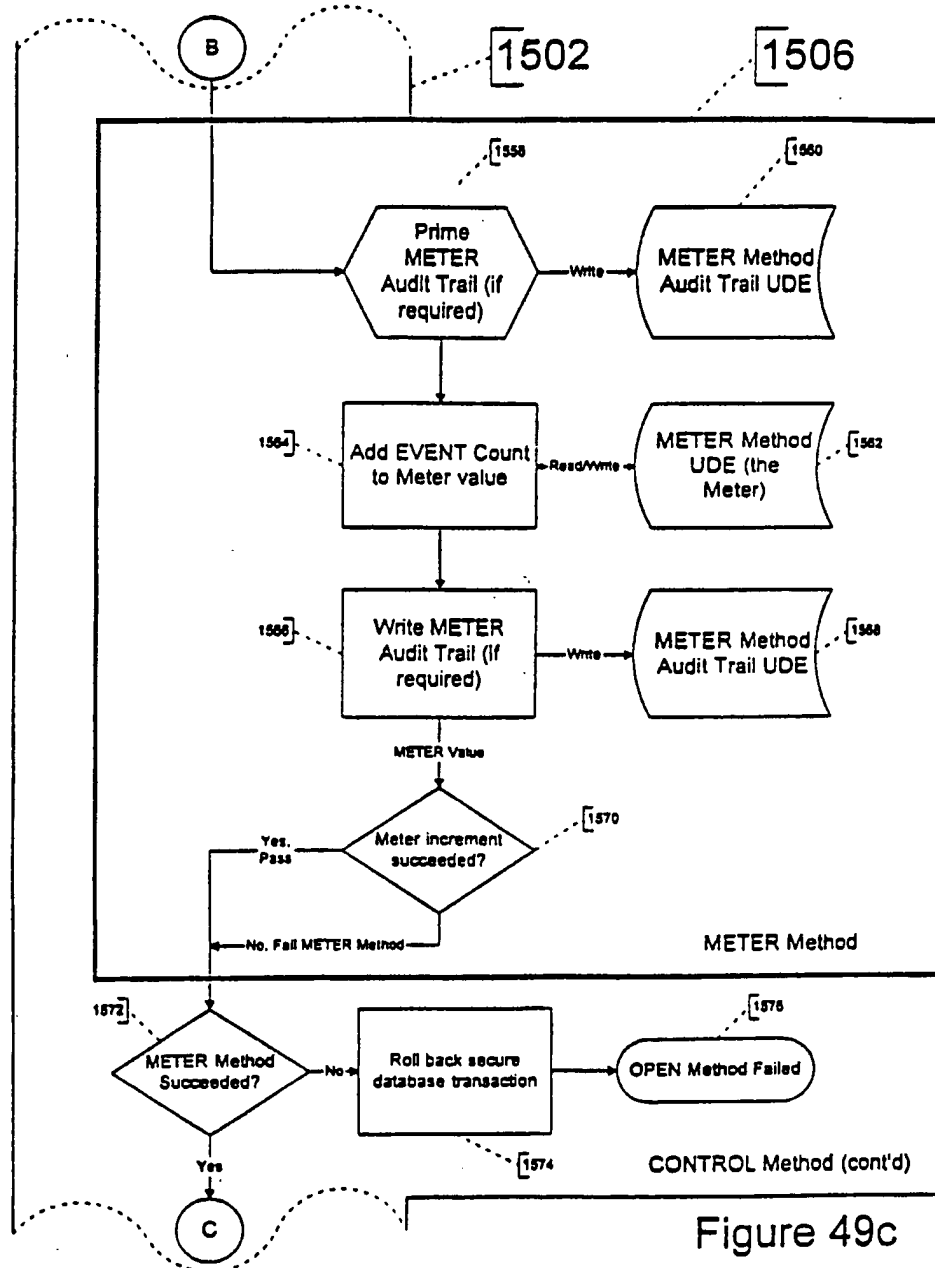


Figure 49c

81/146

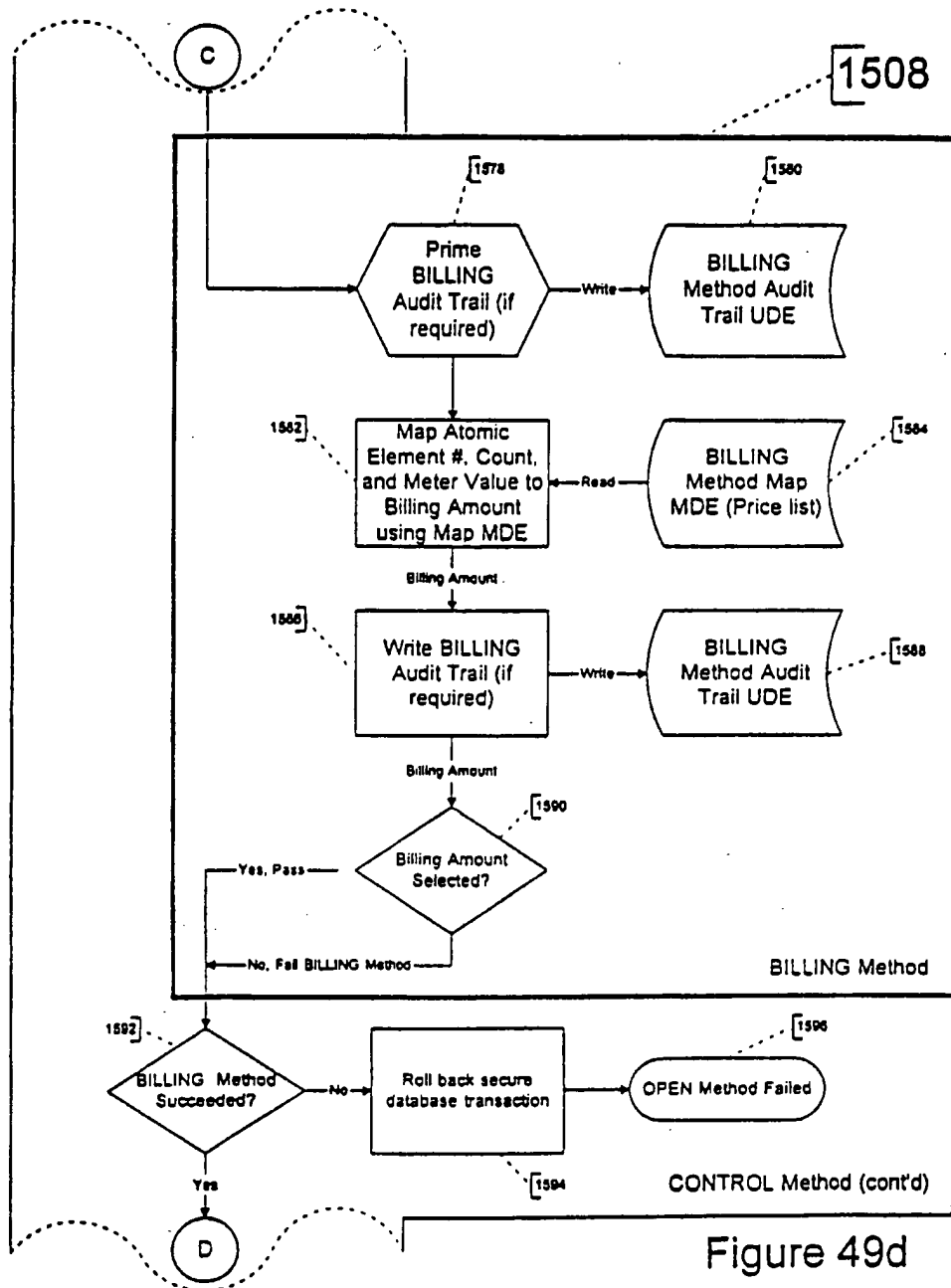


Figure 49d

82/146

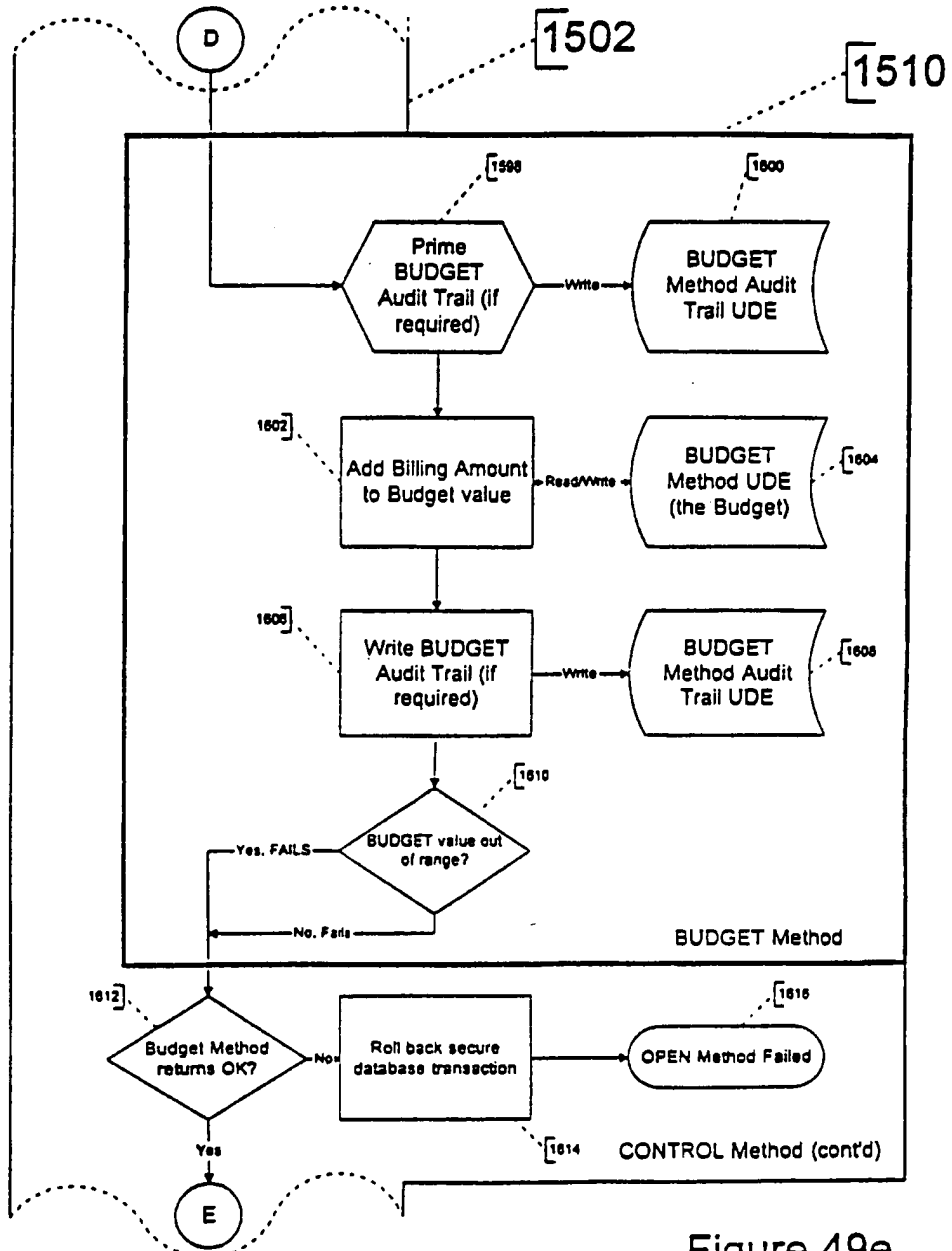


Figure 49e

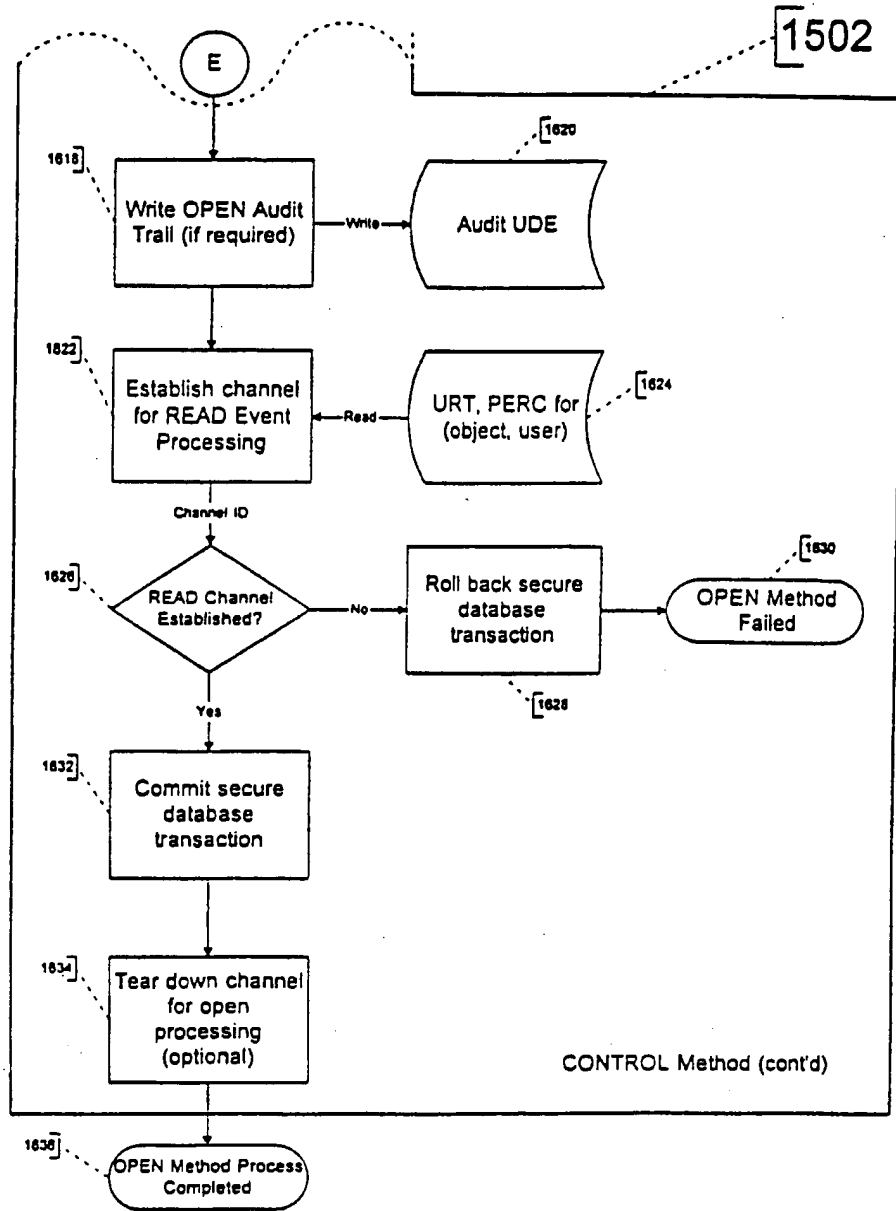


Figure 49f

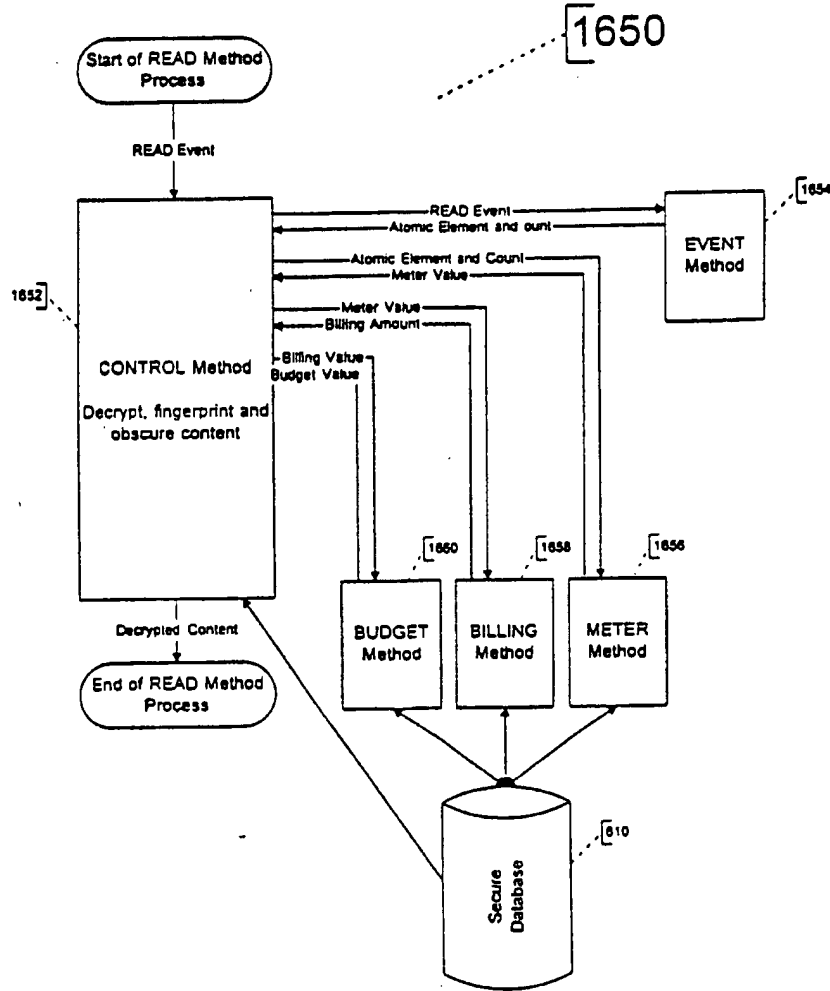


Figure 50

85/146

1650

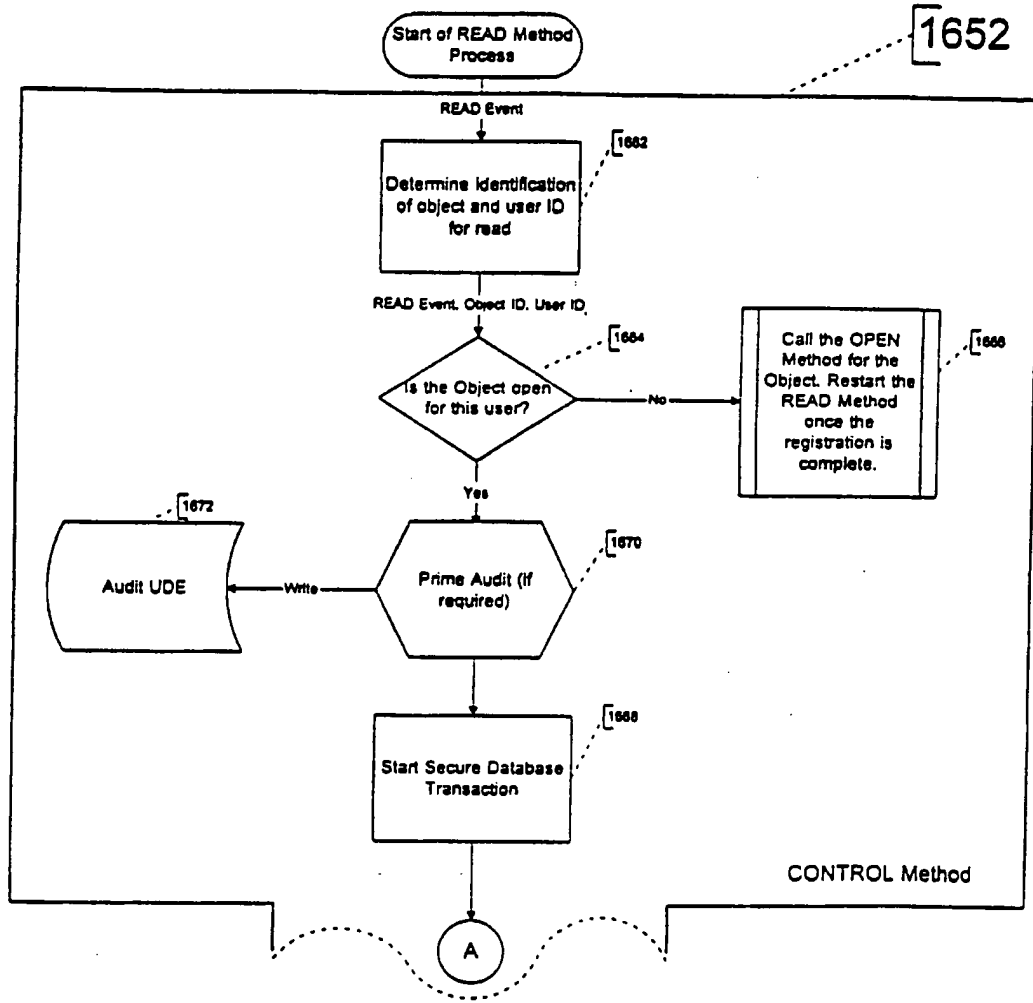


Figure 50a

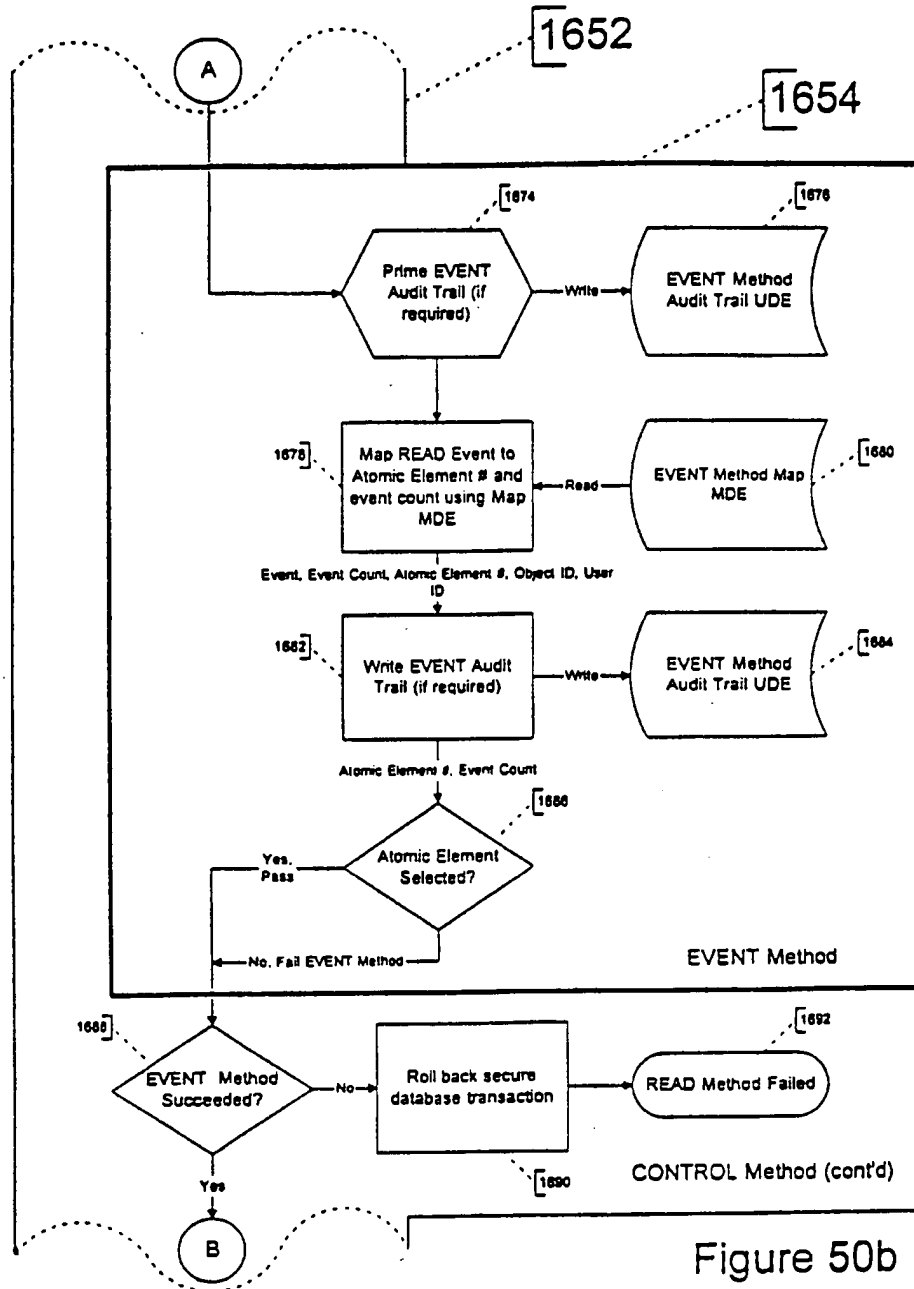


Figure 50b

87/146

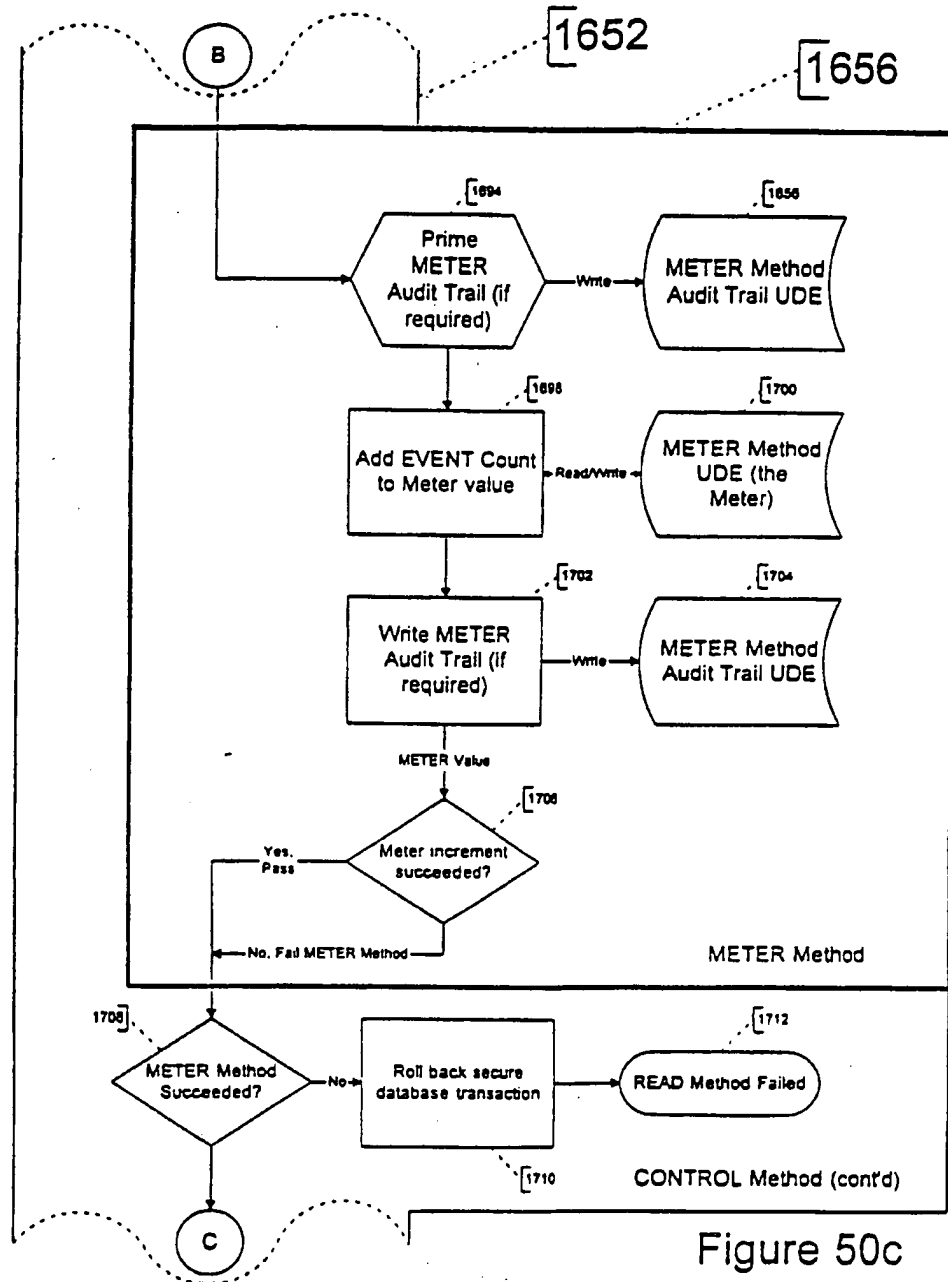


Figure 50c

88/146

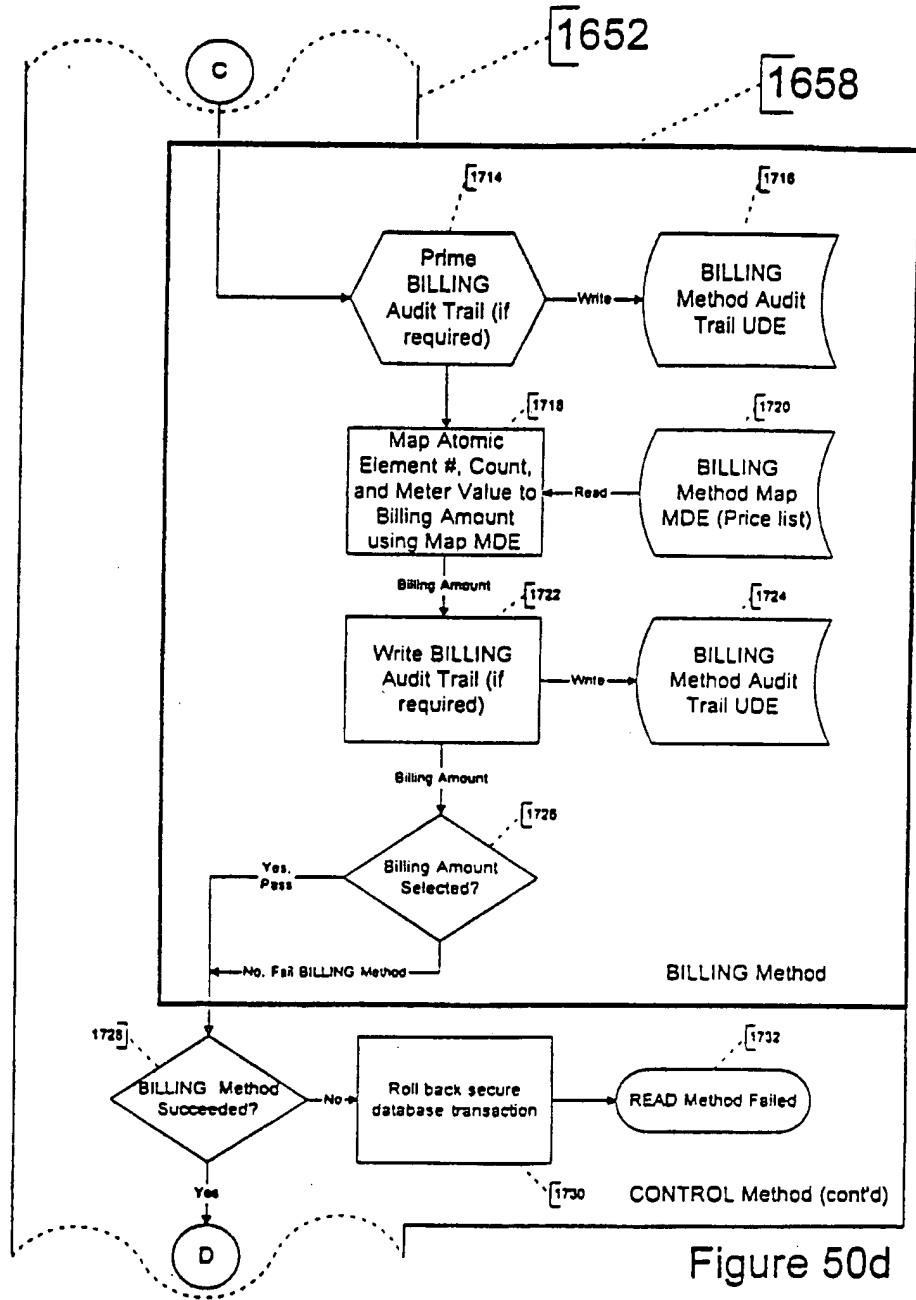


Figure 50d

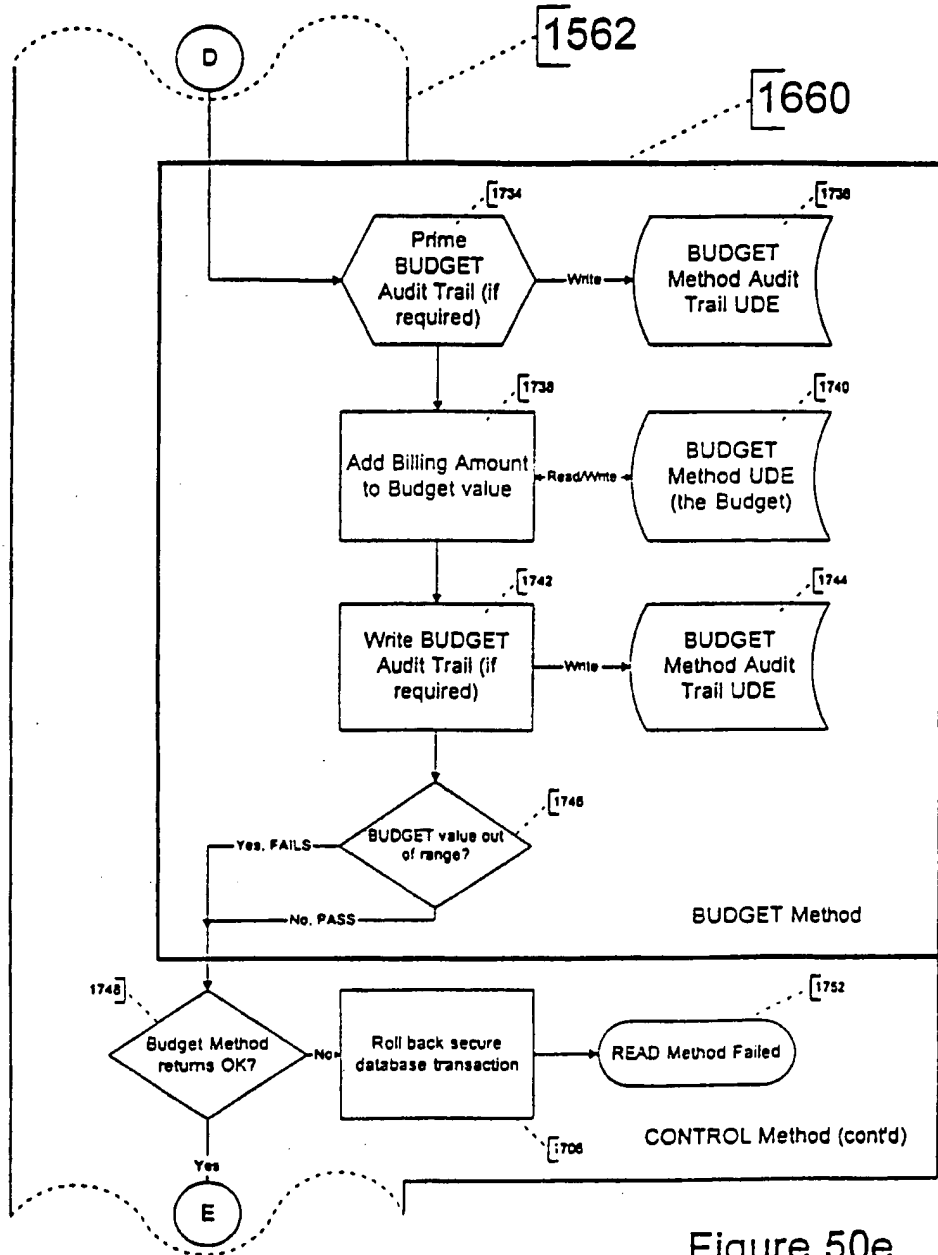
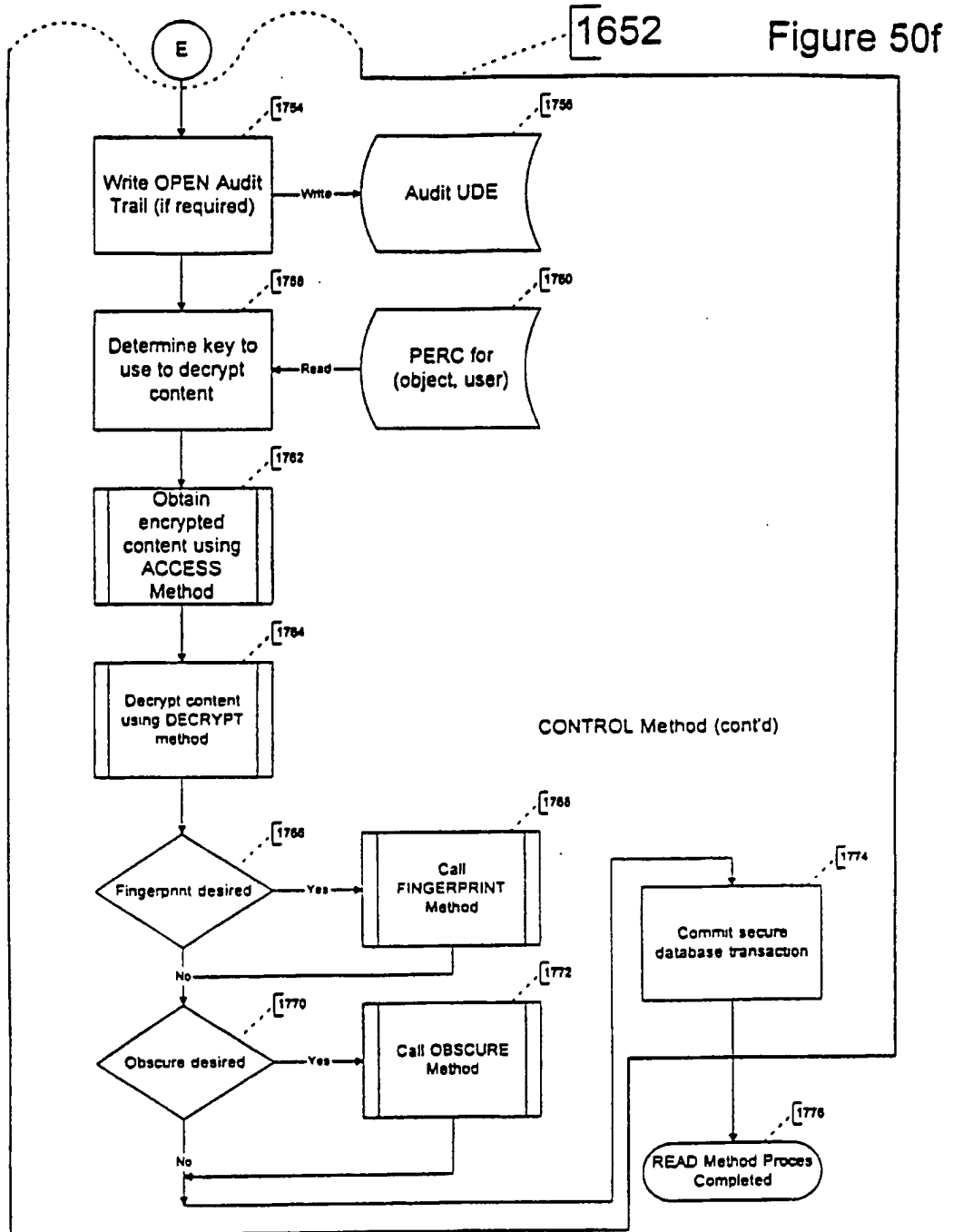


Figure 50e

90/146



SUBSTITUTE SHEET (RULE 26)

91/146

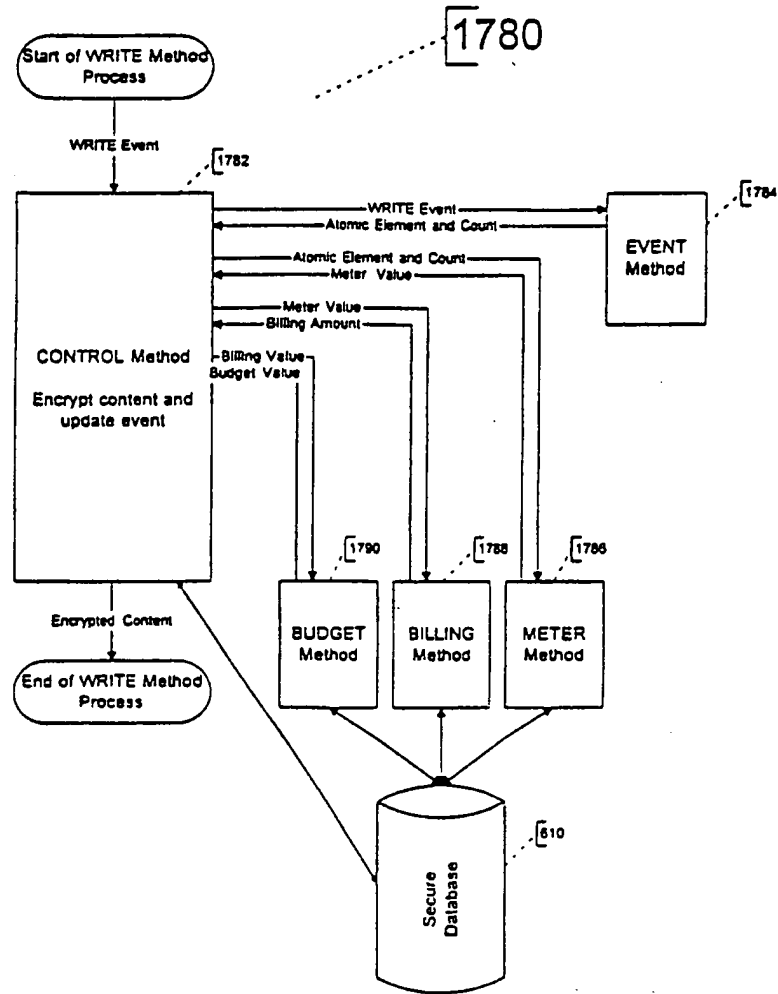


Figure 51

92/146

1780

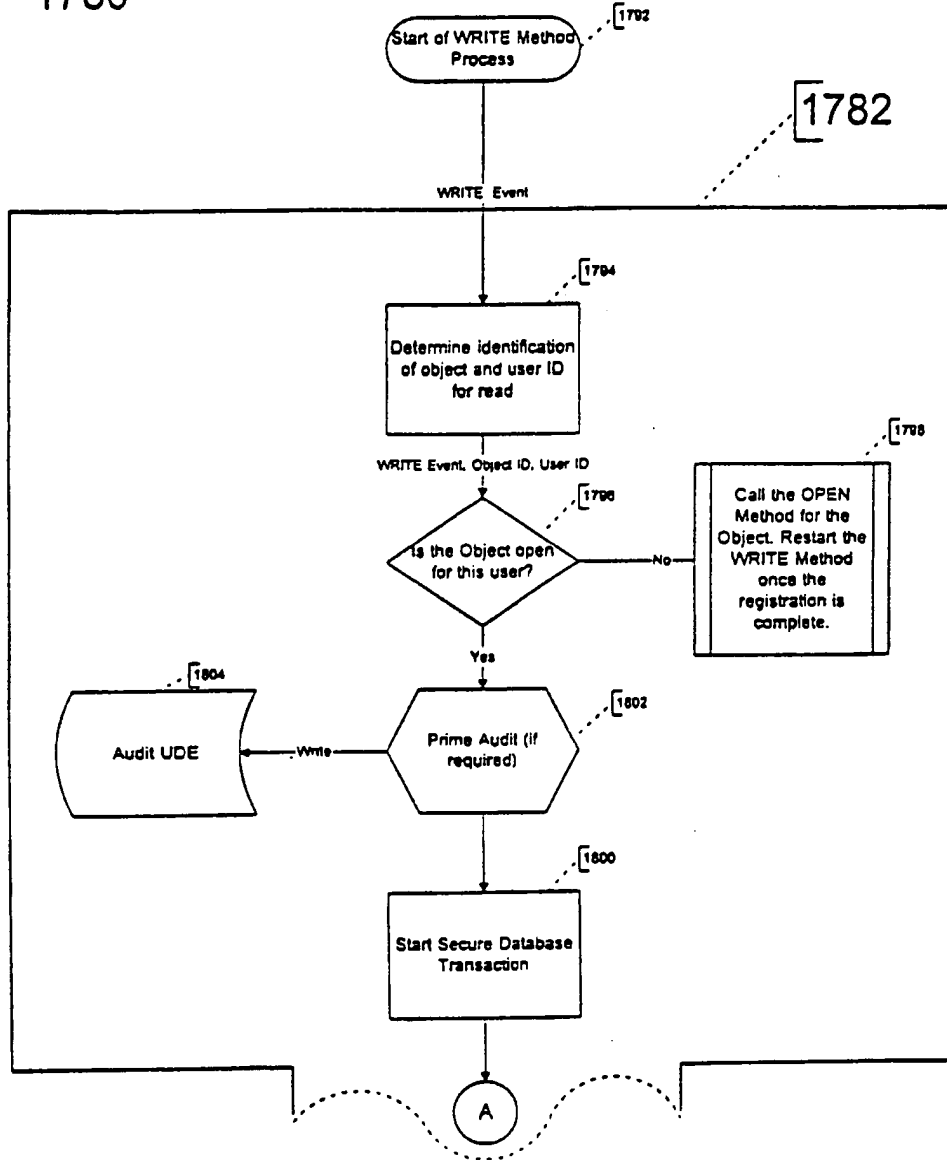


Figure 51a

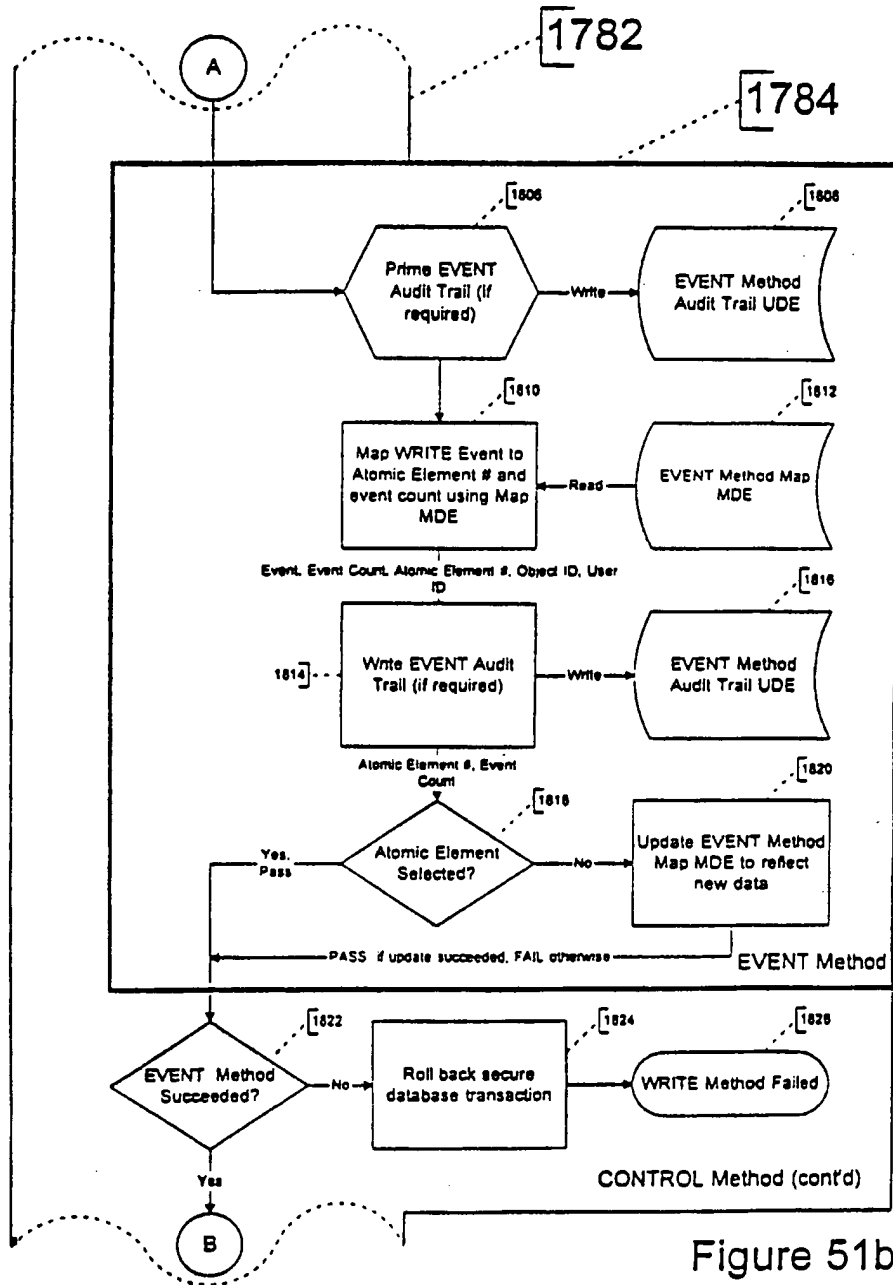


Figure 51b

94/146

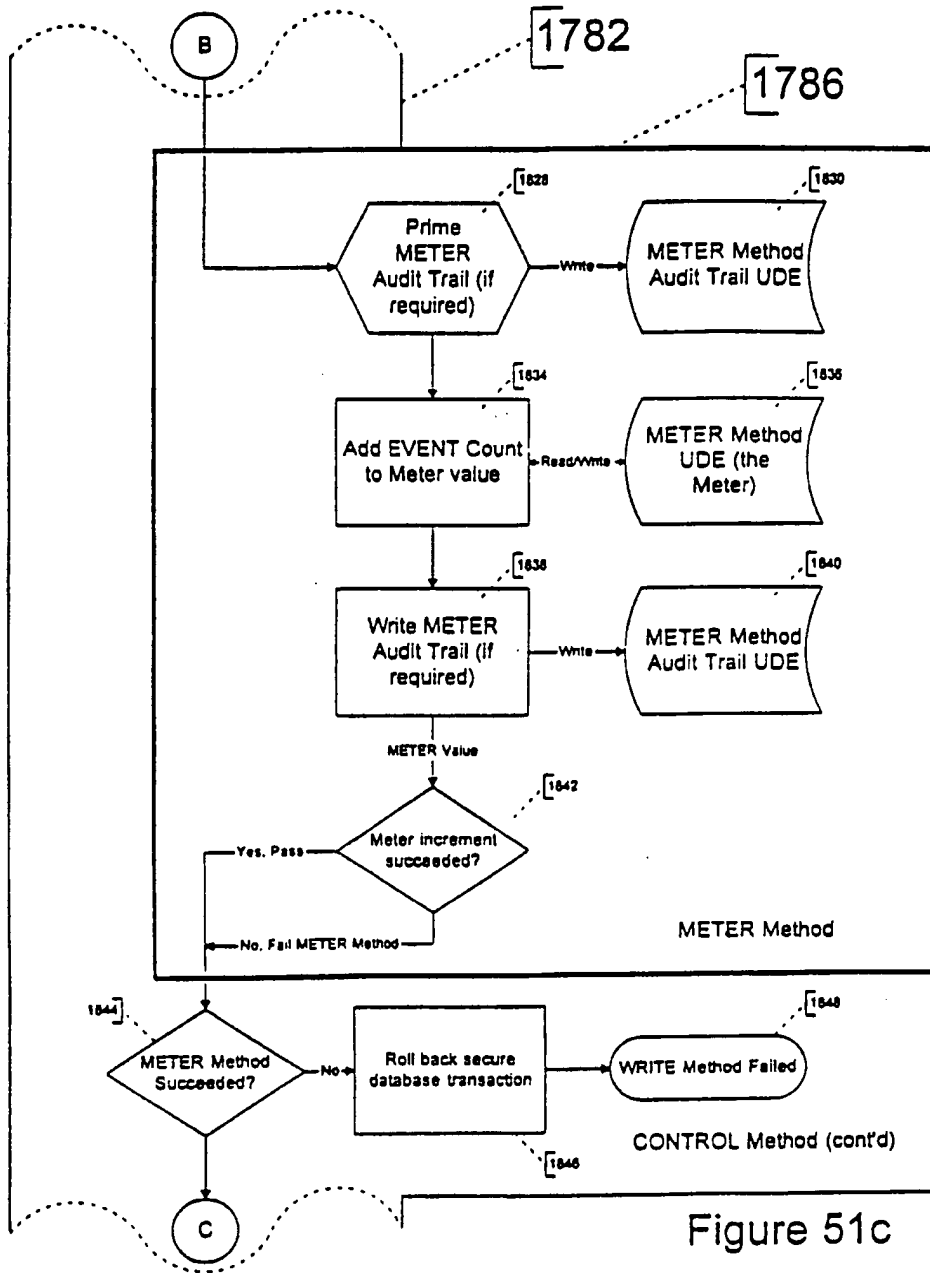
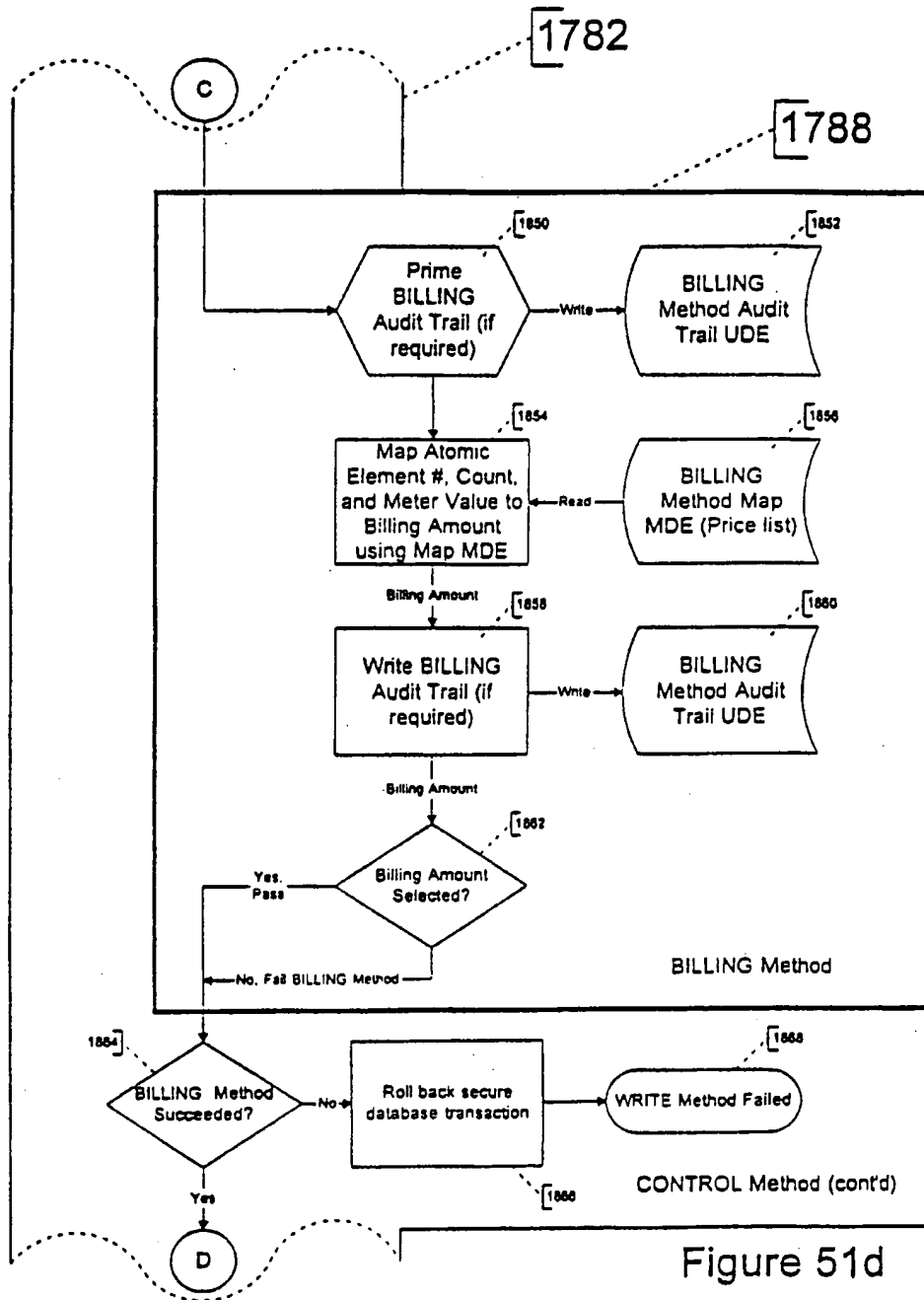


Figure 51c

95/146



96/146

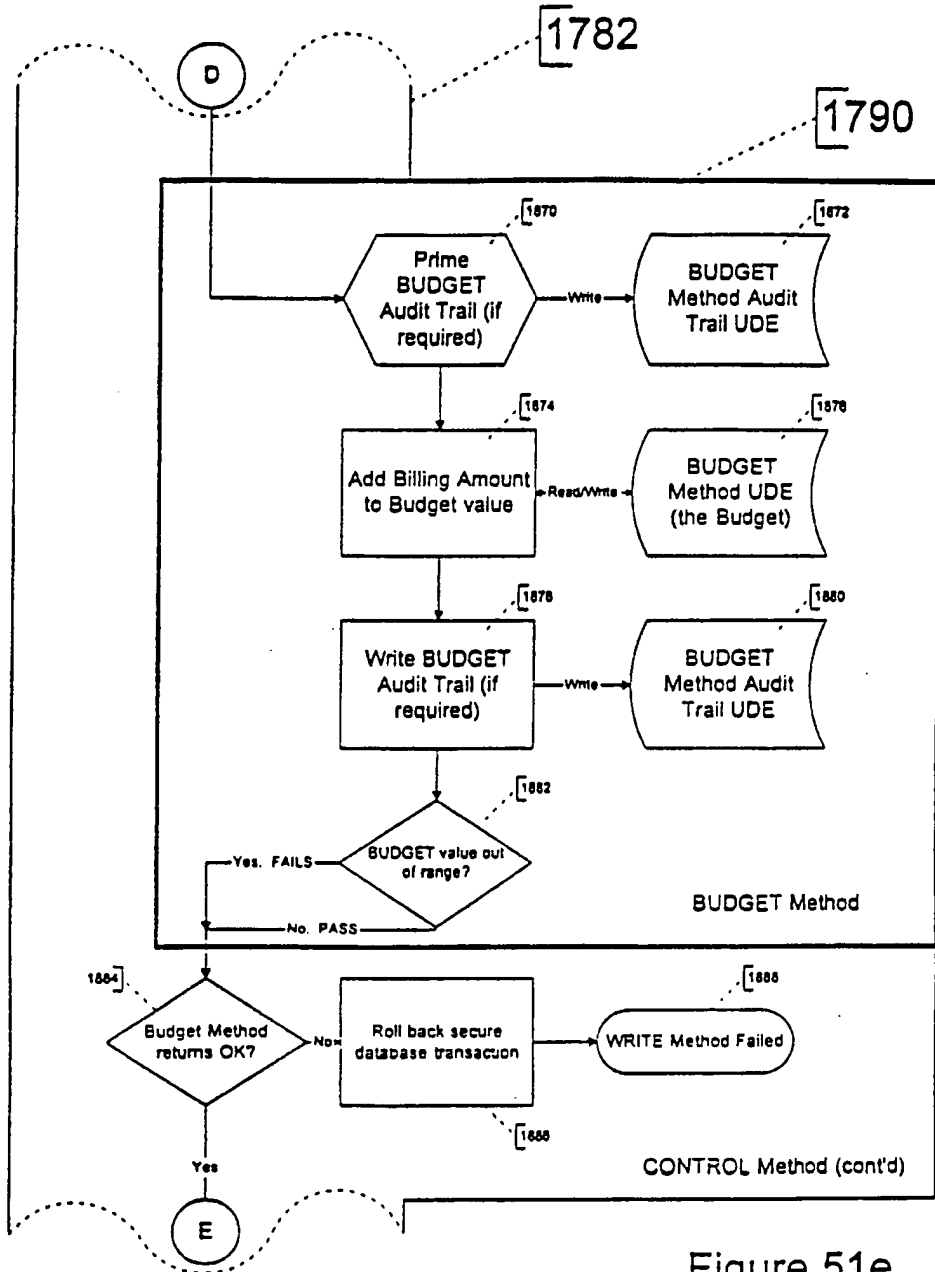


Figure 51e

97/146

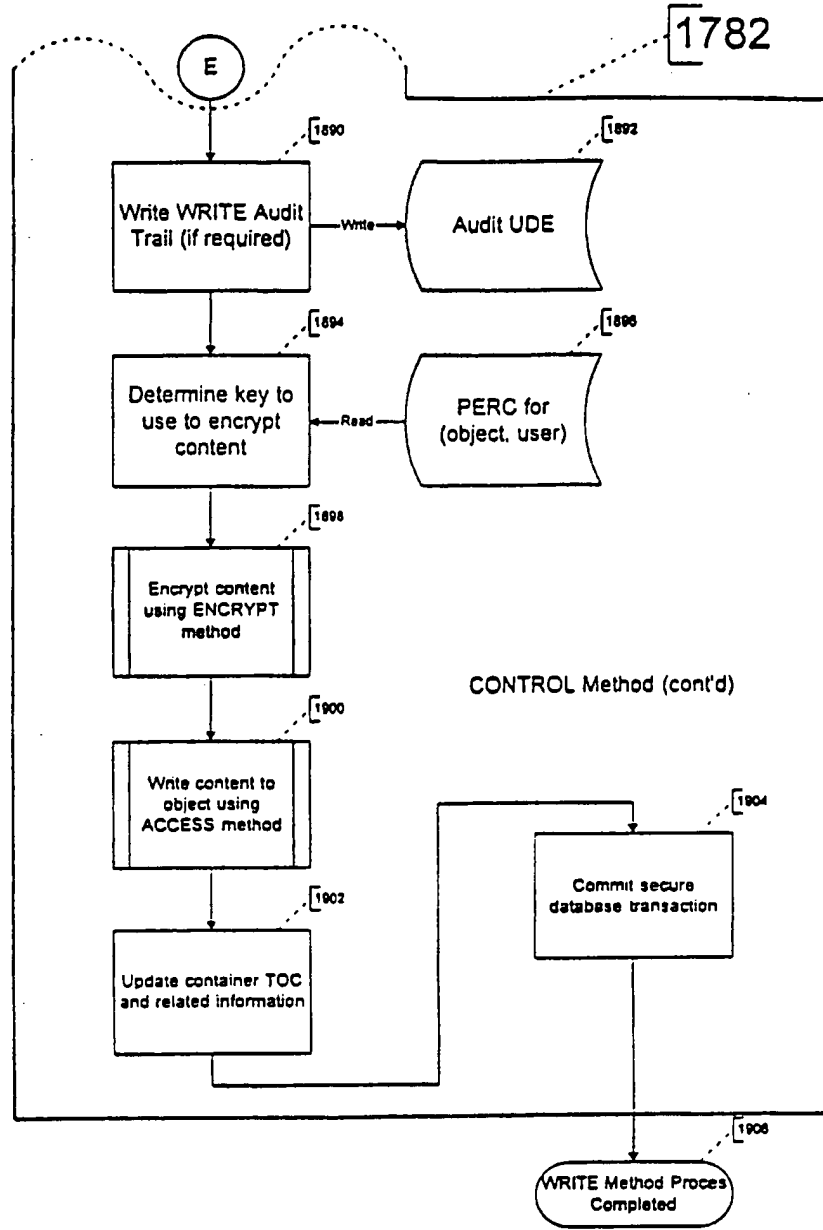


Figure 51f

98/146

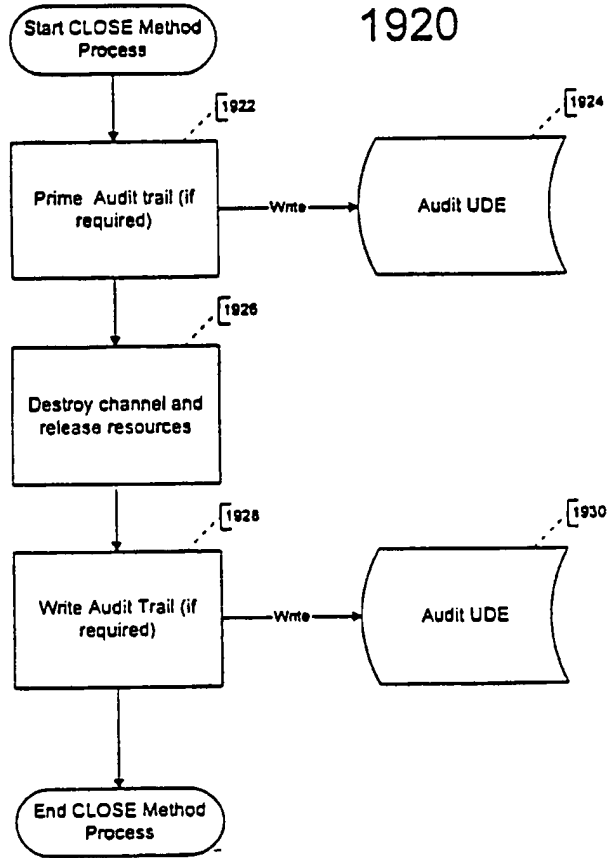


Figure 52

99/146

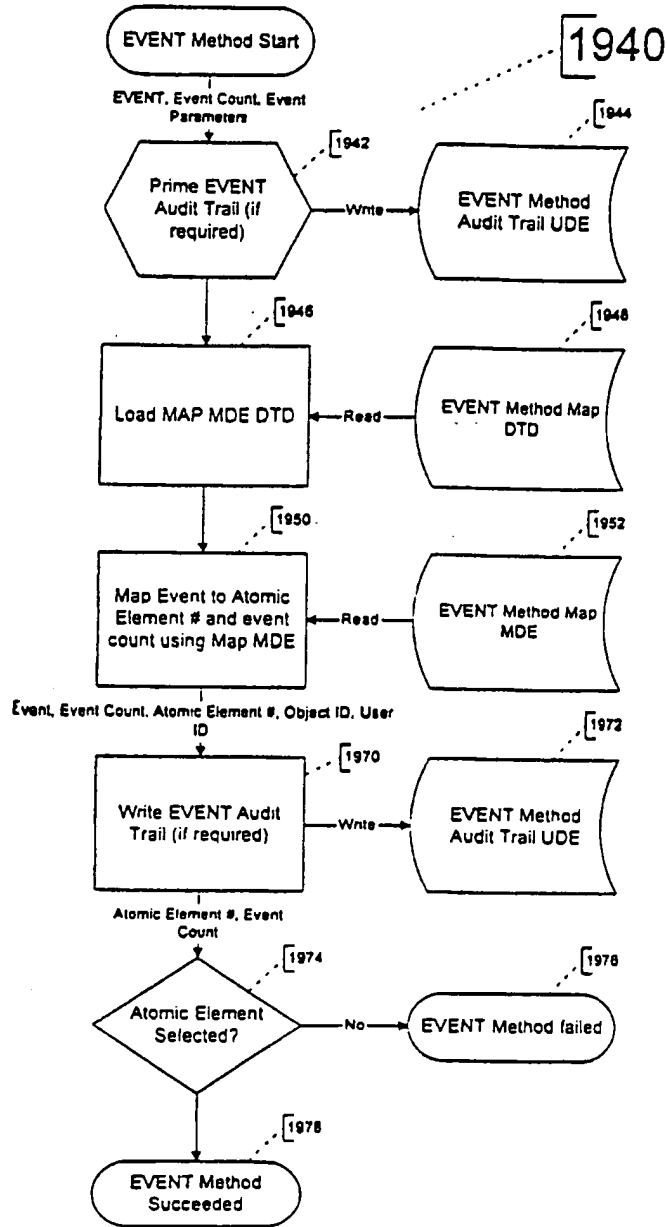


Figure 53a

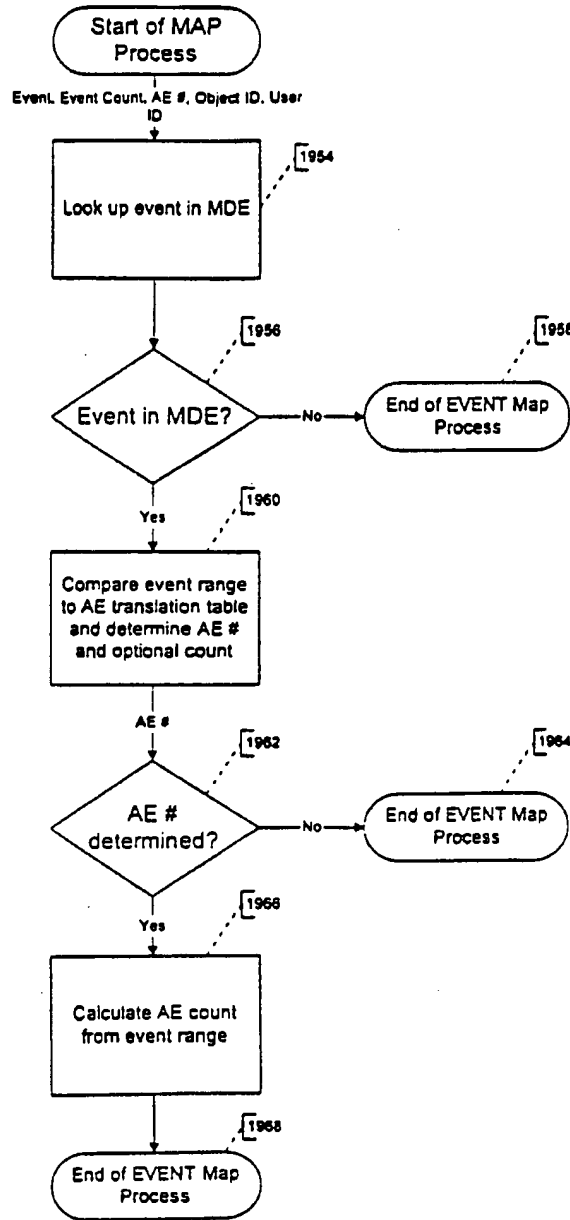


Figure 53b

101/146

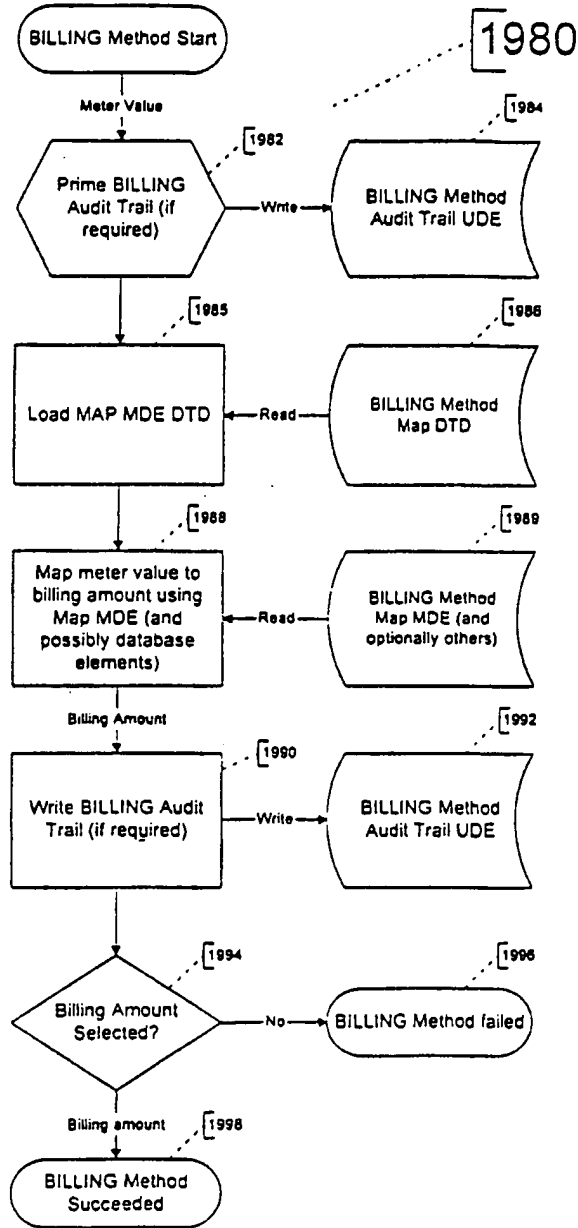


Figure 53c

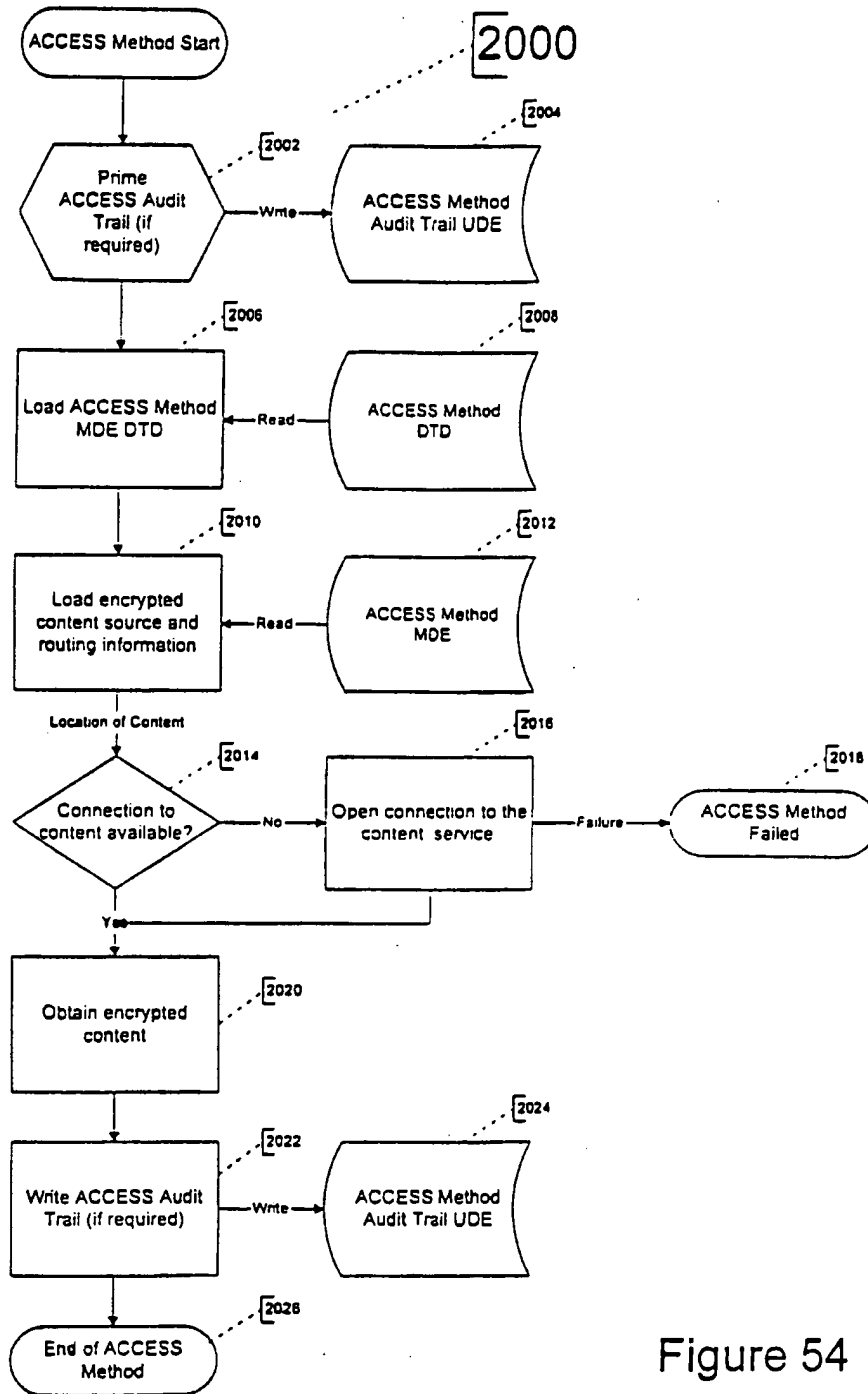


Figure 54

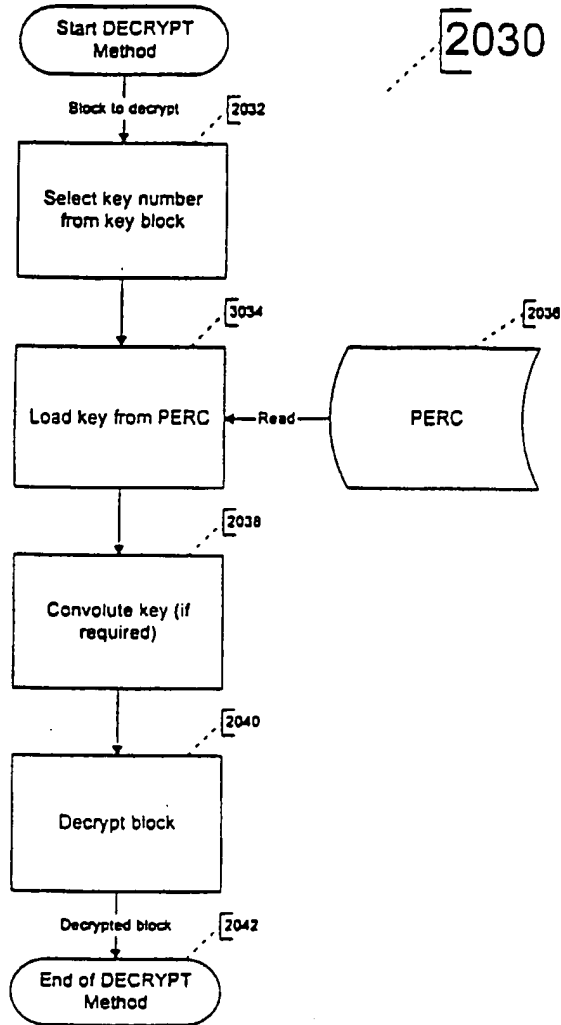


Figure 55a

104/146

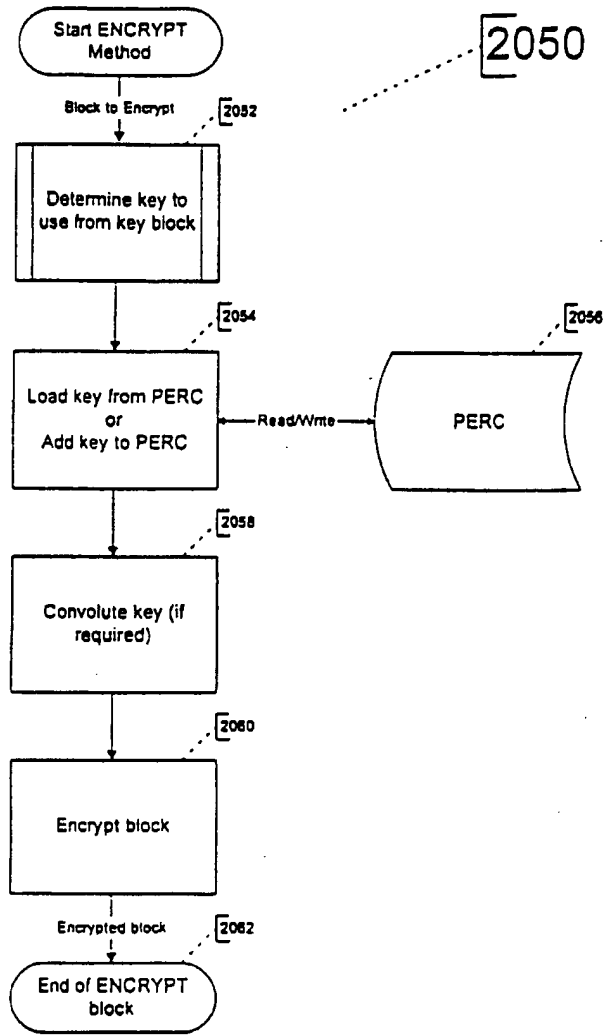


Figure 55b

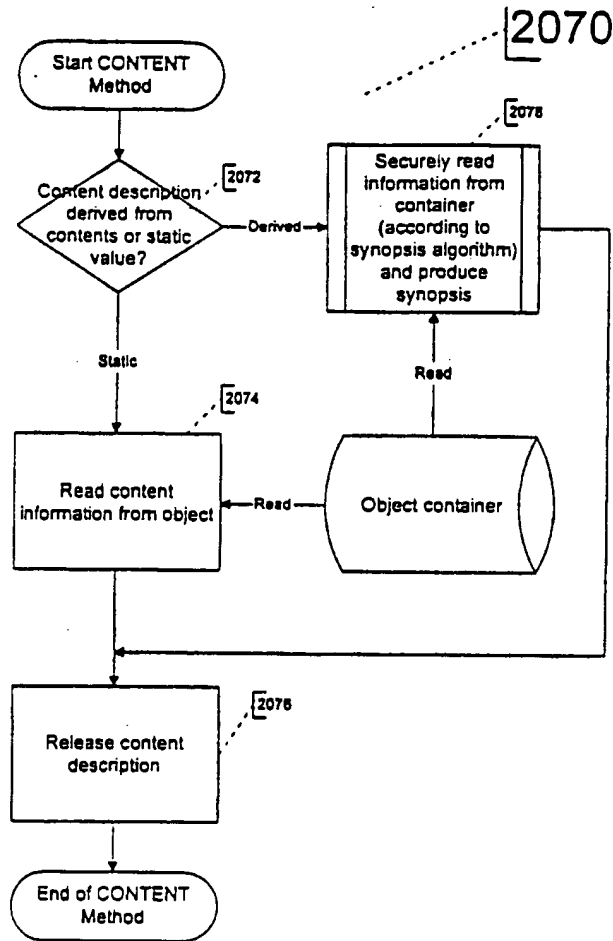


Figure 56

106/146

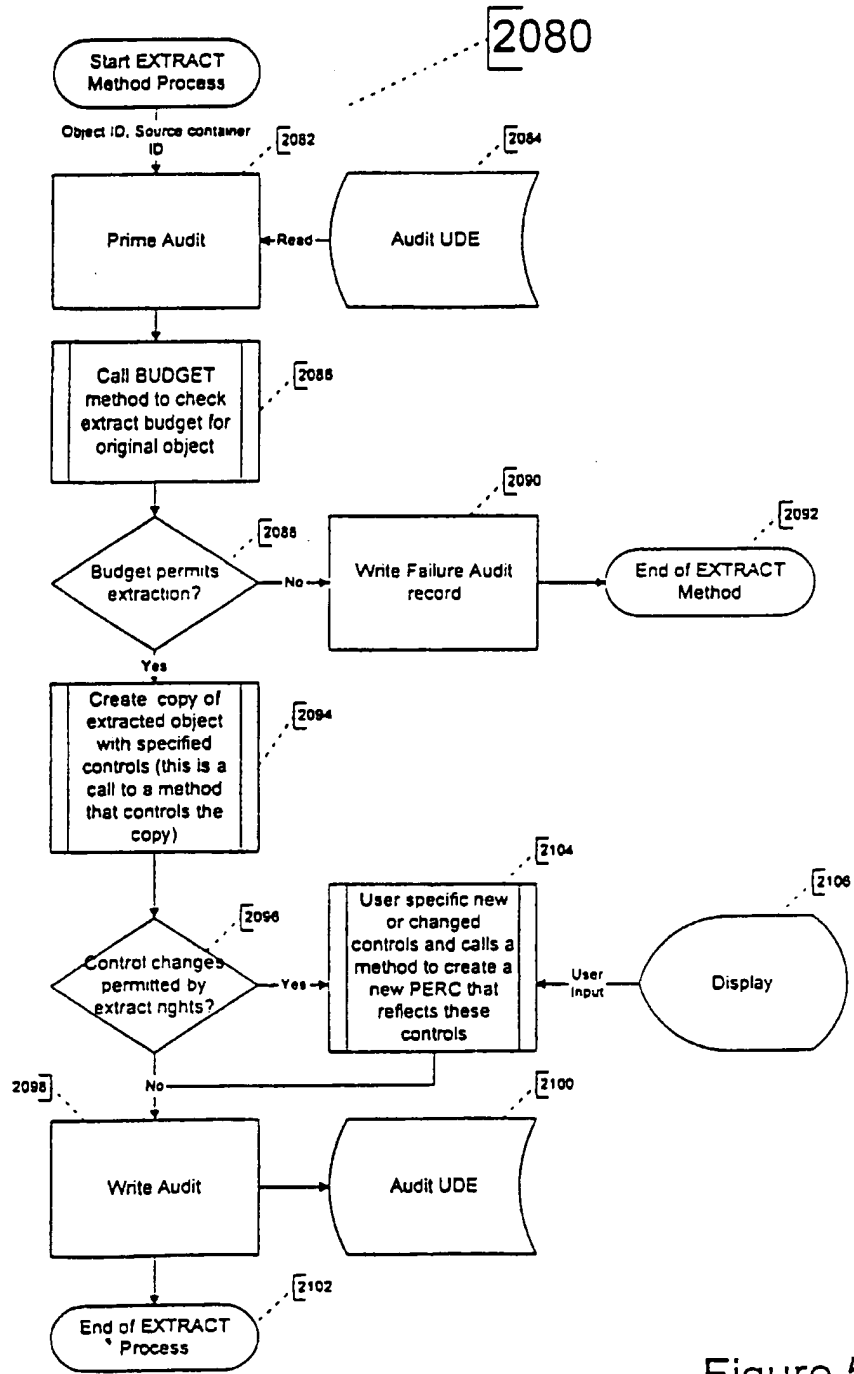


Figure 57a

107/146

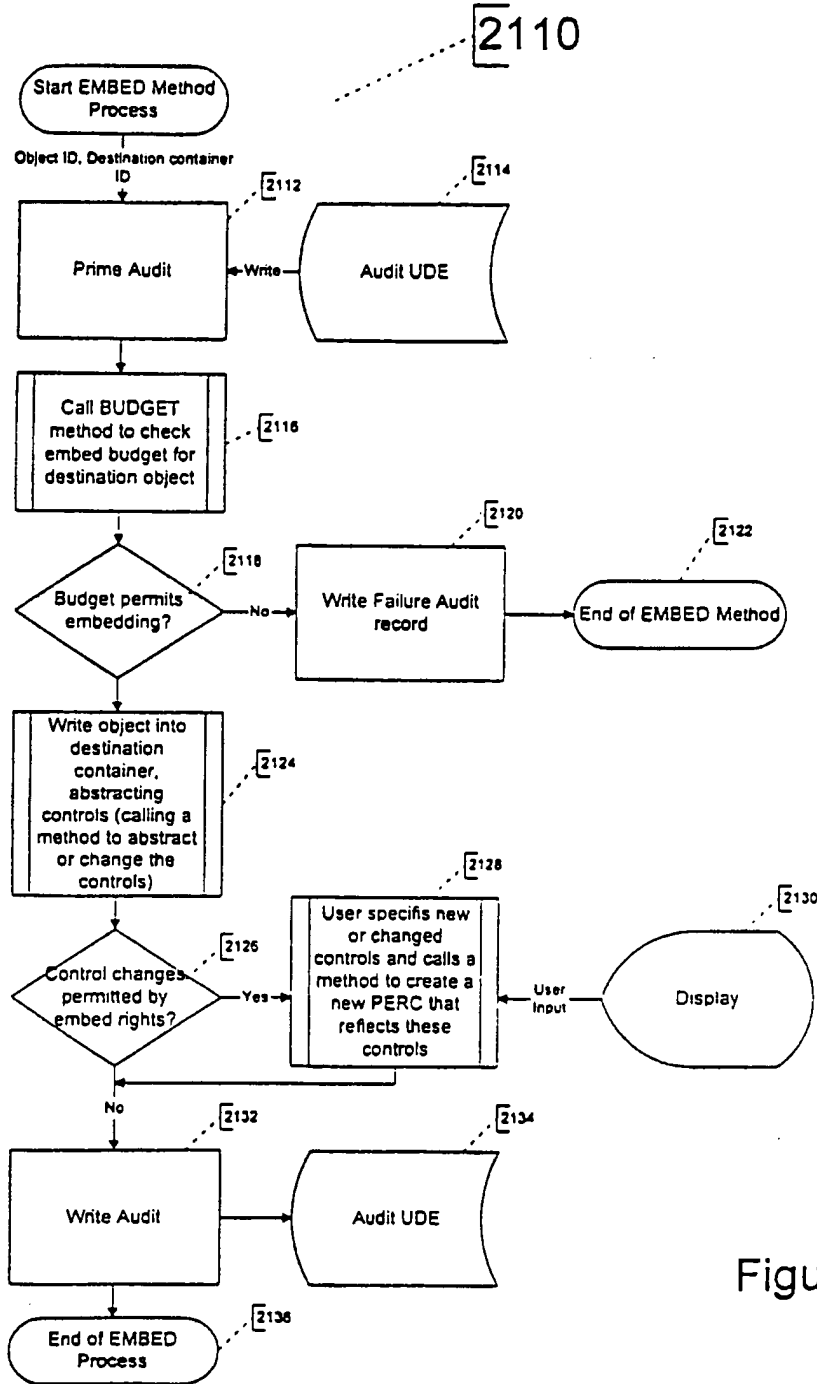


Figure 57b

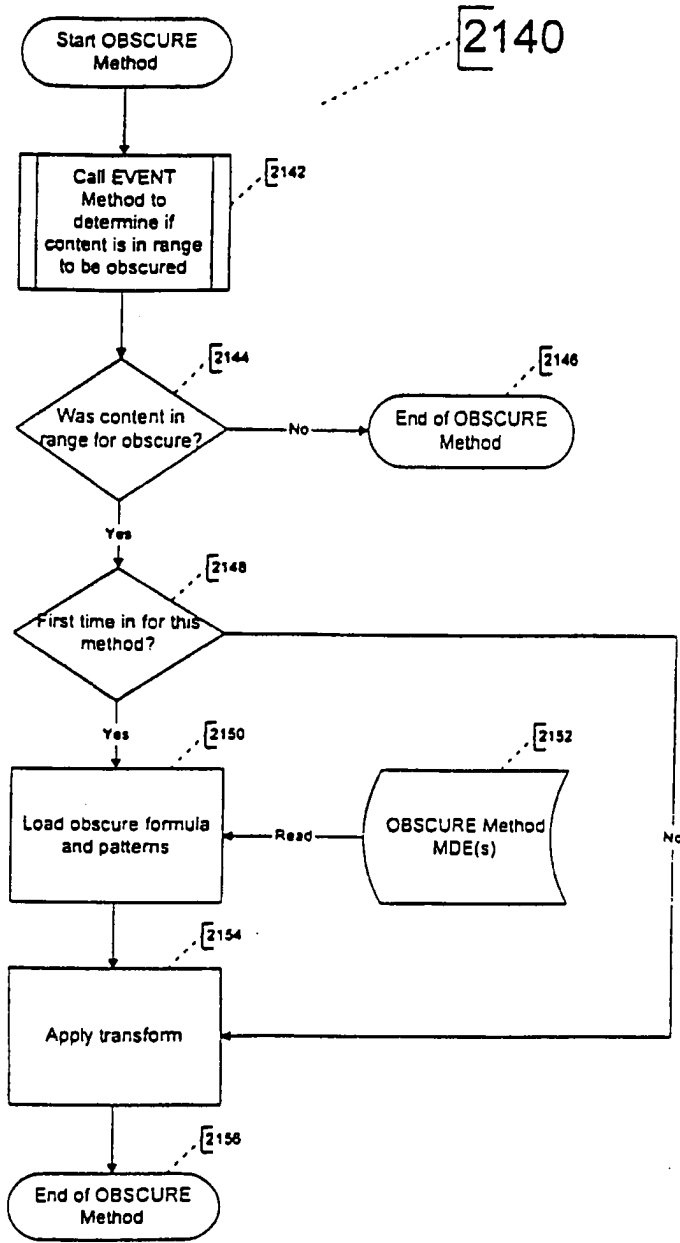


Figure 58a

109/146

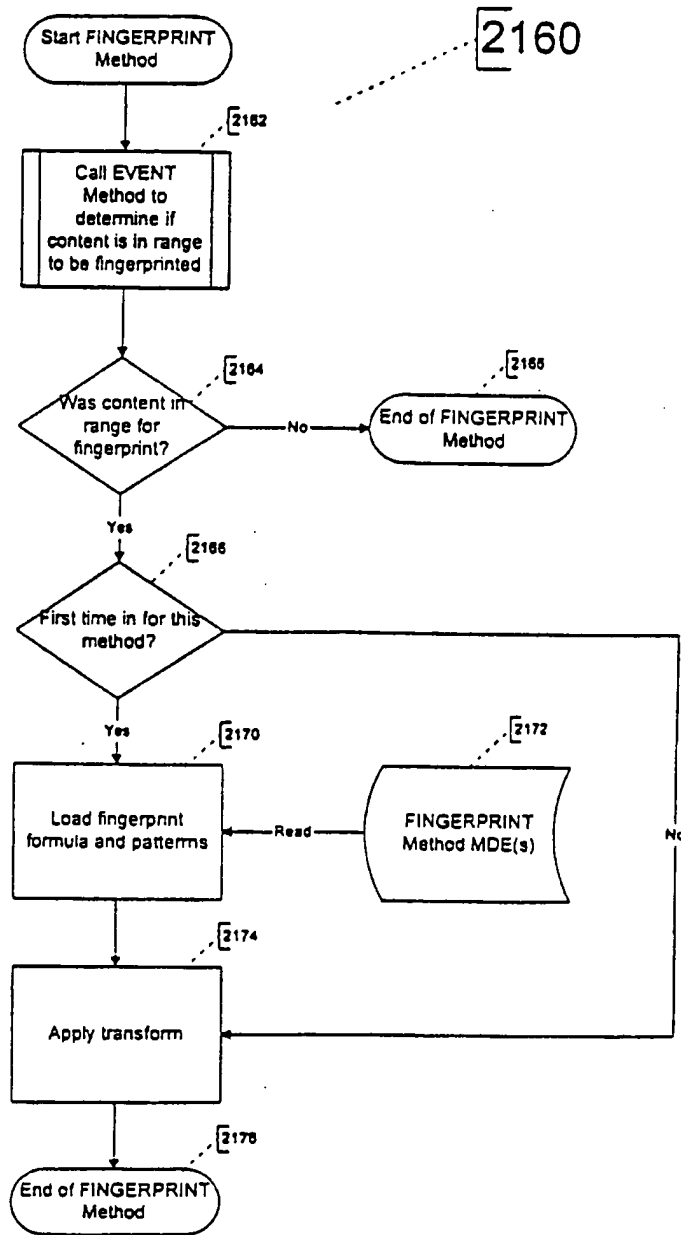


Figure 58b

110/146

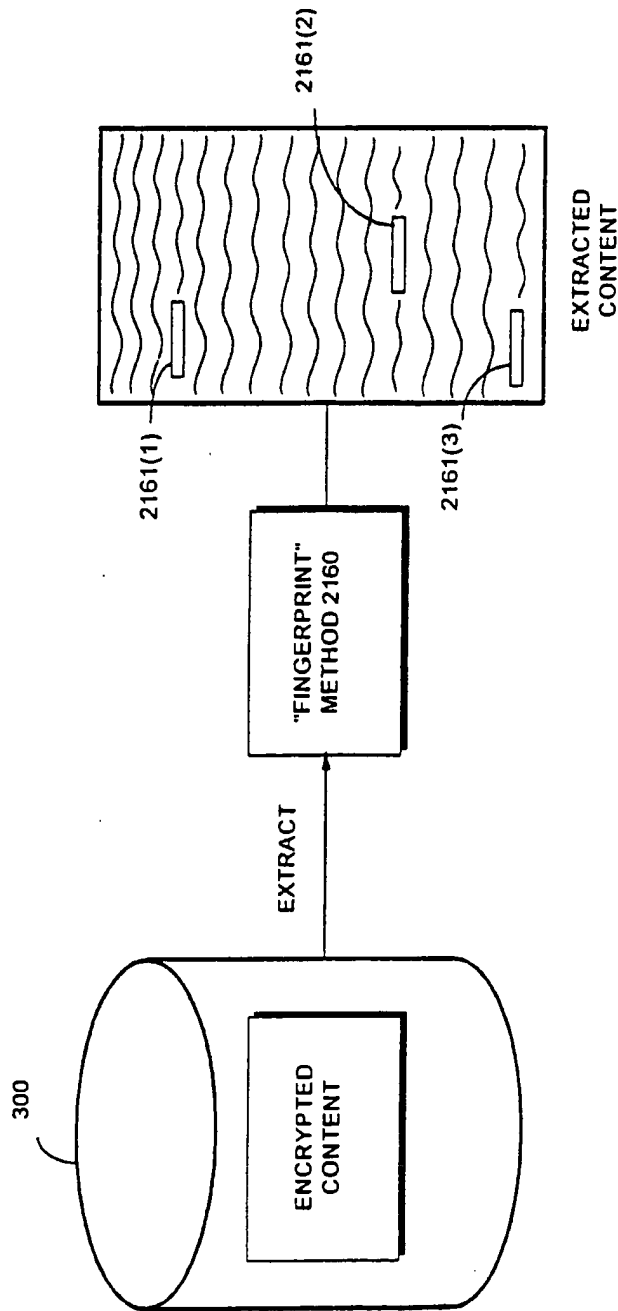


FIG. 58C

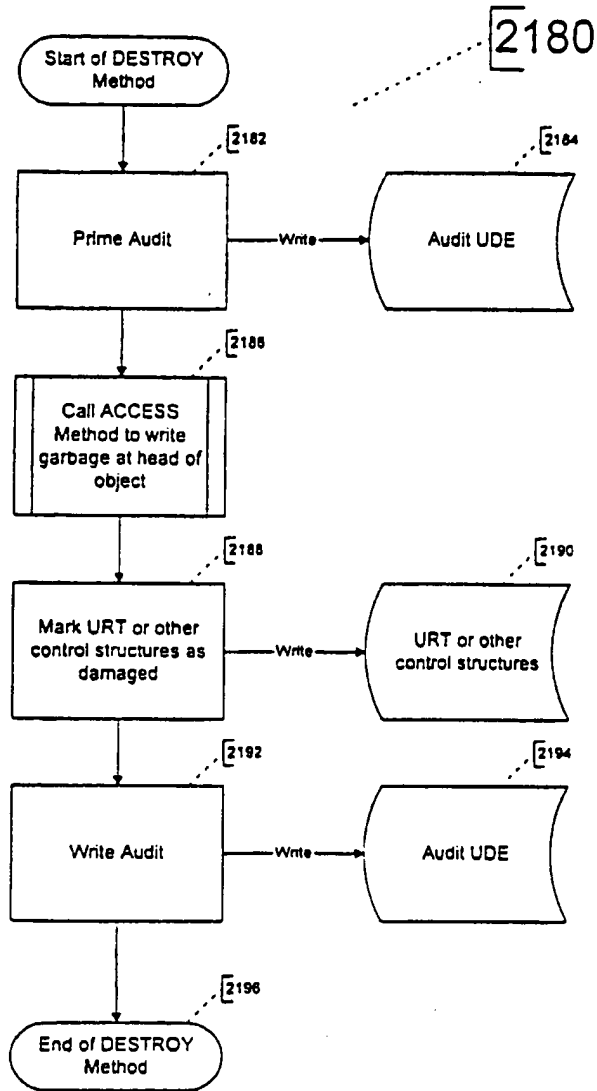


Figure 59

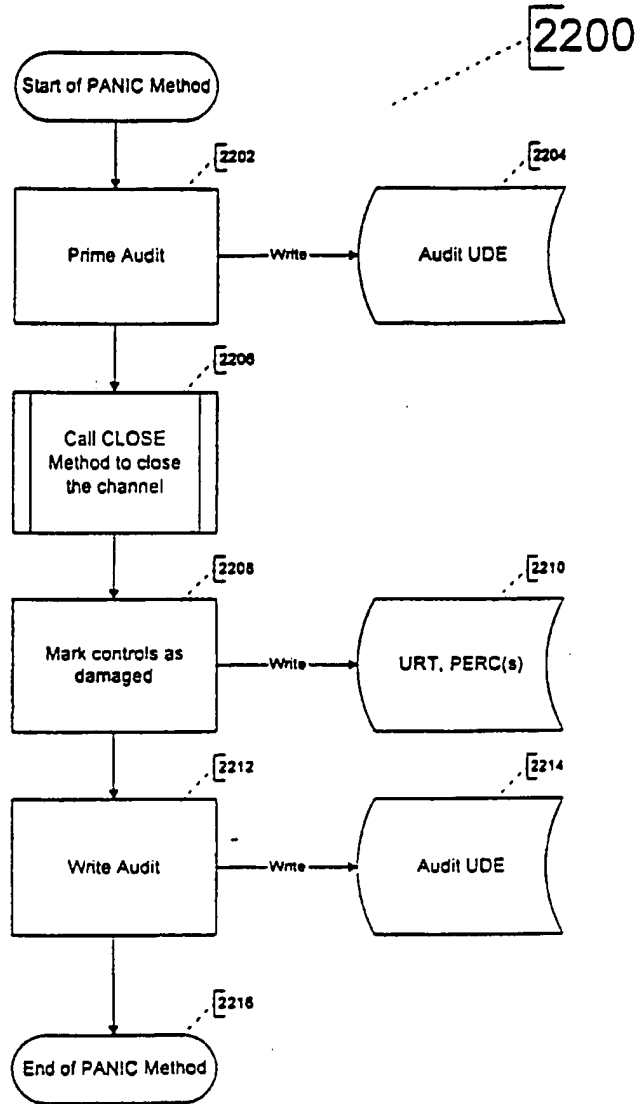


Figure 60

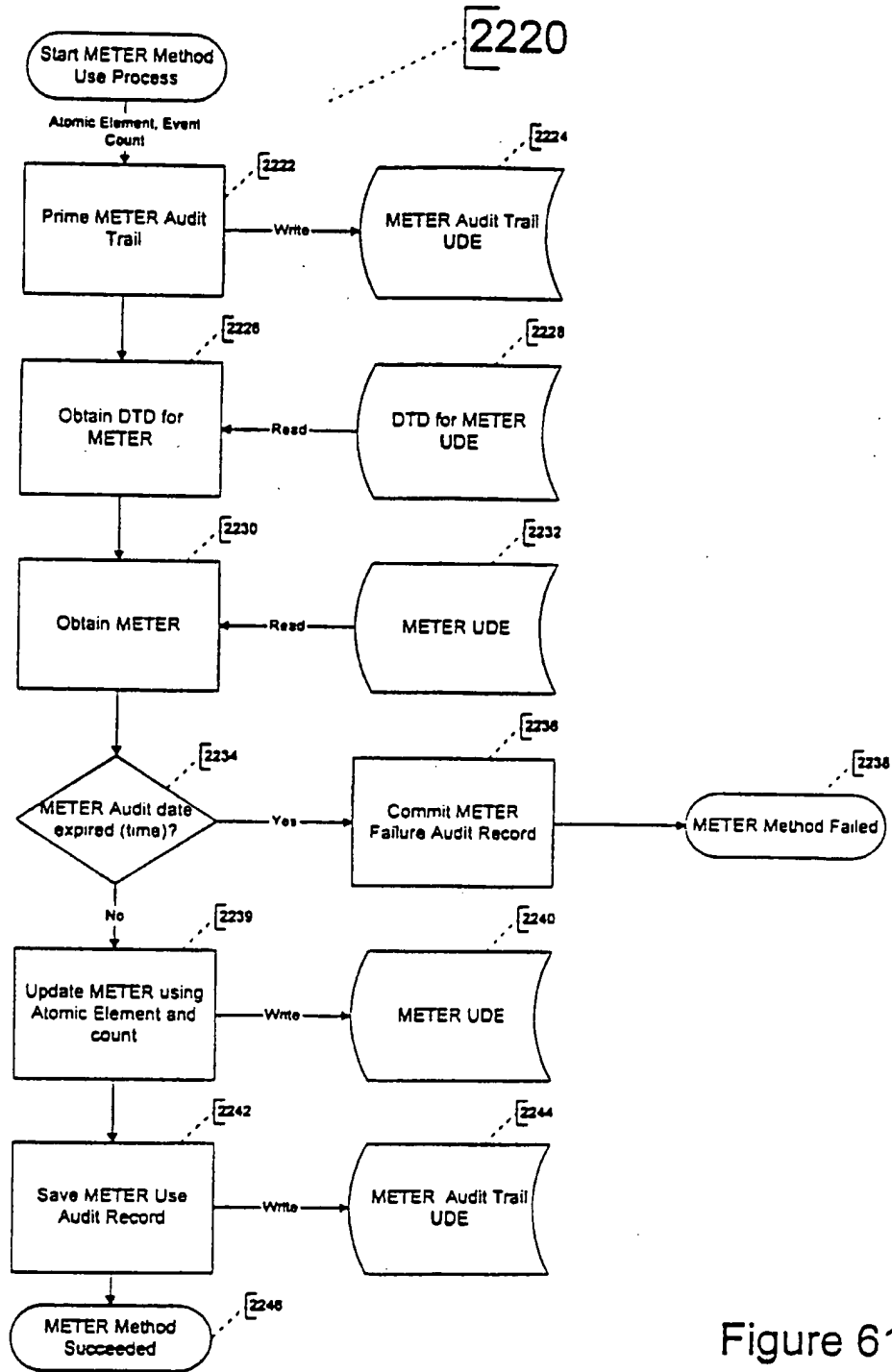
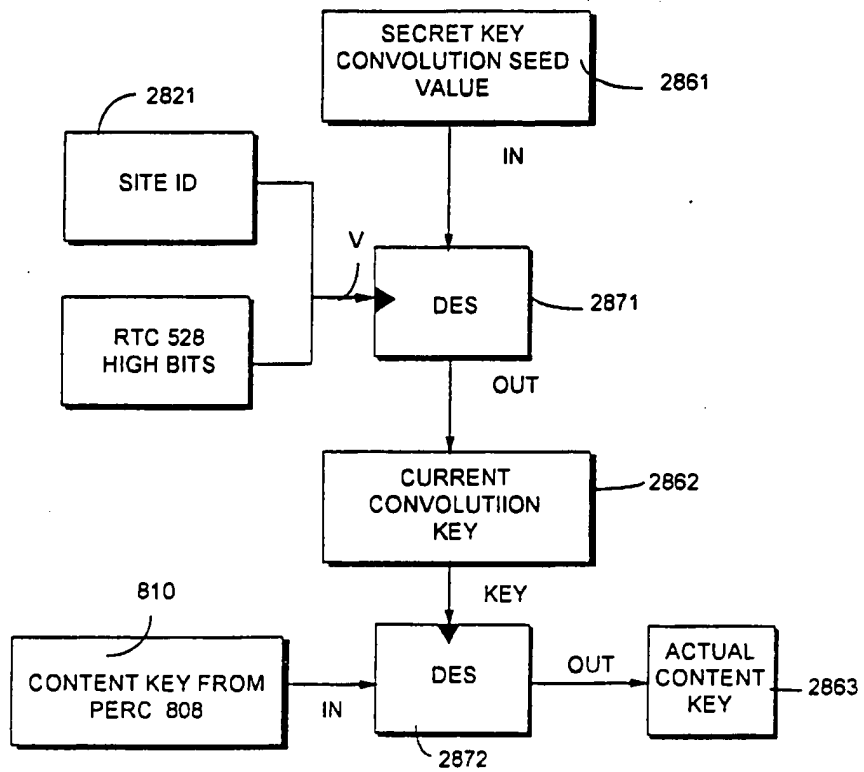


Figure 61

114/146

FIG. 62



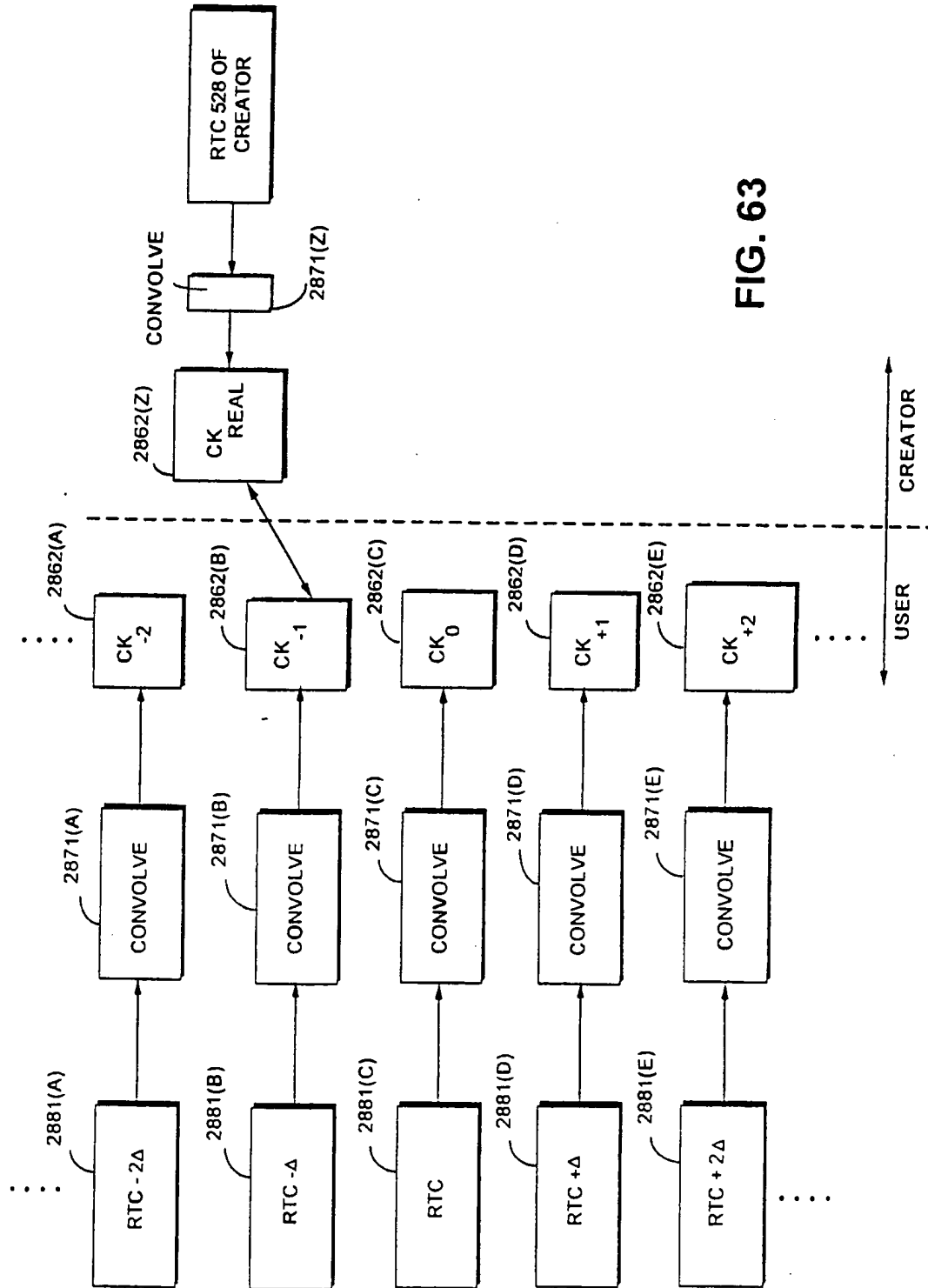
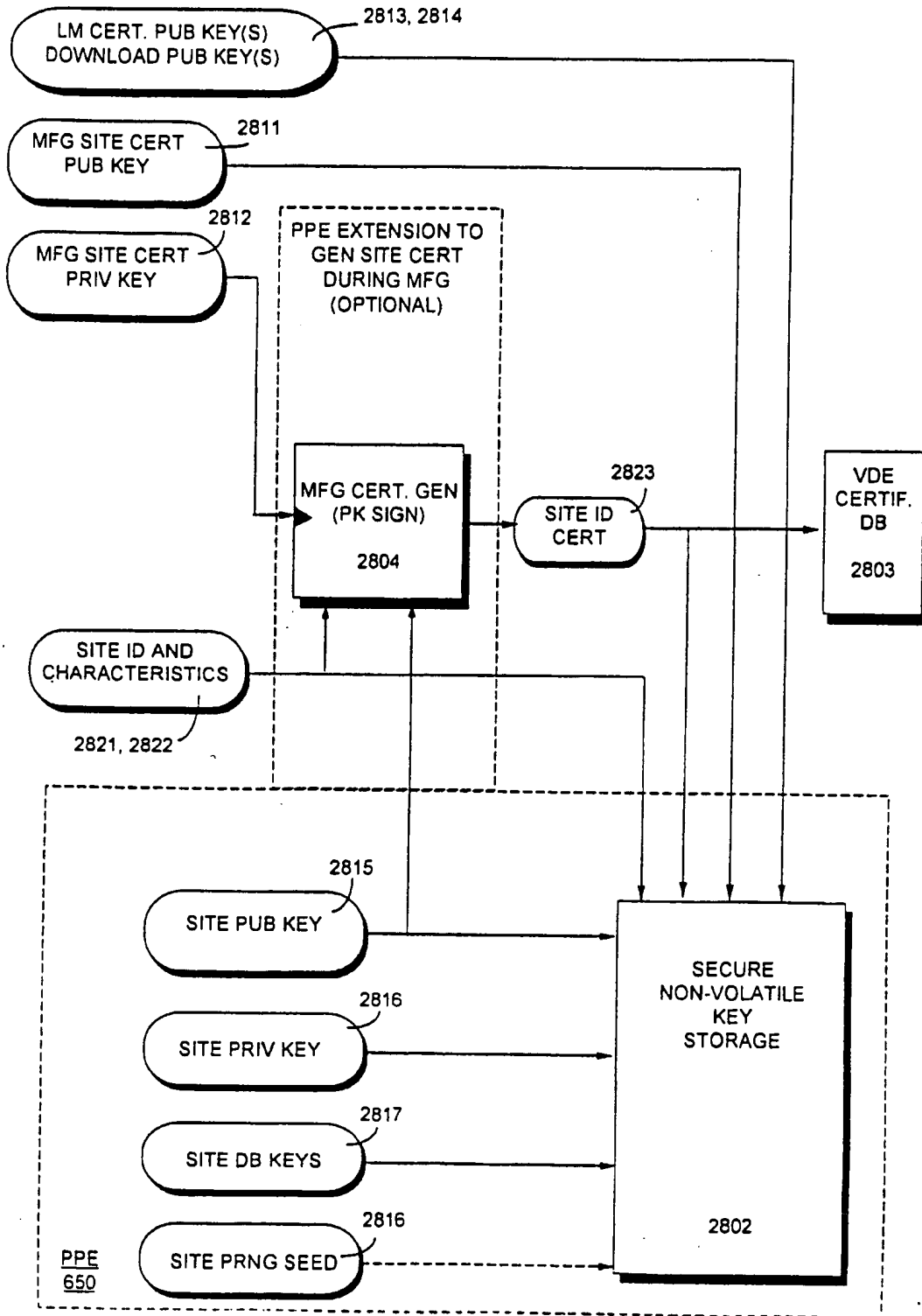


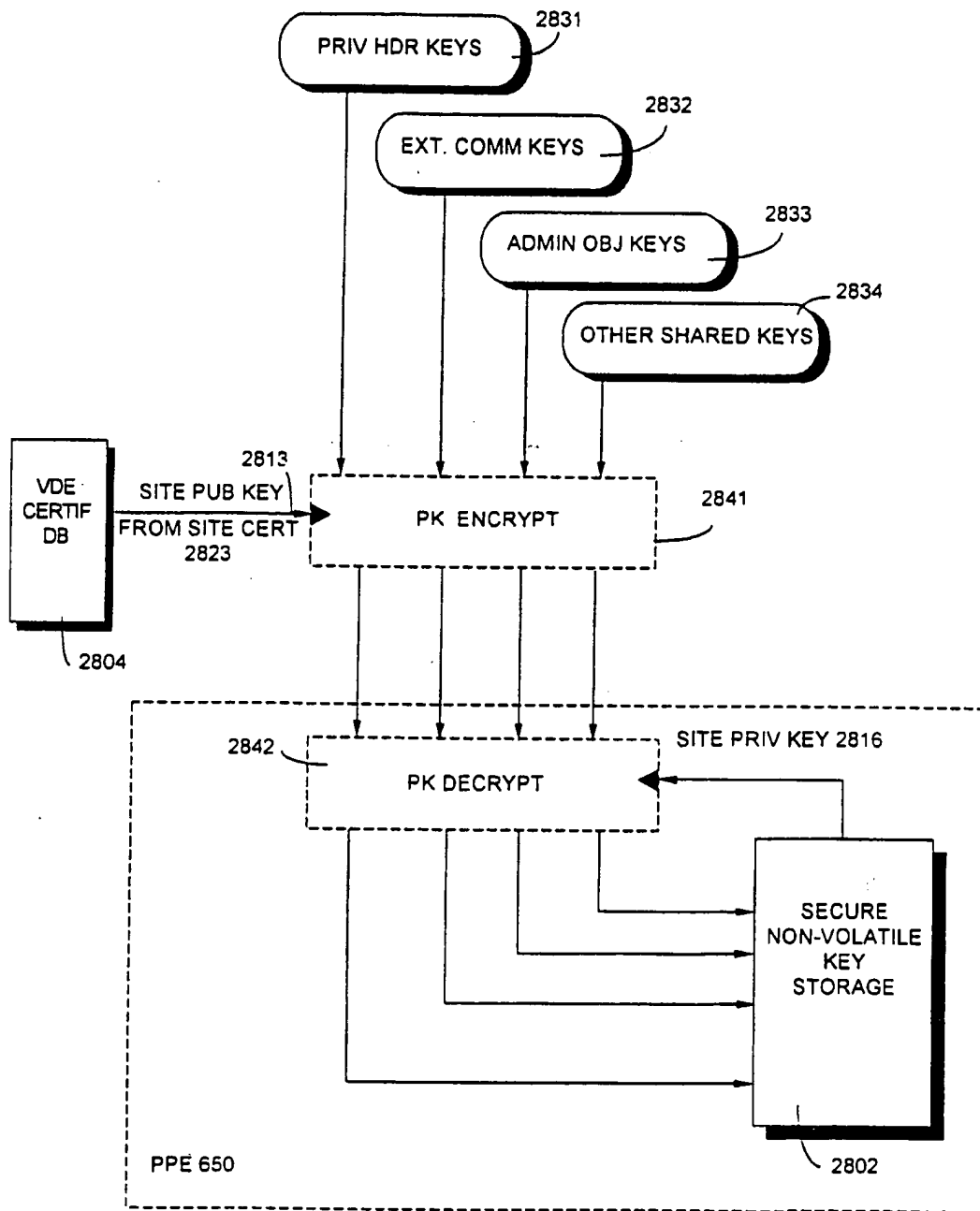
FIG. 63

FIG. 64



117/146

FIG. 65



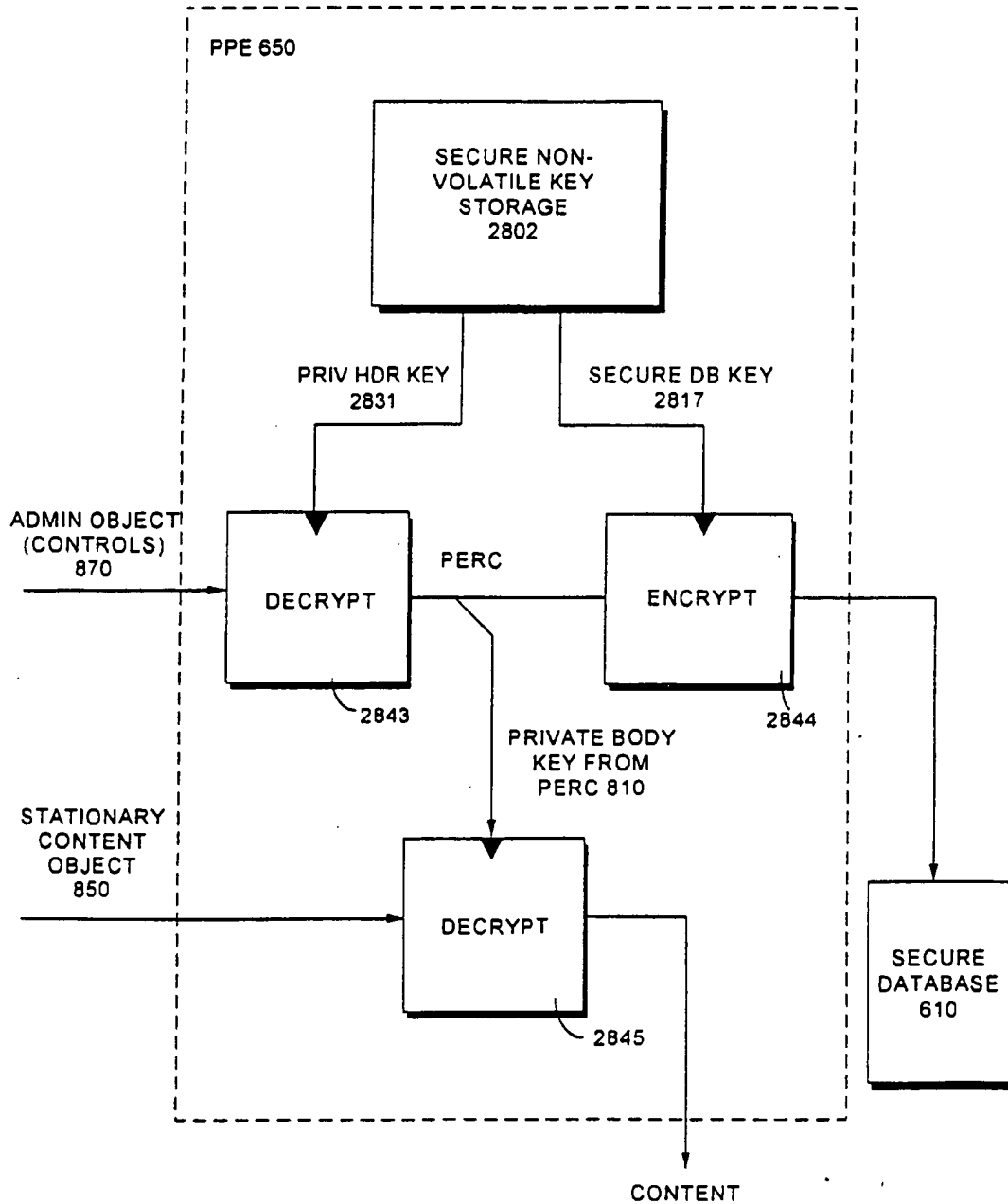


FIG. 66

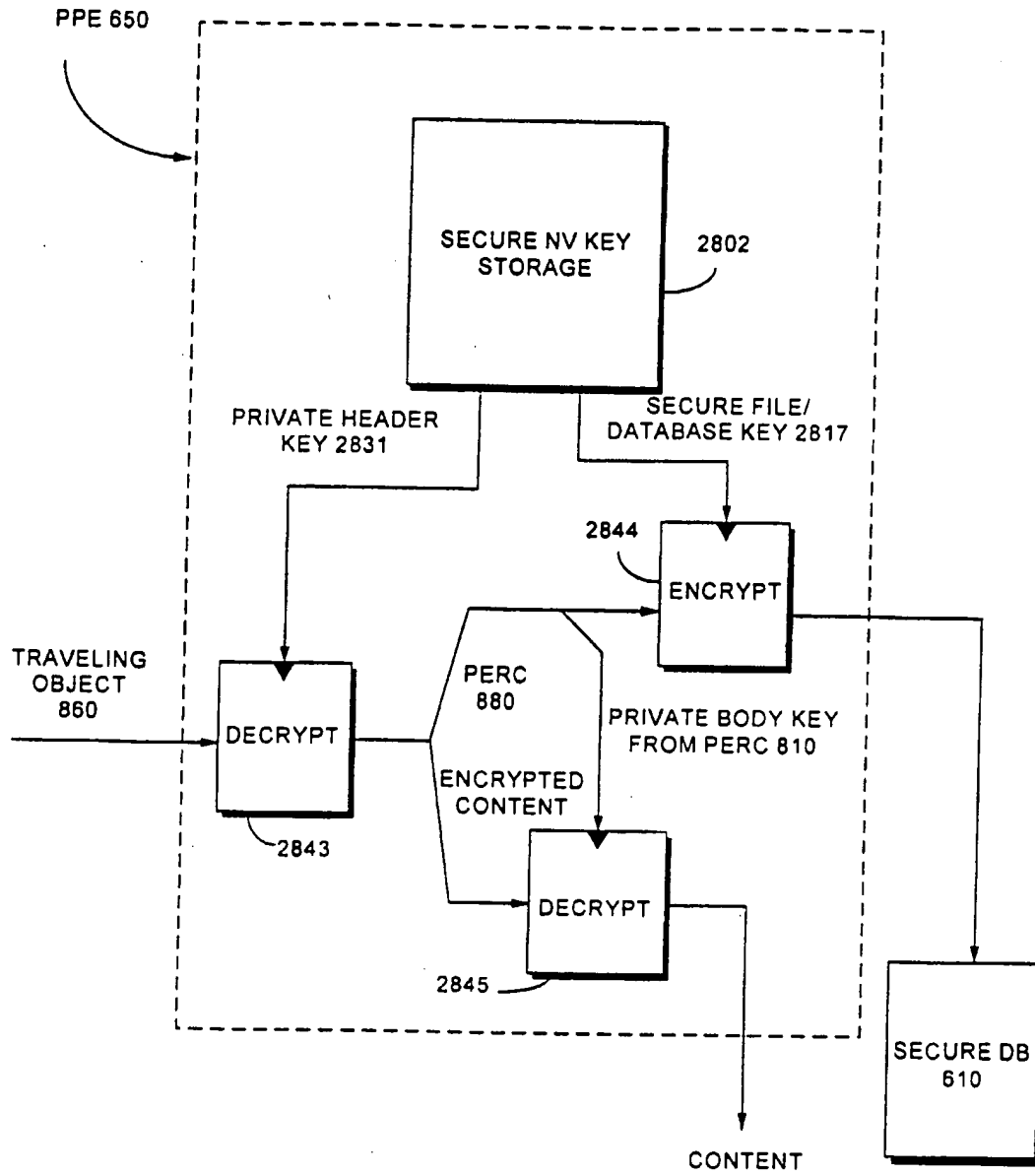
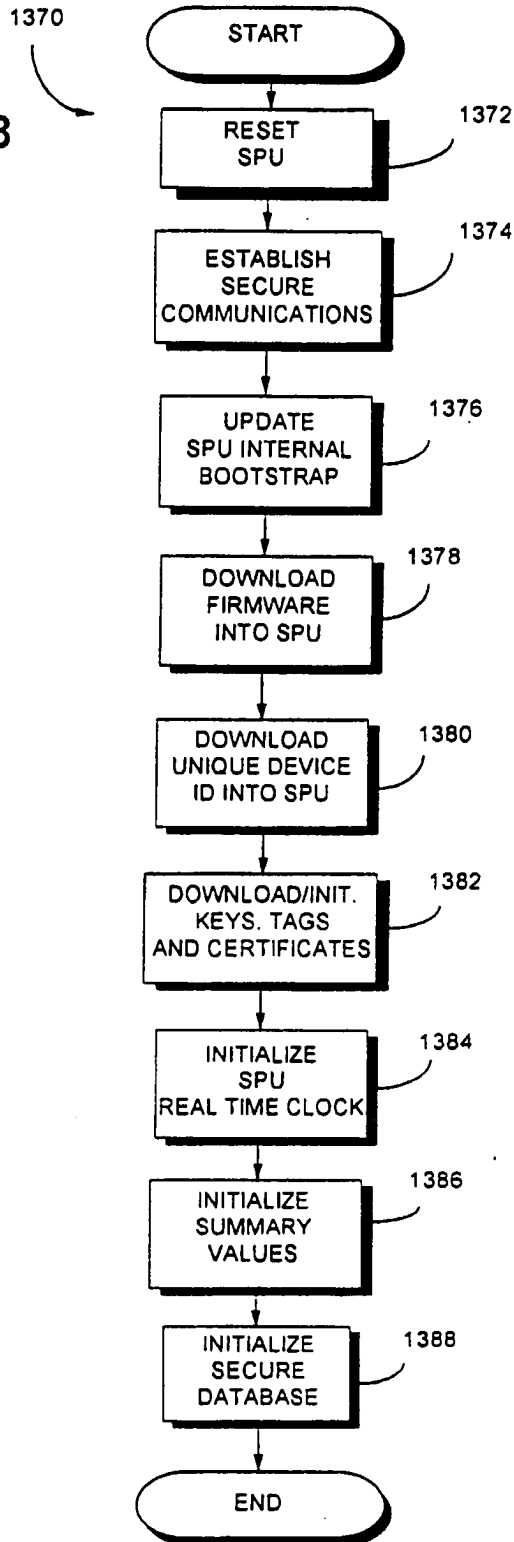


FIG. 67

120/146

FIG. 68



SUBSTITUTE SHEET (RULE 26)

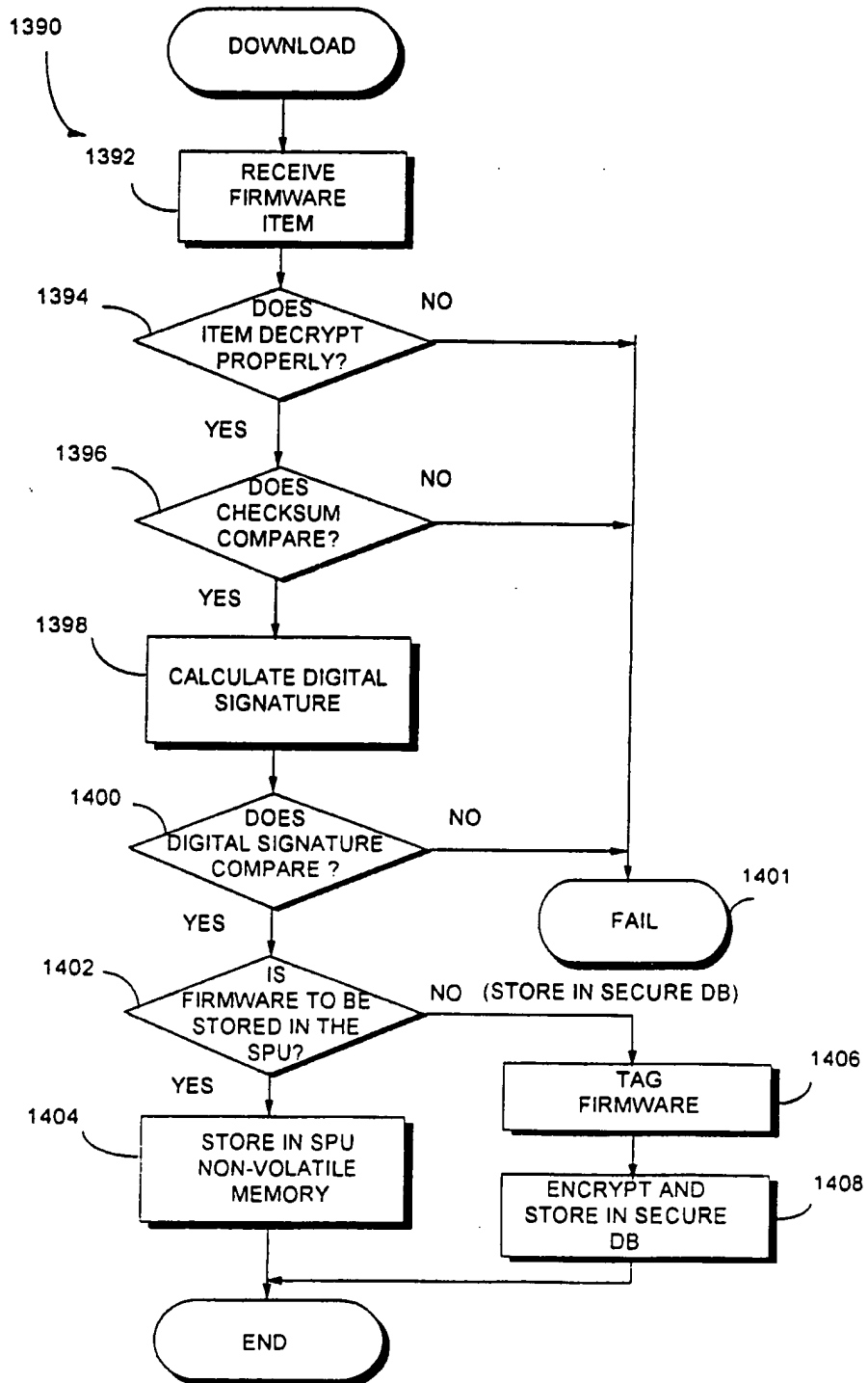


FIG. 69

122/146

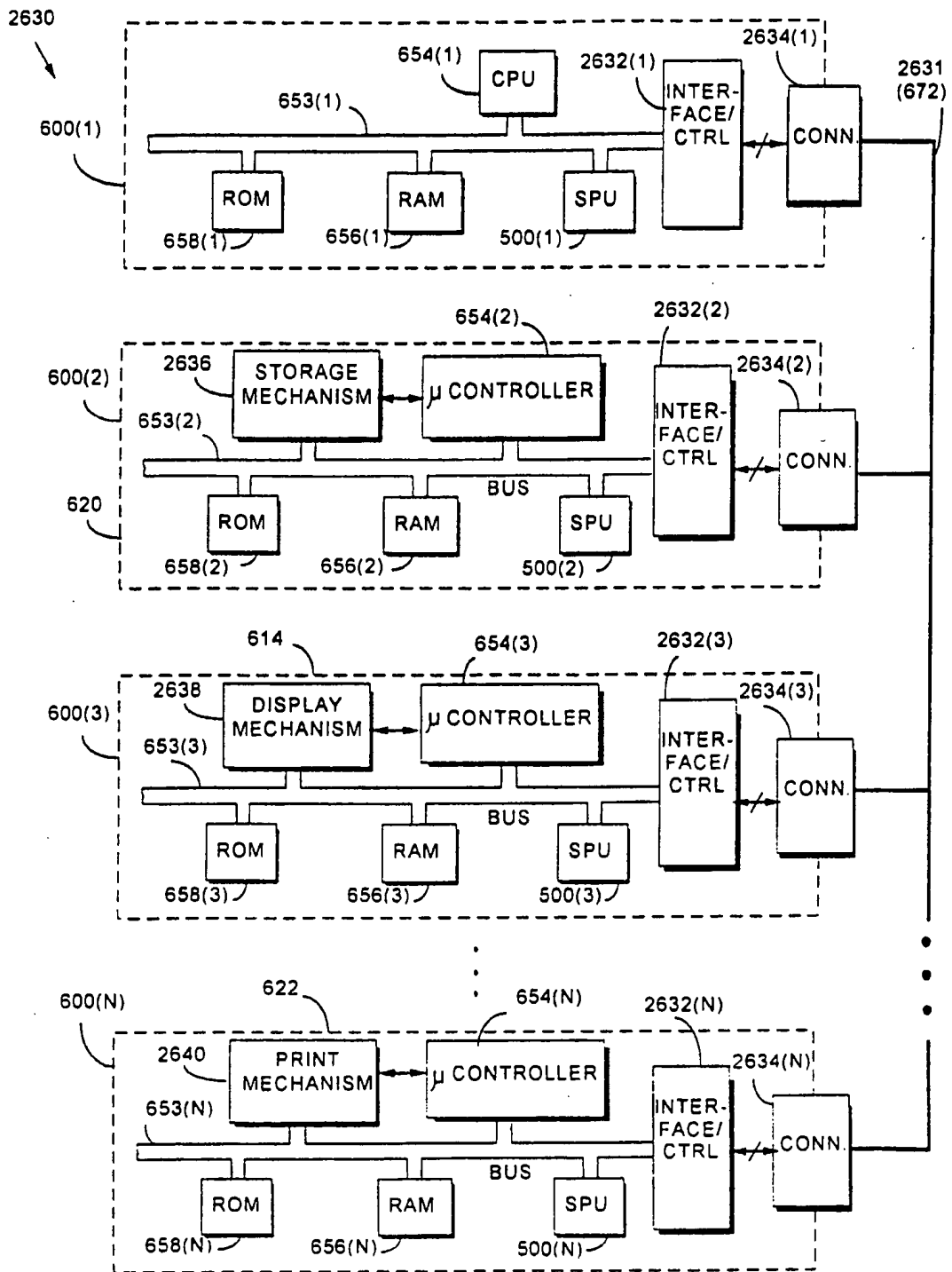
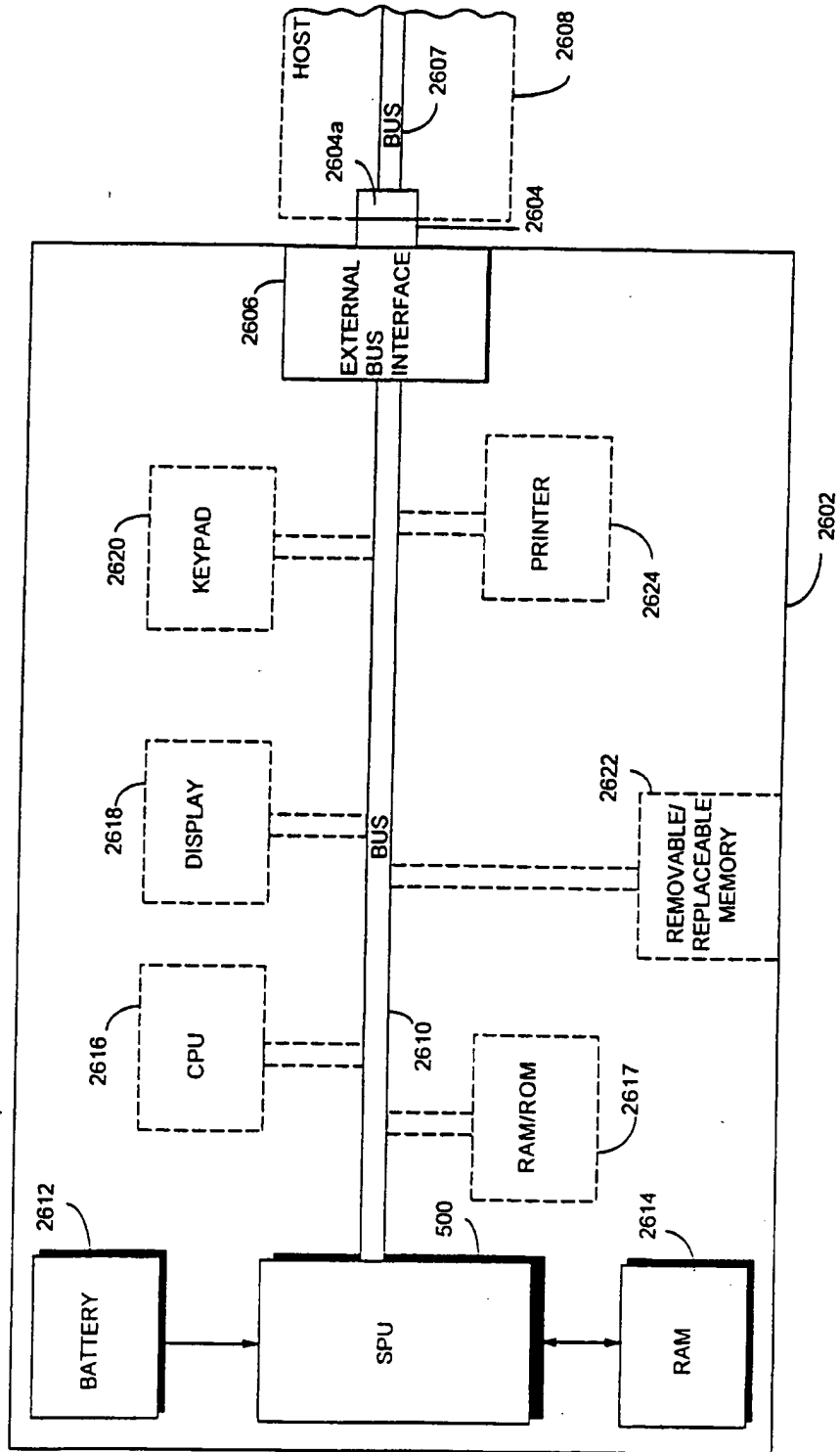


FIG. 70

SUBSTITUTE SHEET (RULE 26)

123/146

FIG. 71



124/146



LOG IN USER INTERFACE 182

USER NAME:	<input type="text" value="SHEAR. V."/>	<input type="button" value="LOGIN"/>
PASSWORD:	<input type="password" value="*****"/>	<input type="button" value="CANCEL"/>
<input type="checkbox"/> LOGIN AT STARTUP		<input type="button" value="HELP"/>

FIG. 72A

FIG. 72B

2660

	YOU HAVE REQUESTED THESE PROPERTIES:	<input type="button" value="CANCEL"/>
<u>LOONEY TUNES NEWS!</u>	<input type="button" value="APPROVE"/> 2662	<input type="button" value="SUSPEND"/>
<input type="button" value="PROPERTY INFO"/>	Your Cost: \$7.50	MORE OPTIONS 

2664

FIG. 72C

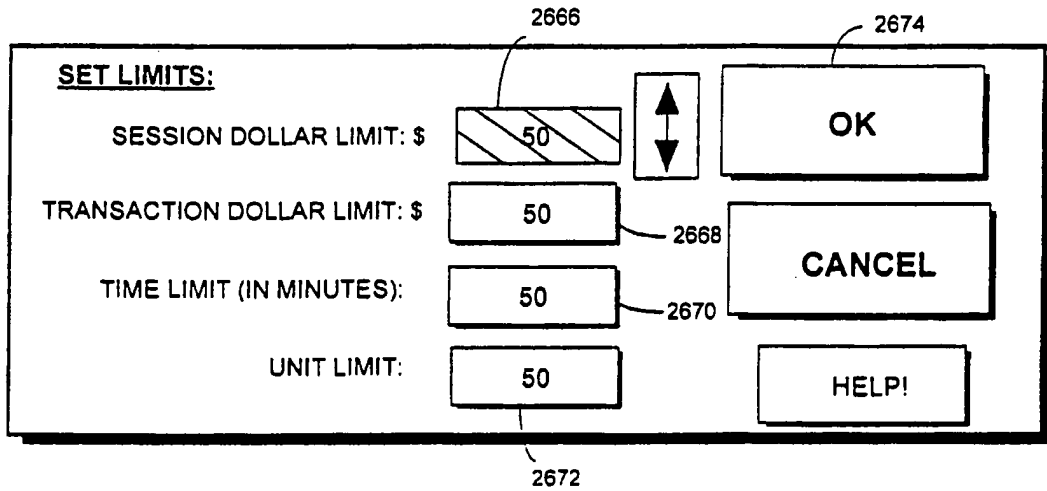



FIG. 72D



YOU HAVE REQUESTED THESE PROPERTIES:

LOONEY TUNE NEWS!

CANCEL

APPROVE

SUSPEND

PROPERTY INFO

YOUR COST : \$7.50

More Options

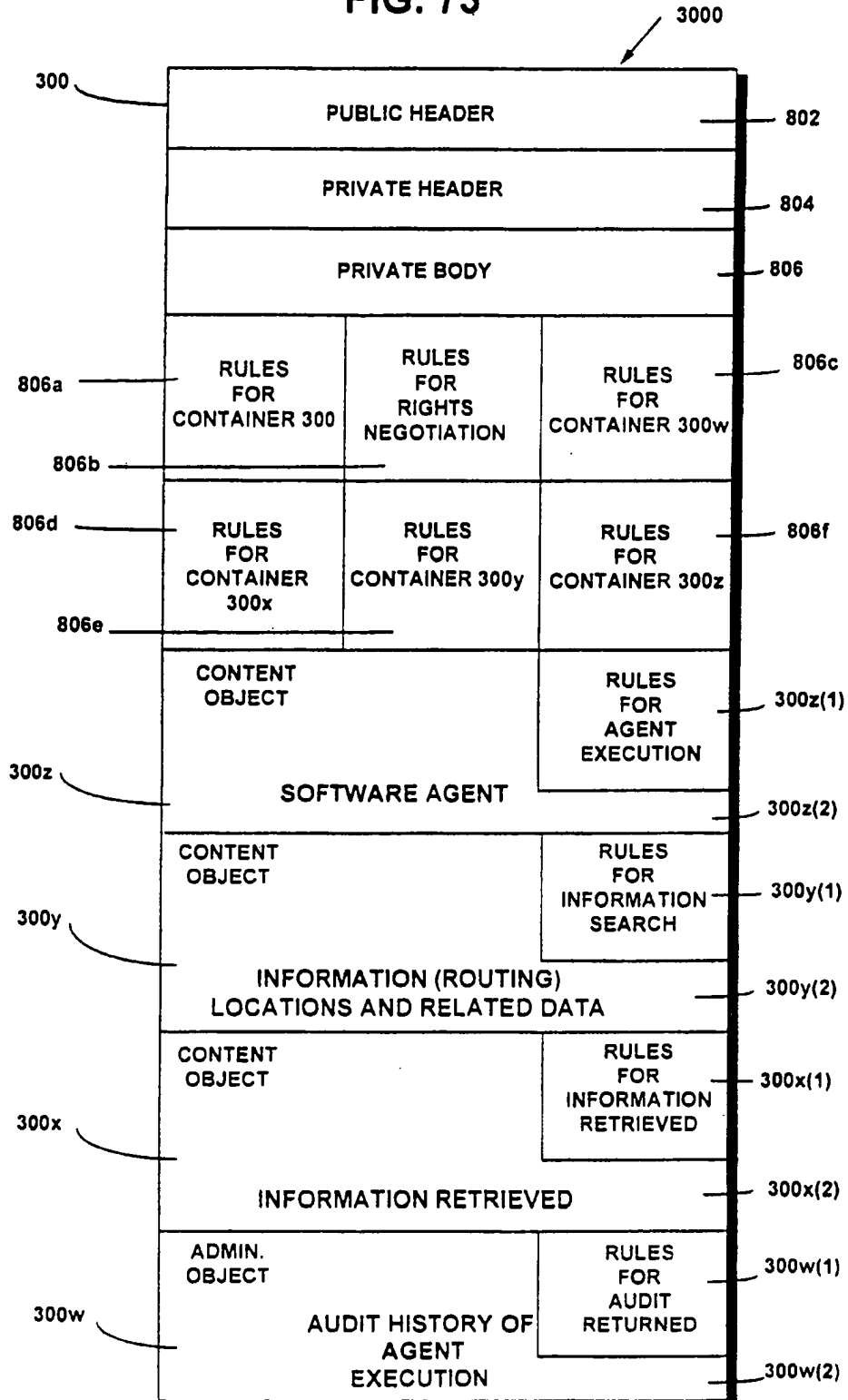
Show Thumbnail

PROPERTY	SIZE	PUBLISHER	AMOUNT	UNITS	COST/UNIT	TYPE	USE?	LINKS	HIST.
CHUCK JONES BIOGRA	256KB	WARNER NEW MEDIA	64	KBYTE	\$1.25	PREVIEW	<input checked="" type="checkbox"/>	●	
▼ BUGS BUNNY..JPE...	1MB	WARNER NEW MEDIA	1	RECORD	\$5.00	DISPLAY	<input checked="" type="checkbox"/>	●	▲
BUGS BUNNY JPEG...	1MB	WARNER NEW MEDIA	10	RECORD	\$3.50	DISPLAY		●	▲
BUGS BUNNY JPEG ..	1MB	WARNER NEW MEDIA	25	RECORD	\$2.50	DISPLAY		●	▲
FRIZ FRELENG BIOGRA	256KB	WARNER NEW MEDIA	120	SECTOR	\$5.00	PRINT			
TEX AVERY BIOGRAP	256KB	WARNER NEW MEDIA	50	PERCENT	\$2.50	COPY			▲
▶ DUCKI RABBITI DU...	64MB	WARNER NEW MEDIA	7.0	MINUTE	\$7.50	COPY-PRO			
MEL BLANC BIOGRAPH	256KB	WARNER NEW MEDIA	1	SPECIAL	\$25.25	INSTALL			▲
LOONEY TUNES DATAB	600MB	WARNER NEW MEDIA	1	OBJECT	\$2000.00	ALL			●

SET LIMITS...
SHOW BUDGETS
ACQUIRE BUDGET...
HISTORY...
TRANSFER...
PREFERENCES...
FEEDBACK...
HELP!

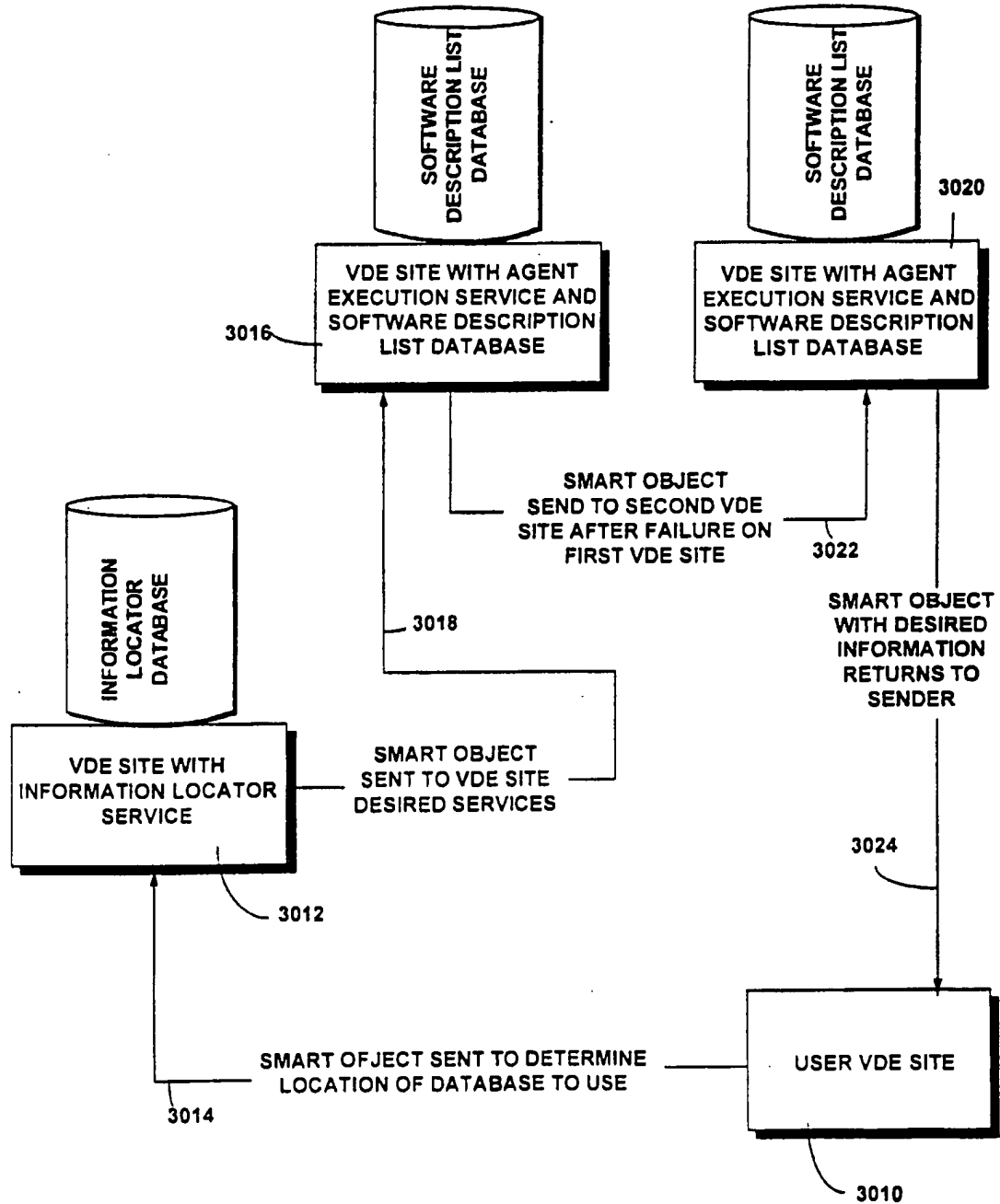
127/146

FIG. 73



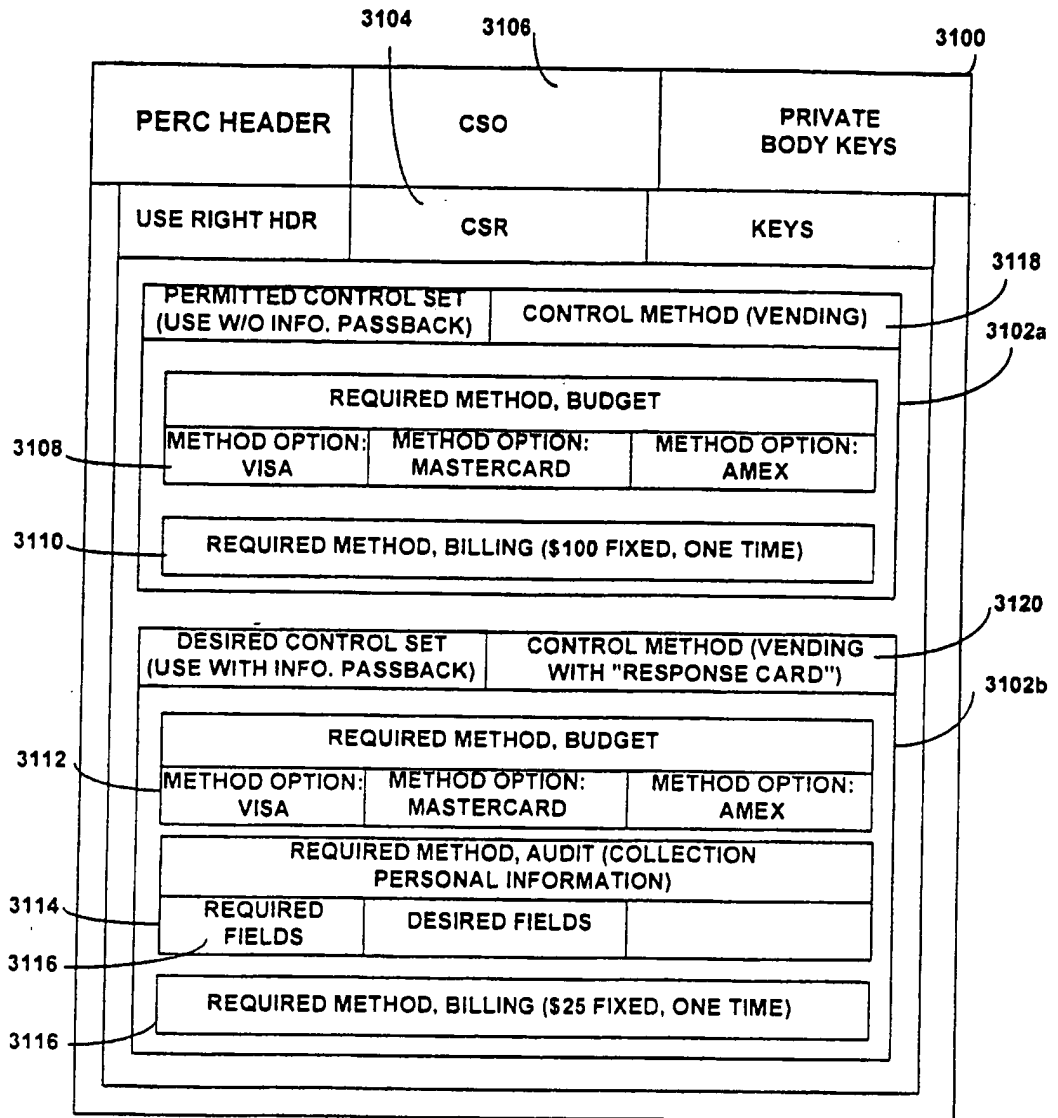
SUBSTITUTE SHEET (RULE 26)

FIG. 74



129/146

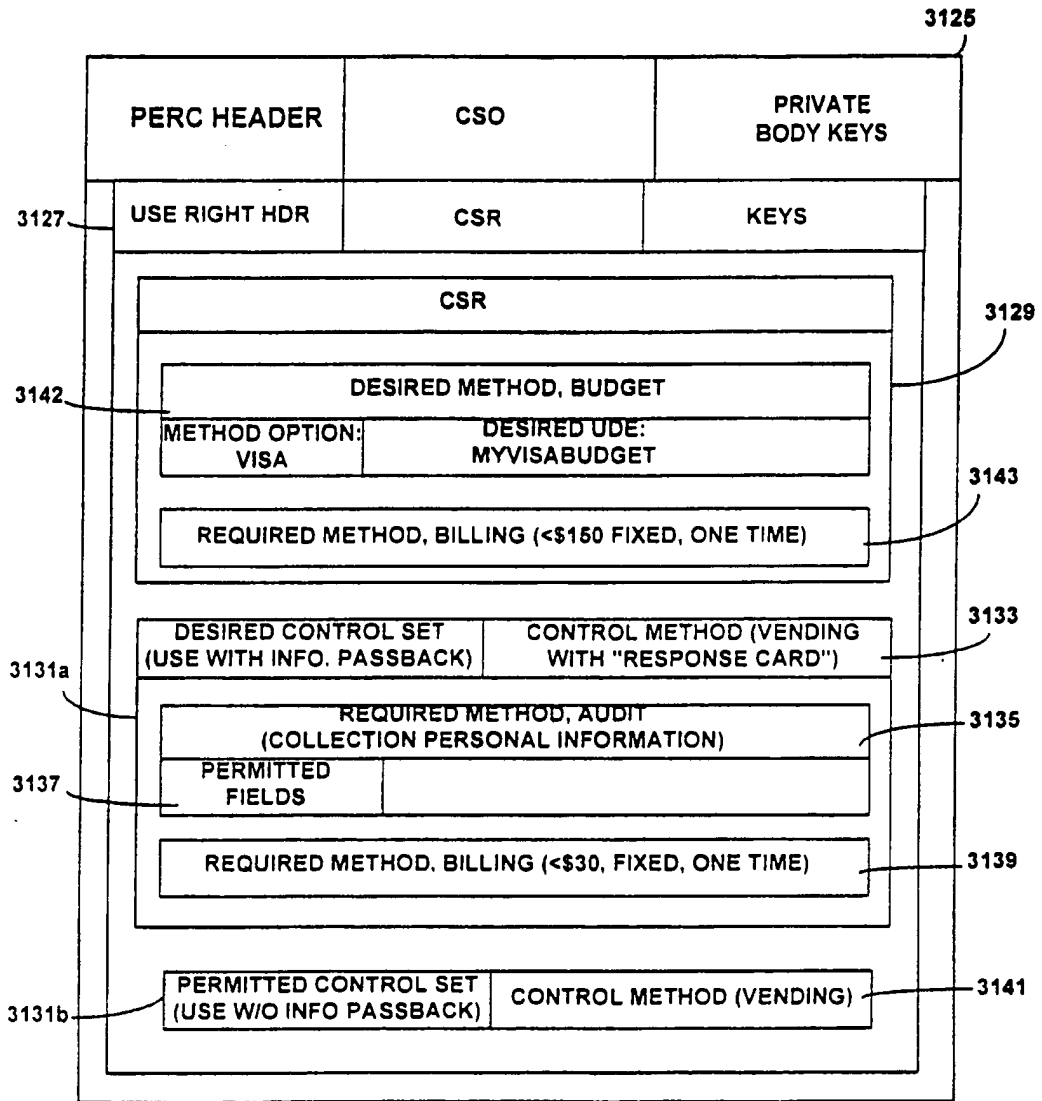
FIG. 75A



SUBSTITUTE SHEET (RULE 26)

130/146

FIG. 75B



SUBSTITUTE SHEET (RULE 26)

FIG. 75C

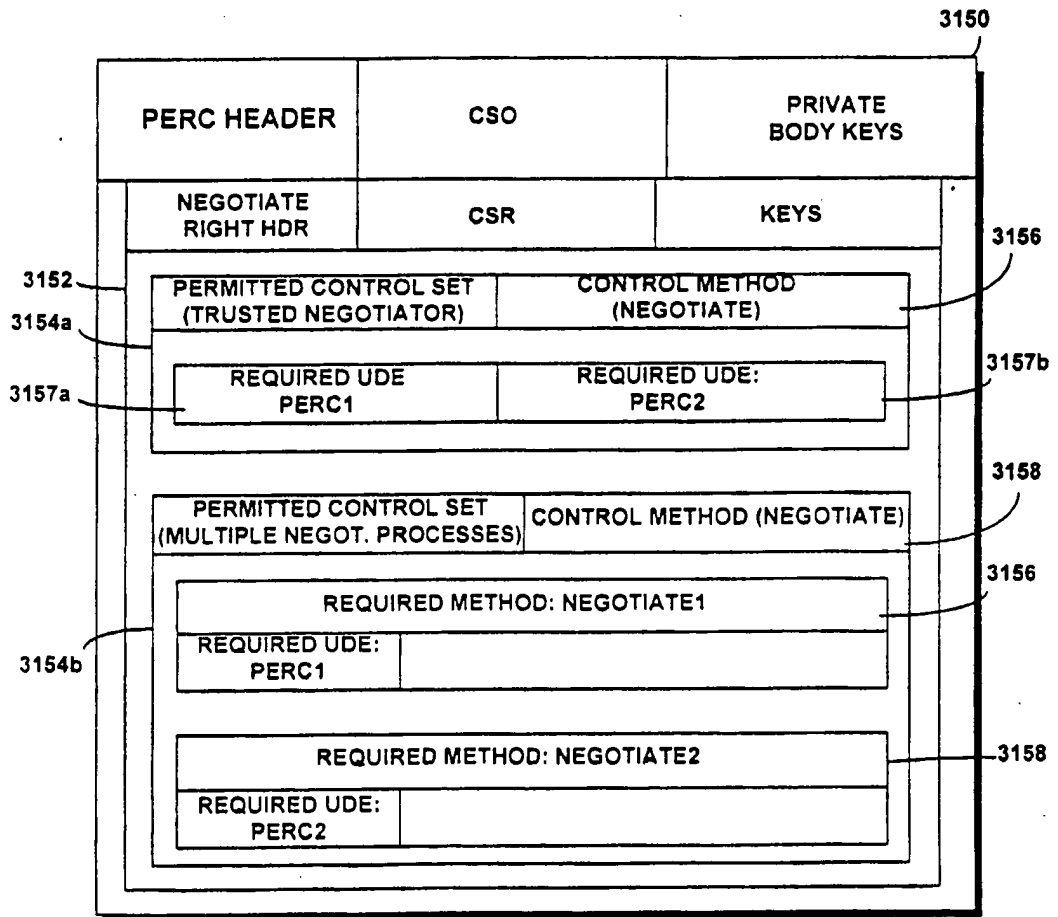
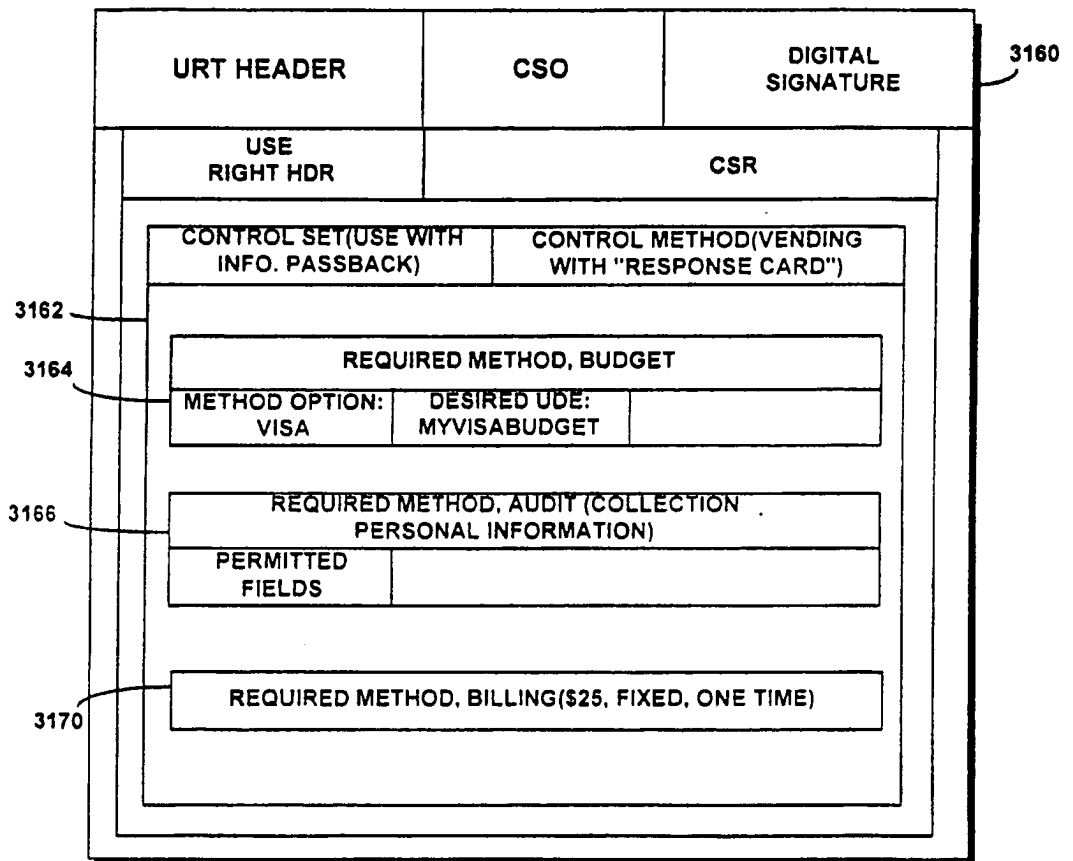


FIG. 75D



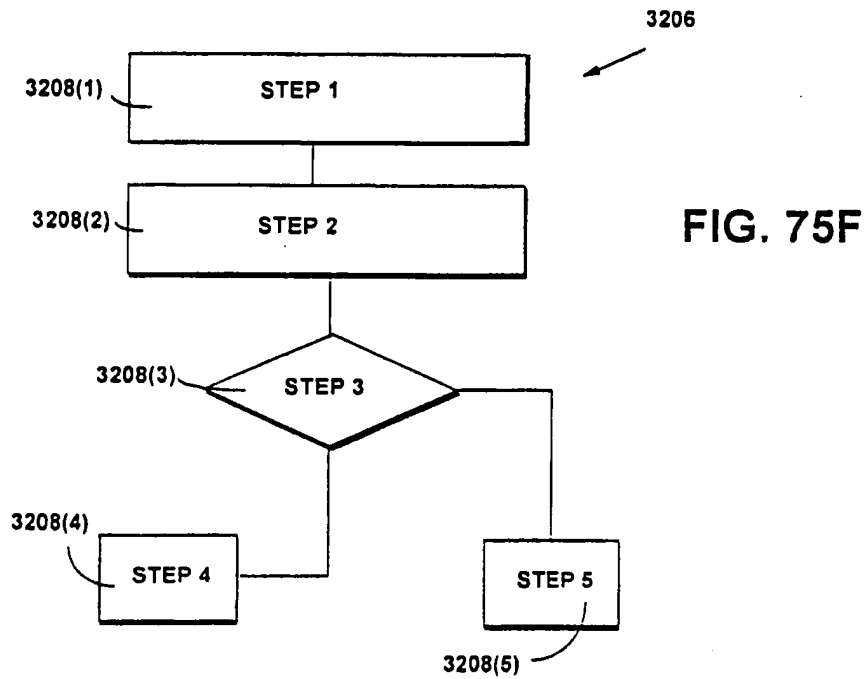
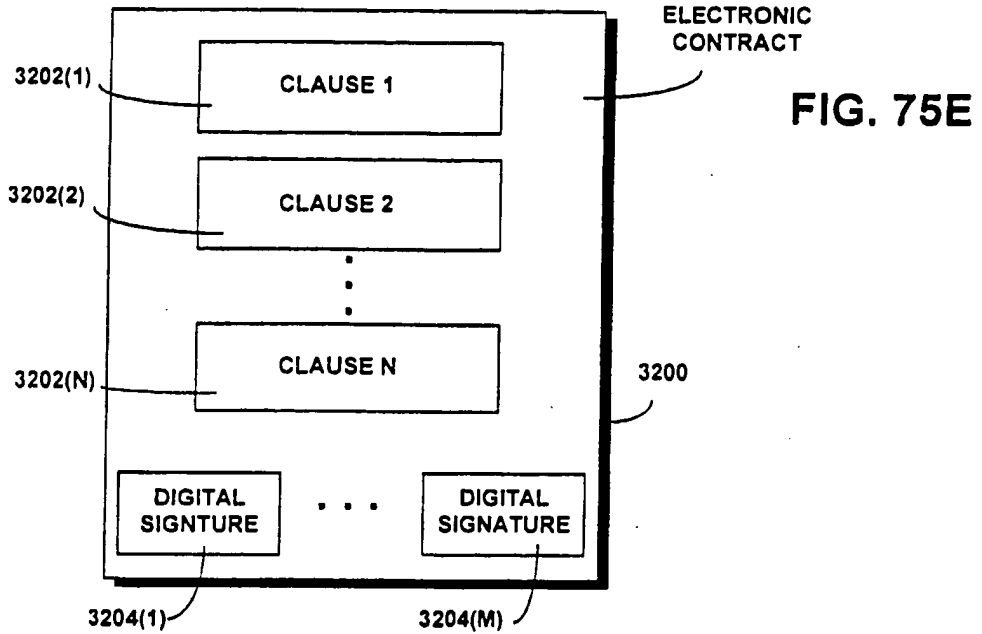


FIG. 76A

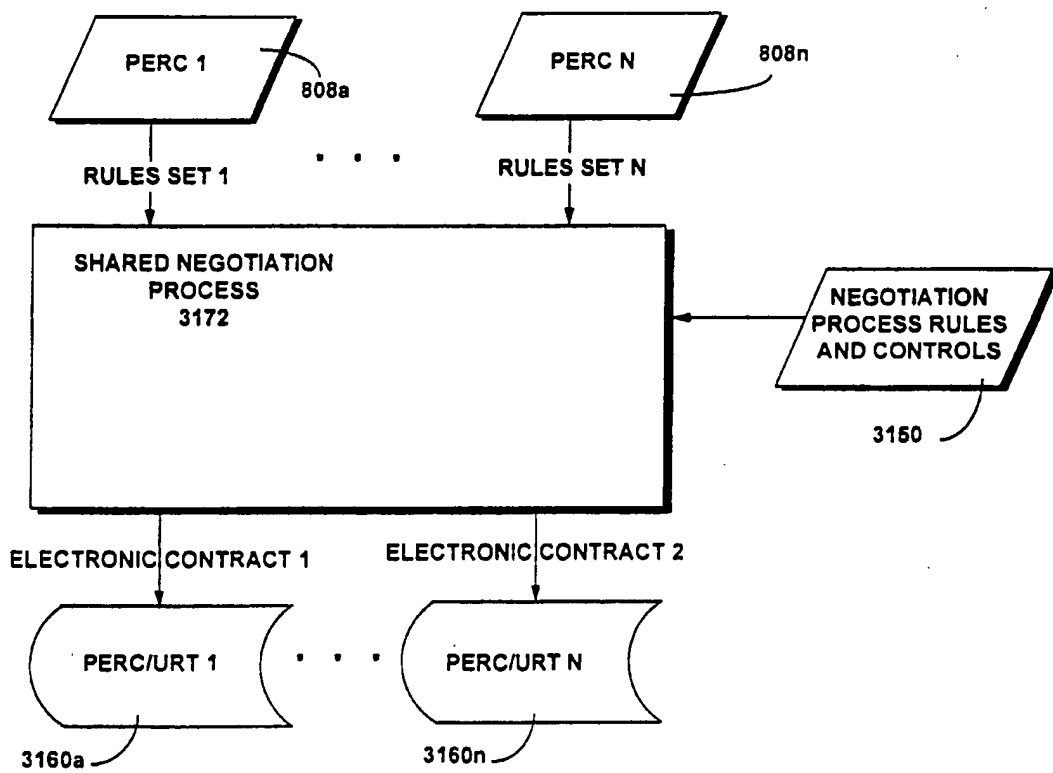
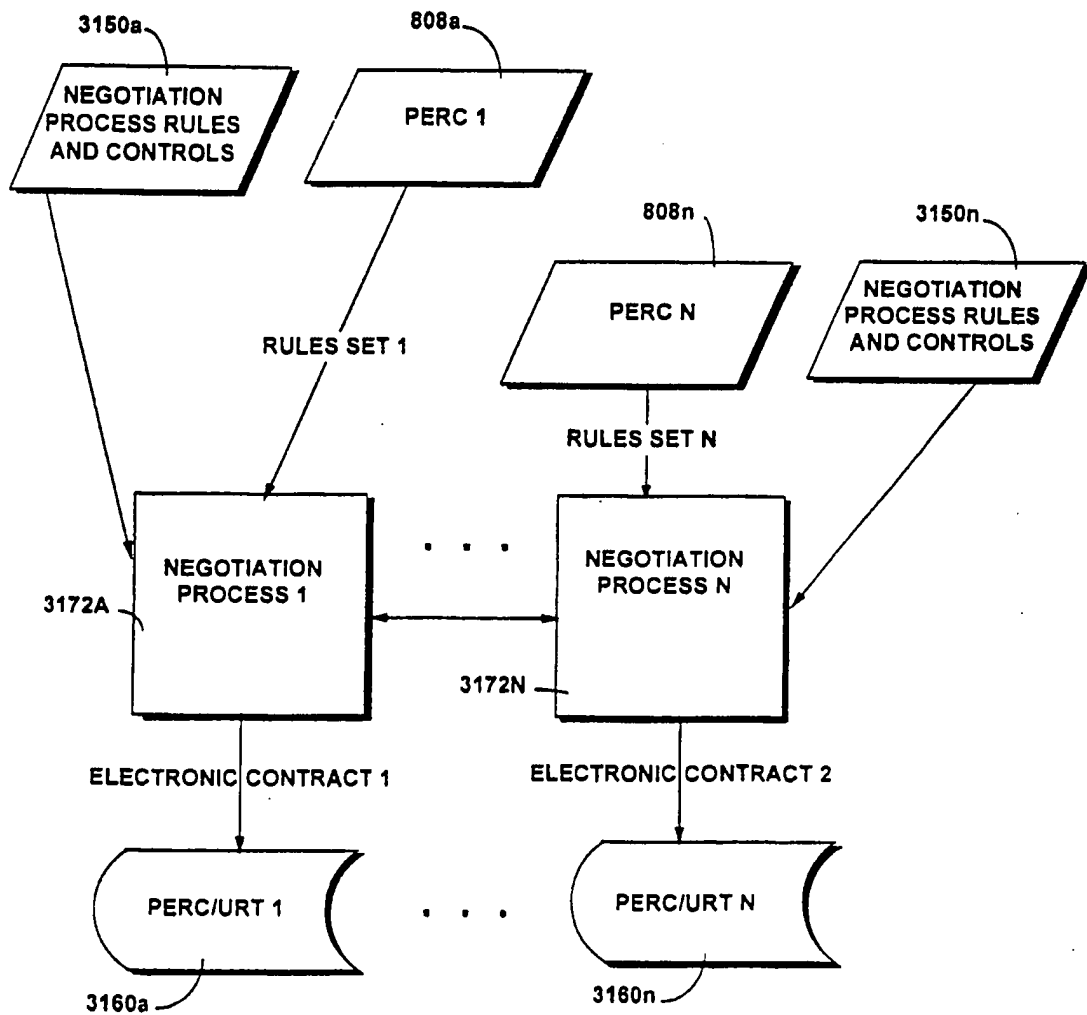
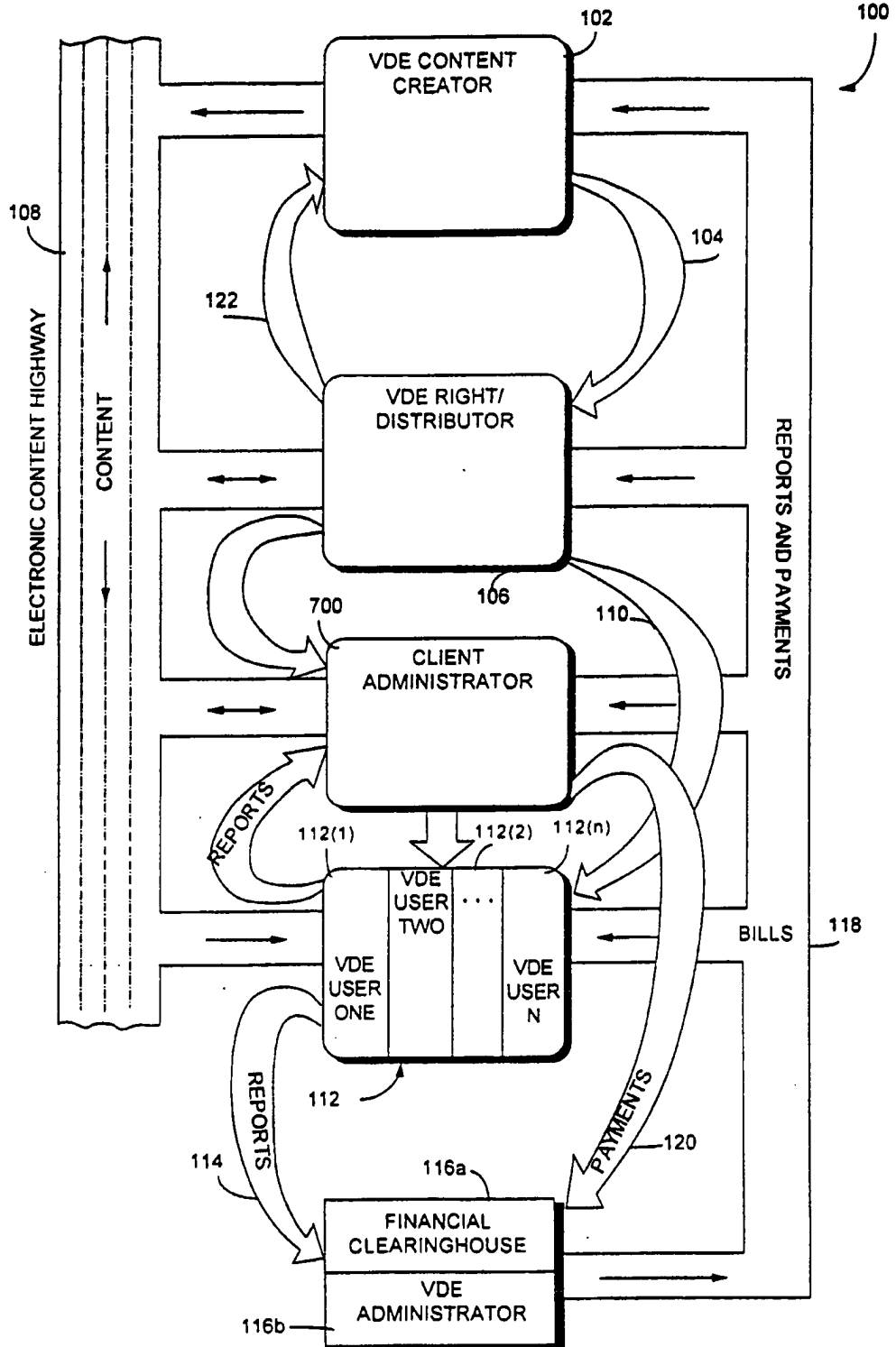


FIG. 76B



136/146

FIG. 77



137/146

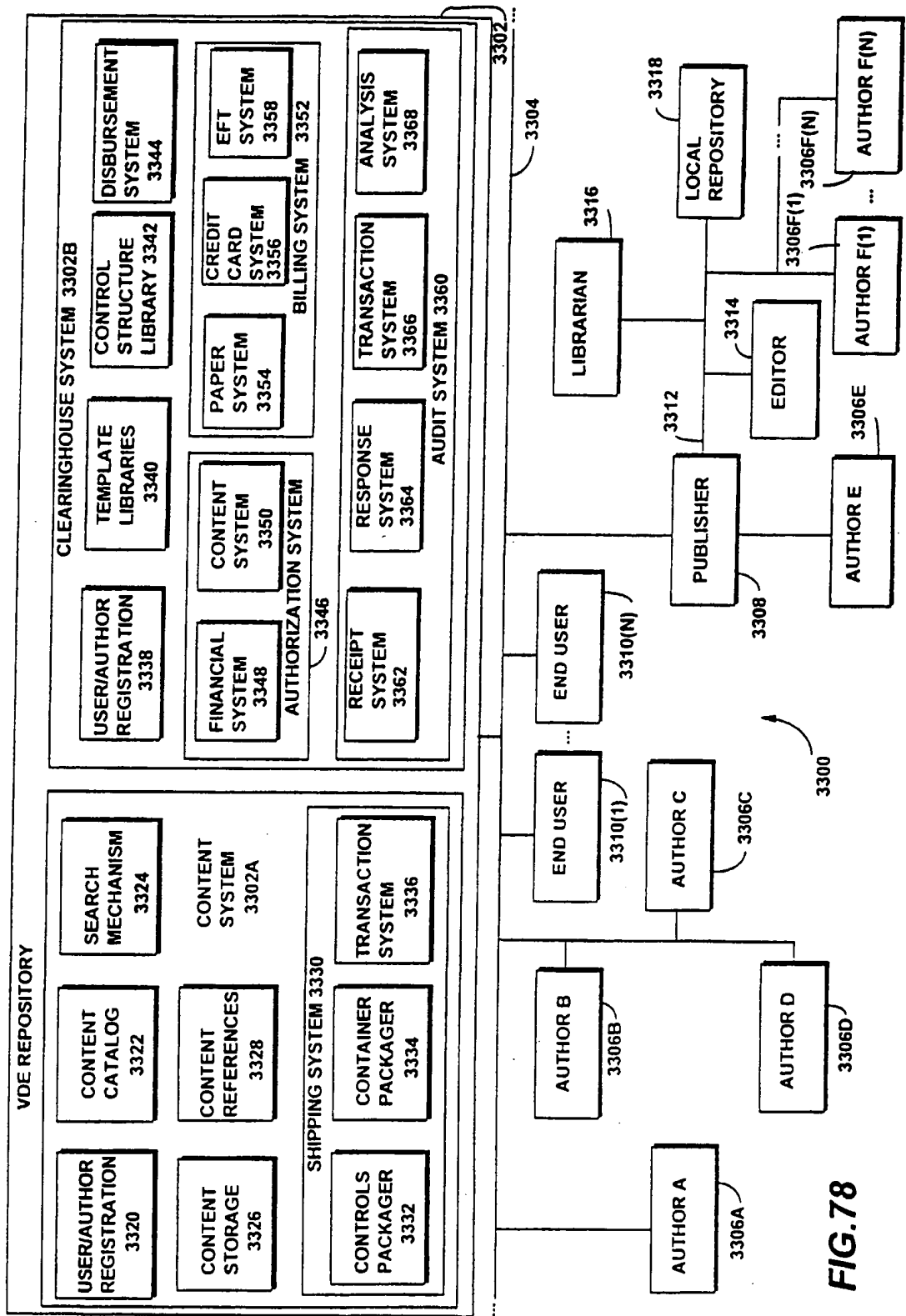
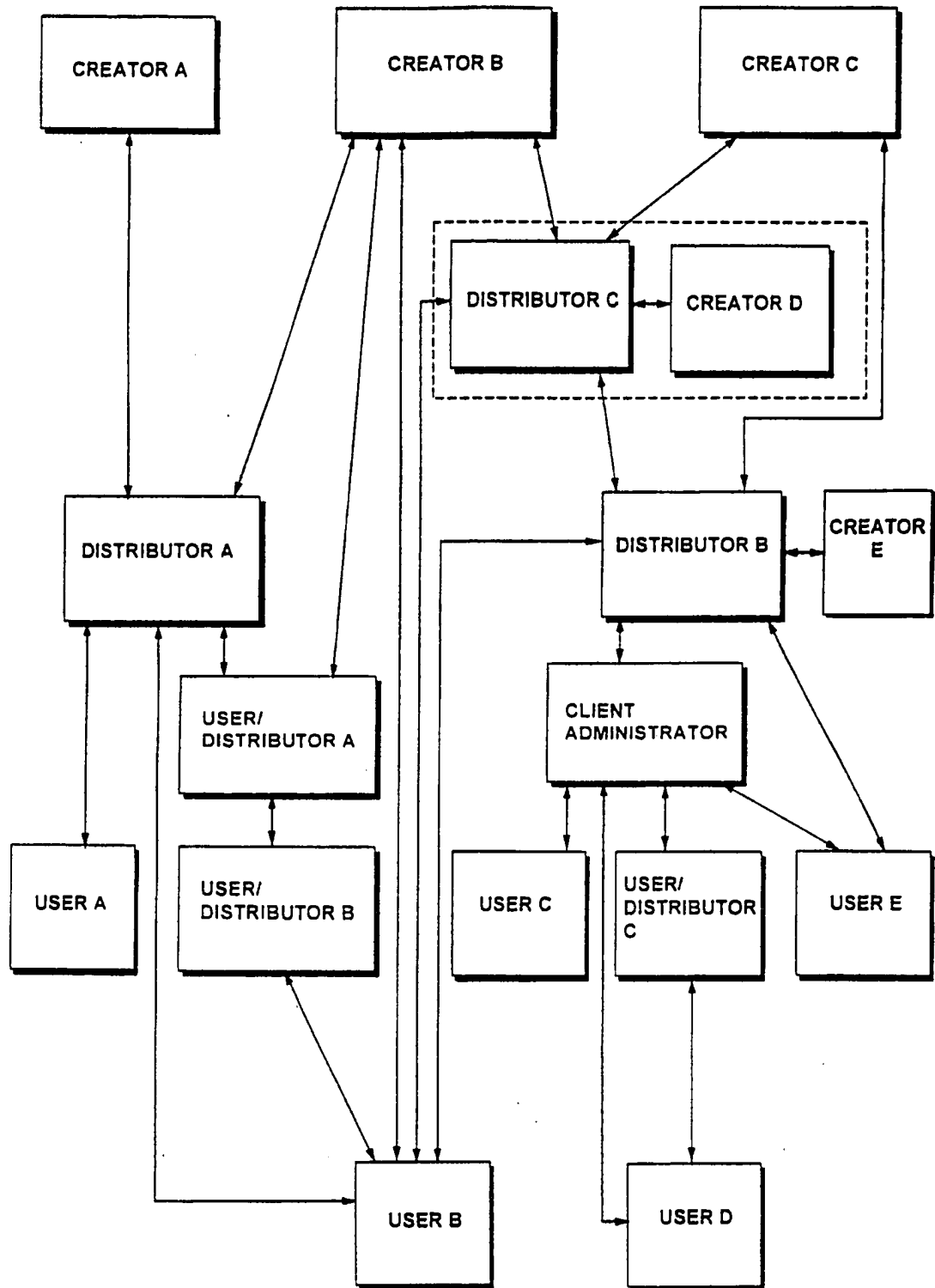


FIG. 78

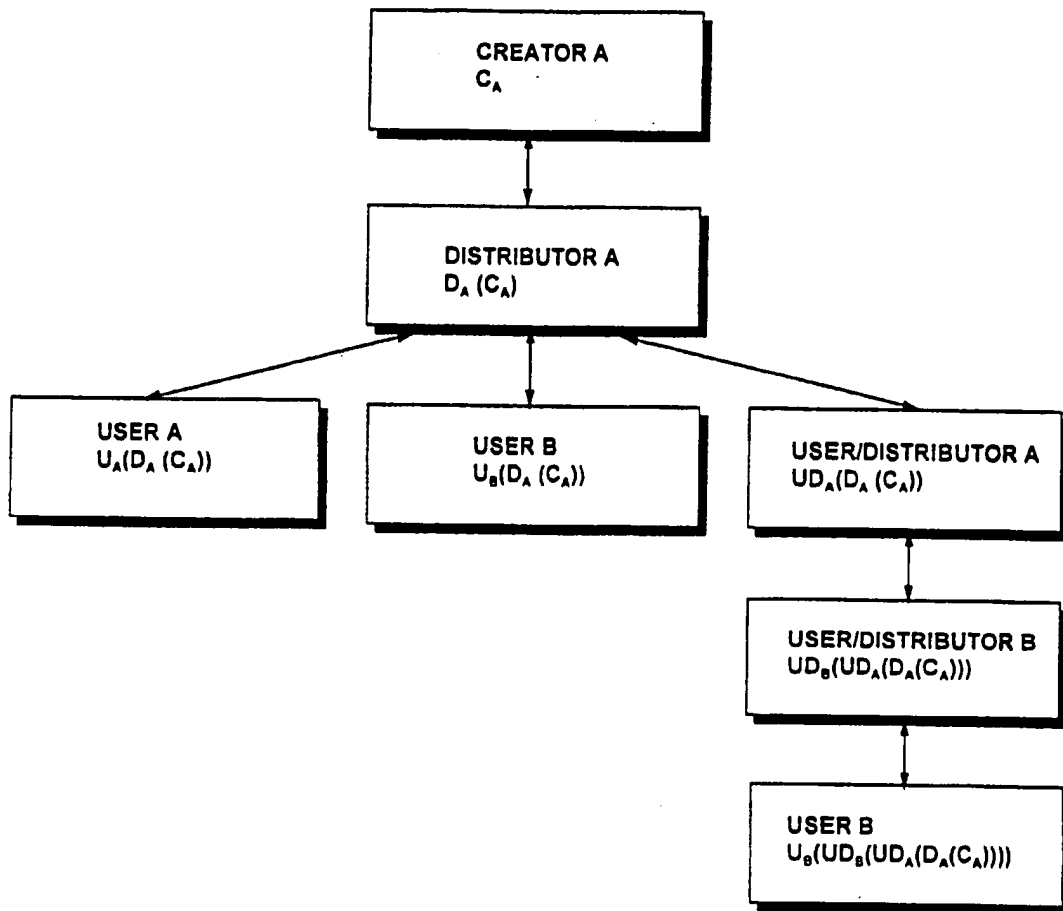
FIG. 79



SUBSTITUTE SHEET (RULE 26)

139/146

FIG. 80



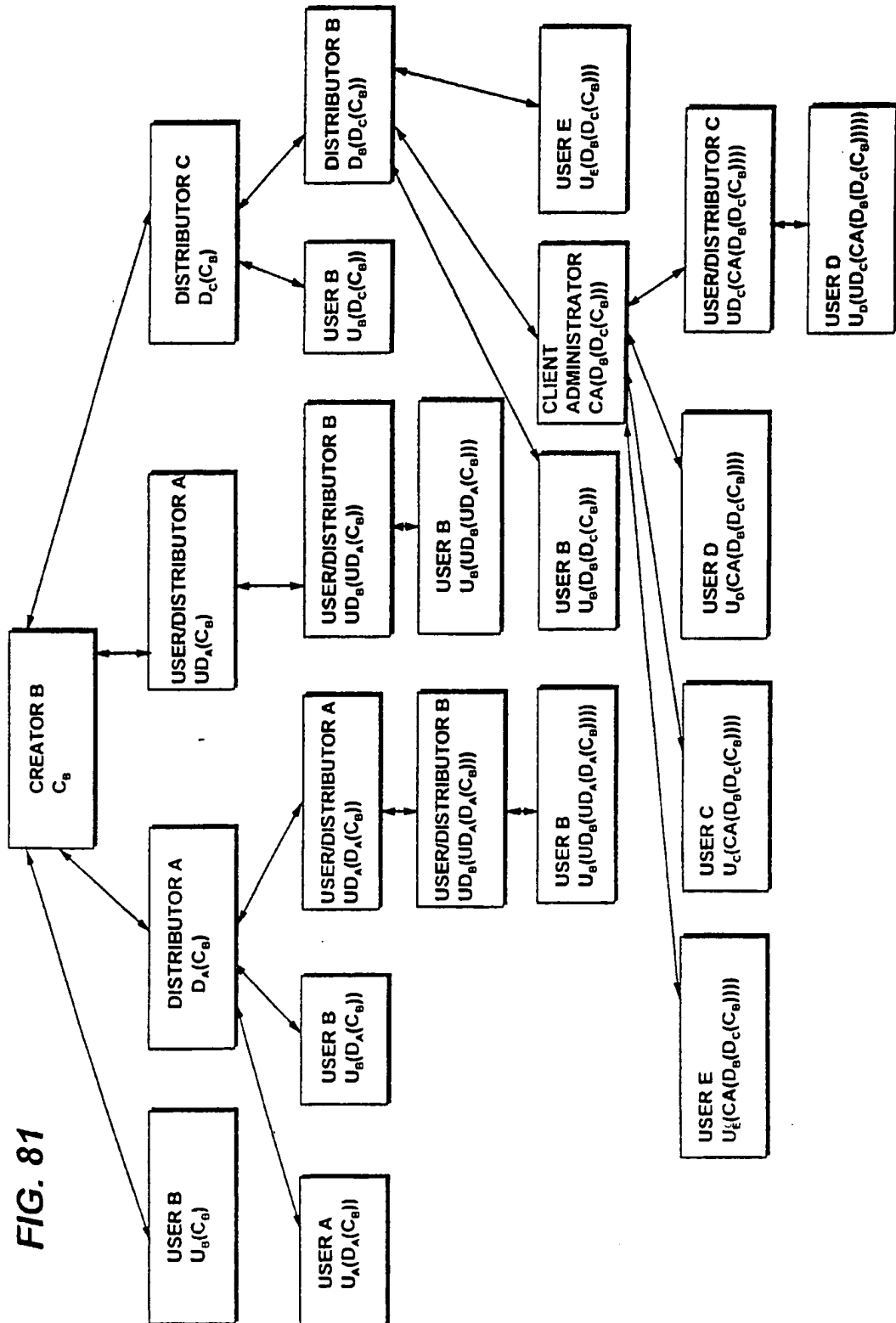


FIG. 81

FIG. 83

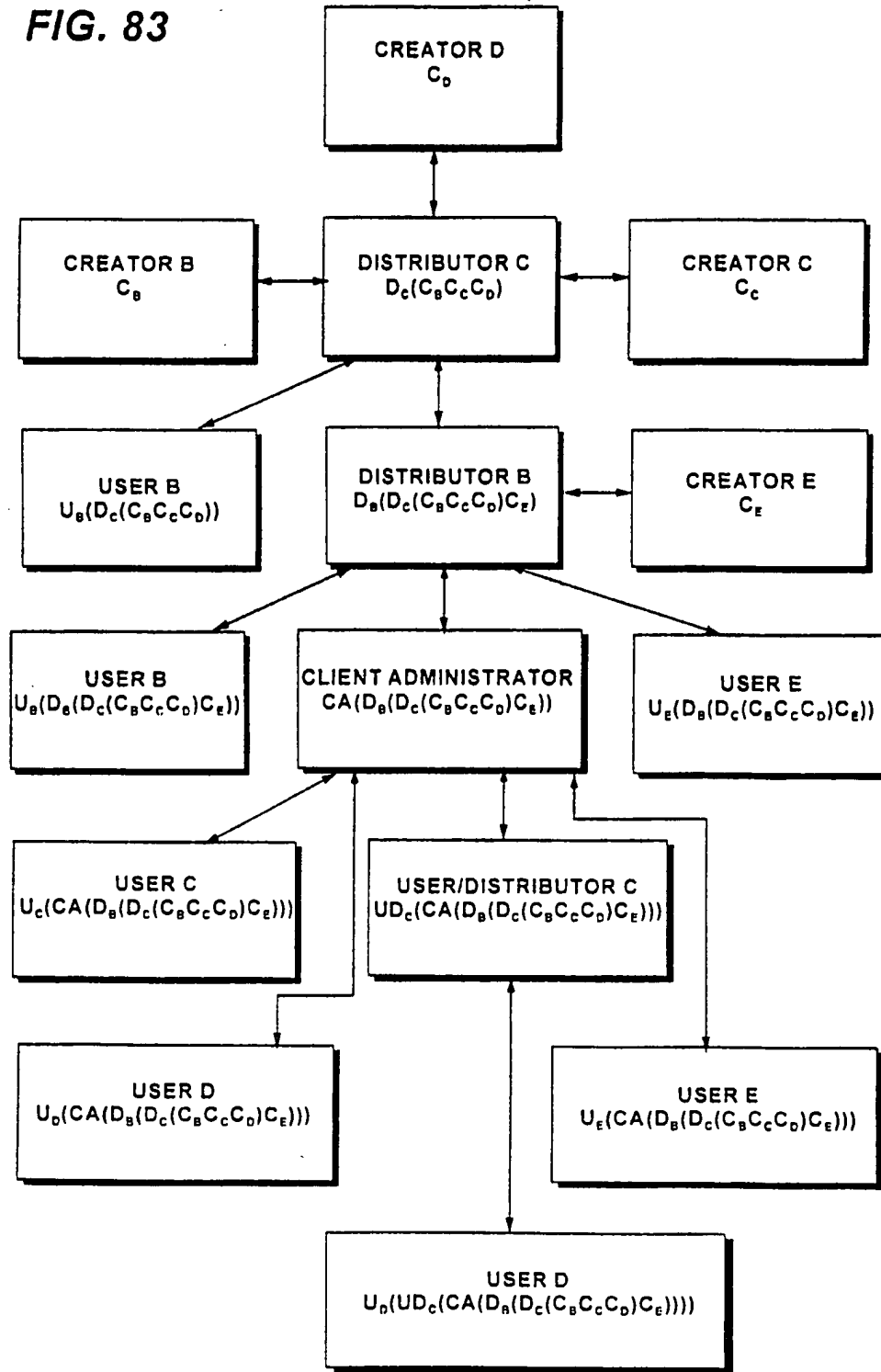


FIG. 84

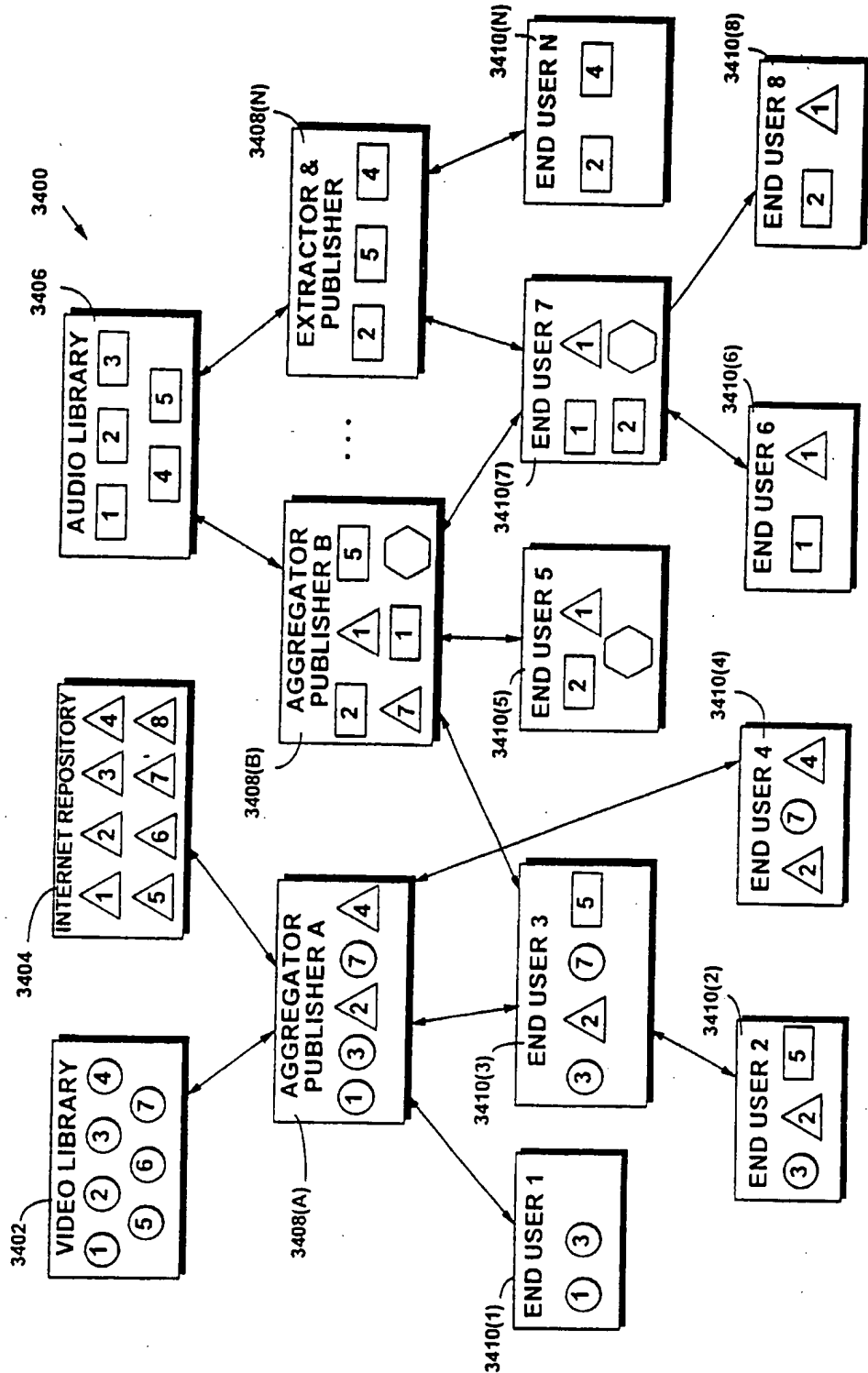
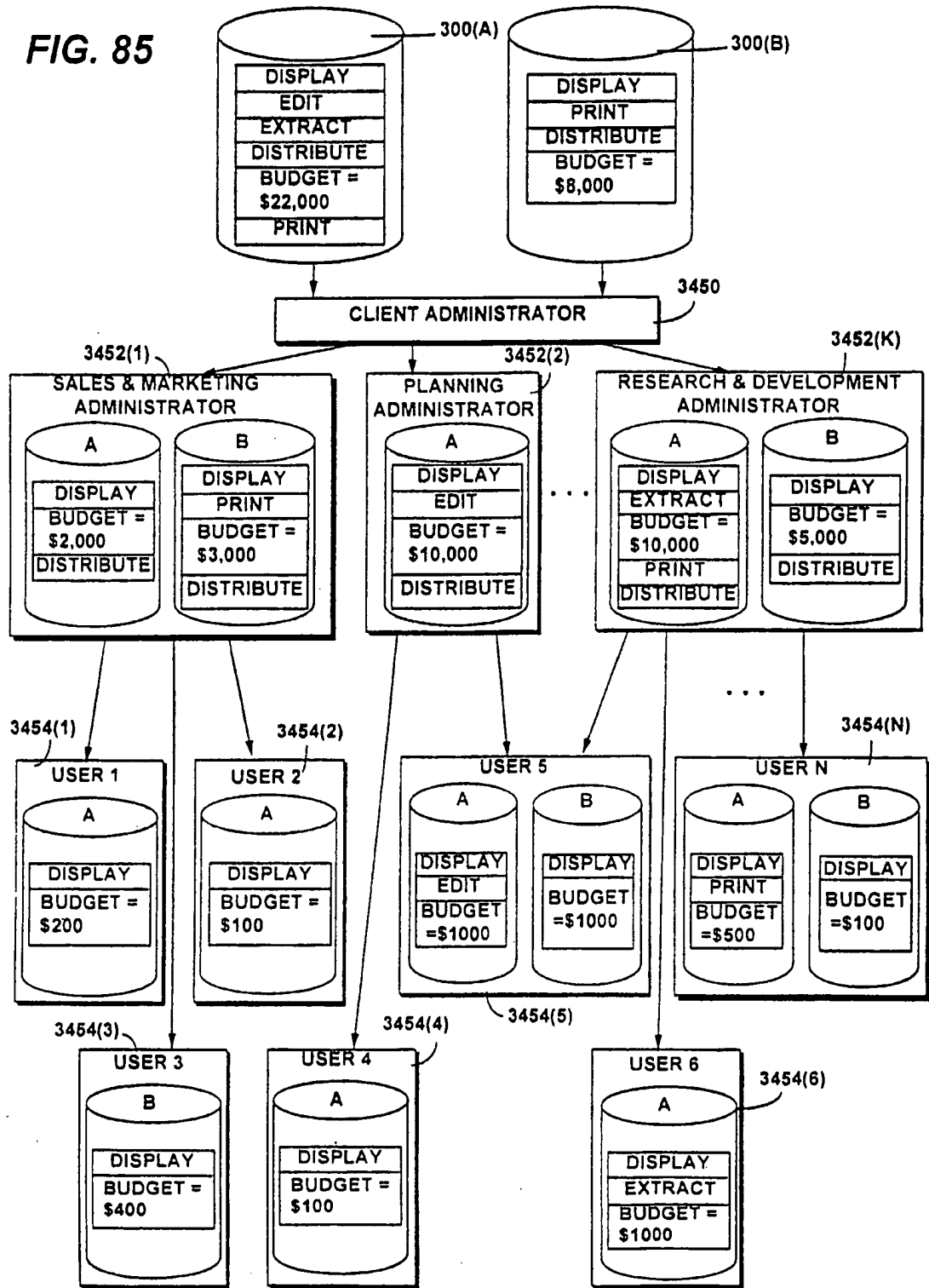


FIG. 85



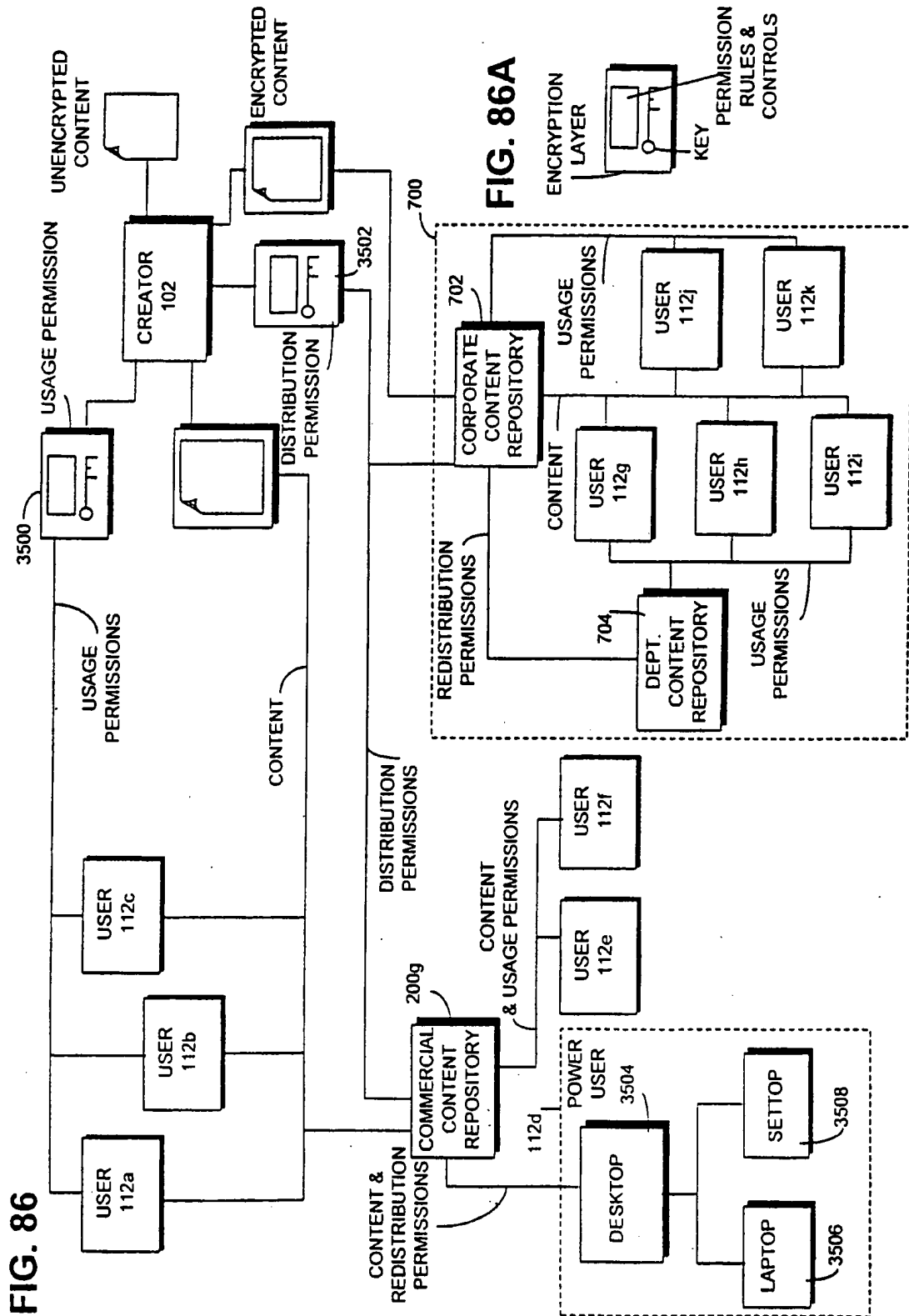


FIG. 86A

FIG. 86

146/146

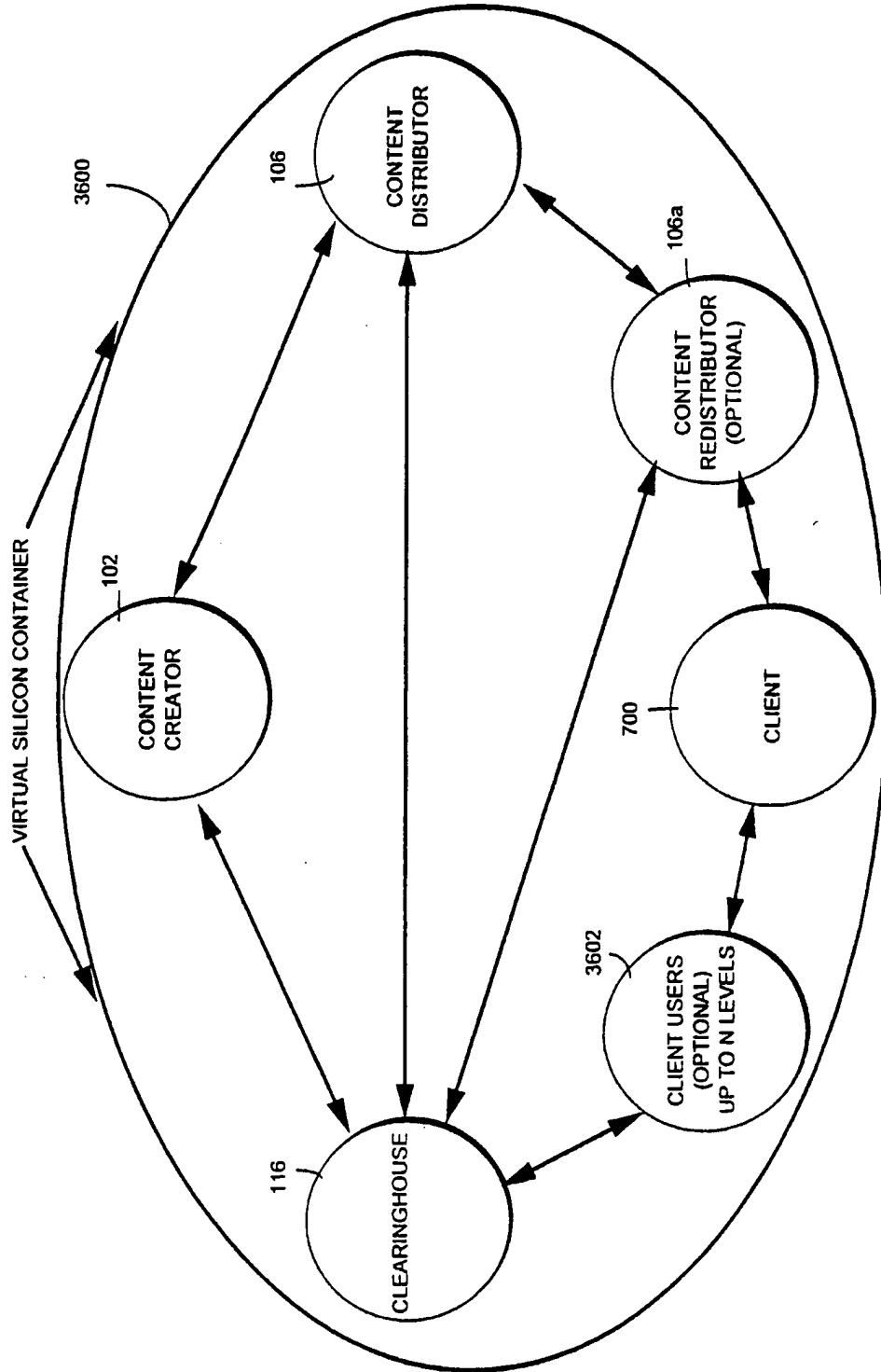


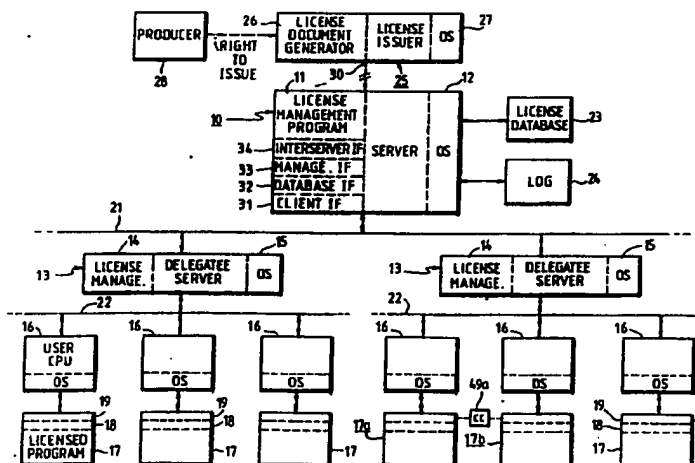
FIG. 87



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

<p>(51) International Patent Classification ⁵ : G06F 1/00</p>	<p>A1</p>	<p>(11) International Publication Number: WO 92/20022 (43) International Publication Date: 12 November 1992 (12.11.92)</p>
<p>(21) International Application Number: PCT/US92/03812 (22) International Filing Date: 6 May 1992 (06.05.92) (30) Priority data: 697,652 8 May 1991 (08.05.91) US 723,456 28 June 1991 (28.06.91) US 722,840 28 June 1991 (28.06.91) US 723,457 28 June 1991 (28.06.91) US (71) Applicant: DIGITAL EQUIPMENT CORPORATION [US/US]; 146 Main Street, Maynard, MA 01754 (US). (72) Inventor: WYMAN, Robert, Mark; 410 Second Avenue, South No. 108, Kirkland, WA 98033 (US).</p>		<p>(74) Agents: NATH, Ram, B. et al.; c/o Joyce D. Lange, Digital Equipment Corporation, 111 Powdermill Road, Maynard, MA 10754 (US). (81) Designated States: AT, AT (European patent), AU, BB, BE (European patent), BF (OAPI patent), BG, BJ (OAPI patent), BR, CA, CF (OAPI patent), CG (OAPI patent), CH, CH (European patent), CI (OAPI patent), CM (OAPI patent), CS, DE, DE (European patent), DK, DK (European patent), ES, ES (European patent), FI, FR (European patent), GA (OAPI patent), GB, GB (European patent), GN (OAPI patent), GR (European patent), HU, IT (European patent), JP, KP, KR, LK, LU, LU (European patent), MC (European patent), MG, ML (OAPI patent), MR (OAPI patent), MW, NL, NL (European patent), NO, PL, RO, RU, SD, SE, SE (European patent), SN (OAPI patent), TD (OAPI patent), TG (OAPI patent). Published <i>With international search report. Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.</i></p>

(54) Title: MANAGEMENT INTERFACE AND FORMAT FOR LICENSE MANAGEMENT SYSTEM



(57) Abstract

A distributed computer system employs a license management system to account for software product usage. A management policy having a variety of alternative styles and contexts is provided. Each licensed product upon start-up makes a call to a license server to check on whether usage is permitted, and the license server checks a database of the licenses, called product use authorizations, that it administers. If the particular use requested is permitted, a grant is returned to the requesting user node. The product use authorization is structured to define a license management policy allowing a variety of license alternatives by values called "style", "context", "duration" and "usage requirements determination method". The license administration may be delegated by the license server to a subsection of the organization, by creating another license management facility duplicating the main facility. The license server must receive a license document (a product use authorization) from an issuer of licenses, where a license document generator is provided. A mechanism is provided for one user node to make a call to use a software product located on another user node; this is referred to as a "calling card", by which a user node obtains permission to make a procedure call to use a program on another node. A management interface allows a license manager at a server to modify the license documents in the database maintained by the server, within the restraints imposed by the license, to make delegations, assignments, etc. The license documents are maintained in a standard format referred to as a license document interchange format so the management system is portable and can be used by all adhering software vendors. A feature of the database management is the use of a filter function.

FOR THE PURPOSES OF INFORMATION ONLY

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

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

MANAGEMENT INTERFACE AND FORMAT FOR LICENSE MANAGEMENT SYSTEM

BACKGROUND OF THE INVENTION

15 This invention relates to methods of operation of computer systems, and more particularly to a method and system for managing the licensing of software executed on computer systems.

20 In U.S. Patent 4,937,863, issued to Robert, Chase and Schafer and assigned to Digital Equipment Corporation, the assignee of this invention, a Software Licensing Management System is disclosed in which usage of licensed software may be monitored in a computer system to determine if a use is within the scope of a license. The system maintains a database of licenses for software products,

- 2 -

delivering the license document may be in the form of a network, or may be a phone line using modems, or may include physical delivery by disks or CD ROMs, for example. Likewise, the method of delivery of the software products being licensed, i.e., the applications programs 17 to be executed on the CPUs 16, is not material to the license management facility of the invention; the products are delivered by some appropriate means, e.g., the communications link 30 and the networks 21 and 22, by CD ROMs or disks physically distributed, etc.

Although shown in Figure 1 as operating on a distributed system, in the simplest case the license management facility of the invention may be operated on a single CPU. The license management program 11 and the applications program 17 may be executing on the same CPU, in which case the license document would be stored in a database 23 as before, on this CPU, and the calls from the unit 18 to the license server would be local instead of RPCs. As in the distributed system, however, the licensed product would still not have access to the license document, but instead could only make inquires to the server program, even if all are executing on the same CPU.

In operation of the distributed system of Figure 1, the producer 28 gives the issuer 25 authority to grant licenses on its behalf (the producer and issuer can be a single entity or multiple entities). The license document generator program 26, under control of a user (a person), generates a license (usually the result of negotiation between the user of program 26 and a user of the server 10). This license is called a product use authorization, and it is transmitted by the link 30 to the server 10. The license management program in the server 10 stores the product use authorization in the database 23, and, if delegation is an authorized option, may distribute parts of the authorized use to the delegatee servers 13,

- 3 -

where it is likewise stored in a database. Thereafter, administration of the license is only in response to inquiries from user nodes 16. When execution of a program 17 begins, the unit 18 is invoked to check on the availability of a license for this particular node. The unit 18 sends (as by an RPC) a request to the license management program 14 (or 11 if there is no delegatee), where the product use authorization stored in database 23 is checked to see if use is authorized. If so, a return is sent to the user node 16, granting permission to continue. When the program 17 has finished executing, the unit 18 again is invoked to signal to the license management program, again by an RPC, that the authorization is released, so the license management program can take appropriate action, e.g., log the use in log 24, etc.

To implement these operations, the license management program 11 or 14 contains several functions, including a client interface 31, a database interface 32, a management interface 33, and an interserver interface 34 for communicating with the delegates 13 (if any). The client interface 31, as described below, handles the requests received from the user nodes 16, and returns resulting from these requests. The database interface 32 handles the storing and retrieval of license information in the database 23, and logging license usage activity to log 24, and retrieval of this data. The management interface 33 handles the tasks of receiving the product use authorizations from the issuer 25 and maintaining the database 23 via the database interface 32. The interserver interface 34 handles the task of communicating with the delegatee servers 13, including transmitting the assigned parts of the product use authorizations, or communicating with other license servers that may be separately executing the license management function; for example, calls for validating calling cards may be made to another such server.

- 4 -

If there are no delegates or no other license servers, then of course the interserver interface 34 has no function, and is idle.

5 The license document or "product use authorization" forming the basis for the license management activity of the program 11 on the server 10 may be illustrated as a data structure containing the information set forth in Figure 2; in actual practice the product use authorization is preferably a more abstract data arrangement, not in such a rigidly structured format as illustrated. For example, the product use authorization as well as similar documents stored in the database 23, or passed between components of the system of Figure 1, may be of the so-called tag-length-value data format, where the data structure begins with an identifying tag (e.g., PUA or product use authorization) followed by a field giving the length, followed by the value itself (the content). One type of data treatment using this tag-length-value format is an international standard referred to as ASN.1 or Abstract Syntax Notation. In any event, the document 35 illustrated in Figure 2 is merely for discussing the various items of data, rather than representing the way the information is stored. Some of the fields shown here exist at some times and not others, and some are optional; the product use authorization may also include additional fields not shown or discussed here. Also it should be noted that copies of parts of this type of document are made for the delegates, so this representation of Figure 2 is a composite of several documents used in the system of Figure 1. The document 35 includes fields 36 identifying the software product by product name, producer, version numbers, release date, etc. The issuer 25 is identified in field 37, and the licensee (usually the owner of the license server 10) identified in field 38. The essential terms of the license grant are then defined in fields 40-46. The start date and end date are specified in fields 40; these store the exact time (date, hour, minute, second, etc.) when the license becomes valid and

- 5 -

when it ends, so licenses may be granted to start at some future time and to end at a particular time. Note that the previous practice has been to specify only the ending date, rather than also a start date as employed here. Each of the nodes, including issuer 25, servers 10 and 13, and user nodes 16, maintain a time value by a local clock referenced to a standard, so inherent in the license management facility is the maintaining of a time standard to compare with the start and end date information in the fields 40. The units granted are specified in field 41; the units are an arbitrary quantitative measure of program usage. In a delegatee server 13, the units field 41 will have some subset of the units field in the original product use authorization. As units are granted to users 16 or delegated, the remaining units available for grant are indicated in a subfield 42 in the copy of the document used by the server. The management policy occupies fields 43-46, and includes style, context, duration and LURDM (license use requirements determination method), as will be explained. The style field 43 specifies whether the licensed units are controlled by an "allocative" style or "consumptive" style, or some other "private" algorithm, where styles are ways used to account for the consumption or allocation of the units. The context field 44 specifies the location and environment in which product use or license management occurs, i.e., a CPU or an individual user or a network, etc. Duration field 45 indicates whether the license granted to a user is by assignment, by transaction, or immediate. The LURDM field 46 indicates the license use requirements determination method, in some cases using a license use requirements table (LURT) seen as field 47, as will be described.

Additional fields 48-54 in the product use authorization 35 of Figure 2 define features such as delegation authorization, calling authorization, overdraft

authorization, combination authorization, token, signature, checksum, etc. These will be described in the following paragraphs.

5 If the delegation field 48 is true, a license server 10 may distribute license units to multiple servers 13. A time limit may be imposed, i.e., units can be delegated to other hardware systems until they time out. Delegation allows an administrator to distribute units to improve response time and increase the resilience of the system. For example, the communication network 21 may include a satellite link to a remote facility where the local server 13 has a number of clients or users 16, in which case the calls to the server 13 would be completed
10 much quicker than would be the case if calls had to be made to the server 10. Also, delegation may be used as a method of allocating licensed units within a budget for administrative purposes. Usually the delegation authorization is a feature that is priced by the issuer, i.e., a license granting 1000 units with delegation authorization is priced higher than without this authorization.

15 The field 49 contains a calling authorization and/or a caller authorization. If the caller authorization in field 49 is true, the product is permitted to receive calls from other named products requesting use of the product, and if conditions are met (identified caller is authorized) the server can grant a calling card, as described below. If the calling authorization is true, the product can make calls
20 to other products. If neither is true, then the product can neither make or receive calls using the calling card feature. Referring to Figure 1, if product 17a wishes to make a remote procedure call to a feature of product 17b running on a different user node 16, it makes a call to its server 13 including a request for a calling card, and, if permitted, the return to product 17a includes a calling card
25 49a. The product 17a then makes a call to product 17b in the usual manner of

5 RPCs, sending along the calling card 49a, which the product 17b then verifies by
a call to its server 13 before executing the called procedure and issuing its return
to product 17a. The feature of calling cards is important for distributed
applications. For example, if a product is able to execute faster in a distributed
10 system by assigning tasks to other CPUs, then the issue is presented of which
license policy is needed, i.e., does every node executing a part of the task have to
be licensed and consume or receive allocation of a unit, or just the one managing
the task? This is resolved for most applications by use of this calling card concept.
The product use authorization for such a product has the calling authorization
15 field 49 enabled, so calling cards can be issued. This feature is typically separately
priced.

The combination authorization field 50 of Figure 2 determines whether or
not license requests from a user node 16 can be satisfied by combining units from
multiple product use authorizations. It may be advantageous to purchase licenses
15 with different policy values, and use units from certain product use authorizations
only for overflow or the like. Or, for other reasons, it may be advantageous to
"borrow" and "lend" units among delegated servers or user nodes. This function
is permitted or denied by the content of field 50.

The overdraft field 51 determines whether or not a requested allocation
20 from a user node 16 will be nevertheless granted, even though the units available
field 42 is zero or too small to permit the requested use. Overdrafts can be
unlimited, or a specific overdraft pool can be set up by a server 10, for a
customer's internal administrative purposes. That is, the overdraft value may be
unlimited in the original license, but limited or zero for internally distributed
25 copies of the license. Thus, the product use authorization sent by the issuer 25 to

5 the customer may have overdrafts permitted by the field 51, but the customer may deny overdraft permission for its own budgeting purposes. In any event, if overdraft is permitted, additional fees have to be paid to the issuer at some accounting period, when the logged usage from log 24 indicates the available units have been exceeded. If overdraft is denied, then the units 18 of the user nodes making request allocations are structured to inform the products 17 that a license grant is not available. The intent is not to prevent the application program from running; the license server merely informs the application whether or not the license manager determines that it is authorized to run. The application can itself be structured to shut itself down if not authorized to run, or it can be structured to shut down certain functions (e.g., ability to save files, ability to print, etc.), or 10 it can be structured to continue in a fully functional manner. The purpose of the license management facility is not that of enforcement, nor that of "copy protection", but instead is merely that of license management.

15 An optional token field 52 is available in the product use authorization 35 of Figure 2. This field can contain comments or other information desired by the issuer or user. For example, a telephone support number may be included in the token field, then when the product 17 shows its "help screen" the number is inserted. This number would be part of the argument, i.e., data transmitted to the user node 16, when the server 10 makes a return following a request allocation message from the user. This field may also be used to store information used in a "private" style, where the information from this field returned to the user node 20 is employed by the application program 17 or the stub 19 to determine if the application can be activated.

The signature field 53 in the product use authorization 35 is a part of a validation mechanism which provides important features. This field contains a digital signature encoded to reflect the data in the license itself, as well as other encoding methods not known to customers, so it cannot be duplicated unless the encoding algorithm is known. In a preferred embodiment, a so-called "public/private key" system of encoding is used for the signature field 53. The encoding algorithm used to generate the signature 53 is known to the issuer 25, using a private key, and anyone knowing the public key can decode the signature to determine if it is valid but cannot determine the encoding algorithm so it cannot produce a forged signature. So, if the server 10 knows the public key which is unique to the issuer 25, it can determine if a license document 35 is genuine, but it cannot itself generate license documents. However, if the server possesses a valid license document that gives it the right to delegate, then it will be assigned its own private key (different from all other issuers or servers) and its delegates 13 will be able to determine if a valid delegated license is delivered to them as they will be given the public key for the servers 13. The field 53 will thus contain both the original signature from the issuer 25 and the license server's signature when delivered to a delegatee 13. The decoding algorithm using a public key for any signatures is thus used by the license server 10 or delegatee 13 to make sure a product use authorization 35 is authentic before it is stored in the database 23. Related to the digital signature 53 is a checksum field 54, which merely encodes a value related by some known algorithm to the data in the product use authorization 35 itself. This field may be used merely to check for corruption of the data as it is stored, recalled, and transmitted within the system. That is, the checksum is used for data validation rather than security.

- 10 -

Two concepts central to the license management system implemented using the license document or product use authorization 35 of Figure 2 are the "license units", specified in field 41 or 42 and the "context", specified in field 44. License units are an abstract numerical measure of product use allowed by the license.

5 When a product 17 (or a function or feature of a product) makes a license-checking request, the license management program 11 on server 10 computes how many license units are required to authorize this particular use of the product, and this is the license units requirement, in some cases using the LURDM field 46.

10 A "context" is a set of tagged values which define the location and environment in which product use or license management occurs. Context values may be specified in field 44 of the product use authorization 35 of Figure 2 to restrict the environments in which the license may be managed and in which product use may occur. A context template may also be specified in the field 44 to indicate which parts of the complete context of product use (sub-contexts) are significant in

15 differentiating product uses for the purposes of unit allocation; when this is specified, it allows separate product uses to share license units in a controlled way.

The two general types of policies specified in field 43 are allocative and consumptive. An allocative policy grants to the holder a specific number of license units (field 41) and specifies the policy which must be used to account for

20 the allocation of these units. A software product 17 which is being managed by an allocative license will require verification that the appropriate number of license units have been allocated to it prior to performing services to the user. Typically, this allocation of units occurs either at the time of activation of the product 17 or at the time that product use is enabled on a particular platform

25 (user CPU 16). The units typically remain allocated to the product 17 throughout the period that the product is running or is enabled to run. Upon termination of

- 11 -

processing or disabling, the allocated units are deallocated and made available for allocation to other instances of the software product 17 (other users 16 activating the product). In general, as long as any license units remain unallocated in field 42, the holder of the license is contractually authorized to increase his utilization of the licensed product. The usage does not deplete the license, however, as the units are returned to the units-available field 42 after a user is finished, and can be granted again to another user.

A consumptive unit based license, indicated in policy field 43, grants to the holder a specific number of initial license units (from field 42) and specifies the policy used to account for the consumption of those units. A software product 17 which is being managed by a consumptive license will cause an appropriate number of license units to be consumed to reflect the services provided by the product. Once consumed, units cannot be reused. Thus, the number of units available for future use declines upon every use of the licensed software product 17. This may also be referred to as a "metered" policy, being conceptually similar to measured consumption of electricity, water, etc. When the number of available units in field 42 reaches zero, the license may require that further use of the product is prohibited, or, the agreement may permit continued decrementing of the number of available units; the result is the accumulation of a negative number of available units in the field 42. It is anticipated that most consumptive unit based licenses will consider negative units to represent an obligation of the license holder to pay the license issuer 25. The transaction log 24 maintains an audit trail for providing a record of the units used in a consumptive license.

Referring to Figure 3, the major elements of the management policy are set forth in a table, where the possible entries for the fields 43, 44, 45 and 46 are

- 12 -

5 listed. For the style entry 43, the possibilities are allocative and consumptive as just described, plus a category called "private" which represents a style of management undefined at present but instead to be created especially for a given product, using its own unique algorithm. It is expected that most licenses may be administered using the named alternatives of Figure 3, but to allow for future expansion to include alternatives not presently envisioned, or to permit special circumstances for unique software, the "private" choices are included, which merely mean that the product 17 will generate its own conditions of use. It is important to note that, except for the "private" alternative, the license management is totally in control of the license management program 11 on the license server 10 (or delegatee 13), rather than at the product 17. All the product 17 does, via the unit 18, is to make the request inquiry to the server 10 via the client interface 31, and report when finished.

15 The context field 44 specifies those components (sub-contexts) of the execution-context name which should be used in determining if unit allocations are required. License data is always used or allocated within, or for the benefit of, some named licensing context, and context can include "platform contexts" and "application contexts". Platform contexts are such things as a specific network, an execution domain, a login domain, a node, a process ID or a process family, a user name, a product name, an operating system, a specific hardware platform, as listed in Figure 3. Applications contexts are information supplied from the application (the product 17), such as may be used in a "private" method of determining license availability. The context name can use several of these, in which case the context name is constructed by concatenating the values of all subcontexts into a single context name, e.g., a VAX 3100 platform using VMS operating system.

20

25

The duration field 45 defines the duration of an allocation of license units to a specific context or the duration of the period which defines a valid consumptive use. For durations of type "Assignment," the specification of a reassignment constraint is also provided for, as discussed below. There are three types of duration, these being "transaction," "assignment" and "immediate" as seen in Figure 3.

The transaction duration type, when specified for an allocative policy, indicates that license units should be allocated to the specified context upon receipt of a license request and that those units should be deallocated and returned to the pool of available units upon receipt of a corresponding license release from a user node 16. Abnormal termination of the process or context having made the original license request will be semantically equivalent to a license release. On the other hand, when specified for a consumptive policy, this duration type indicates that license units should be allocated to the specified context upon receipt of a license request and permanently removed from the available units pool (field 42) upon receipt of a license release which reflects successful completion of the transaction. Upon receipt of a license release which carries an error status or upon abnormal termination of the processor context having made the original license request, the allocated units will be deallocated and returned to the pool of available units (field 42).

The assignment duration type in Figure 3 (field 45 of Figure 2) imposes the constraint that the required units must have been previously assigned to a specific context. The sub-contexts which must be specified in the assignment are those given in the context-template. A "reassignment constraint" may be imposed, and this is a limitation on how soon a reassignment can be made. For example, a

reassignment constraint of 30-days would require that units assigned to a specific context could not be reassigned more often than every 30-days; this would prevent skirting the intent of the license by merely reassigning units whenever a user of another context made a request allocation call for the product. Related to this assignment constraint, a "reallocation limit" may also be imposed, to state the minimum duration of an allocation; where there is a context template of process, the intent is to count the number of uses of the software product at a given time, but where software runs in batch rather than interactive mode it may run very quickly on a powerful machine, so a very few concurrent uses may permit almost unlimited usage - by imposing a reallocation constraint of some time period, this manner of skirting the intent of the license may be constrained.

The immediate duration type (field 45 of Figure 2) is used to indicate that the allocation or consumption of an appropriate number of license units from the pool of available units (field 42) should be performed immediately upon receipt of a license request. Receipt of license release or abnormal terminations will then have no impact on the license management system. When specified as the duration for an allocative policy, the effect will be simply to check if an appropriate number of license units are available at the time of a license request. When specified as the duration for a consumptive policy, the effect will be to deduct the appropriate number of license units from the available pool at the time of a license request, and, thereafter, abnormal termination, such as a fault at the user CPU 16 or failure of the network link, will not reinstate the units.

The LURDM or license unit requirement determination method, field 46, has the alternatives seen in Figure 3 and stores information used in calculating the number of units that should be allocated or consumed in response to a license

- 15 -

request. If this field specifies a table lookup kind, this means license unit requirements are to be determined by lookup in the LURT (field 47) which is associated with the current license. If a constant kind is specified, this indicates that the license units requirements are constant for all contexts on which the licensed product or product feature may run. A private LURDM specifies that the license unit requirements are to be determined by the licensed product 17, not by the license management facility 11. The license unit requirements tables (LURTs) provide a means by which issuers of licenses can store information describing the relation between context (or row selector) and unit requirements. The license units requirements determination method (LURDM) must specify "table lookup" for the LURT to be used, and if so a row selector must be specified, where a valid row selector is any subcontext, e.g., platform ID, user name, time of day, etc. An example of an LURT fragment is shown in Figure 4, illustrating the license unit requirements table mechanism. In this example, the row selector is "platform-ID" so the platform-ID value determines which row is used. The issuer of this LURT of Figure 4 has established three unit requirement tiers for use in determining the unit requirements for that issuer's products. The reason for the tiers is not mandated by the license management system, but the issuer 25 (actually the user of the program 26) would probably be establishing three pricing tiers, each reflecting a different perspective on the relative utility of different platforms in supporting the use of various classes of product 17. The first column in Figure 4, Column A, specifies the use requirements for a class of products whose utility is highly sensitive to the characteristics of the specific platform on which they are run. This can be seen by observing that the unit requirements are different for every row in Column A. Products which use the second column (Column B) appear to have a utility which is more related to the class of platform on which they run. This is indicated by the fact that all the PC

platforms share a single value which is different from that assigned to the VAX platform. The final column (Column C) is for use with a class of products which is only supported on the VAX platform. Figure 4 is of course merely an example, and the actual LURT created by the license document generator 26 and stored in the license database 23 (as field 47 of the product use authorization 35) can be of any content of this general format, as desired by the license issuer.

Instead of always selecting the rows in LURT tables according to the platform ID of the execution platform, in order to handle the breadth of business practices that need to be supported by the license management facility, the LURT mechanism is extended by providing a "row selector" attribute in the LURT class structure. No default is provided although it is expected that the normal value for the row selector attribute will be "platform ID."

In the system of patent 4,937,863, a concept similar to that of the LURT of Figure 4 was provided, with rows selected by the platform ID and columns selected by some arbitrary means, typically according to product type. The system of this invention allows flexibility in the selection of both LURT row and column while continuing to provide backwards compatibility for licenses defined within the constraints of patent 4,937,863.

Some examples will illustrate potential uses for the row selector attribute. A customer may only want to pay for the use of a product during one or two months of the year; the product may be FORTRAN and the reason for this request may be that the company has a fairly stable set of FORTRAN subroutines that are given regular "annual maintenance" only during the months of May and June. To handle this customer's needs, the FORTRAN product would generate

- 17 -

an application subcontext which would contain a value representing the month of the year. Then, a LURT table would be defined with twelve rows, one for each month of the year. In some column, probably column A, a negative one (-1) would be placed in each month except for May and June. These two months would contain some positive number. The product use authorization would then have a LURDM field specifying a LURT for use to determine the units requirement, and would name this custom LURT table. The effect would be that the PUA could only be used during the months of May and June since negative one is interpreted by license managers to mean "use not authorized." This mechanism could also be used to do "time of day" charging. Perhaps charging fewer units per use at night than during the day. Also, if a subcontext was used that contained a year value, a type of license would be provided that varied in its unit requirements as time passed. For instance, it might start by costing 10-units per use in 1991 but then cost one unit less every year as time passed, eventually getting to the point where the unit requirement was zero.

Another example is font names. A specific customer may purchase a license giving it the right to concurrent use of 100-units of a large font collection; some of the fonts may cost more to use than others. For instance, Times Roman might cost 10-units per use while New Century Schoolbook costs 20-units per use. The problem is, of course, making sure that charges are properly made. The solution is to build a LURT table with a specified application subcontext as its row selector. A row is then created for each font in the collection and in Column A of the LURT, the number of units required to pay for use of the font would be specified. The print server would then specify the name of a font as the value of the application subcontext whenever it does an *lm_request_allocation()* call. This will allow charges to be varied according to font name.

5 A further example is memory size. Some products are more or less valuable depending on the size of memory available to support them. A software vendor wishing to determine unit requirements based on memory size will be able to do so by building LURT tables with rows for each reasonable increment of memory (probably 1-megabyte increments). Their applications would then sense memory size (using some mechanism not part of the license management facility) and pass a rounded memory size value to the license manager in a private context.

10 Other examples are environment and operating system. Some products may be valued differently depending on whether they are being run in an interactive mode or in batch. This can be accomplished by building LURT rows for each of the standard platform subcontexts that specify environment. Regarding operating system, it has been considered desirable by many to have a single product use authorization permit the use of a product on any number of operating systems, this conflicts with some vendors policies who do not want to have to create a single price for a product that applies to all operating systems. 15 Thus, if an operating system independent license were offered for a C compiler, the price would be the same on MS-DOS, VMS, and/or UNIX. Clearly, it can be argued that the value of many products is, in part, dependent on the operating system that supports them. By using a row selector of operating system (one of the standard platform subcontexts), license designers could, in fact, require 20 different numbers of units for each operating system. However, it might be more desirable to base the row selection on a private application subcontext that normally had the same value as the operating system subcontext. The reason for this is that the license designer might want to provide a default value for operating system names that were unknown at the time the LURT rows were defined. If 25 this is the case, the product would contain a list of known operating systems and

- 19 -

pass the subcontext value of "Unknown" when appropriate. The LURT row for "Unknown" would either contain a negative one (-1) to indicate that this operating system was unsupported or it would contain some default unit requirement.

5 Another example is variable pricing within a group. One of the problems with a "group" license is that there is only one unit requirements field on the PUA for a group. Thus, all members of the group share a single unit requirement. However, in those cases where all members of the group can be appropriately licensed with a constant unit requirement yet it is desired to charge different amounts for the use of each group member, a LURT can be built that has rows defined for each group member. The row selector for such a group would be the standard platform subcontext "product name."

10

Many different types of license can be created using different combinations of contexts, duration and policy from the table of Figure 3. As examples, the following paragraphs show some traditional licensing styles which can be implemented using the appropriate values of the product use authorization fields 43-46.

15

A "system license" as it is traditionally designated is a license which allows unlimited use of a product on a single hardware system. The correct number of units must be allocated to the processor in advance and then an unlimited product use is available to users of the system. The product use authorization would have in the context field 44 a context template for a node name, the duration field would be "assignment" and the policy style field 43 would be "allocative".

20

5 A "concurrent use" license is one that limits the number of simultaneous uses of a licensed product. Concurrent use license units are only allocated when the product is being used and each simultaneous user of the licensed product requires their own units. In this case the context template, field 44, is a process ID, the duration field is "transaction" and the policy style 43 is "allocative".

10 A "personal use" license is one that limits the number of named users of a licensed product. This style of licensing guarantees the members of a list of users access to a product. Associated with a personal use type of product use authorization there is a list of registered users. The administrator is able to assign these users as required up to the limit imposed by the product use authorization; the number of units assigned to each user is indicated by the LURDM. It may be a constant or it may vary as specified in a LURT. The context template is "user name", the duration is "assignment", and the policy is "allocative".

15 A "site license" is one that limits the use of a licensed product to a physical site. Here the product use authorization contains for the context template either "network name" or "domain name", the duration is "assignment" and the policy style field 43 is "allocative".

20 Generally, a license to use a software product is priced according to how much benefit can be gained from using the product, which is related to the capacity of the machine it will run on. A license for unlimited use on a large platform such as a mainframe, where there could be thousands of potential users at terminals, would be priced at a high level. Here the style would be "allocative", the context template = "node", the duration = "assignment" and the LURDM may be "Column A" - the units, however, would be large, e.g., 1000. At the other end

- 21 -

of the scale would be a license for use on a single personal computer, where the field values would be the same as for the mainframe except the units would be "1". If a customer wanted to make the product available on the mainframe but yet limit the cost, he could perhaps get a license that would allow only five users at any given time to use the product; here the fields in the product use authorization would be: units = 5; style = allocative; context template = process; duration = transaction; LURDM = constant, 1-unit. This would still be priced fairly high since a large number of users may actually use the product if a session of use was short. A lower price would probably be available for a personal use license where only five named persons could use the product, these being identified only in the license server 10, not named by the license issuer 25. Here the fields in the product use authorization are: units = 5; style = allocative; context template = user name; duration = transaction; LURDM = constant, 1-unit.

An additional feature that may be provided for in the product use authorization 35 is license combination. Where there are multiple authorizations for a product, license checking requests sent by user nodes 16 may be satisfied by combining units from multiple authorizations. Individual product use authorizations may prohibit combined use. Thus, a licensee may have a license to use a product 17 on an allocative basis for a certain number of units and on a consumptive basis for another number of units (this may be attractive from pricing standpoint); there might not be enough units available for a particular context from one of these licenses, so some units may be "borrowed" from the other license (product use authorization), in which case a combination is made.

The interface between the program executing on the client or user 16 and the license server 10 or its delegates 13 includes basically three procedure calls:

- 22 -

a request allocation, a release allocation and a query allocation. Figure 5 illustrates in flow chart form some of the events occurring in this client interface. The request allocation is the basic license checking function, a procedure call invoked when a software product 17 is being instantiated, functioning to request an allocation of license units, with the return being a grant or refusal to grant. Note that a product may use request allocation calls at a number of points in executing a program, rather than only upon start-up; for example, a request allocation may be sent when making use of some particular feature such a special graphics package or the like. The release allocation call is invoked when the user no longer needs the allocation, e.g., the task is finished, and this return is often merely an acknowledge; if the style is consumptive, the caller has the opportunity via the release allocation call to influence the number of units consumed, e.g., decrease the number due to some event. The query allocation call is invoked by the user to obtain information about an existing allocation, or to obtain a calling card, as will be described.

The request allocation, referred to as *lm_request_allocation()*, is a request that license units be allocated to the current context. This function returns a grant or denial status that can be used by the application programmer to decide whether to permit use of the product or product feature. The status is based on the existence of an appropriate product use authorization and any license management policies which may be associated with that product use authorization. License units will be allocated or consumed, if available, according to the policy statement found on the appropriate product use authorization. The product would normally call this function before use of a licensed product or product feature. The function will not cause the product's execution to be terminated should the request fail. The decision of what to do in case of failure to obtain allocation of license

- 23 -

units is up to the programmer. The arguments in a request allocation call are the product name, producer name, version, release date, and request extension. The product name, producer name, version and release date are the name of the software product, name of producer, version number and release date for specifically identifying the product which the user is requesting an allocation be made. The request extension argument is an object describing extended attributes of the request, such as units required, LURT column, private context, and comment. The results sent back to the calling node are a return code, indicating whether the function succeeded and, if not, why not, and a grant handle, returned if the function completes successfully, giving an identifying handle for this grant so it can be referred to in a subsequent release allocation call or query allocation call, for example.

The release allocation, referred to as *lm_release_allocation()*, is an indication from a user to the license manager to release or consume units previously allocated. This function releases an allocation grant made in response to a prior call to request allocation. Upon release, the license management style 38 determines whether the units should be returned to the pool of available units or consumed. If the caller had specified a request extension on the earlier call to request allocation which contained a units-required-attribute, and the number of units requested at that time are not the number of units that should be consumed for the completed operation, the caller should state with the units-consumed argument how many units should be consumed. The arguments of the release allocation are: grant handle, units consumed, and comment. The grant handle identifies the allocation grant created by a previous call to request allocation. The units-consumed argument identifies the number of units which should be consumed if the license policy is consumptive; this argument should only be used

in combination with an earlier call to request allocation which specified a units requirement in a request extension. Omission of this argument indicates that the number of units to be consumed is the same as the number allocated previously. The comment argument is a comment which will be written to the log file 24 if release units are from a consumptive style license or if logging is enabled. The result is a return code indicating if the function succeeded, and, if not, why not.

The query allocation, or *lm_query_allocation()*, is used by licensed products which have received allocations by a previous request allocation call. The query is to obtain information from the server 10 or delegatee server 13 about the nature of the grant that has been made to the user and the license data used in making the grant, or to obtain a calling card (i.e., a request that a calling card be issued). Typically, the item read by this query function is the token field 52 which contains arbitrary information encoded by the license issuer and which may be interpreted as required by the stub 19 for the licensed product software 17, usually when a "private" allocation style or context is being employed. The arguments in this procedure call are the grant handle, and the subject. The grant handle identifies the allocation grant created by a previous call to request allocation. The subject argument is either "product use authorization" or "calling card request"; if the former then the result will contain a public copy of the product use authorization. If this argument is a calling card request and a calling card which matches the previous constraints specified in that request can be made available, the result will contain a calling card. If the subject argument is omitted, the result will contain an instance of the allocation. The results of the query allocation call are (1) a return code, indicating whether the function succeeded, and, if not, why not, and (2) a result, which is either an allocation, a product use authorization or a calling card, depending on type and presence of the subject argument.

- 25 -

Referring to Figure 5, the flow chart shows the actions at the client in its interface with the server. When the software product 17 is to be invoked, the unit 18 is first executed as indicated by the block 60, and the first action is to make a request allocation call via the stub 19, indicated by the block 61. The client waits for a return, indicated by the loop 62, and when a return is received it is checked to see if it is a grant, at decision block 63. If not, the error code in the return is checked at block 64, and if a return code indicates a retry is possible, block 65, control passes back to the beginning, but if no retry is to be made then execution is terminated. If the policy is to allow use of the product 17 without a license grant, this function is separately accounted for. If the decision point 63 indicates a grant was made, the grant handle is stored, block 66, for later reference. The program 17 is then entered for the main activities intended by the user. During this execution of product 17, or before or after, a query allocation call can be made, block 67, though this is optional and in most cases not needed. When execution of the program 17 is completed, the grant handle is retrieved, block 68, and a release allocation call is made, block 69. A loop 70 indicates waiting for the return from the server, and when the return received it is checked for an error code as before, and a retry may be appropriate. If the release is successfully acknowledged, the program exits.

Referring to Figure 6, the actions of the server 10 or delegatee server 13 in executing the license management program 11 or 14, for the client interface, are illustrated in flow diagram form. A loop is shown where the server program is checking for receipt of a request, release or query call from its clients. The call would be a remote procedure call as discussed above, and would be a message communicated by a network, for example. This loop shows the decision blocks 71, 72 and 73. If a release allocation call is received, a list of products for which

authorizations are stored is scanned, block 74, and compared to the product identity given in the argument of the received call, block 75. If there is no match, an error code is returned to the client, block 76, and control goes back to the initial loop. If the product is found, the authorization is retrieved from the database 23, block 77 (there may be more than one authorization for a given product, in which case all would be retrieved, but only one will be referred to here) and all of the information is matched and the calculations made depending upon the management policy of Figures 3 and 4, indicated by the decision block 78. If a grant can be made, it is returned as indicated at block 79, or if not an error code is returned, block 80. If a release allocation call is received, indicated by a positive at the decision block 72, the grant handle in the argument is checked for validity at block 81. If no match is found, an error code is returned, block 82, and control passes back to the initial loop. If the handle is valid, the authorization for this product is retrieved from the database 23 at block 83, and updated as indicated by the block 84. For example, if the license management style is allocative, the units are returned to the available pool. Or, in some cases, no update is needed. The authorization is stored again in the database, block 85, and a return made to the client, block 86, before control passes back to the initial loop. If the decision block 73 indicates that a query allocation call is received, again the grant handle is checked at block 87, and an error code returned at block 88 if not valid. If the grant handle matches, the authorization is retrieved from the database 23, at block 89, and a return is made to the client giving the requested information in the argument, block 90.

The basic allocation algorithm used in the embodiment of the license management system herein described, and implemented in the method of Figures 5 and 6, is very simple and can handle a very large proportion of known license

unit allocation problems. However, it should be recognized that a more elaborate and expanded algorithm could be incorporated. Additions could be made in efforts to extend the allocation algorithm so that it would have specific support for optimizing unit allocation in a wider variety of situations. Particularly, sources of non-optimal allocations occurring when using the basic allocation algorithm are those that arise from combination and reservation handling.

The first step is formation of full context. The client stub 19 is responsible for collecting all specified platform and application subcontexts from the execution environment of the product 17 and forwarding these collected subcontexts to the license management server 13 or 10. The collection of subcontexts is referred to as the "full context" for a particular license unit allocation request.

The next step is retrieval of the context template. When the license manager receives an *lm_request_allocation()*, it will look in its list of available product use authorizations (PUA) to determine if any of them conform to the product identifier provided in the *lm_request_allocation()* call. The product identifier is composed of: product name, producer, version, release date. If any match is found, the license manager will extract from the matching PUA the context template. This template is composed of a list of subcontexts that are relevant to the process of determining unit requirements. Thus, a context template may indicate that the node-ID subcontext of a specific full context is of interest for the purposes of unit allocation. The context template would not specify any specific value for the node-ID; rather, it simply says that node-ID should be used in making the allocation computation.

5 The next step is masking the full context. Having retrieved the context template, the license manager will then construct an "allocation context" by filtering the full context to remove all subcontexts which are not listed in the context template. This allocation context is the context to be used in determining allocation requirements.

10 Then follows the step of determining if the request is new. The license manager maintains for each product use authorization a dynamic table which includes the allocation contexts of all outstanding allocations for that PUA (i.e., allocations that have been granted but have not yet been released). Associated with each entry in this table is some bookkeeping information which records the number of units allocated, the full context, etc. To determine if a recent *lm_request_allocation()* requires an allocation of units to be made, the license manager compares the new allocation context with all those allocation contexts in the table of outstanding allocations and determines if an allocation has already
15 been made to the allocation context. If the new allocation context does not already exist in the table, an attempt will be made to allocate the appropriate number of units depending on the values contained in the LURDM structure of the PUA and any LURTs that might be required. If an allocation context similar to that specified in the new allocation request does exist in the table, the license
20 manager will verify that the number of units previously allocated are equal to or greater than the number of units which would need to be allocated to satisfy the new allocation request. If so, the license manager will return a grant handle to the application which indicates that the allocation has been made (i.e., it is a "shared allocation" - the allocated units are shared between two requests.) If not,
25 the license manager will attempt to allocate a number of units equal to the

difference between the number previously allocated and the number of units required.

5 The step of releasing allocations (Fig. 6, blocks 84-85) occurs when the license manager receives an *lm_release_allocation()* call; it will remove the record in its dynamic allocation table that corresponds to the allocation to be released. Having done this, the license manager will then determine if the allocation to be removed is being shared by any other allocation context. If so, the units associated with the allocation being released will not be released. They will remain allocated to the remaining allocation contexts. Some of the units might
10 be released if the license manager determines that the number of allocated units exceeds the number needed to satisfy the outstanding allocation contexts. If this is the case, the license manager will "trim" the number of allocated units to an appropriate level.

15 In summary, the two things that make this algorithm work are (1) the basic rule that no more than one allocation will be made to any single allocation context, and (2) the use of the context template to make otherwise dissimilar full contexts appear to be similar for the purposes of allocation.

20 The license designer's task, when defining basic policy, is then to determine which contexts should appear to be the same to the license manager. If the license designer decides that all contexts on a single node should look the same (context template = node-ID), then any requests that come from that node will all share allocations. On the other hand, a decision that all contexts should be unique (i.e., context template = process-ID) will mean that allocations are never shared.

and stores a unit value indicating the number of licensing units for each product. When a user wishes to use a licensed product, a message is sent to the central license management facility requesting a license grant. In response to this message, the facility accesses the database to see if a license exists for this product, and, if so, whether units may be allocated to the user, depending upon the user's characteristics, such as the configuration of the platform (CPU) which will execute the software product. If the license management facility determines that a license can be granted, it sends a message to the user giving permission to proceed with activation of the product. If not, the message denies permission.

While the concepts disclosed in the patent 4,937,863 are widely applicable, and indeed are employed in the present invention, there are additional functions and alternatives that are needed in some applications. For example, the license management system should allow for simultaneous use of a wide variety of different licensing alternatives, instead of being rigidly structured to permit only one or only a few. When negotiating licenses with users, vendors should have available a wide variety of terms and conditions, even though a given vendor may decide to narrow the selection down to a small number. For example, a software product may be licensed to a single individual for use on a single CPU, or to an organization for use by anyone on a network, or for use by any users at terminals in a cluster, or only for calls from another specific licensed product, or any of a large number of other alternatives. A vendor may have a large number of products, some sold under one type of license and some under others, or a product may be a composite of a number of features from one or more vendors having different license policies and prices; it would be preferable to use the same license management system for all such products.

5 Distributed computing systems present additional licensing issues. A distributed system includes a number of processor nodes tied together in a network of servers and clients. Each node is a processor which may execute programs locally, and may also execute programs or features (subparts of programs) via the network. A program executing on one node may make remote procedure calls to procedures or programs on other nodes. In this case, some provision need be made for defining a license permitting a program to be executed in a distributed manner rather than separately on a single CPU, short of granting a license for execution on all nodes of a network.

10 In a large organization such as a company or government agency having various departments and divisions, geographically dispersed, a software license policy is difficult to administer and enforce, and also likely to be more costly, if individual licenses are negotiated, granted and administered by the units of the organization. A preferred arrangement would be to obtain a single license from
15 the software producer, and then split this license into locally-administered parts by delegation. The delays caused by network communication can thus be minimized, as well as budgetary constraints imposed on the divisions or departments. Aside from this issue of delegation, the license management facility may best be operated on a network, where the licensing of products run on all
20 nodes of the network may be centrally administered. A network is not necessary for use of the features of the invention however, since the license management can be implemented on a single platform.

25 Software products are increasingly fragmented into specific functions, and separate distribution of the functions can be unduly expensive. For example, a spreadsheet program may have separate modules for advanced color graphics, for

accessing a database, for printing or displaying an expanded list of fonts, etc. Customers of the basic spreadsheet product may want some, none or all of these added features. Yet, it would be advantageous to distribute the entire combination as one package, then allow the customer to license the features separately, in various combinations, or under differing terms. The customer may have an entire department of the company needing to use the spreadsheet every day, but only a few people who need to use the graphics a few days a month. It is advantageous, therefore, to provide alternatives for varied licensing of parts or features of software packages, rather than a fixed policy for the whole package.

Another example of distribution of products in their entirety, but licensing in parts, would be that of delivering CD ROMs to a customer containing all of the software that is available for a system, then licensing only those parts the customer needs or wishes to pay fees for rights to use. Of course, the product need not be merely applications programs, operating systems, or traditional executable code, but instead could also include static objects such as printer fonts, for example, or graphics images, or even music or other sound effects.

As will be explained below, calling and caller authorizations are provided in the system according to one feature of the invention, in order to provide technological support for a number of business practices and solve technical problems which require the use of what is called "transitive licensing." By "transitive licensing" is meant that the right to use one product or feature implies a right to use one or more other products or features. Transitive licenses are similar to group licenses in that both types of license consist of a single instrument providing rights of use for a plurality of products. However, transitive licenses differ from group licenses in that they restrict the granted rights by specifying that

5 the licensed products can only be used together and by further specifying one or more permitted inter-product calling/caller relationships. Some examples may help to clarify the use and nature of a transitive license: the examples to be explained are (1) two products sold together, (2) a give-away that results from narrow choices of licensing alternatives, (3) a client licensing method in a client/server environment, (4) impact of modular design, and (5) the impact of distributed design.

10 A software vendor might have two products for sale: the first a mail system, and the second a LEXISTM-like content-based text retrieval system. Each of these products might be valued at \$500 if purchased separately. Some customers would be satisfied by purchasing the rights to use only one of these products. others might find that they can justify use of both. In order to increase the likelihood that customers will, in fact, purchase both products, it would not be surprising if the software vendor offered his potential customers a volume discount, offering the two products for a combined price of \$800. The customers
15 who took advantage of this combined offer would find that they had received two products, each of which could be exploited to its fullest capabilities independently from the other. Thus, these customers would be able to use the content based retrieval system to store and retrieve non-mail documents. However, from time to time, the vendor may discover that particularly heavy users of mail wish to be able to use the content based retrieval system only to augment the filing capabilities provided by the standard mail offering. It is likely that many of these potential customers would feel that \$800 is simply too much to pay for an extended mail capability. The vendor might then consider offering these
20 customers a license that grants mail users the right to use the content-based retrieval system only when they are using mail and prohibits the use of content
25

based retrieval with any other application that might be available on the customers system. This type of license is referred to below a "transitive license," and it might sell for \$600.

5 Another example is a relational database product (such as that referred to as Rdb™) designed for use on a particular operating system, e.g., VMS. This relational database product has two components: (1) A user interface used in developing new databases, and (2) a "run-time" system which supports the use of previously developed databases. The developers of the database product might spend quite a bit of effort trying to get other products made by the vendor of the database product to use it as a database instead of having those other products
10 build their own product-specific databases. Unfortunately, the other product designers may complain that the cost of a run-time license for the database product, when added to the cost of licenses for their products, would inevitably make their products uncompetitive. Thus, some mechanism would be needed that would allow one or another of the vendor's products to use the run-time system for the relational database product in a "private" manner while not giving
15 unlicensed access to products of other vendors. No such mechanism existed, prior to this invention; thus, the vendor might be forced to sell the right to use its run-time system for the database product with its proprietary operating system license. Clearly, this combined license would make it possible for the vendor's products to
20 use its database product without increasing their prices; however, it also would make it possible for any customers and third-parties to use the database product without paying additional license fees. However, had the system of the invention been available, the vendor could have granted transitive licenses for the run-time
25 component of its database product to all the vendor's products. Essentially, these licenses would have said that the database run-time could be used without an

- 35 -

additional license fee if and only if it was used in conjunction with some other of the vendor's products. Any customer wishing to build a new relational database application or use a third-party application that relied on the vendor's database product would have had to pay the vendor for its database run-time license.

5 A proposed client/server licensing method provides yet another example of a problem which could be solved by transitive licensing. Typically, a client is only used by one user at a time, while a server can support an arbitrary number of clients depending on the level of client activity and the capacity of the machine which is supporting the server. While traditionally, server/client applications have
10 been licensed according to the number of clients that a server could potentially support, this may not be the most appropriate method for licensing when the alternatives afforded by the invention are considered. The business model for the proposed client/server method requires that each client be individually licensed and no explicit licensing of servers is required to support properly licensed clients.
15 Such a licensing scheme makes it possible to charge customers only for the specific number of clients they purchase. Additionally, it means that a single client can make use of more than one server without increasing the total cost of the system. The solution to this transitive licensing problem would be to provide a mechanism that would allow the clients to obtain license unit allocations and then pass a
20 "proof" of that allocation to any servers they may wish to use. Servers would then support any clients whose proofs could be verified to be valid. On the other hand, if a client that had not received a proof of allocation attempted to use a server, the server would obtain a license allocation for that client session prior to performing any services. Such a solution has not been heretofore available.

- 36 -

As the complexity and size of the software systems provided to customers increases, it is found that the actual solution provided to customers is no longer a single product. Rather, customers are more often now offered solutions which are built up by integrating an increasing number of components or products, each of which can often stand alone or can be part of a large number of other solutions. In fact, a product strategy may rely almost exclusively on the vendor's engineering and selling a broad range of specialized components that can only be fully exploited when combined together with other components into a larger system. Such components include the relational database runtime system mentioned above, mail transport mechanisms, hyperinformation databases, document format conversion services, time services, etc. Because these components are not sold on their own merits, but rather on their ability to contribute to some larger system, it is unlikely that any one customer will be receiving the full abstract economic value of any one of the components once integrated into a system. Similarly, it can be observed that the value of any component once integrated into a larger system varies greatly from system to system. Thus, it may be found that a mail transport mechanism contributes a large part of a system whose primary focus is mail, however, it will contribute proportionally less of the value of a system that provides a broader office automation capability. As a result of these observations, the job of the business analyst who is attempting to find the "correct" market price for each component standing on its own, is more complex. In reality, the price or value of the component can only be determined when considering the contribution of that component to the full system or solution in which it is integrated. Attempting to sell the components at prices based on their abstract, independent values will simply result in overpriced systems.

- 37 -

Transitive license styles are particularly suited to dealing with pricing of modular components, since component prices can be clearly defined in relation to the other components or systems which they support. Thus, a vendor can charge a price of \$100 for the right to use a mail transport system in conjunction with one product, yet charge \$200 for the use of the same mail transport system when used by another product.

In addition to the "business" reasons for wanting to support transitive licensing, there is also a very good technical reason that arises from the growing tendency of developers to build "distributed products" as well as the drive toward application designs that exploit either tightly or loosely coupled multiprocessor systems; the availability and growing use of remote procedure calls has contributed to this tendency. This technical problem can be seen to arise when considering a product which has a number of components, each of which may run in a different process space and potentially on a different computer system. Thus, there might be a mail system whose user interface runs on one machine, its "file cabinet" is supported by a second machine and its mail transport system runs on yet a third machine. The simple question which arises is: "Which of the three components should check for licenses?" Clearly it must be ensured that no single component can be used if a valid license is not present. Thus, the answer to the question will probably be that all three components should check for licenses. However, the question is then presented: "Where are the licenses to be located?". This can become more complex.

Increasingly, the distributed systems being built are being designed so that it is difficult to predict on which precise machine any particular component will run. Ideally, networks are supposed to optimize the placement of functions

5 automatically so that the machine with the most available resource is always the one that services any particular request. This dynamic method of configuring the distribution of function servers on the network makes it very difficult for a system or network manager to predict which machines will run any particular function and thus very difficult for him to decide on which machines software licenses should be loaded.

10 Even if a system manager could predict which machines would be running the various application components and thus where the license units should be loaded, the situation would still be less than ideal. The problem arises from the fact that each of the components of the application would be independently making requests for license unit allocations. This behavior will result in a difficult problem for anyone trying to decide how many license units are required to support any one product. Given the mail example, the problem wouldn't exist if it were assumed that all three components (i.e., user interface, file cabinet, and transport system) were required by the design of the mail system to be in use simultaneously. If this were the case, it could be simply assumed that supporting a single activation of the mail system would require three units. However, in a real mail system, it will be inevitably discovered that many users will only be using just the user-interface and file-cabinet components of the system at one time. Thus, there will be some unused units available which could be used to authorize additional users. This situation might not be what is desired by the software vendor.

25 The problem of providing license support to multi-component products which are dynamically configured could be solved by viewing each of the product components as a distinct licensable product and by treating the problem as one

of transitive licensing, but a mechanism for accomplishing this has not been available. Essentially, a single license document would be created that stated that if any one of the components had successfully obtained a license to run, it could use this grant to give it the right to exploit the other components. Thus, in the
5 example above, the user might start the mail system by invoking its user interface. This user interface code would then query the license management facility for a license allocation and once it has received that allocation, it would pass a proof of allocation to the other mail components that it uses. Each of the other components would request that the license management system validate that the
10 "proof" is valid prior to performing any service; however, none of the other components would actually require specific allocations to be made to them. In this way, the complexity of licensing and managing networks of distributed applications can be significantly reduced.

SUMMARY OF THE INVENTION

15 In accordance with one embodiment of the invention, a license management system is used to account for software product usage in a computer system. The system employs a license management method which establishes a management policy having a variety of simultaneously-available alternative styles and contexts. A license server administers the license, and each licensed product
20 upon start-up makes a call to the license server to check on whether usage is permitted, in a manner similar to that of patent 4,937,863. The license server maintains a store of the licenses, called product use authorizations, that it administers. Upon receiving a call from a user, the license server checks the product use authorization to determine if the particular use requested is

permitted, and, if so, returns a grant to the requesting user node. The license server maintains a database of product use authorizations for the licensed products, and accesses this database for updating and when a request is received from a user. While this license management system is perhaps of most utility on a distributed computer system using a local area network, it is also operable in a stand-alone or cluster type of system. In a distributed system, a license server executes on a server node and the products for which licenses are administered are on client nodes. However, the license management functions and the licensed products may be executing on the same processor in some embodiments.

The product use authorization is structured to define a license management policy allowing a variety of license alternatives by components called "style", "context", "duration" and "usage requirements determination method". The style may be allocative or consumptive. An allocative style means the units of the license may be allocated temporarily to a user when a request is received, then returned to the pool when the user is finished, so the units may be reused when another user makes a request. A consumptive style means the units are deducted from an available pool when a user node makes a valid request, and "consumed", not to be returned for reuse. The context value defines the context in which the use is to be allowed, such as on a particular network, by a particular type of CPU, by a particular user name, by a particular process, etc. The duration value (used in conjunction with the style component) concerns the time when the license units are to be deducted from the available pool of units, whether at the time of request, after a use is completed, etc. A usage requirements determination method may be specified to define or provide information concerning the number of license units charged in response to a license request from a user node; for example, some CPU platforms may be charged a larger number of license units

than others. A table may be maintained of usage requirements, and the determination method may specify how to access the table, for example. The important point is that the user node (thus the software product) can only make a request, identifying itself by user, platform, process, etc., and the license management facility calculates whether or not the license can be granted (that is, units are available for allocation), without the user node having access to any of the license data or calculation. There is a central facility, the license server, storing the license documents, and, upon request, telling the licensed products whether they can operate under the license terms.

An important feature of one embodiment is that the license administration may be delegated to a subsection of the organization, by creating another license management facility duplicating the main facility. For example, some of the units granted in the product use authorization may be delegated to another server, where the user nodes serviced by this server make requests and receive grants.

The license management facility cannot create a license itself, but instead must receive a license document (a product use authorization) from an issuer of licenses. As part of the overall license management system of the invention, a license document generator is provided which creates the product use authorizations under authority of the owner of the software, as negotiated with customers. Thus, there are three distinct rights in the overall license management facility of the invention: (1) the right to issue licenses, (2) the right to manage licenses, and (3) the right to use the licensed products. Each one of these uses the license document only in prescribed ways. The license issuer can generate a license document. The license manager (or license server as referred to herein) can grant products the right to use under the license, and can delegate parts of the

- 42 -

licensed units for management by another server, as defined by the license document; the way of granting rights to products is by responding to certain defined calls from the products. And, the licensed products can make certain calls to the license server to obtain grants of rights based upon the license document, inquire, or report, but ordinarily cannot access the document itself.

As explained above, transitive licensing is an important feature of one embodiment. This is the provision of a mechanism for one user node to get permission to use another software product located on another user node; this is referred to as a calling authorization and a caller authorization, using a "calling card," and these are examples of the optional features which must be specifically permitted by the product use authorization. A user node must obtain permission to make a procedure call to use a program on another node; this permission is obtained by a request to the license server as before, and the permission takes the form of a calling card. When a calling card is received by a second node (i.e., when the procedure call is made), a request is made by the second node to the license server to verify (via the product use authorization) that the calling card is valid, and a grant sent to the user node if allowed. In this manner, all nodes may have use of a program by remote calls, but only one consumes license units.

Another important feature of one embodiment is a management interface which allows a license manager to modify the license policy components of a license document maintained by at a license server in its database. Usually the license manager can only make modifications that restrict the license policy components to be more restrictive than originally granted. Of course, the management interface is used to make delegations and assignments, if these are authorized.

- 43 -

The license document interchange format is an important feature, in that it allows the license management system to be used with a wide variety of software products from different vendors, so long as all follow the defined format. The format uses data structures that are defined by international standards.

5 An important function is the filter function, used in the management interface and also in the client interface to select among elements in the data structures.

BRIEF DESCRIPTION OF THE DRAWINGS

10 The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as other features and advantages thereof, will be best understood by reference to the detailed description of specific embodiments which follows, when read in conjunction with the accompanying drawings, wherein:

15 Figure 1 is a diagram in block form of a distributed computer system which may be used to implement the license management operations according to one embodiment of the invention;

 Figure 2 is a diagram of the content of a license document or "product use authorization" generated by the license document generator and stored by the license server in the system of Figure 1;

Figure 3 is a diagram of the alternatives for license style, context and duration making up the license management policy implemented in the system of Figure 1, according to one embodiment of the invention;

5 Figure 4 is a diagram of an example of a fragment of a license use requirements table (LURT) used in the system of Figure 1, according to one embodiment of the invention;

Figure 5 is a logic flow chart of a program executed by a user node (client), in the system of Figure 1, according to one embodiment of the invention;

10 Figure 6 is a logic flow chart of a program executed by a license server, in the system of Figure 1, according to one embodiment of the invention; and

Figure 7 is a diagram of the calls and returns made in an example of use of calling cards in the system of Figure 1.

Figure 8 is a diagram of an LDIF document identifier, according to an standard format;

15 Figure 9 is a syntax diagram of an LDIF document;

Figure 10 is a diagram of an LDIF document structure;

Figures 11, 13, 15, 17, 18, 19, 21-28 and 31-43 are syntax diagrams for elements of various ones of the LDIF data structures;

Figure 16 is a diagram of a license data structure;

Figures 12, 14 and 20 are examples of descriptions of data elements using a standard notation;

5 Figures 29 and 30 are examples of context templates used in the license management system;

Figures 44 and 45 are tables of attributes specific to filter and filter item type; and

Figure 46 is notation in a standard format for an example of a filter.

DETAILED DESCRIPTION OF SPECIFIC EMBODIMENTS

10 Referring to Figure 1, a license management facility according to one example embodiment of the invention is centered around a license server 10, which typically includes a CPU located in the customer's main office and executing a license management program 11 as will be described, under an operating system 12. The license server 10 communicates with a number of delegates 13 which
15 likewise include CPUs in departments or divisions of the company or organization, each also executing a license management program 14 under an operating system 15. The license management program 14 is the same as the program 11 executing on the main server 10; the only difference in the functions of server 10 and servers 13 is that the latter have a delegated subset of the license units granted to the
20 server 10, as will be described. The CPUs 13 are in turn servers for a number of

- 46 -

users 16, which are CPU nodes where the licensed programs 17 are actually executed. The programs 17 executing on the user CPUs 16 are applications programs (or operating systems, etc.) which have added to them units 18 and 19, according to the invention, allowing them to make inquiry to the their server 13 (or 10) before executing and to report back after executing, using a client stub 19 in the manner of remote procedure calls, in one embodiment. A user node 16 may have many different programs 17 that may be executed, and the various user nodes 16 would usually each have a set of programs 17 different from the other user nodes, all of which would be administered by the license management program 14 or 11. The terms "program" and "licensed product" are used in reference to the element 17, but it is understood that the products being administered may be segments of programs, or functions or features called by another program, or even merely data (such as printer fonts), as well as complete stand-alone applications programs. The license server 10 communicates with the delegatee servers 13 by a network 21, as is usual in large organizations, and the delegatee servers 13 each communicate with their user nodes 16 by networks 22; these networks may be of the Ethernet, token ring, FDDI types or the like, or alternatively, the user nodes 16 may be merely a cluster of terminals on a multiuser system with the delegatee being a host CPU. The particular hardware construction of the user nodes, server nodes, communication networks, etc., and the operating systems 12 or 15, are of no concern regarding the utility of the features of the invention, the only important point being that the user CPUs 16 of the software products 17 in question can communicate readily and quickly with their respective server nodes 13 or 10. In one embodiment, remote procedure calls (RPCs) are used as the communication medium for the interfaces between components of the system, handling the inquiries and grants as will be described.

- 47 -

A remote procedure call is similar to a local procedure call but is made to a procedure located on a remote node, by way of a communications network.

The function of the unit 19 is that of a client stub, in a remote procedure call sense. The calls to the license server 10 are made through this stub 19, and returns are received by the stub 19 and passed on to the program 17. The stub 19 is responsible for obtaining the network addresses of other nodes on the network, such as the server 10. Also, the stub 19 is responsible for determining the context (as defined below) for passing on to the server 10. The unit 18 functions to execute a "private" type of license availability determination if this is used, rather than this task being done by the application program 17, but if the ordinary method of determination is employed (using the license server) as is usually the case, the unit 18 is merely code that starts the execution and passes calls and returns back and forth between the program 17 and the unit 19.

The license server 10, using the license management program 11, maintains a license data file 23 comprising a number of license documents or licenses (product use authorizations), and also maintains a log 24 which is a record of the usage activity of all of the user CPUs 16 of each of the licensed programs. The delegatee servers 13 would maintain similar license databases and logs. The license server 10 has no authority to originate a license, but instead must receive a license from a license issuer 25. The issuer 25 is again a CPU executing a license document generator program 26 under an operating system 27. The license issuer 25 may be under control of the producer 28 of the programs or software products being licensed, or may be controlled by a distributor who has received the authority to grant licenses from the producer or owner 28. The communications link 30 between the license issuer 25 and the license server 10 for

- 48 -

5 This mechanism permits the system of the invention to dispose of the cumbersome, explicit support of license types having different scope such as the cluster licenses, node licenses, and process licenses found in prior license management systems including that of patent 4,937,863. Instead of defining a limited set of scopes (cluster, node, etc.), the system of this invention provides a general mechanism which allows an effectively unlimited range of allocation scopes to be defined.

10 Transitive licensing, as referred to above, is supported by the system of the invention by (1) calling authorizations, which are statements made in field 49 of the product use authorization 35 for one product (the "caller") to permit that product to call another product (the "callee"), and, (2) caller authorizations, which are statements made in field 49 of the product use authorization for one product (the "callee") to permit it to be called by another product (the "caller").

15 If calling or caller authorizations are to be exploited by products, then whenever one product calls another product, it must pass the callee a calling card 49a. This calling card 49a is an encoding of an identification of the caller as well as a statement by the license management system that a license unit allocation has been made to the caller which is passing the calling card. This calling card is then passed by the callee to the license management system for validation and, if the
20 either the product use authorization of the caller carries an appropriate calling authorization or the product use authorization of the callee carries an appropriate caller authorization, the use of the callee by the caller will be authorized without requiring any additional license unit allocations.

Referring to Figure 7, the intercomponent interactions that occur when either calling or caller authorizations are being used are illustrated. This figure shows a license management server 10, a caller product 17a named "Product-1" and a callee product 17b named "Product-2". When Product-1 starts to run, it will make an *lm_request_allocation()* call to the license management server 10 to obtain a grant handle for an allocation of some number of units of the Product-1 license. Either immediately, or at some later time, but always prior to making a call to Product-2, Product-1 will call *lm_query_allocation()*, passing the grant handle received earlier and specifying that it wants a calling card for the product named "Product-2." If the field 49 of the product use authorization 35 used to satisfy the grant represented by the grant handle carries a calling authorization in field 49 naming "Product-2," the license manager will create a calling card 49a which includes the statement that a calling authorization exists and pass this calling card back to Product-1. If the calling authorization does not exist, the calling card passed to Product-1 will contain a statement to that effect.

Once Product-1 has successfully obtained a calling card 49a from the license manager, it will then make a call to Product-2, passing the calling card along with any other initialization parameters that would normally be used when starting Product-2. Product-2 will then pass that calling card to the license manager as part of its *lm_request_allocation()* call and the license manager will determine if the calling card is valid. Note that calling cards become invalid once the process which received the calling card makes an *lm_release_allocation()* call or terminates abnormally. If the calling card is valid, and it indicates that a calling authorization is present, the license manager will verify this statement and if found to be true, will return a grant handle to Product-2. If, on the other hand, the calling card carries an indication that no calling authorization is present, the

- 50 -

license manager will attempt to find a product use authorization for Product-2 that contains a caller authorization naming Product-1 as an authorized caller. If the caller authorization is found, a grant handle will be passed back to Product-2. If not, the license manager will ignore the calling card and proceed with the normal
5 *lm_request_allocation()* logic.

The requirement to be passing calling cards between products requires that both the caller and the callee be "aware" of the fact that calling and caller authorizations may be used. This is one of the few examples of a requirement for a product 17 to become actively involved in the licensing problem when using the
10 licensing management system of the invention. However, since the use of calling/caller authorizations is a fairly "sophisticated" and powerful feature, it is considered acceptable to impose this burden on application coders.

MANAGEMENT INTERFACE

Referring to Figure 1, the license management program 11 executing on a
15 server 10 includes a license management interface 33 which functions to allow a user at a console for the server 10 CPU or at a remote terminal to implement certain necessary operations. The management interface 33 is essentially the tools or mechanisms available to the license manager at the licensee's site to (a) load the various licenses received from issuers 25 into the database 23 and make them
20 available for request allocation calls from the users, (b) remove the licenses from the machine when expired, (c) to make delegations if permitted, (d) to make assignments, (e) to make reservations, etc. Whatever the license manager is allowed to do to modify the license for his special circumstances (within the

- 51 -

original grant, of course), he does it by the mechanism of the management interface 33. Some licenses are not modified at all, but merely loaded. In a multiple machine environment, as on a network, there is considerable modification, as it is necessary to make sure the correct number of units are distributed onto the correct machines, the right people have access, other people don't have access, etc. Thus, in a network environment, there is extensive use of the management interface 33.

In reference to the terminology used in describing the management interface, as well as the license management system in general, it is helpful to note that the documentation conventions, data declarations, macro declarations, etc., for the object management used in one embodiment of the invention are according to the standards set forth in *OSI Object Management API Specification, Version 2.0*, X.400 API Association and X/Open Company Limited, 24 August 1990, a published document.

The specific operations available to the management interface 33 are to allow a manager to open and close a management session, register (load) objects in the license database 23, obtain a list of objects in the license database 23, and control a cursor (a cursor is a movable pointer to a member of a list of items). Once an object in the license database 23 is identified with the cursor, certain changes may be made in the object by a write function. For example, certain fields of a license document of Figure 2 or an LURT of Figure 4 may be changed in only specified ways as will be explained.

The operation of opening a session goes by the name of *lm_open_session()* and is used to establish a license management service session between a

management client and the service. Opening a session also creates a workspace to contain objects returned as a result of functions invoked within the session. Object management Objects can be created and manipulated within this workspace. Objects created within this workspace, and only such objects, may be used as Object arguments to the other license management service management functions used during the session established by a call to this function. More than one session may exist simultaneously.

The arguments that go with a *lm_open_session()* call are (a) the binding handle, which is binding information that defines one possible binding (a client-server relationship), and (b) a comment which will be inserted in the log file if logging is enabled. The results from a *lm_open_session()* call are (a) a return code indicating whether the function succeeded, and, if not, why not, (b) a session, which is an established license management session between the management client and the license management service, and (c) a workspace that will contain all objects returned as a result of functions invoked in the session.

The close session call is referred to by *lm_close_session()* and functions to terminate the lm session. This function terminates the license service management session and makes the argument unavailable for use with other interface functions. The arguments that go with a *lm_close_session()* call are (a) the session which identifies the established lm session between the management client and the license management service, and (b) a comment which will be inserted in the log file if logging is enabled. The result of the call is a return code indicating whether the function succeeded, and, if not, why not.

- 53 -

The list function returns a set of selected objects in the license database 23, and uses the name *lm_list_licenses()*. This function is used to search the license database 23 and return a cursor which represents the first of one or more objects which match the specified filter. The specified filter will be applied to each object in the license database 23; all objects for which the filter evaluates true will be included in the object list accessible by the *set_cursor* function. The arguments that go with *lm_list_licenses()* are (a) session which identifies an established session between the management client and the license management service, and (b) a filter which is an object used to select license database 23 objects; license database objects will only be included in the object list headed by the cursor if they satisfy the filter - the constant no-filter may be used as the value of this argument if all license data objects are to be included in the object list. The results of the *lm_list_licenses()* call are (a) a return code indicating whether the function succeeded, and, if not, why not, and (b) a license list upon successful completion of this call containing a cursor which represents the first of one or more objects in the current license database 23 for which the specified filter evaluates true.

The register function is to register objects in the license database 23, and uses the name *lm_register()*. This function is used to register (i.e., load or create) new objects, or modify existing objects, in the license database 23; the objects which may be registered include only those which are subclasses of the license data class or history objects. The arguments are (a) session, which identifies an established session between the management client and the license management service, (b) license data object which is to be registered; if this argument is omitted, the comment argument is a required argument and a history object containing the comment will be registered in the license database 23, and (c)

comment, which will be inserted in the log file if logging is enabled. The result is a return code indicating whether the function succeeded, and, if not, why not. The errors possible when it does not succeed include data-expired, duplicate-object, no-such-session, memory-insufficient, network-error, etc., indicated by this return code.

5

The set cursor function establishes a new cursor, and is called by *lm_set_cursor()*. The arguments are (a) session, which identifies an established session between the management client and the license management service, (b) forward, which is a boolean value indicating if the direction in which the cursor is to be moved is forward or reverse, (c) filter which is used to eliminate cursors from the search for the next cursor that are not wanted; a new cursor will only be set if it satisfies the filter - the constant no-filter may be used as the value of this argument if any cursor is to be considered as the target cursor, and (d) the cursor which is to be used as the starting point in searching for the new cursor. The results are (a) a return code indicating whether the function succeeded, and, if not, why not, and (b) next-cursor, which is the requested cursor. The error codes in the return code may be end-of-list, not-a-cursor, etc.

10

15

20

25

After a session is opened, and an object such as a product use authorization or a LURT has been identified by the cursor, using the functions explained above, the management interface 33 is able to execute certain object management interface functions such as write or copy. By this mechanism, the management interface can modify certain limited attributes. None of these attributes can be modified in such a way that they reduce constraints established by corresponding attributes in the license data objects. The more important attributes which can be modified by the management interface 33 using this mechanism are:

- 55 -

(a) assignment: an assignment of some or all of the units granted on the associated product use authorization;

(b) reservation: a reservation of some or all of the units granted on the associated product use authorization;

5 (c) delegation: a delegation of the right to manage some or all of the units granted on the associated product use authorization, or if the associated license data is not a product use authorization, the delegation is of the right to use that license data;

10 (d) backup delegation: a statement of the right to manage some or all or the units granted on the associated product use authorization; this right is only active at times when the delegating server is not available;

(e) allocation: an allocation of units to a specific context;

15 (f) allocation period: the minimum duration of a single allocation - all allocated units cannot be allocated to a new context until a time period equal to the allocation period has passed since the units were last allocated;

(g) termination date: a date which is to override the value specified as the end date of the product use authorization 40 - this date must be earlier than specified;

20 (h) delegation permitted: an override of the delegation permitted flag of the associated license data;

(i) overdraft: the current overdraft level;

(j) overdraft logging: an override of the overdraft logging attribute of the associated product use authorization;

25 (k) comment: a comment created by the licensee;

(l) extended info: information not defined by the architecture which may be of use in managing the license data.

5 It will be noted that an assignment and a reservation are identical, the only difference being that a reservation is something optional, while an assignment is something that is required. If the duration is Assignment in the policy declaration of Figure 3, the license manager must assign some or all of the units before units can be allocated. Reservations, on the other hand, are made by the license manager using the management interface, regardless of the policy.

10 Thus, there are certain attributes that can be changed by a license administrator using the management interface at the server 10, but none of these can result in obtaining more extensive rights to use than granted by the product use authorization. In each case, the license administrator can limit the rights which will be allocated to users in some way that may be appropriate for the administrator for control purposes.

LICENSE DOCUMENT INTERCHANGE FORMAT

15 The major structural components of an ASN.1 encoded document which conforms to the specifications for the license management system discussed above will be described. The object identifier that is assigned to this data syntax, according to one embodiment, is that specified in ASN.1 as seen in Figure 8. The International Standards Organization or ISO, as it is referred to, defines how bit patterns are chosen to uniquely identify an object type, so the bit pattern set forth
20 in Figure 8 would precede each document used in the license management system so the document could be identified as being a document conforming to the prescribed License Document Interchange Format.

5 A document encoded according to this format is represented by a value of a complex data type called "license document interchange format document" of LDIFDocument, in this embodiment. A value of this data type represents a single document. This self-describing data structure is of the syntax defined in the international standard ASN.1 referred to above. The X/Open standard referred to above defines the conventions that must be used in employing this syntax, while the syntax itself is described in an OSI (Open Systems Interconnect, a standard administered by ISO) document identified as X.409 (referenced in the X/Open document identified herein).

10 The LDIFDocument data type consists of an ordered sequence of three elements: the document descriptor, the document header, and the document itself. Each of these elements are in turn composed of other elements. The overall structure of the LDIFDocument data type will be described, and the nature of the document descriptor and document header types. Then, the document content
15 elements will be described in detail, as well as the various component data types used in the definition of the descriptor, the header and the content.

The LDIFDocument represents a single license document, with the syntax being shown in Figure 9 and the high-level structure of an LDIF document in graphical form being seen in Figure 10. The DocumentDescriptor of Figure 9 is
20 a description of the document encoding, the DocumentHeader contains parameters and processing instructions that apply to the document as a whole, and the DocumentContent is the content of the document, all as explained below.

Referring to Figure 9, what this says is that an LDIFDocument is composed of (::= means "is composed of") a number of elements, the first thing in an

LDIFDocument is a bit pattern (tag) according to an international standard, indicating a certain type of document follows, which is indicated here to be "private" or vendor selected, the number 16373 in this case. Following the bit pattern which functions as a "starting delimiter" it is "implicit" that a "sequence" of elements must follow, where a sequence is distinguished from a set. A sequence is one or more of the elements to follow, whereas a set is exactly one of the elements to be listed. Implicit means that any file identified as LDIFDocument must have a sequence data type, rather than some other type. In the case of Figure 9, the sequence is document-descriptor, document header and document content; the document-content is mandatory, whereas the first two are optional. If an element in the sequence begins with a "0" it is a document-descriptor, "1" means a document-header, and "2" means it is a document-content. Again, it is implicit that the data following is of the format DocumentDescriptor, etc., in each case, and these are defined in Figure 11, Figure 13 and Figure 15.

Each file is in the tag-length-value format mentioned above, and also each element of a file containing multiple elements is of the tag-length-value format. The data stream could be examined beginning at any point, and its content determined by first looking for a tag, which will tell what data structure this is, then a length field will say how long it is, then the content will appear. These structures are nested within one another; a document containing several product-use-authorizations would be an LDIFDocument of the format of Figure 9, with a number of DocumentContent elements of Figure 15 following, with the length given for the LDIFDocument spanning the several PUAs, and the length given for each PUA being for the one PUA.

In Figure 11, the elements major-version and minor-version are seen to be "implicit integer". This means that because the element is of the type major-version, etc.. it must be an integer. Various other implicit types are given in other syntax diagrams, such as character-string, boolean, etc.

5 In Figure 15, the license body is identified as being of the type "choice" meaning it can be one of PUA, LURT, GroupDefinition, KeyRegistration, etc. Thus, knowing this is a license-body does not mean the data type of the object is known; it is a bit further where the kind of a license-body becomes known. The definition of a license body is not implicit, but instead is a choice type.

10 The contents of the various data elements will now be described in detail with reference to Figures 11-43. Using these detailed descriptions, the exact format of each of the elements used in the LDIF can be interpreted.

15 The license document descriptor or DocumentDescriptor consists of an ordered sequence of four elements which specify the version level of the LDIF encoding and identify the software that encoded the document, with the syntax being shown in Figure 11. An example of the way a product called PAKGEN V1.0 is expressed in the DocumentDescriptor encoding is shown in Figure 12. The fields in the DocumentDescriptor syntax are major-version, minor-version, encoder-identifier and encoder-name. The major-version field is the primary
20 indicator of compatibility between LDIF processors and the encoding of the present document; this major-version field is updated if changes are made to the system encoding that are not backward compatible. The minor-version field is the revision number of the system encoding. The encoder-identifier field is a registered facility mnemonic representing the software that encoded the document;

- 60 -

the encoder-identifier can be an acronym or abbreviation for the encoder name -
this identifier is constant across versions of the encoder. The encoder-identifier
should be used as a prefix to Named Value Tags in Named Value Lists to identify
the encoder of the named value. The encoder-name field is the name of the
5 product that encoded the document; the encoder-name string must contain the
version number of the product.

The document header or `DocumentHeader` contains data that pertains to
the document as a whole, describing the document to processors that receive it;
the syntax is shown in Figure 13. An example of a document header is shown in
10 Figure 14, using the hypothetical product `PAKGEN V1.0` of Figure 12. The
`private-header-data` contains the global information about the document that is not
currently standardized; all interpretations of this information are subject only to
private agreements between parties concerned, so a processor which does not
understand private header data may ignore that data. The `Title` field is the user-
15 visible name of the document. The `Author` field is the name of the person or
persons responsible for the information content of the document. The `Version`
field is the character string used to distinguish this version of the document from
all other versions. The `Date` field is the date associated with this document. Note
that the nature and significance of the `Title`, `Author`, `Version`, and `Date` fields can
20 vary between processing systems.

The content of an LDIF document is represented by a value of a complex
data type called `DocumentContent`. An element of this type contains one or more
`LicenseData` content element using a syntax as shown in Figure 15. There are no
restrictions on the number, ordering or context of `LicenseData` elements. The
25 structure of a `LicenseData` element is represented in Figure 16. No restrictions

- 61 -

are made on the number, ordering, or context of LicenseData elements. The license-data-header field of Figure 16 specifies that data, common to all types of license data, which describes the parties to the licensing agreement, the term of the agreement, and any constraints that may have been placed on the management of the license data encoded in the license body. The license-body is an element that contains one content element, including: product use authorizations, license unit requirements tables, group definitions, key registrations, and various forms of delegations. The Management-Info is an element that contains information concerning the current state of the license data; this element is not encoded by Issuers.

The license data header, called LicenseDataHeader, is represented as a syntax diagram in Figure 17. The license-id field provides a potentially unique identification of the encoded license data, so issuers of license data can generate unique license-ids to distinguish each issuance of license data; however, the architecture does not require this to be the case, since the only architectural restriction is that no two objects in any single license management domain may have the same value for license-id. The licensee field identifies the party who has received the rights reflected in the license data; there are at least two parties involved in all transfers of license data, first, the issuer of the license data, and second, the licensee or recipient of that data - it is anticipated that individual licensees will specify to those issuing them licenses what the licensee fields on their license data should contain. the term field identifies the term during which the license data may be used; the validity of license data can be limited by issuers to specific time ranges with given starting and ending dates, which are carried in the term element - attempts to use license data or products described by that data either before the start date or after the end date will result in conforming license

- 62 -

managers denying access to the license. Management-constraints identifies constraints placed on the right to manage the associated license data; these constraints can include (a) limiting the set of contexts permitted to manage the data, (b) limiting the set of platforms which may benefit from that management, and (c) limiting the right to backup and delegate the managed data. The signature provides the digital signature used by the issuer to sign the license data and identifies the algorithm used in encoding the signature. Issuer-comment is a comment provided by the issuer and associated with the license data.

The IssuerComment is of an informational nature and does not impact the process of authorizing product or feature use. This field is not included in the fields used to generate the signature for a license, thus, even if specified by an issuer, the IssuerComment can be omitted from a license without invalidating the license. If specified, the IssuerComment should be stored in the appropriate license data base with the associated license data. The IssuerComment can be retrieved by products which use the system and may be of particular utility to products in the "Software Asset Management" domain which are intended to extend or augment the administrative or accounting facilities or basic system components. Some examples of potential uses for this field are order information, additional terms and conditions, and support information. For order information, some issuers may wish to include with their loadable license data some indication of the purchase order or orders which caused the license data to be issued; licensees may find it useful to include this data in their license databases to assist in the license management process. For additional terms and conditions, the system will never provide automatic means for the management of all possible license terms and conditions, and so some issuers may wish to include summaries of non-system managed terms and conditions in the comment as a reminder. For

support information, the IssuerComment could be used to record the phone numbers or addresses of the responsible individuals within the issuing organization who should be contacted if there are problems with the data as issued.

5 A product use authorization as previously discussed in reference to Figure 2 is used to express the issuance of a right to use some product, product feature, or members of some product group. As such, it records the identity of the product for which use is authorized and specifies the means that will be used by the license manager to ensure that the licensee's actual use conforms to the terms and conditions of the license. Figure 18 illustrates a syntax diagram for a
10 ProductUseAuthorization. Product-id identifies the name of the producer of the product or product feature of which usage rights are being granted as well as the name of that product; in addition, issuers of product use authorizations may specify a range of versions and/or releases whose use is controlled by the specific product use authorization. Units-granted - Contains the number of units of
15 product use which are granted by the license. Management-policy defines the policy which is to be used in managing the granted software usage rights; this definition specifies the Style, Context-Template, Duration, and License Unit Requirements Determination Method which must be used. The calling-authorizations and caller-authorizations are as explained above in reference to
20 calling cards. The execution-constraints field identifies constraints placed on the characteristics of execution contexts which may be authorized to benefit from the units granted by this Product Use Authorization. The product-token field contains product specific data not interpreted in any way by any processors conformant with the architecture; software product producers 28 use this array to augment the
25 capabilities of conformant license managers.

Some anticipated uses of the token field include language support, detailed feature authorizations, and product support number. For language support, a token could be constructed which contains a list of local language interface versions whose use is authorized; thus, if a product were available in English, German, French and Spanish, a token could be constructed listing only English and German as the authorized languages. For detailed feature authorizations, some license issuers will wish to have very fine control over the use of features in a complex product; however, they may not wish to issue a large number of individual Product Use Authorizations to "turn on" each feature, so these vendors could construct tokens which contain lists of the features authorized or whose use is denied. For product support number, some issuers may wish to include on the product use authorization, and thus make available to the running product, some information concerning the support procedures for the product; for example, an issuer might include the telephone number of the support center or a support contract number, and the product could be designed to retrieve this data from the license manager and display it as part of Help dialogues.

The LURT's or license use requirements tables of Figure 4 provide a means by which issuers of licenses, whose LURDM is dependent on the type of platform on which the product is run, can store information describing the relationship between the platform type and unit requirements. A syntax diagram for a LURT is shown in Figure 19. In Figure 20, an example of how the LURT of Figure 4 might be encoded is illustrated. Lurt-name specifies the name by which the LURT is to be known to conforming license managers. The rows field models a list of multicolumn lurt rows. Platform-id identifies the platform for which this LurtRow provides license unit requirements. The lurt-columns field provides a list of one or more lurt column values; the first value provided is

- 65 -

5 assigned to column-1 of the lurt-row, the second value provided is assigned to column-, etc. A lurt column value of -1 indicates that use of the product or feature is not authorized, while a lurt column value of 0 or greater indicates the number of units that must be allocated in order to authorize product use on the platform described by this lurt-row. All unspecified columns (e.g., columns whose number is greater than the number of column values provided in the lurt columns element) will be considered to contain the value -1.

10 In reference to Figure 19, to use the row-selector feature mentioned above, the platform-ID element would be replaced with *row-selector* which would be implicit of Context. Also, in Figure 34 described below, in the lurdm-kind element, *row-selector* would be included if the row-select feature is to be used.

15 As discussed above, Figure 4 provides an example of a hypothetical LURT, illustrating the LURT mechanism, where the issuer of this LURT table has established three unit requirement tiers for use in determining the unit requirements for that issuer's products. Figure 20 provides an example of how the LURT presented in Figure 4 might be encoded.

20 A group definition is used to define and name a license group. Once so defined, the name of this group can be used on product use authorizations in the same manner as a product name. Since a single product use authorization specifies the management policy for all members of the group, the members of that group must be compatible in their licensing styles, i.e., a personal use type product can not be mixed with a concurrent use product in the same group. Figure 21 shows a group definition syntax diagram. Group-name is the name which must appear on Product Use Authorizations for this group. Group-version

5 specifies the current version of this group; the requirements for matching between the version information on a product use authorization and that on a specified group definition are the same as those rules which require matching between produce use authorizations and the Release Date data provided by products. Group-members lists those products or features which are components of the named group.

10 A key registration is used by a producer 28 or issuer 25 who have been registered as authorized license issuers and provided with an appropriate public and private key pair. The key registration identifies the public key which is to be used by conforming license managers 10 in evaluating signatures 53 created by the named issuer 25 or producer 28. A key registration syntax diagram is shown in Figure 22. Key-owner-name provides the name which must be used in either of, or both, of the Producer and Issuer fields of license data generated by the issuer; the key-owner-name must be identical to that specified in the Issuer field of the header record. Key-algorithm identifies the registered algorithm that is to be used 15 when producing digital signatures with this key. Key-value identifies the public key.

20 An issuer delegation is typically issued by a producer 28 and authorizes the named issuer 25 to issue licenses for products produced by the producer. An issuer delegation syntax diagram is shown in Figure 23. Delegated-issuer-name identifies the name which must appear in the Issuer field of any Product Use Authorization generated using the License Issuer Delegation. Delegated-product-id identifies the products whose licenses the named issuer is authorized to issue. Delegated-units-granted, if specified, indicates that the use of this IssuerDelegation 25 is to be managed in the style of a consumptive license; the value of this attribute

- 67 -

gives the number of units for which license documents may be generated (i.e., if granted 1000 units by a Producer, an Issuer can only issue 1000 units.) Template-authorization provides a "template" Product Use Authorization whose attribute values must be included on any Product Use Authorization generated using this IssuerDelegation; in the case of attributes which have a scalar value (i.e., Version, Release Date, etc.), the Issuer may issue licenses with more restrictive values than those specified on the Template Authorization. Sub-license-permitted indicates whether the Issuer identified on this IssuerDelegation may issue an IssuerDelegation for the delegated-product-id.

A license delegation, as shown in a syntax diagram of Figure 24, is used to delegate the right to manage license data. Such delegations are created by the licensee (by the license manager), if authorized by the issuer 28. A backup delegation, also shown in Figure 24, is used by one license management facility to authorize another to manage the delegated rights in the case that the delegating license manager is not running. The delegated-units field specifies the number of units whose management is being delegated; this may only be specified when a product use authorization is being delegated. Delegation-distribution-control defines the mechanisms by which the distribution and refreshing of the delegation will be accomplished. Delegatee-execution-constraints identifies any constraints which are placed on the execution-context of the Delegatee; these constraints are applied in addition to those which are a part of the delegated License Data. Assignment-list identifies any assignments of the delegated units that must be respected by the delegatee. Delegated-data stores a copy of the LicenseData received from the issuer that is the subject of the delegation; the delegated data is not provided when the LicenseDelegation element is included in a DelegationList.

5 The management information or ManagementInfo element records
 information concerning the current state of the LicenseData with which it is
 associated. A syntax diagram of the ManagementInfo element is shown in Figure
 25. The assignments field identifies a list of one or more assignments which may
 be outstanding for the units on the associated product use authorization.
 10 Reservations identifies a list of one or more reservations which may be
 outstanding for the units on the associated product use authorization. Delegations
 identifies a list of all outstanding delegations. Backup-delegations identifies all
 outstanding backup delegations. the allocations field provides detailed
 15 information about outstanding allocations which involve units from the associated
 product use authorization. Registration-date is the date on which the LicenseData
 was registered in the license database. Registrar is the context which caused the
 LicenseData to be registered. Local-comment is a comment field. Termination-
 date means a license defined date after which the license data may not be used;
 this date must be earlier than the end-date specified in the license data's term
 record. The extended-info field allows additional information concerning the state
 of the LicenseData and its handling by the license manager that is not
 standardized.

20 The defined types of elements will now be described. These defined type
 are:

- | | | |
|----|----------------------|------------------|
| 25 | Allocation | ManagementPolicy |
| | Assignment | Member |
| | Context | NamedValue |
| | DistributionControl | NamedValueList |
| | ExecutionConstraints | ProductID |
| | IntervalTime | Signature |

- 69 -

LicenseID	Term
LUDRM	Version
ManagementConstraints	

5 The allocation element records the information concerning a single unit allocation, and is shown in a syntax diagram in Figure 26. Allocation-context specifies the context to which the allocation was made. The allocation-lur field specifies the license unit requirement which applies to the allocation-context; this license unit requirement is calculated without consideration of any allocation sharing which may be possible. The allocation-group-id field identifies the
10 "allocation-group" for the current allocation, in which an unshared allocation will always have an allocation group id of 0; allocations which utilize shared units will have an allocation group id which is shared by all other allocations sharing the same units.

15 The assignment element is shown in syntax diagram in Figure 27. Assigned-units identifies the number of units which are assigned. Assignment-term identifies the start and end of the assignment period. Assignee identifies the context to which the assignment is made.

20 The context element is shown in syntax diagram in Figure 28. The SubContext-type field identifies the type of subcontext, and this type can be either standard or private; if standard, the type value will be taken from the standard-subcontext-type enumeration: (a) network-subcontext means the subcontext value identifies a network; (b) execution-domain-subcontext means the subcontext value is the name of the management domain within which the caller is executing; (d) login-domain-subcontext means the subcontext value is the name of the

management domain within which the user of the caller was originally authenticated or "logged in"; (d) node-subcontext means the subcontext value is the name of a node; (e) process-family-subcontext means the subcontext value is an implementation specific identifier for a group of related processes; (f) process-ID-subcontext means the subcontext value is an implementation specific process identifier; (g) user-name-subcontext means the subcontext value is a user name; (h) product-name-subcontext means the subcontext value is the same as the product name found on the Product Use Authorization; (i) operating-system-subcontext means the subcontext value is a character string representation of the name of the operating system; (j) platform-ID-subcontext means the subcontext value is an identifier that describes the hardware platform supporting the context. The subcontext-value field is the value of the subcontext.

As discussed above, license data is always used or allocated within, or for the benefit of, some named licensing context. This context name is constructed by concatenating the values of all subcontexts into a single context name. A Context Template specifies those components of the context name which should be used in calculating license unit requirements. The management system determines the need to perform a unit allocation each time license units are requested. The full context on whose behalf the allocation should be made is obtained for each requested authorization. The system will mask the full context to exclude all sub-contexts not specified in the context template and then determine if the resulting context already has units allocated to it. If not, units will be allocated according to the specification of the LURDM, otherwise, the units previously allocated will be shared by the new context. Thus, if a given product authorization contains a context specification of NODE + USER_NAME, each context which requests license unit allocations and which has a unique pair

of NODE + USER_NAME subcontext values will require an explicit grant of license units to be made. On the other hand, any contexts which share the same pair of NODE and USER_NAME subcontext values will be able to "share" a single allocation of license units. The requirement for specific allocations of units and the ability to share units is exhibited in Figure 29 which attempts to provide a "snapshot" of the units allocated for the product FOOBAR V4.1 at a particular instance. It is seen from the figure that although presented with five unique full contexts, only four of them are unique when looking only at those portions of each context which are described by the Context Template (ie: NODE + USER_NAME). A unit allocation must be made for each of the four instances of unique contexts, when masked by the Context Template. The fifth context can share allocated units with another context. Thus, the total requirement to support product use as described in this example would be 40-units (ie: four allocations of ten units each). Significant changes in the unit requirements can be achieved by making small modifications to the Context Template. Figure 30 shows the same contexts as in Figure 29 but a Context_Template of NODE. The total unit requirement for this example would be three units (three allocations of ten units each) rather than the forty units required in the previous example.

The distribution control element defines the mechanism that will be used for distributing the subject delegation and records some status information concerning the distribution of that delegation. A syntax diagram of the distribution control element is shown in Figure 31. Distribution-method identifies the means by which the delegation will be distributed, and the alternatives are refresh-distribution, initial=distribution-only, and manual-distribution. Refresh-distribution means the license manager shall be responsible for the initial distribution of the delegation and for ensuring that refresh delegations are

properly distributed. Initial-distribution-only means the license manager shall be responsible for the initial distribution of the delegation, however, distribution of refresh delegations will be made by some other means. Manual-distribution means the distribution of the delegation will be under the control of some other mechanism (perhaps a license asset manager). Current-start-date is the time that the last successful initial or refresh delegation distribution was performed. Current-end-date identifies the last date on which the most recent delegation distribution was performed. Refresh-interval identifies the period of time between attempts to refresh the delegation; the refresh-interval may not be longer than the maximum-delegation-period and should normally be less than that in order to ensure that refresh delegations are distributed prior to the expiration of the previous delegations that they are replacing. Retry-interval identifies the amount of time to wait for an unsuccessful distribution attempt to try again. Maximum-retry-count identifies the maximum number of times that an unsuccessful distribution attempt may be retried. Retries-attempted records the number of unsuccessful retry attempts which have been made since the last successful initial or refresh delegation distribution was performed.

The execution constraints elements place limits on the environments and contexts which may receive allocations. A syntax diagram of the execution constraints element is shown in Figure 32. Operating-system contains a list of zero or more operating systems on which the use of the subject license is authorized; if no operating systems are specified, it is assumed that license use is authorized on all operating systems. Execution-context specifies a list of zero or more full or partial context names which identify the contexts within which products described by the license data may be executed; if no context names are specified, the licensed products may be executed in any context controlled by the licensee.

- 73 -

Environment-list identifies those environments within which the licensed product may be used.

The interval time element is defined by the syntax `IntervalTime ::= UTCTime`.

5 The license ID element uniquely identifies the license data it is associated with, and is described by the syntax diagram of Figure 33. Here issuer uniquely identifies the issuer of the license data as well as the name space within which the LicenseID Number is maintained. While the issuer name will typically be the same as the name of the issuer's company or personal name, this is not a
10 requirement. For instance: The issuer name for Digital Equipment Corporation is "DEC," an abbreviation of the corporate name. Valid contents of the Issuer field are maintained in the an Issuer Registry. The serial-number provides a unique identification or serial number for the license data. The amendment field is an integer which is incremented each time license data is amended by its issuer,
15 with the first version of any license data carries the amendment number 0; an amendment can only be applied to license data if that license data has identical Issuer and Number values and an amendment number less than the number of the amendment to be applied.

20 The license units requirements determination method or LURDM element is shown in syntax diagram in Figure 34. The combination-permitted field indicates whether conforming license managers are permitted to combine together into a common pool the units from different product use authorizations if those produce use authorizations have the same product record value; for example, if combination is permitted and a single license manager discovers in its database

-74-

two 500-unit authorizations for the use of DEC Cobol, the license manager would be permitted to combine these two authorizations into a logical grant of 1000 units. The overdraft-limit modifies the behavior of a conforming license management facility in those cases where it is found that there are zero or fewer license units available for use at the time of a request for the allocation or consumption of additional license units. Operation of overdraft is different depending upon whether allocative, or consumptive style is being used. In using with allocative style, an allocation is granted even though the remaining units are zero or less, up to the overdraft-limit. In using with consumptive style, the license is authorized to accumulate a negative balance of license units, up to the overdraft-limit. Overdraft-logging-required indicates whether all license grants which are the result of overdraft use must cause a log record to be generated. When the allocation-size field is non-zero, then all unit allocations and delegations must be made in sizes which are whole number multiples of the allocation-size value. Lurdm-kind identifies the method by which license unit requirements will be calculated once the requirement for an allocation has been discovered, the permitted alternatives being (a) LURT which specifies that license unit requirements are to be determined by lookup in the LURT which is associated with the current license, (b) Constant which specifies that license unit requirements are constant for all platforms on which the licensed product or product feature may run, and (c) Private-LURDM which specifies that license unit requirements are to be determined by the licensed product, not by the license management facility. The named-lurt-id specifies the name of the LURT table to be used in determining license unit requirements if the LURDM-kind is specified as LURT; if the LURDM-kind is specified as LURT and no table is explicitly named, the name of the table to be used is constructed from the issuer name on the product use authorization. Lurdm-value specifies the LURT column to be

-75-

5 used when LURDM-kind = LURT; however, when LURDM-kind = Constant, the Lurdm-value field contains the precise number of units to be allocated or consumed. Default-unit-requirement specifies the unit requirement value to be used when the appropriate LURT does not have a row corresponding to the appropriate platform ID; when specified on a product use authorization with Style = Allocative, the context template will change to Process + Product_Specific and the Duration will change to Transaction in cases of unrecognized Platform ID's.

10 The management constraints element is shown in a syntax diagram in Figure 35. The management-context field specifies a list of zero or more partial context names which identify the specific contexts within which the license data may be managed. If no management contexts are specified, the license data may be managed within any context controlled by the licensee. The contexts used in specifying Management Context Constraints may only contain the Network, Domain, and Node subcontexts. Specifying a list of management contexts does not effect whether or not the license data can be used within other contexts. For example, unless otherwise restricted, license data with a specified management context can be remotely accessed from or delegated to other nodes in a network. The management-scope field defines the maximum permitted size of the license management domain within which the license data may be managed or distributed, these being single-platform, management-domain, or entire-network. Single-platform constrains the license management domain for the subject license data to be no larger than a single platform. Management-domain constrains the license management domain for the subject license data to be no larger than a single management domain. Entire-network constrains the license management domain for the subject license data to be no larger than a single wide area network; that

15

20

25

network which contains the platform on which the license units were initially loaded. Although technology may not exist to detect the interorganizational boundaries of a wide area network (i.e., what is on the Internet as opposed to being on a company's own network), the assumption is that interorganization and internetwork sharing of licenses will normally be considered a violation of license terms and conditions. The backup-permitted field indicates if the Issuer has authorized the use of backup delegations for this data. Delegation-permitted indicates if the Issuer has authorized the licensee to delegate this data. Maximum-delegation-period identifies the longest interval during which a delegation may be valid; by default, delegations have a life of 72-hours.

The major elements of the management policy specification are shown in Figure 3, as previously discussed. A syntax diagram for the management policy element is shown in Figure 36. For the Style field, three fundamental styles of license management policy are supported, allocative, consumptive, and private-style, as explained above. Only one of these styles may be assigned to any single product use authorization. The Context-template specifies those components (sub-contexts) of the execution-context name which should be used in determining if unit allocations are required. The Duration defines the duration of an allocation of license units to a specific context or the duration of the period which defines a valid consumptive use. For durations of type "Assignment," the specification of a Reassignment Constraint is also provided for. Three types of Duration_Kind are supported, these being Transaction, Assignment and Immediate, as explained above. The lur-determination-method stores information used in calculating the number of units that should be allocated or consumed in response to a license request. The allocation-sharing-limit identifies the largest number of execution contexts that may share an allocation made under this management policy; an

allocation-sharing-limit of 0 indicates that the number of execution contexts that may share an allocation is unlimited. The reassignment-constraint specifies a minimum duration of assignment; although there is normally no constraint placed on how frequently granted units may be reassigned, an issuer may constrain reassignment by specifying this minimum duration of an assignment, in which case reassignment of assigned units will not be supported until the amount of time specified in the Reassignment Constraint has passed. If an assignment of some particular set of units has been delegated and the delegation period for that delegation has not terminated, cancellation of the delegation must be performed prior to reassignment.

The member element identifies a specific licensed product which may be part of a calling authorization or group definition, and is shown in syntax diagram in Figure 37. Member-product identifies the product which is a member. Member-signature is constructed from the product and token fields of the called member structure as well as the product and issuer fields of the calling product. Member-token provides the data which should be used as the product token for this member.

Named values are data elements with a character string tag that identifies the data element, and have a syntax as shown in Figure 38, which also shows the syntax for ValueData and named value list. A named value list models a list of named values, with an example being shown in Figure 39. In Figure 38, Value-Name uniquely identifies the value; no standard value names are defined, and the period character can be used as a part of the value name to form a hierarchical tag registry at the discretion of the issuer. Value-data is the data that has been named; data types are selected from the possible Value Data types, seen in the

Figure. Value-boolean means the named data is a boolean value. Value-integer means the named data is an integer value. Value-text means the named data is a StringList value. Value-general means the named data is a stream of bytes in any format. Value-list means the named data is a list of named data values.

5 The product ID explicitly identifies the product which is the subject of the license data with which it is associated, with the syntax for ProductID being shown in Figure 40. The version and release date fields provide a mechanism for defining which specific instances of the licensed product are described in the associated license data. The Producer field is a registered name which identifies
10 the producer of the licensed feature; in the case of Group Names, the Producer is always also the Issuer of the group. The Product-name identifies a licensed software feature. The First-version identifies the earliest version of the product whose use is authorized. The Last-version identifies the latest version of the product whose use is authorized. The First-release-date identifies the earliest
15 release of the product whose use is authorized. The Last-release-date identifies the latest release of the product whose use is authorized. Conforming license managers are required to interpret the contents of these fields in the most restrictive way possible. Thus, if a license is issued with Last-version = 3.0 and a Last-release-Date of 1-Jan-1991, then the use of version 2.0 of the licensed
20 product would be unauthorized if it had a release date of 2-Jan-1991. If either a First-version or First-release-date is specified without a matching Last-version or Last-release-date, use of the produce is authorized for all versions or release dates following that specified. Similarly, if either a last-version or Last-release-date is specified without a matching First-version or First-release-date, use of the produce
25 is assumed to be authorized for all versions or release dates prior to that specified. Issuers should typically only specify one of either First-version or First-release-

date. This is the case since it is anticipated that these fields will typically refer to events which occurred prior to the moment of license data issuance. Thus, it should normally be possible for the issuer to state unambiguously with only one of these two fields which is the oldest implementation of the product that is to be authorized. The architecture does permit, however, both fields to be used in a single product authorization.

The signature element is used to establish the integrity and authorship of the license data with which it is associated. A syntax diagram for the signature element is shown in Figure 41. The Signature-algorithm field identifies the registered algorithm that was used to produce the digital signature. Signature-parameters are the values of the algorithm's parameters that are to be used; the need for and syntax of parameters is determined by each individual algorithm. Signature-value is an enciphered summary of the information to which the signature is appended; the summary is produced by means of a one-way hash function, while the enciphering is carried out using the secret key of the signer (Issuer).

The term element defines an interval during which the license data is valid, and is shown in syntax diagram form in Figure 42. The fields are start-date and end-date. Start-date identifies the first date of the term; if not specified, the license data is considered valid on any date prior to the end-date. End-date identifies the last date of the term; if not specified, the license data is considered valid on any date after the Start-date. While the Start-date is always either omitted or specified as an absolute date, the End-date can be either absolute or relative. If the End-date is specified as a relative or "interval" date and the Start-date has been omitted, the date of license registration will be used as the effective

- 80 -

5 start date in computing the valid term of the license data. It should be noted that the system does not specify the mechanism by which system dates are maintained by platforms supporting system components. Instead, the system always accepts that system time returned to it as correct. Thus, the reliability of the management of license data which specifies terms is dependent on the time management function of the underlying platform.

10 The version element identifies a four-part version of the licensed software product or feature. A syntax diagram of the version element is shown in Figure 43. The schematics of each of the four parts is not detailed, but it is required that producers who wish to permit version ranges to be specified on product use authorizations ensure that the collating significance of the four parts is maintained. When comparing versions, Part-1 is considered first, then Part-2, then Part-3, and finally, Part-4. Part-1 identifies a major modification to the versioned object. Part-2 identifies a modification to the versioned object which is less significant than a modification which would cause a change in the Part-1 value. Part-3 identifies a modification to the versioned object which is less significant than a modification which would cause a change in the Part-2 value. Part-4 identifies a modification to the versioned object which is less significant than a modification which would cause a change in the Part-3 value.

20 FILTERS

An important feature is the use of filters in the license management program 11, including the client interface 31 and the management interface 33. A filter is used to select items in the license database 23, for example. Various

- 81 -

5 selection mechanisms are used in picking out or doing lookups in database technology; filters are one of them. The filter engine used in the license management system 11 of Figure 1 is generally of a known construction, with the exception of the select filter item type as will be described, which allows a complex rather than a flat data format to be selected from. The feature that is of importance to this embodiment is the way of specifying items as an input to the filter function , rather than the filter function itself. Thus, there is described below a template for specifying input to the filter engine. This is as if a form were used as the input, with blanks on the form; by filing in certain blanks these would be the items selected on, the blanks not filled in would be "don't care".

10 An instance of the class *filter* is a basis for selecting or rejecting an object on the basis of information in that object. At any point in time, a filter has a value relative to every object - this value is false, true or undefined. The object is selected if and only if the filter's value is true. This concrete class has the attributes of its superclass - *Object* - and the specific attributes listed in the table of Figure 44.

20 A filter is a collection of simpler filters and elementary filter-items together with a Boolean operation. The filter value is undefined if and only if all the component filters and filter-items are undefined. Otherwise, the filter has a Boolean value with respect to any object, which can be determined by evaluating each of the nested components and combining their values using Boolean operation (components whose value is undefined or ignored). The attributes specific to *filter* as shown in Figure 44 are (a) *filter items* which are a collection of assertions, each relating to just one attribute of an object, (b) *filters* which are a

collection of simple filters, and (c) *filter type* which is the filter's type, of one of the following values: And, Or, Not.

5 An instance of the class *filter item* is a component of a *filter*. It is an assertion about the existence or values of a single attribute of a license data object or one or its subobjects. This concrete class has the attributes of its superclass - *object* - and the specific attributes listed in the table of Figure 45.

10 The value of a filter item is undefined if: (a) the Attribute Types are unknown, or (b) the syntax of the Match Value does not conform to the attribute syntax defined for the attribute type, or (c) a required Attribute is not provided. The attributes specific to *filter item* as shown in Figure 45 are (a) *filter item type* which identifies the type of filter item and thereby the nature of the filter, and its value must be one of

	equality	less
	inequality	present
15	greater or equal	select
	less or equal	request candidates
	greater	simulate request

20 (b) *attribute type* which identifies the type of that attribute whose value or presence is to be tested; the value of All Attributes may be specified, (c) *match value* which is the value which is to be matched against the value of the attribute, (d) *filter* which identifies the filter to be used in evaluating a selected subobject of the current object; the filter is ignored if the *filter item type* is not *select* or if the specified attribute type is not present in the object, and upon evaluation of the *filter* the value of *filter item* will be set to that of the *filter*, (e) *initial substring*, if
25 present, this is the substring to compare against the initial portion of the value of

the specified attribute type, (f) *substring*, if present, this is the substring(s) to compare against all substrings of the value of the specified attribute type, (g) *final substring*, if present, this is the substring to compare against the final portion of the value of the specified attribute type, and (h) *license request*, if present, this is license request against which the appropriate license data objects should be evaluated; this attribute may only be specified if the value of the filter item type is either Request Candidates or Simulate Request.

An instance of enumeration syntax *Filter Type* identifies the type of a filter. Its value is chosen from one of the following: (a) *And* means the filter is the logical conjunction of its components; the filter is true unless any of the nested filters or filter items is false, or if there are no nested components, the filter is true; (b) *Or* means the filter is the logical disjunction of its components; the filter is false unless any of the nested filters or filter items is true, or, if there are no nested components, the filter is false; (c) *Not* means the result of the filter is reversed; there must be exactly one nested filter or filter item, and the filter is true if the enclosed filter or filter item is false, and is false if the enclosed filter or filter item is true.

An instance of enumeration syntax *Filter Item Type* identifies the type of a filter item. Its value is chosen from one of the following: (a) *Equality* which means the filter item is true if the object contains at least one attribute of the specified type whose value is equal to that specified by Match Value (according to the equality matching rule in force), and false otherwise; (b) *Inequality* which means the filter item is true if the object contains at least one attribute of the specified type whose value is not equal to that specified by Match Value (according to the equality matching rule in force), and false otherwise; (c) *Greater*

5 *or Equal* which means the filter item is true if the object contains at least one attribute of the specified type whose value is equal to or greater than the value specified by Match Value (according to the matching rule in force), and false otherwise; (d) *Less or Equal* which means the filter item is true if the object contains at least one attribute of the specified type whose value is equal or less than the value specified by Match Value (according to the matching rule in force), and false otherwise; (e) *Greater* which means the filter item is true if the object contains at least one attribute of the specified type whose value is greater than the value specified by Match Value (according to the matching rule in force), and false otherwise; (f) *Less* which means the filter is true if the object contains at least one attribute of the specified type, whose value is less than the value specified by Match Value (according to the matching rule in force), and false otherwise; (g) *Present* which means the filter item is true if the object contains at least one attribute of the specified type, and false otherwise; (h) *Select* which means the filter item is true if the object contains at least one attribute of the specified type which has an object syntax and when the Filter is evaluated against the attributes of that object the Filter is true, and false otherwise; (i) *Request Candidates* which means the filter item is true if the object against which it is evaluated is one which could be used to provide some or all of the units requested by the specified License Request; the evaluation is made independently of any outstanding allocations or preallocations; and (j) *Simulate Request* which means the filter item is true if the object against which it is evaluated is one which would be used to provide some or all of the units requested by the specified License Request.

25 The Request Candidates and Simulate Request filter item types are of special use in testing and prototyping of systems by a license manager at a

licensee's site. For example, the license manager can simulate the effect of potential assignments, the effect of a population of certain types on a network, etc.

As an example, Figure 46 shows how a filter may be constructed to identify "All Product Use Authorizations issued by Digital for the Product 'Amazing Graphics System' which contains a calling authorization for Digital's 'Amazing Database' Product". This example is in the international standard format referred to as X.409 as mentioned above.

Filters can also used in a request allocation, being specified in a request extension as explained above. That is, a filter is one of the optional items in a request extension. For example, if a user wanted to use a version of WordPerfect with French language extension, and there were version with and without on the network, his request allocation would have a request extension that specified a filter for "French" in the token field. In this manner, a product can describe itself more richly. The filter in the request extension can be a Required filter or a Preferred filter, meaning the feature such as "French" is either absolutely necessary, or merely the preferred.

While this invention has been described with reference to specific embodiments, this description is not meant to be construed in a limiting sense. Various modifications of the disclosed embodiments, as well as other embodiments of the invention, will be apparent to persons skilled in the art upon reference to this description. It is therefore contemplated that the appended claims will cover any such modifications or embodiments as fall within the true scope of the invention.

- 86 -

WHAT IS CLAIMED IS:

1 1. A method of managing use of licensed software items, said
2 software items separately executable on a computer system or
3 accessible by said computer system, the computer system including
4 a processor and one or more nodes, comprising the steps of:

5 maintaining by said processor a store of license
6 authorizations for said software items; each license authorization
7 including an indication of license management policy for a software
8 item, said indication having a plurality of sets of policy
9 components, said sets of policy components granting alternatives of
10 specified restrictive rights to selectively access and execute said
11 software items in said system; said indication of license
12 management policy being in the format of an encoded document of a
13 data type consisting of an ordered sequence of elements;

14 accessing said store by said processor to modify in said store
15 one or more of said specified restrictive rights of said policy
16 components of an identified license authorization;

17 accessing said store by said processor using a filter to
18 obtain information from said license authorization for a selected
19 software item, in response to a request from a node, and

20 comparing an identification of said node and said software
21 item with said information, to produce and send to said node a
22 grant or refusal of said request.

1 2. A method according to claim 1 including the step of
2 receiving said license authorizations , for storing in said store,

1 from a license grantor external to said processor, and wherein said
2 step of accessing said store to modify in said store one or more of
3 said specified restrictive rights employs management functions
4 executable on said processor but not on said nodes or said license
5 grantor to identify a license authorization in said store.

1 3. A method according to claim 1 wherein said indication is
2 in the format of an encoded document of a data type consisting of
3 an ordered sequence of three elements, the three elements including
4 a document descriptor, a document header and the document content.

1 4. A method according to claim 1 wherein said filter
2 specifies one or more of said attributes and a Boolean operator for
3
4 each selected attribute.

1 5. A method according to claim 2 wherein said step of
2 accessing said store to modify one or more of said policy
3 components is to allow grant of rights to use which are more
4 restrictive than said specified restrictive rights.

1 6. A method according to claim 2 including the steps of:

2

3

4 sending a request by a user of one of said software items to
5 obtain permission to use said software item; said request
identifying the user and said software item;

1 accessing said store to obtain information from said license
2 authorization for said software item, in response to said request,
3 and comparing said identification of said user and said software
4 item with said information, to produce a grant or refusal of said
5 request for sending to said user.

1 7. A method according to claim 6 wherein said store is
2 maintained by a license server, and said request is sent to said
3 server and wherein said request is in the form of a remote
4 procedure call, and said grant or refusal sent to said user is a
5 return of said procedure call.

1 8. A method according to claim 7 wherein said license
2 authorization is a data arrangement specified as a product use
3 authorization, and said product use authorization is received by
4 said server from an issuer, and wherein said server and said users
5 are nodes on a computer network.

1 9. A method according to claim 2 wherein said policy
2 components include a termination date, and said management
3 functions can modify said termination date to an earlier
4 termination date and wherein said policy components include a right
5 of delegation of a right to grant said requests to another server,
6 and said management functions can modify said right of delegation
7 to remove said right of delegation.

1 10. A method according to claim 2 including storing in
2 association with said license authorization a number of management
3 attributes, and said management functions being able to modify said
4 management attributes.

1 11. A method according to claim 10 wherein said management
2 attributes include a reservation of units of license use granted by
3 said license authorization so that said units will not be granted
4 to a user in response to said request, and wherein said management
5 attributes include an allocation of units of license use to a
6 specific context.

1 12. A method according to claim 10 wherein said management
2 attributes include an allocation period which is the minimum
3 duration of an allocation of units, and wherein said management
4 attributes include permission to enable a backup delegation of the
5 right to grant said requests.

1 13. A system for managing use of licensed software products,
2 comprising: means for maintaining a store of license documents, one
3 for each said product; each license document including an
4 indication of license policy having plurality of sets of policy
5 components granting specified restrictive rights to use said
6 software products, said policy components in each set providing
7 alternatives;

8 a management interface for accessing said store to modify

1 selected ones of said components of an identified license
2 authorization.

1 14. A system according to claim 13 including:

2 means for sending a request from a user of one of said
3 products to obtain permission to use said product; said request
4 identifying the user and said product;

5 means for accessing said store to obtain information from said
6 license document for said product, in response to said request, and
7 for comparing said identification of said user and said product
8 with said information, and with constraints imposed by said policy
9 components, to produce a grant or refusal of said request and send
10 said grant or refusal to said user.

1 15. A system according to claim 13 wherein said management
2 interface can modify said selected ones of said components to allow
3 grant of rights to use which are more restrictive than said
4 specified restrictive rights and wherein said means for
5 maintaining, and said means for accessing and sending to said user
6 are all located at a server on a distributed network, and said
7 means for sending a request is located at a user node on said
8 network.

1 16. A system according to claim 14 wherein said request is in
2 the form of a remote procedure call, and said grant or refusal sent
3 to said user is a return of said procedure call, and wherein said

1 license document is a data arrangement specified as a product use
2 authorization, and said product use authorization is received by
3 said server from a license issuer.

1 17. A system according to claim 13 wherein said policy
2 components include a termination date, and said management
3 functions can modify said termination date to an earlier
4 termination date, and wherein said policy components include a
5 right of delegation of a right to grant said requests to another
6 server, and said management functions can modify said right of
7 delegation to remove said right of delegation.

1 18. A system according to claim 15 including means for storing
2 in association with said license authorization a number of
3 management attributes, wherein said management functions are able
4 to modify said management attributes and wherein said management
5 attributes include a reservation of units of license use granted by
6 said license authorization so that said units will not be granted
7 to a user in response to said request.

1 19. A system according to claim 18 wherein said management
2 attributes include an allocation of units of license use to a
3 specific context.

1 20. A system according to claim 18 wherein said management
2 attributes include an allocation period which is the minimum

1 duration of an allocation of units, and include permission to
2 enable a backup delegation of the right to grant said requests.

1 21. A method according to claim 3 wherein said document
2 descriptor includes an encoding method version number, and encoder-
3 identifier and an encoder-name, and wherein said document-header
4 includes a title, an author, a version and a date for the software
5 item.

1 22. A method according to claim 3 wherein said document
2 content includes at least one of the following:

- 3 a product-use-authorization;
- 4 a license-use-requirements-table;
- 5 a group-definition;
- 6 a key-registration;
- 7 a delegation.

1 23. A method according to claim 3 wherein said document-
2 content includes a license-data-header, and said license-data-
3 header describes the parties to the license document, the term of
4 the agreement and constraints that may have been placed on
5 management of the license data.

1 24. A method according to claim 3 wherein said document-
2 content includes management-info, where the management-info may
3 include at least one of the following:

1 an assignment;
2 a reservation;
3 a delegation;
4 a backup delegation;
5 an allocation;
6 a registration date;
7 a registrar;
8 a comment;
9 a termination-date.

1 25. A method according to claim 3 wherein:

2 said document descriptor includes an encoding method
3 version and a date for the software item;

4 said document content may include at least one of the
5 following: a product-use-authorization, a license-use-requirements-
6 table, a group-defination, a key-registration, and a delegation;

7 said document-content selectively includes a license-
8 data-header, and said license-data-header describes the parties to
9 the license document, the term of the agreement and constraints
10 that may have been placed on management of the license data;

11 said document-content may have been placed on management
12 of the license data;

13 said document-content selectively includes management-
14 info, where the management-info may include at least one of the
15 following: an assignment, a reservation, a delegation, a backup
16 delegation, an allocation, a registration date, a registrar, and a

1 comment.

1 26. A method according to claim 3 wherein said store is
2 maintained by a license server, and said request is sent to said
3 server, and wherein said server and said users are nodes on a
4 computer network.

1 27. A method according to claim 3 wherein said request is in
2 the form of a remote procedure call, and said grant or refusal sent
3 to said user is a return of said procedure call, and wherein said
4 license authorization is received by said server from an issuer.

1 28. A method according to claim 3 including the steps of:
2 sending a request by a user of one of said software items to obtain
3 permission to use said software item; said request identifying the
4 user and said software item;
5 sending said grant or refusal to said user.

1 29. Apparatus for managing use of licensed software items,
2 comprising:
3 means for maintaining a store of license authorizations
4 for said software items; each license authorization including an
5 indication of license management policy for a software item, said
6 indication being in the format of an encoded document of a data
7 type consisting of an ordered sequence of three elements, the three
8 elements including a document descriptor, a document header and the

1 document content;

2 means for sending a request by a user of one of said
3 software items to obtain permission to use said software item; said
4 request identifying the user and said software item;

5 means for accessing said store to obtain information from
6 said license authorization for said software item, in response to
7 said request, and comparing said identification of said user and
8 said software item with said information, to produce a grant or
9 refusal of said request;

10 means for sending said grant or refusal to said user.

1 30. Apparatus according to claim 29 wherein said document
2 descriptor includes an encoding method version number, and an
3 encoder-identifier and an encoder-name, and wherein said document-
4 header includes a title, an author, a version and a date for the
5 software item.

1 31. Apparatus according to claim 29 wherein said document
2 content includes at least one of the following:

3
4 a product-use-authorization;
5 a license-use-requirements-table;
6 a group-definition;
7 a key-registration;
8 a delegation.
9

1 32. Apparatus according to claim 29 wherein said document-
2 content includes a license-data-header, and said license-data-
3 header describes the parties to the license document, the term of
4 the agreement and constraints that may have been placed on
5 management of the license data.

1 33. Apparatus according to claim 29 wherein said document-
2 content includes management-info, where the management-info may
3 include at least one of the following:

4 an assignment;
5 a reservation;
6 a delegation;
7 a backup delegation;
8 an allocation;
9 a registration date;
10 a registrar;
11 a comment;
12 a termination-date.

1 34. Apparatus according to claim 29 wherein:
2 said document descriptor includes an encoding method
3 version number, and encoder-identifier and an encoder-name;
4 said document-header includes a title, an author, a
5 version and a date for the software item;
 said document content may include at least one of the
following: a product-use-authorization, a license-use-requirements-

table, a group-definition, a key-registration, and a delegation;

said document-content may include a license-data-header, and said license-data-header describes the parties to the license document, the term of the agreement and constraints that may have been placed on management of the license data;

said document-content may include management-info, where the management-info may include at least one of the following: an assignment, a reservation, a delegation, a backup delegation, an allocation, a registration date, a registrar, and a comment.

1

2

35. Apparatus according to claim 29 wherein said store is maintained by a license server, and said request is sent to said server, and wherein said request is in the form of a remote procedure call, and said grant or refusal sent to said user is a return of said procedure call.

3

4

5

6

1

36. Apparatus according to claim 29 wherein said license authorization is received by said server from an issuer, and wherein said server and said users are nodes on a computer network.

2

3

1

37. A method of storing license documents by a server for a license management system, comprising the steps of:

2

3

maintaining a store of license documents for software items; each license document including an indication of license management policy for a software item, said indication being in the format of an encoded document of a data type consisting of an ordered

4

5

6

1 sequence of three elements, the three elements including a document
2 descriptor, a document header and the document content;

3 accessing said store to obtain information from a selected one
4 of said license documents for a software item, in response to a
5 request, and referencing said indication of license management
6 policy, to produce a grant or refusal of said request.

1 38. A method according to claim 37 wherein said document
2 descriptor includes an encoding method version number, an encoder-
3 identifier and an encoder-name, and wherein said document-header
4 includes a title, an author, a version and a date for the software
5 item.

1 39. A method according to claim 37 wherein said document
2 content includes at least one of the following:

- 3 a product-use-authorization;
4 a license-use-requirements-table;
5 a group-definition;
6 a key-registration;
7 a delegation.

1 40. A method according to claim 4 wherein said step of
2 selecting by a filter may select on one or more of the attributes:
3 issuer, producer, product name, product use authorization, calling
4 authorization, and wherein said store is maintained by a license
5 server, and said request is sent to said server.

1 41. A method according to claim 4 wherein said request is in
2 the form of a remote procedure call, and said grant or refusal sent

1 to said user is a return of said procedure call.

1 42. A method according to claim 40 wherein said license
2 authorization is a data arrangement specified as a product use
3 authorization, and said product use authorization is received by
4 said server from an issuer, and wherein said server and said users
5 are nodes on a computer network.

1 43. Apparatus for managing use of licensed software items,
2 comprising:

3 means for maintaining a store of license authorizations for
4 said software items; each license authorization including an
5 indication of license management policy for a software item, said
6 indication being an encoded document containing a number of
7 attributes defining said license policy;

8 filter means for selecting from said store, said filter means
9 specifying one or more of said attributes and a Boolean operator
10 for each selected attribute;

11 means for sending a request by a user of one of said software
12 items to obtain permission to use said software item; said request
13 identifying the user and said software item;

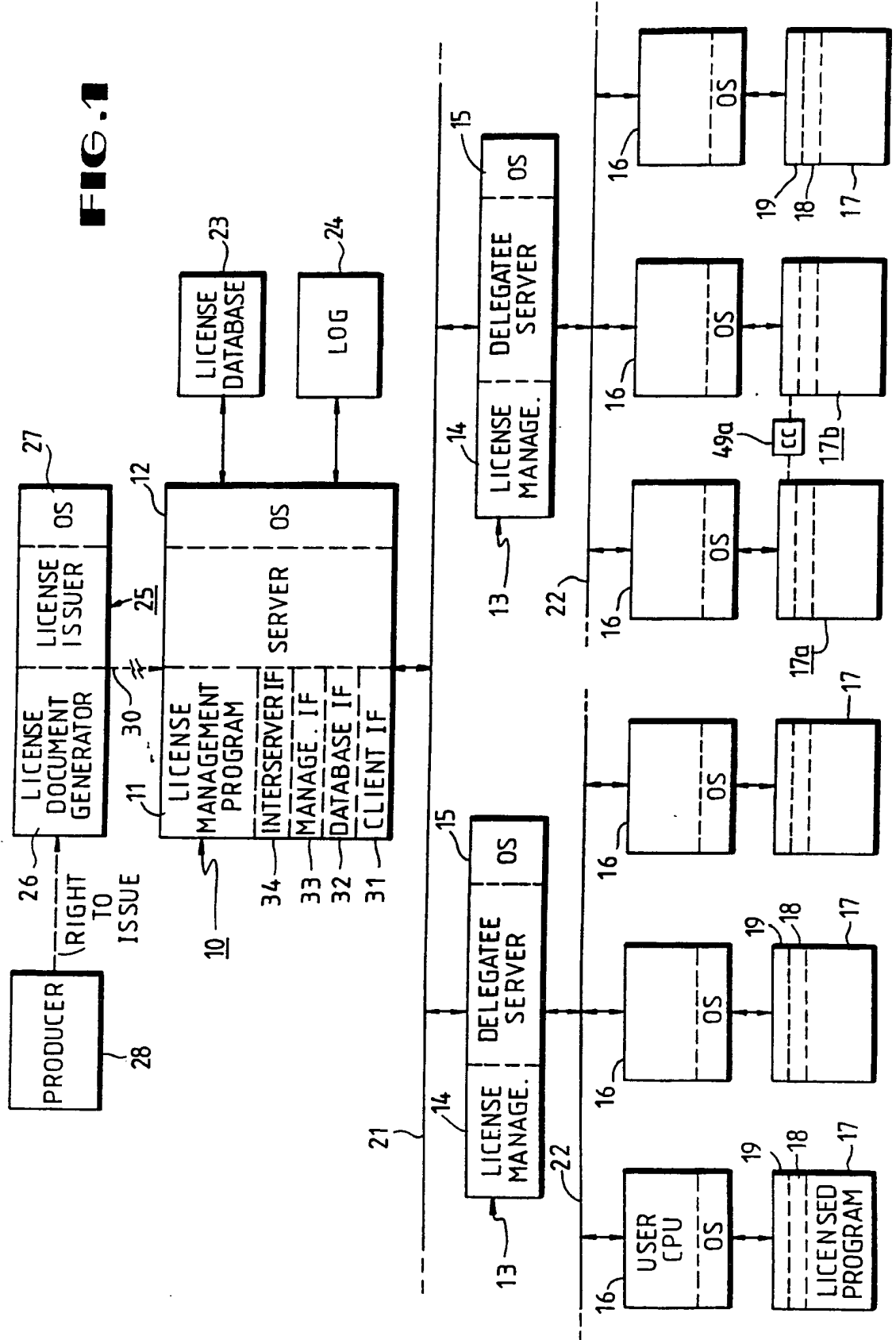
14 means for accessing said store to obtain information from said
15 license authorization for said software item, in response to said
16 request, and comparing said identification of said user and said
17 software item with said information, to produce a grant or refusal
18 of said request; and

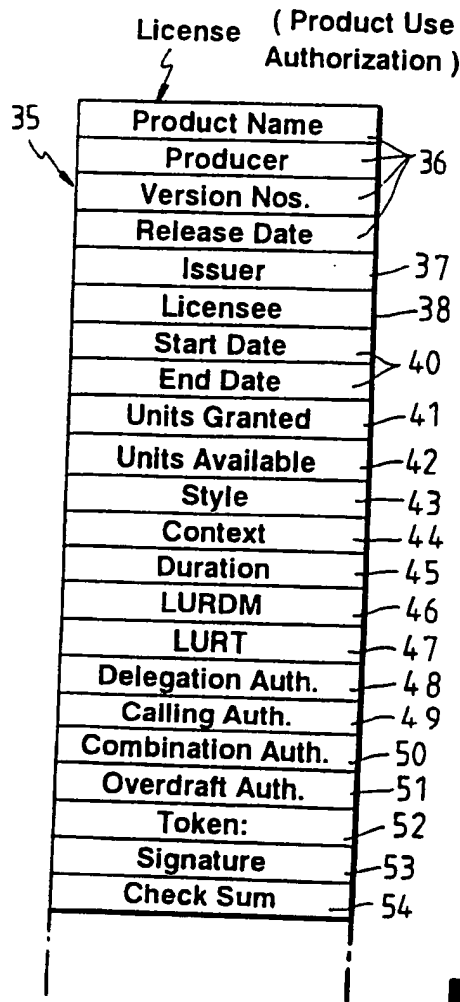
1 means for sending said grant or refusal to said user.

1 44. Apparatus according to claim 43 wherein said filter means
2 may select on one or more of the attributes: issuer, producer,
3 product name, product use authorization, calling authorization, and
4 wherein said store is maintained by a license server, and said
5 request is sent to said server, and wherein said request is in the
6 form of a remote procedure call, and said grant or refusal sent to
7 said user is a return of said procedure call.

1 45. Apparatus according to claim 43 wherein said license
2 authorization is a data arrangement specified as a product use
3 authorization, and said product use authorization is received by
4 said server from an issuer, wherein said server and said users are
5 nodes on a computer network.

FIG. 1





License Unit Requirements Table			
Row Selector	Columns		
Platform ID	A	B	C
PC-0	10	230	-1
PC-1	12	230	-1
VAX 6210	158	300	150

FIG. 4

FIG. 2

43 Style	44 Context	45 Duration	46 LURDM
Allocative	Network	Transaction	Constant
Consumptive	Execution_Domain	Assignment	Table Lookup
Private	Login_Domain	Immediate	Private
	Node_ID		
	Process_Family		
	Process		
	User_Name		
	Product_Name		
	Operating_System		
	Platform_ID		
	Private		

FIG. 3

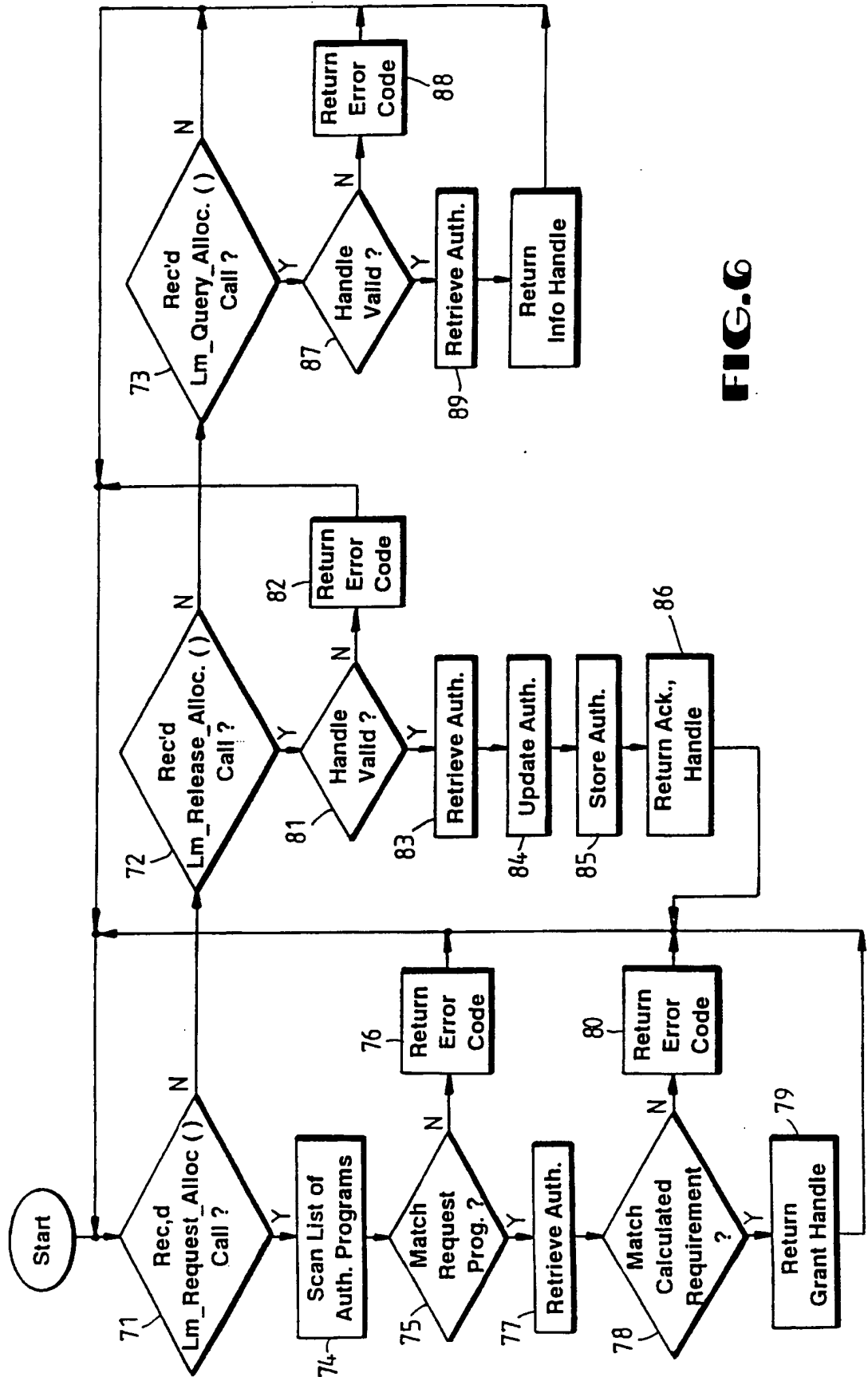


FIG. 6

SUBSTITUTE SHEET

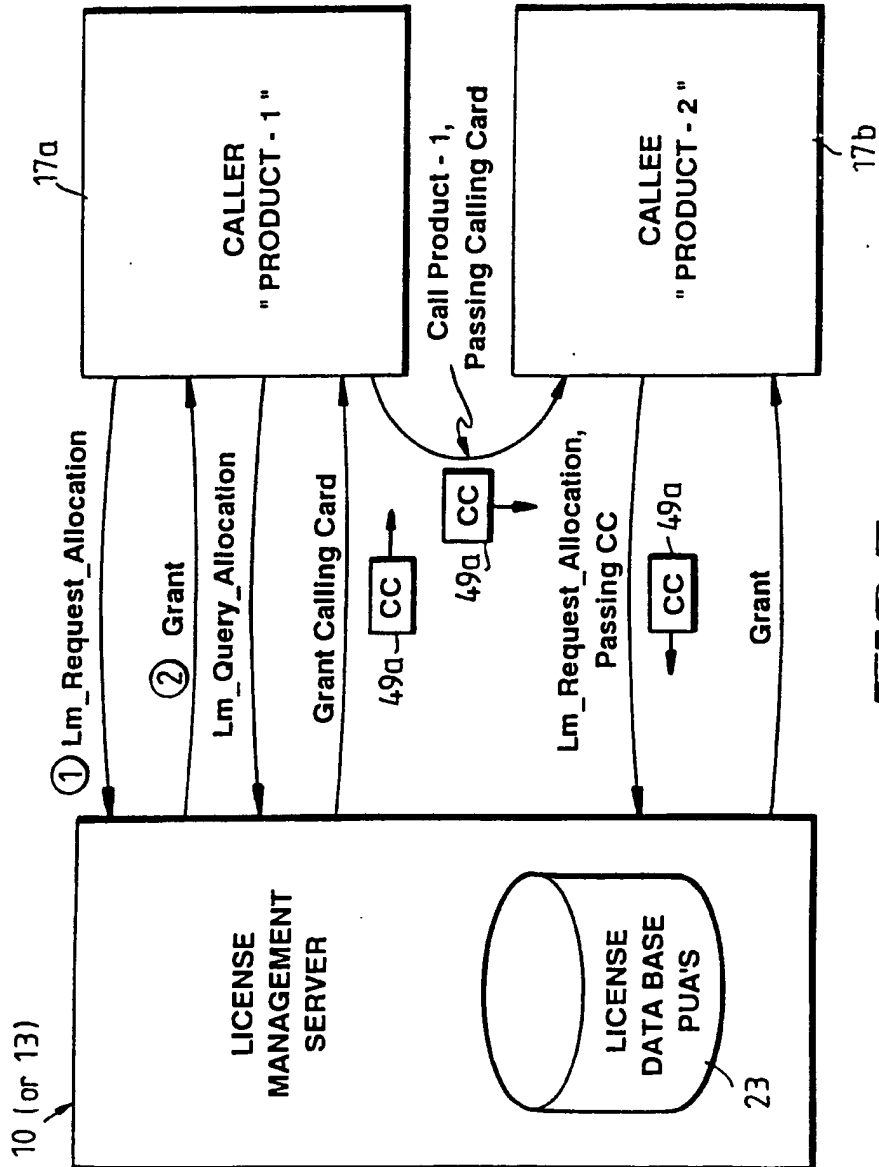


FIG.7

SUBSTITUTE SHEET

```

Object Identifier Value ::= {
    iso(1)
    identified-organization(3)
    icd-ecma(12)
    member-company(2)
    dec(1011)
    data-syntaxes(1)
    cda(3)
    ldif(17)
}

Object Identifier Encoding ::= {
    0x6, 0x8, 0x2B, 0xC, 0x2,
    0x87, 0x73, 0x1, 0x3, 0x11
}

```

FIG. 8 LDIF Object Identifier

```

LDIFDocument ::= [PRIVATE 16373] IMPLICIT SEQUENCE {
  document-descriptor [0] IMPLICIT DocumentDescriptor OPTIONAL,
  document-header [1] IMPLICIT DocumentHeader OPTIONAL,
  document-content [2] IMPLICIT DocumentContent
}

```

FIG. 9 LDIF Document Syntax Diagram

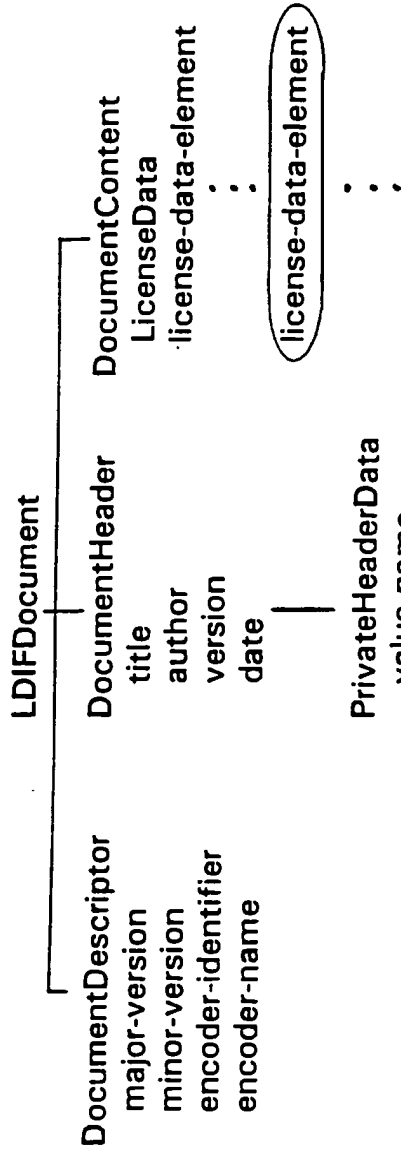


FIG. 10 LDIF Document Structure

```

DocumentDescriptor ::= SEQUENCE {
    major-version [0] IMPLICIT INTEGER OPTIONAL,
    minor-version [1] IMPLICIT INTEGER OPTIONAL,
    encoder-identifier [2] IMPLICIT Character-String OPTIONAL,
    encoder-name [3] IMPLICIT Character-String OPTIONAL
}
    
```

FIG. 11 Document Descriptor Syntax Diagram

```

Pakgen DocumentDescriptor ::= {
    major-version 1,
    minor-version 0,
    encoder-identifier "PAKGEN",
    encoder-name {Character-String "PAK Generator V1.0"}
}
    
```

FIG. 12 Document Descriptor Example

```

DocumentHeader ::= SEQUENCE {
  private-header-data
    title [0] IMPLICIT NamedValueList OPTIONAL,
    author [1] IMPLICIT Character-String OPTIONAL,
    version [2] IMPLICIT Character-String OPTIONAL,
    date [3] IMPLICIT Character-String OPTIONAL,
    [4] IMPLICIT UTCTime OPTIONAL
}

```

FIG. 13 Document Header Syntax Diagram

```

example-header-document-header ::= {
  title {Character-String "PAKGEN Licenses with Associated LURT data"}
  author {Character-String "Tom Jones, Foobar, Inc. License Department"}
  version {Character-String "VO.1"}
  date "198801021100-0500"
}

```

FIG. 14 Document Header Example


```

Document Content ::= SEQUENCE OF LicenseData

LicenseData ::= SEQUENCE {
  license-data-header [0] IMPLICIT LicenseDataHeader,
  license-body [1] CHOICE {
    product-use-authorization [0] IMPLICIT ProductUseAuthorization,
    license-units-requirements-table [1] IMPLICIT LURT,
    group-definition [2] IMPLICIT GroupDefinition,
    key-registration [3] IMPLICIT KeyRegistration,
    issuer-delegation [4] IMPLICIT IssuerDelegation,
    license-delegation [5] IMPLICIT LicenseDelegation,
    backup-delegation [6] IMPLICIT BackupDelegation
  },
  management-info [2] IMPLICIT ManagementInfo OPTIONAL
}

```

FIG. 15 Document Content Syntax Diagram

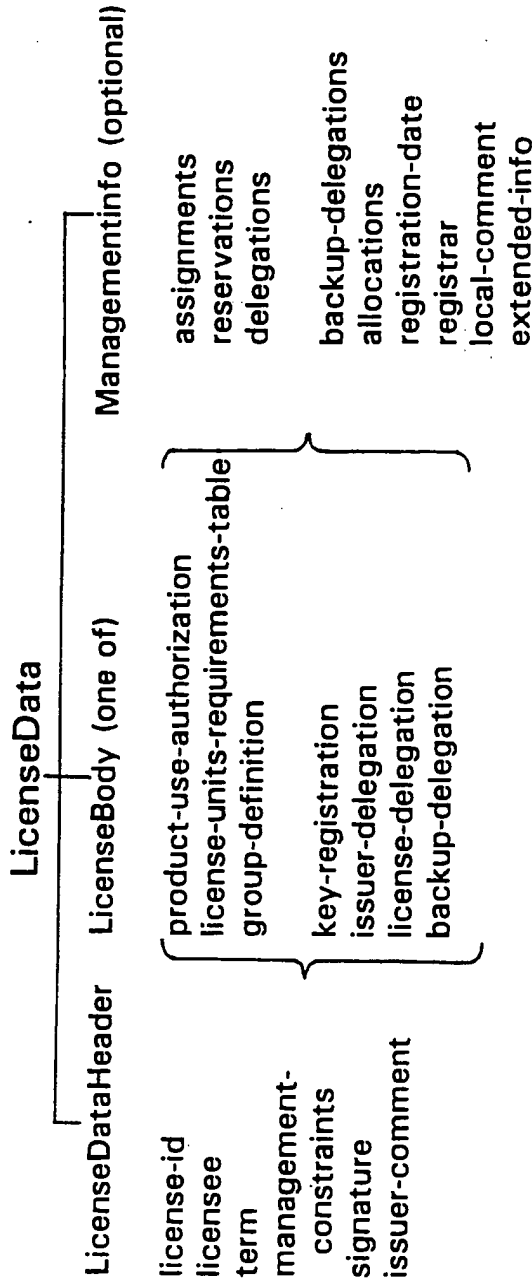


FIG. 16 License Data Structure

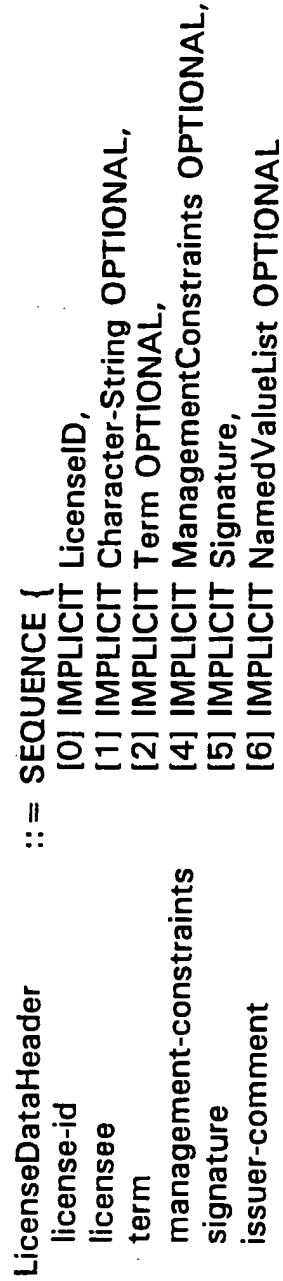


FIG. 17 License Data Header Syntax Diagram

```

ProductUseAuthorization ::= SEQUENCE {
  product-id          [0] IMPLICIT ProductID,
  units-granted       [1] IMPLICIT INTEGER,
  management-policy   [2] IMPLICIT ManagementPolicy,
  calling-authorizations [3] IMPLICIT SEQUENCE OF Member OPTIONAL,
  caller-authorizations [4] IMPLICIT SEQUENCE OF Member OPTIONAL,
  execution-constraints [5] IMPLICIT ExecutionConstraints OPTIONAL,
  product-token       [6] IMPLICIT NamedValueList OPTIONAL
}

```

FIG. 18 Product Use Authorization Syntax Diagram

```

LURT ::= SEQUENCE {
  lurt-name      [0] IMPLICIT Character-String,
  rows           [1] IMPLICIT RowList
}
RowList ::= SEQUENCE OF LurtRow

LurtRow ::= SEQUENCE {
  platform-id    [0] IMPLICIT Character-String,
  lurt-columns   [1] IMPLICIT SEQUENCE OF INTEGER
}

```

FIG. 19 License Unit Requirement Table Syntax Diagram

```

Example LURT ::= {
  lurt-name { Character-String "Example LURT" }
  rows {
    LurtRow {
      {Character-String "PC-0"}
      {{10} {230} {-1}}
    }
    LurtRow {
      {Character-String "PC-1"}
      {{12} {230} {-1}}
    }
    LurtRow {
      {Character-String "VAX 6210"}
      {{158} {300} {150}}
    }
  }
}

```

FIG. 20 Example Encoding of LURT

```

Group Definition
group-name
group-version
group-release-date
group-members
 ::= SEQUENCE {
    [0] IMPLICIT Character-String,
    [1] IMPLICIT Version,
    [2] IMPLICIT UTCTime,
    [3] IMPLICIT SEQUENCE OF Member
 }

```

FIG. 21 Group Definition Syntax Diagram

```

KeyRegistration
key-owner-name
key-algorithm
key-value
 ::= SEQUENCE {
    [0] IMPLICIT Character-String,
    [1] IMPLICIT Character-String,
    [2] IMPLICIT OCTET STRING
 }

```

FIG. 22 Key Registration Syntax Diagram

SUBSTITUTE SHEET

```

IssuerDelegation
  delegated-issuer-name
  delegated-product-id
  delegated-units-granted
  template-authorization
  sub-license-permitted
  ::= SEQUENCE {
    [0] IMPLICIT Character-String,
    [1] IMPLICIT SEQUENCE OF Member,
    [2] IMPLICIT INTEGER OPTIONAL,
    [3] IMPLICIT ProductUseAuthorization OPTIONAL,
    [4] IMPLICIT BOOLEAN DEFAULT FALSE
  }

```

FIG. 23 Issuer Delegation Syntax Diagram

```

LicenseDelegation
  delegated-units
  delegated-distribution-control
  delegatee-execution-constraints
  assignment-list
  delegated-data
  ::= SEQUENCE {
    [0] IMPLICIT INTEGER OPTIONAL
    [1] IMPLICIT DistributionControl,
    [2] IMPLICIT ExecutionConstraints OPTIONAL,
    [3] IMPLICIT AssignmentList OPTIONAL,
    [4] IMPLICIT LicenseData OPTIONAL
  }

```

FIG. 24 License Delegation & Backup Delegation Syntax Diagrams

```

ManagementInfo
  assignments
  reservations
  delegations
  backup-delegations
  allocations
  registration-date
  registrar
  local-comment
  termination-date
  extended-info
  ::= SEQUENCE {
    [0] IMPLICIT AssignmentList OPTIONAL,
    [1] IMPLICIT AssignmentList OPTIONAL,
    [2] IMPLICIT DelegationList OPTIONAL,
    [3] IMPLICIT DelegationList OPTIONAL,
    [4] IMPLICIT AllocationList OPTIONAL,
    [5] IMPLICIT UTCTime,
    [6] IMPLICIT Context,
    [7] IMPLICIT NamedValueList OPTIONAL,
    [8] IMPLICIT UTCTime OPTIONAL,
    [9] IMPLICIT NamedValueList OPTIONAL
  }

```

FIG. 25 ManagementInfo Syntax Diagram

SUBSTITUTE SHEET

```

AllocationList ::= SEQUENCE OF Allocation
Allocation ::= SEQUENCE {
  allocation-context [0] IMPLICIT Context,
  allocation-lur [1] IMPLICIT INTEGER,
  allocation-group-id [2] IMPLICIT INTEGER OPTIONAL
}

```

FIG. 26 Allocation Syntax Diagram

```

AssignmentList ::= SEQUENCE OF Assignment
Assignment ::= SEQUENCE {
  assigned-units [0] IMPLICIT INTEGER,
  assignment-term [1] IMPLICIT Term,
  assignee [2] IMPLICIT Context
}

```

FIG. 27 Assignment Syntax Diagram


```

ContextList ::= SEQUENCE OF Context
Context ::= SEQUENCE OF Subcontext
SubContext ::= SEQUENCE {
  sub-context-type [0] SubContextType,
  subcontext-value [1] ValueData
}
SubContextType ::= CHOICE {
  standard-subcontext-type [0] IMPLICIT INTEGER {
    network-subcontext(1),
    execution-domain-subcontext(2),
    login-domain-subcontext(3),
    node-subcontext(4),
    process-family-subcontext(5),
    process-id-subcontext(6),
    user-name-subcontext(7),
    product-name-subcontext(8),
    operating-system-subcontext(9),
    platform-id-subcontext(10)
  }
  private-subcontext [1] IMPLICIT INTEGER {first(0),last(255)}
}

```

FIG. 28 Context Syntax Diagram

SUBSTITUTE SHEET

FOOBAR V4.1 Allocated Units			
Units	Context Template		Full Context Specifications
	Node	User_Name	
10	BLUE	WYMAN	ENET, AA_Cluster, BLUE, PID-1..., WYMAN
10	RED	OLSEN	ENET, BB_Cluster, RED, PID-1..., OLSEN
10	RED	WYMAN	ENET, BB_Cluster, RED, PID-2..., WYMAN
10	GREEN	WYMAN	ENET, AA_Cluster, GREEN, PID-1..., WYMAN
	GREEN	WYMAN	ENET, AA_Cluster, GREEN, PID-2..., WYMAN

FIG. 29 Only unique contexts require explicit unit allocations.

FOOBAR V4.1 Allocated Units		
Units	Context Template	Full Context Specifications
	Node	
10	BLUE	ENET, AA_Cluster, BLUE, PID-1..., WYMAN
10	RED	ENET, BB_Cluster, RED, PID-1..., OLSEN
	RED	ENET, BB_Cluster, RED, PID-2..., WYMAN
10	GREEN	ENET, AA_Cluster, GREEN, PID-1..., WYMAN
	GREEN	ENET, AA_Cluster, GREEN, PID-2..., WYMAN

FIG. 30 Modification of Context_Template impacts units requirements.

```

DistributionControl ::= SEQUENCE {
  distribution-method [0] IMPLICIT INTEGER {
    refresh-distribution(1),
    initial-distribution-only(2),
    manual-distribution(3)
  },
  current-start-date [1] IMPLICIT UTCTime OPTIONAL,
  current-end-date [2] IMPLICIT UTCTime OPTIONAL,
  refresh-interval [3] IMPLICIT IntervalTime OPTIONAL,
  retry-interval [4] IMPLICIT IntervalTime OPTIONAL,
  maximum-retry-count [5] IMPLICIT INTEGER OPTIONAL,
  retries-attempted [6] IMPLICIT INTEGER OPTIONAL
}

```

FIG. 31 Distribution Control Syntax Diagram

```

ExecutionConstraints ::= SEQUENCE {
  operating-system      [0] IMPLICIT SEQUENCE OF Character-String OPTIONAL,
  execution-context    [1] IMPLICIT ContextList OPTIONAL,
  environment-list     [2] IMPLICIT SEQUENCE OF EnvironmentKind OPTIONAL
}
EnvironmentKind ::= INTEGER {
  batch(1),
  interactive(2),
  local(3),
  network(4),
  remote(5)
}

```

FIG. 32 Execution Constraints Syntax Diagram

```
LicenseID      ::= SEQUENCE {  
    issuer      [0] IMPLICIT Character-String,  
    serial-number [1] IMPLICIT Character-String,  
    amendment   [2] IMPLICIT INTEGER DEFAULT 0  
}
```

FIG. 33 License ID Syntax Diagram

```

LURDM ::= SEQUENCE {
  combination-permitted [0] IMPLICIT BOOLEAN DEFAULT TRUE,
  overdraft-limit [1] IMPLICIT INTEGER DEFAULT 0,
  overdraft-logging-required [2] IMPLICIT BOOLEAN DEFAULT FALSE,
  allocation-size [3] IMPLICIT INTEGER OPTIONAL,
  lurdm-kind [4] IMPLICIT INTEGER {
    lurdm(1),
    constant(2),
    private-lurdm(3)
  },
  named-lurdm-id [5] IMPLICIT Character-String OPTIONAL,
  lurdm-value [6] IMPLICIT INTEGER OPTIONAL,
  default-unit-requirement [7] IMPLICIT INTEGER OPTIONAL
}

```

FIG. 34 License Unit Requirements Determination Method Syntax Diagram

```

ManagementConstraints ::= SEQUENCE {
  management-context          [0] IMPLICIT ContextList OPTIONAL,
  management-scope           [1] IMPLICIT INTEGER {
    single-platform(1),
    management-domain(2),
    entire-network(3)
  } OPTIONAL,
  backup-permitted           [2] IMPLICIT BOOLEAN DEFAULT TRUE,
  delegation-permitted       [3] IMPLICIT BOOLEAN DEFAULT TRUE,
  maximum-delegation-period [4] IMPLICIT IntervalTime OPTIONAL
}

```

FIG. 35 Management Constraints Syntax Diagram

SUBSTITUTE SHEET


```

ManagementPolicy ::= SEQUENCE {
  style
    [0] IMPLICIT INTEGER {
      allocative(1),
      consumptive(2),
      private-style(3)
    },
  context-template
    [1] IMPLICIT SEQUENCE OF SubcontextType
    OPTIONAL,
  duration
    [2] IMPLICIT INTEGER {
      transaction(1),
      assignment(2),
      immediate(3)
    } OPTIONAL,
  lur-determination-method
  allocation-sharing-limit
  reassignment-constraint
}

```

FIG. 36 Management Policy Syntax Diagram

```

Member
member-product
member-signature
member-token

 ::= SEQUENCE {
    [0] IMPLICIT ProductID,
    [1] IMPLICIT Signature,
    [2] IMPLICIT NamedValueList OPTIONAL
}

```

FIG. 37 Member Syntax Diagram

```

NamedValue
value-name
value-data

 ::= SEQUENCE {
    Character-String,
    ValueData
}

ValueData
value-boolean
value-integer
value-text
value-general
value-list

 ::= CHOICE {
    [0] IMPLICIT BOOLEAN,
    [1] IMPLICIT INTEGER,
    [2] IMPLICIT SEQUENCE OF Character-String
    [3] IMPLICIT OCTET STRING,
    [4] IMPLICIT SEQUENCE OF ValueData
}

```

```

NamedValueList
 ::= SEQUENCE OF NamedValue

```

FIG. 38 Named Value, Value Data & Named Value List Syntax Diagrams

```

ExampleList NamedValueList ::= {
  NamedValue {
    value-name {Character-String "Purchase Order"}
    value-data {INTEGER 154493}
  }
  NamedValue {
    value-name {Character-String "Telephone Support #"}
    value-data {Character-String { + 1 (999) 555-1234}
  }
}

```

FIG. 39 Named Value List Example

```

ProductID
  producer
  product-name
  first-version
  last-version
  first-release-date
  last-release-date
} ::= SEQUENCE {
  [0] IMPLICIT Character-String,
  [1] IMPLICIT Character-String,
  [2] IMPLICIT Version OPTIONAL,
  [3] IMPLICIT Version OPTIONAL,
  [4] IMPLICIT UTCTime OPTIONAL,
  [5] IMPLICIT UTCTime OPTIONAL
}

```

FIG. 40 Product ID Syntax Diagram

```

Signature
signature-algorithm
signature-parameters
signature-value
 ::= SEQUENCE {
   [0] IMPLICIT Character-String,
   [1] IMPLICIT NamedValueList OPTIONAL,
   [2] IMPLICIT OCTET STRING
 }

```

FIG. 41 Signature Syntax Diagram

```

Term
start-date
end-date
 ::= SEQUENCE {
   [0] IMPLICIT UTCTime OPTIONAL,
   [1] IMPLICIT UTCTime OPTIONAL,
 }

```

FIG. 42 Term Syntax Diagram

```

Version
  part-1
  part-2
  part-3
  part-4
  ::= SEQUENCE {
    [0] IMPLICIT INTEGER,
    [1] IMPLICIT INTEGER DEFAULT 0,
    [2] IMPLICIT INTEGER DEFAULT 0,
    [3] IMPLICIT INTEGER DEFAULT 0
  }

```

FIG. 43

Attributes Specific to Filter				
Attribute	Value Syntax	Value Length	Value Number	Value Initially
Filter Items	Object(Filter Item)	-	0 or more	-
Filters	Object(Filter)	-	0 or more	-
Filter Type	Enum(Filter Type)	-	1	-

FIG. 44

SUBSTITUTE SHEET

Attributes Specific to Filter					
Attribute	Value Syntax	Value Length	Value Number	Value Initially	
Filter Item Type	Enum(Filter Item Type)	-	1	-	-
Attribute Type	Type	-	1	-	-
Match Value	any	-	0-1	-	-
Filters	Object(Filter)	-	0-1	-	-
Initial Substring	String(*)	1 or more	0-1	-	-
Substring	String(*)	1 or more	0 or more	-	-
Final Substring	String(*)	1 or more	0-1 or more	-	-
License Request	Object(License Request)	-	0-1	-	-

FIG. 45

```

Filter {
  Filter-Type AND
  Filter-Item {
    Filter-Item-Type SELECT
    Attribute-Type Product-Use-Authorization
    Filter {
      Filter-Type AND
      Filter-Item{
        Filter-Item-Type SELECT
        Attribute-Type Calling-Authorization
        Filter{
          Filter-Type AND
          Filter-Item {
            Filter-Item-Type EQUALITY
            Attribute-Type Producer
            Match-Value "Digital"
          }
          Filter-Item {
            Filter-Item-Type EQUALITY
            Attribute-Type Producer
            Match-Value "Amazing Database"
          }
        }
      }
    }
  }
  Filter-Item {
    Filter-Item-Type EQUALITY
    Attribute-Type Producer
    Match-Value "Digital"
  }
  Filter-Item{
    Filter-Item-Type EQUALITY
    Attribute-Type Issuer
    Match-Value "Digital"
  }
  Filter-Item {
    Filter-Item-Type EQUALITY
    Attribute-Type Product-Name
    Match-Value "Amazing Graphics System"
  }
}

```

FIG. 46 Example Filter Value Notation

III. DOCUMENTS CONSIDERED TO BE RELEVANT (CONTINUED FROM THE SECOND SHEET)		
Category ^a	Citation of Document, with indication, where appropriate, of the relevant passages	Relevant to Claim No.
Y	IBM TECHNICAL DISCLOSURE BULLETIN. vol. 31, no. 8, 1 January 1989, NEW YORK US pages 195 - 198; 'METHOD FOR MANAGING CLIENT/SERVER RELATIONSHIP IN THE AIX OPERATING SYSTEM'	1-3, 6-19, 22, 24, 26-29, 31-33, 35-37, 39 43-45
Y A	see the whole document ---	21

**ANNEX TO THE INTERNATIONAL SEARCH REPORT
ON INTERNATIONAL PATENT APPLICATION NO. US 9203812
SA 60557**

This annex lists the patent family members relating to the patent documents cited in the above-mentioned international search report. The members are as contained in the European Patent Office EDP file on The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information. 09/09/92

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP-A-0332304	13-09-89	US-A- 4937863 JP-A- 2014321	26-06-90 18-01-90

EPO FORM P0479

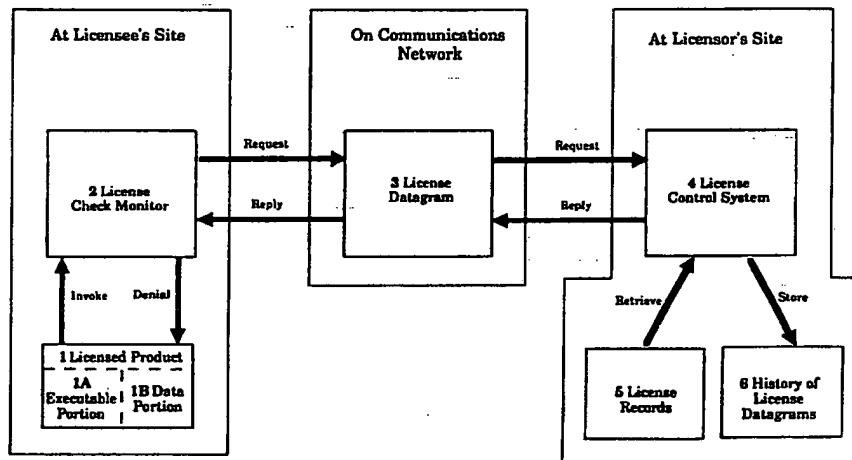
For more details about this annex : see Official Journal of the European Patent Office, No. 12/82



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

<p>(51) International Patent Classification ⁵ : G06F 11/34, H04L 9/00</p>	<p>A1</p>	<p>(11) International Publication Number: WO 93/01550 (43) International Publication Date: 21 January 1993 (21.01.93)</p>
<p>(21) International Application Number: PCT/US92/05387 (22) International Filing Date: 30 June 1992 (30.06.92) (30) Priority data: 724,180 1 July 1991 (01.07.91) US 907,934 29 June 1992 (29.06.92) US (71) Applicant: INFOLOGIC SOFTWARE, INC. [US/US]; 1223 Peoples Avenue, Suite 5405, Troy, NY 12180 (US). (72) Inventor: GRISWOLD, Gary, N. ; 1937 Regent Street, Schenectady, NY 12309 (US). (74) Agents: LAZAR, Dale, S. et al.; Cushman, Darby & Cush- man, Ninth Floor, 1100 New York Avenue, N.W., Wash- ington, DC 20005-3918 (US).</p>	<p>(81) Designated States: AT, AU, BB, BG, BR, CA, CH, CS, DE, DK, ES, FI, GB, HU, JP, KP, KR, LK, LU, MG, MN, MW, NL, NO, PL, RO, RU, SD, SE, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IT, LU, MC, NL, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, SN, TD, TG). Published With international search report.</p>	

(54) Title: LICENSE MANAGEMENT SYSTEM AND METHOD



(57) Abstract

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

FOR THE PURPOSES OF INFORMATION ONLY

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

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

- 1 -

LICENSE MANAGEMENT SYSTEM AND METHOD

BACKGROUNDField of the Invention

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

Description of Related Art

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

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

SUBSTITUTE SHEET

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

SUMMARY OF THE INVENTION

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

BRIEF DESCRIPTION OF THE DRAWINGS

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

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

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

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

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

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

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

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

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

DETAILED DESCRIPTION OF THE
PRESENTLY PREFERRED EXEMPLARY EMBODIMENT

15 As shown in FIGURE 1, a licensed product 1 is located at a licensee's site. Product 1 may include a data portion 1B and a functional portion 1A such as computer software product or any other kind of information product used to control use of data
20 portion 1B. If data portion 1B is CD-ROM database information, functional portion 1A should enable the licensee to search indexes and display text. If data portion 1B is video information, functional portion 1A should control the display of the video information.
25 For audio information, functional portion 1A should play the audio information. If data portion 1B is an electronic book, functional portion 1A should display and turn pages. The above examples show some of the ways functional portion 1A can control data portion
30 1B; however, they are hardly exhaustive.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Number of processes running 26 is the number of concurrent uses of product 1 at the time datagram 3 is sent. Authorization code 27 is used on reply datagrams 3 to indicate an approval or a denial. 5 Message text 28 contains a message which will be displayed to the user upon a denial.

License database 29 at the licensor's site holds records of information about customers, licenses, and license usage. The types of information within 10 license database 29 of the present embodiment are shown in FIGURE 2. However, a specific license management system may require its license database to hold types of information other than those in FIGURE 2. For example, licensee name and address may be 15 incorporated as a part of a license database 29.

License record 5 contains information on licenses. Licensee network address 30 identifies a precise network node which is licensed to use product 1. If request datagrams are received which do not 20 originate from known licensee network addresses 30, reply datagrams containing denial messages are transmitted. Product model number 31 is the model number of a licensed product. Termination date 32 is the expiration date of a license. When the license of 25 a product is issued for an unlimited duration, termination date 32 should reflect a date very far into the future, relative to the licensing date.

The present embodiment allows licenses to be paid for in a lease-like or rental fashion. If a licensee 30 were to rent or lease product 1, paid through date 33 would reflect the date through which the licensee has paid for using the product. Grace period 34 is the time interval for which the licensee is allowed to be

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

device such as a CRT screen. Step 105.3.3 terminates the current use of product 1. This step may be implemented by subroutine or function call to a simple exit that saves any current user data to a file.

5 Alternatively, product 1 may be designed so that, upon being directed to terminate further execution, it first gives the user an opportunity to save their data.

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

10

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

15

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

20

25

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

30

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

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

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

If the number of processes running 26 exceeds a licensed number of concurrent uses of product 1 (at a particular node), then step 108.6 sets authorization code 27 to 4 (i.e. concurrent process usage limit is exceeded).
20

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

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

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

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

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

Step 109.1 encrypts authorization code 27 and writes the encrypted code into datagram 3. Next, step 10 109.2 writes message text 28 corresponding to authorization code 27 into datagram 3.

Step 109.2 may be replaced with the following method for relaying proper messages to a product user. Proper messages corresponding to each authorization 15 code is stored in monitor 2 at each licensee's site. Upon reception of a reply datagram 3, monitor 2 would locate within itself the proper message corresponding to the authorization code, and use the message for various purposes. This method would reduce the size 20 of reply datagrams 3. However, if the licensor wanted to implement new denial codes, each product would need to somehow incorporate the new message associated with the new denial code into itself. The list of messages, one of which may be written as message text 25 28, are as follows:

AUTHORIZATION CODE	TEXT MESSAGE
30 1	This product is not licensed to run at this location. Please contact the licensor to either license this product, or move an existing license of your organization to this location. Use of this product at this

- 23 -

location is discontinued until this problem is resolved.

2
5 Your license on this product has expired. Please contact licensor in order to have your license extended. Use of this product is discontinued until this problem is resolved.

10 3 Payment on this licensed product is over due and past your grace period. Please have your accounting department send payment in order to continue your license. Use of this product is discontinued until this problem is resolved.

20 4 Your current use of this licensed product exceeds limits for the number of uses your organization has licensed. Please try again later.

25 5 A reply from licensor to numerous authorization requests was never received. This product must be connected to a communications network in order to function.

0 Authorization is OK. There is no message.

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

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

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

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

15 Also, the presently described embodiment includes a product 1 which is at the licensee's site. This implies that product 1 is on some physical media such as diskette, tape, or CD. However, product 1 can be electronically delivered over communications lines to
20 the licensee and therefore might exist in the memory of the licensee's machine, rather than any physical media. In the case of a product such as music, radio programs and the like, product 1 may even be broadcast to the licensee's site for playback; thus, the product
25 1 would not even be "resident" in the licensee's machine.

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

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

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

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

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

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

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

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

WHAT IS CLAIMED IS:

1. A method for monitoring the use of a licensed product, comprising the steps of:
 - generating, at regular time intervals,
5 datagrams including an address in a communications facility, said facility address identifying a licensee;
 - automatically sending said datagrams from at least one licensee's site over said facility to a
10 licensor's site while said licensed product is in use;
 - receiving said datagrams at said licensor's site;
 - storing an indication of receipt of each of said datagrams; and
15 counting said datagrams from each licensee as an indication of the use by the licensee of said licensed product.
2. A method as in claim 1 further wherein:
 - said generating step includes the step of
20 incorporating a model number of said product in said datagrams; and
 - said counting step includes the step of separately counting datagrams for each product model number for each licensee.
- 25 3. A method as in claim 1, wherein said generating step includes the step of automatically obtaining said facility address that identifies said licensee from said facility without any data being provided by said licensee.

- 28 -

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

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

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

receiving said request datagram at said licensor's site;

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

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

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

5. A method as in claim 4, wherein:

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

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

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

- 29 -

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

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

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

9. A method as in claim 4, wherein:
said generating step includes the step of providing a datagram identification code within said datagram;
said reply datagram sending step includes the step of inserting the same datagram identification code in said reply datagram; and
said reply receiving step rejects said reply authorizing datagram if the datagram identification code included in said reply authorizing datagram does

SUBSTITUTE SHEET

- 30 -

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

10. A method as in claim 4, wherein:

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

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

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

12. A method as in claim 4, wherein:

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

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

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

14. A method as in claim 4, wherein:

SUBSTITUTE SHEET

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

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

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

16. A method as in claim 4, wherein:

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

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

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

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

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

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

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

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

21. A method as in claim 4, wherein:
said licensed product comprises an executable portion and a data portion; and
said method further comprises a step of controlling use of said data portion with said executable portion.

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

23. A system for controlling licensed product comprising:
a communications facility to which at least one licensee having a license for operating a licensed product from the licensor is connected;
monitoring means, connected to said facility at a site of each said licensee, for generating a request datagram including an address of said licensee

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

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

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

15 24. A system as in claim 23, wherein:
 said monitoring means sends request datagrams at regular time intervals during use of said licensed product; and

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

 25. A system as in claim 23 wherein:
25 said monitoring means incorporates a model number for said product in said request datagram; and
 said controlling means comprises means for counting datagrams for each product model number for each licensee, in order to compute an amount to be
30 billed to each licensee.

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

5 27. A system as in claim 23, wherein:
 said controlling means sends a reply denial datagram to said licensee's site if use of said product is not approved; and

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

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

 29. A system as in claim 23, wherein:
 said monitoring means incorporates a unique identification code in said request datagram;

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

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

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

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

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

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

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

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

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

25 36. A system as in claim 23, wherein:
said monitoring means includes in said request datagram data indicative of the number of

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

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

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

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

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

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

25 41. A system as in claim 23, wherein:
 said licensed product comprises an executable portion and a data portion; and

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

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

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

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

20 45. A system for monitoring product comprising:
a communications facility to which at least one licensee having a license for operating a licensed product from a licensor is connected;
monitoring means, connected to said facility at a site of each said licensee, for generating datagrams including an address of said licensee on said facility and transmitting said datagrams at periodic intervals over said facility to a site of
25 said licensor; and
control means, connected to said facility at said licensor's site, for receiving said request datagrams, storing an indication of receipt of each of said datagrams and counting said datagrams from each

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

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

47. A system as in claim 45, wherein:
said monitoring means incorporates a product model number in said request datagrams; and
said controlling means separately counts request datagrams for each product model number for each licensee.

48. A method for monitoring the use of a licensed product comprising the steps of:
generating, at regular time intervals, datagrams including an address in a communications facility, said facility address identifying a licensee; and
automatically sending said datagrams from at least one licensee's site over said communications facility to a licensor's site while said licensed product is in use.

49. A method as in claim 48 further wherein:
said generating step includes the step of incorporating a model number of said product in said datagrams.

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

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

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

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

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

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

52. A method as in claim 51 wherein:

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

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

54. A method as in claim 51, wherein:

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

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

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

56. A method as in claim 51, wherein:
said generating step includes the step of providing a datagram identification code within said datagram; and
said reply receiving step rejects said reply authorizing datagram if the datagram identification code included in said reply authorizing datagram does not match the datagram identification code included in said request datagram.

57. A method as in claim 51, wherein:
said generating step includes the step of incorporating in said datagram data indicative of the number of processes currently using said product at said licensee's site.

58. A method as in claim 51, further comprising the steps of:
receiving a reply denial datagram; and
displaying, at a licensee's site, a reason for denial when said reply denial datagram is received.

59. A method as in claim 51, wherein:
said licensed product comprises an executable
portion and a data portion; and
said method further comprises a step of
5 controlling use of said data portion with said
executable portion.

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

10 61. A system for controlling a licensed product
comprising:

a communications facility to which at least
one licensee is connected;
monitoring means, connected to said
15 communications facility at a site of each said
licensee, for generating a request datagram including
an address of said licensee on said communications
facility and transmitting said request datagram over
said communications facility, and for receiving and
20 processing a reply authorizing datagram; and
means for denying use of said product when no
reply authorizing datagram is received.

62. A system as in claim 61, wherein:
said monitoring means sends request
25 datagrams at regular time intervals during use of said
licensed product.

63. A system as in claim 61 wherein:

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

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

65. A system as in claim 61, wherein:
said monitoring means resends said request datagram if no reply authorizing datagram and no reply denial datagram is received within a predetermined period of time after said requesting datagram is sent.

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

67. A system as in claim 61, wherein:
said monitoring means incorporates a unique identification code in said request datagram; and
said monitoring means rejects any reply authorizing datagram which does not include the same identification code as included in said request datagram.

68. A system as in claim 61, wherein:
said monitoring means includes in said request datagram data indicative of the number of processes, at a licensee's site, currently using said product.

69. A system as in claim 61, wherein:

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

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

10 71. A system as in claim 61, wherein:
 said licensed product comprises an executable portion and a data portion; and
 said system further comprises means for controlling use of said data portion with said executable portion.

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

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

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

75. A system for monitoring a licensed product comprising:

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

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

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

77. A system as in claim 75, wherein:
said monitoring means incorporates a product model number in said request datagrams.

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

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

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

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

79. A method as in claim 78 further wherein:
said datagrams include a model number of
each product; and

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

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

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

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

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

81. A method as in claim 80 wherein:

 said datagrams include a model number of
said product;

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

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

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

5 83. A method as in claim 80, wherein:
 said datagrams include a datagram
 identification code; and
 said reply datagram sending step includes
 the step of inserting the same datagram identification
10 code in said reply datagram.

 84. A method as in claim 80, wherein:
 said comparing step includes the step of
 comparing said facility address that identifies said
 licensee with a list of valid licensee addresses to
15 determine if said facility address is a valid address;
 and
 said reply authorizing datagram is not sent
 if said facility address that identifies said licensee
 is not valid.

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

 86. A method as in claim 80, wherein:
25 said comparing step includes the step of
 comparing a license expiration date with a date at
 which said datagram is received; and

- 47 -

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

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

88. A method as in claim 80, wherein:
said comparing step includes the step of
10 checking currentness of payments from said license;
and
said reply authorizing datagram is not sent if payment is overdue.

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

90. A method as in claim 80, wherein:
said datagrams include data indicative of
the number of processes currently using said product
20 at said licensee's site;
said comparing step includes the step of comparing a number of processes using said product to an authorized number; and
said reply authorizing datagram is not sent
25 if said number of processes using said product exceeds said authorized number.

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

number of processes using said product exceeds said authorized number.

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

93. A system for controlling a licensed product comprising:

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

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

25 94. A system as in claim 93, wherein:

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

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

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

95. A system as in claim 93 wherein:

5 said datagrams include a model number for said product; and

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

96. A system as in claim 93, wherein:

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

15 97. A system as in claim 93, wherein:

 said datagrams include a unique identification code; and

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

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

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

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

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

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

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

104. A system as in claim 93, wherein:
said datagrams include data indicative of
25 the number of processes, at a licensee's site, currently using said product; and
said controlling means does not generate a reply authorizing datagram, thus denying a use of said

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

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

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

15 107. A system as in claim 93, wherein:
said licensed product comprises an executable portion and a data portion; and
said system further comprises means for
controlling use of said data portion with said
executable portion.

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

20 109. A system for monitoring a licensed product comprising:

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

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

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

110. A system as in claim 110, wherein:
said request datagrams include a product model number; and
said controlling means separately counts request datagrams for each product model number for each licensee.

FIG. 1

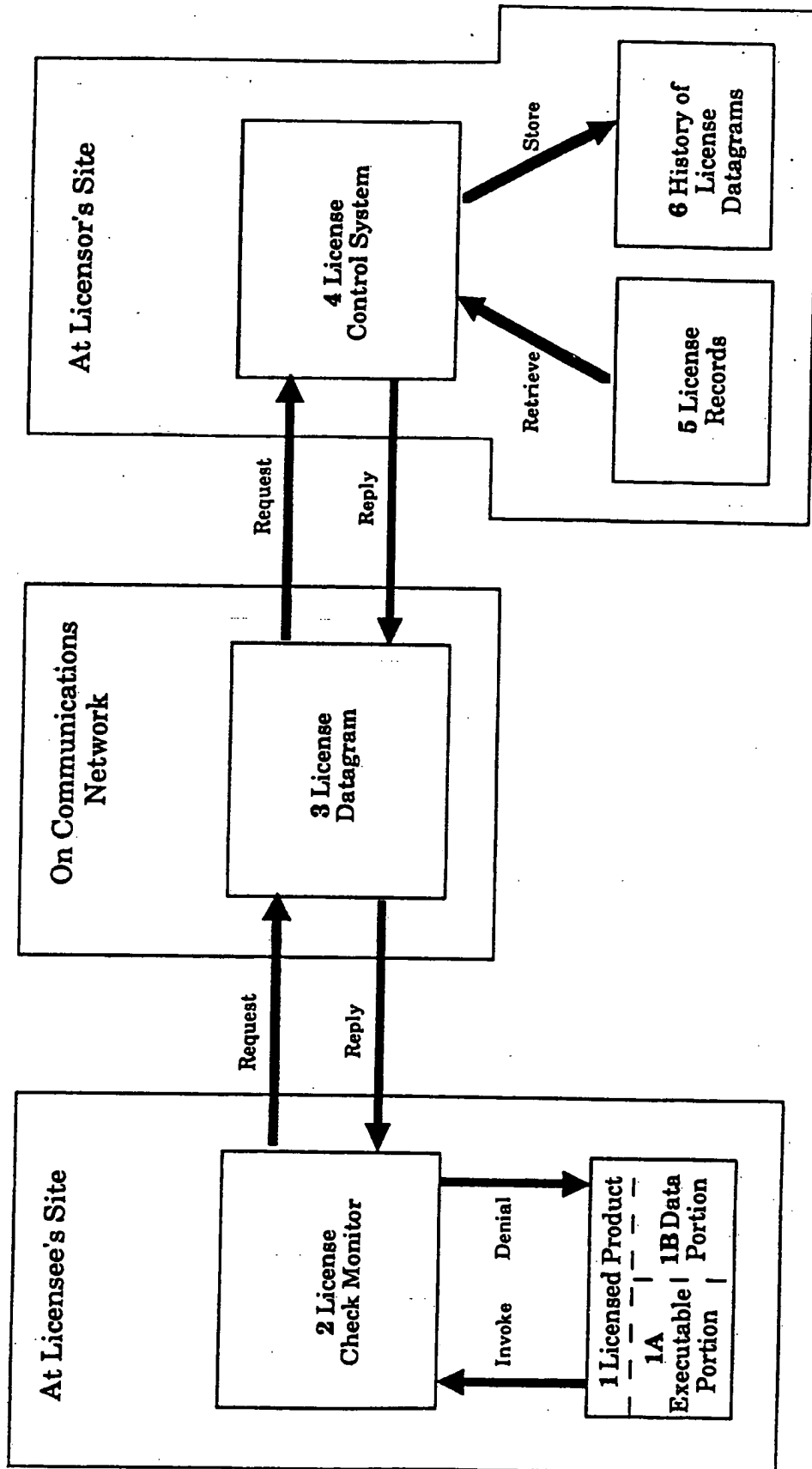


FIG. 2

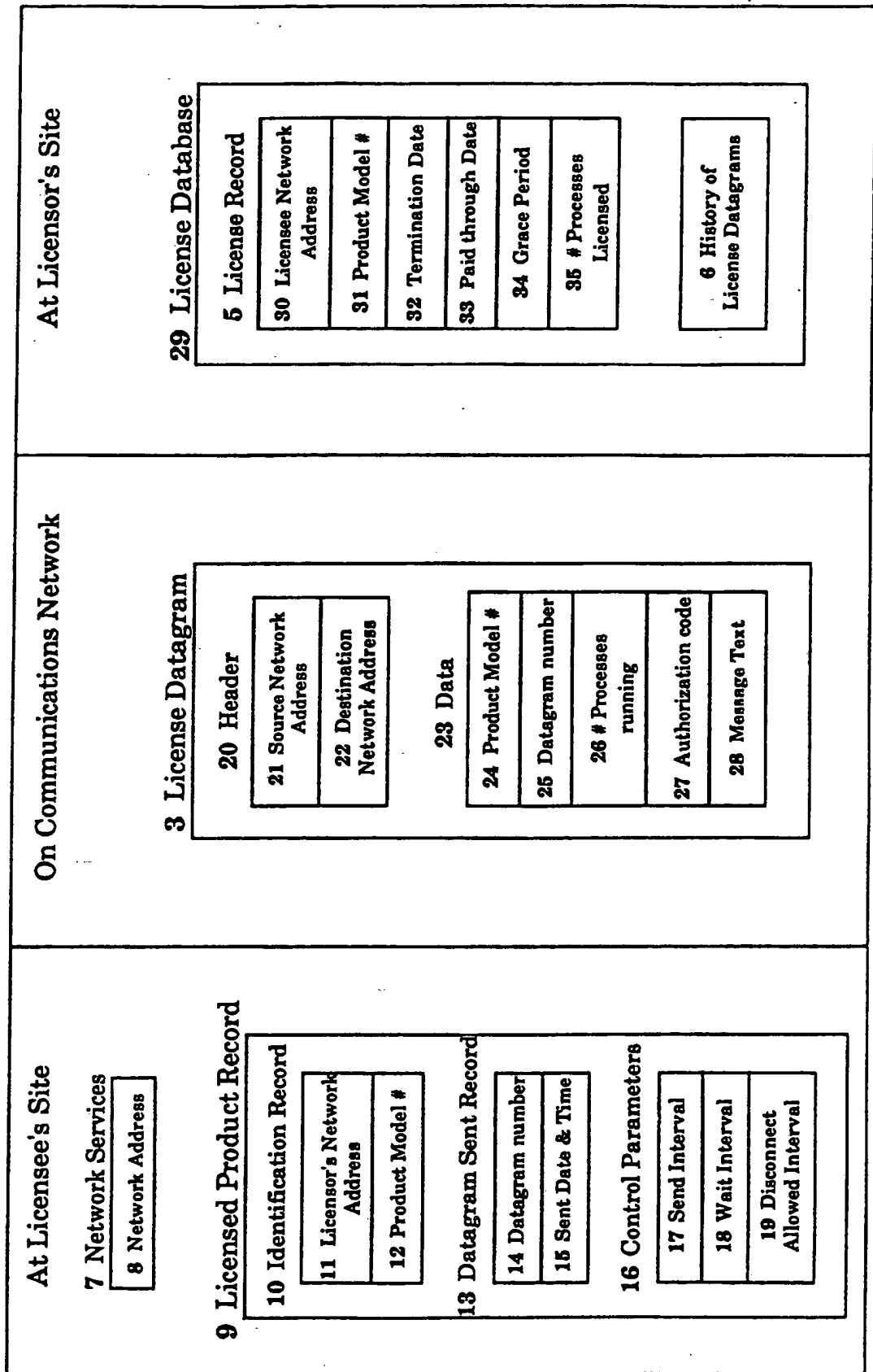


FIG. 3

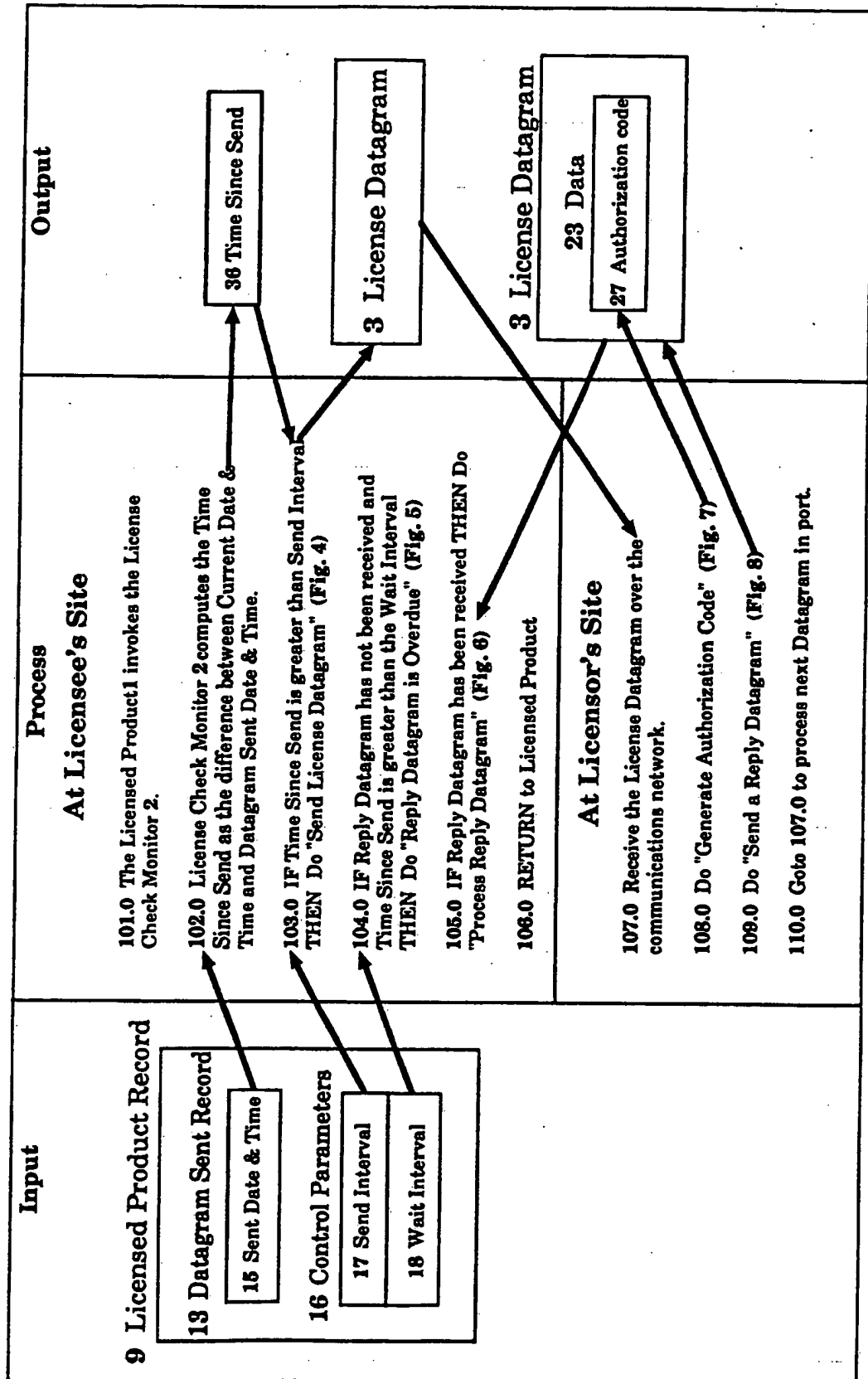


FIG. 4

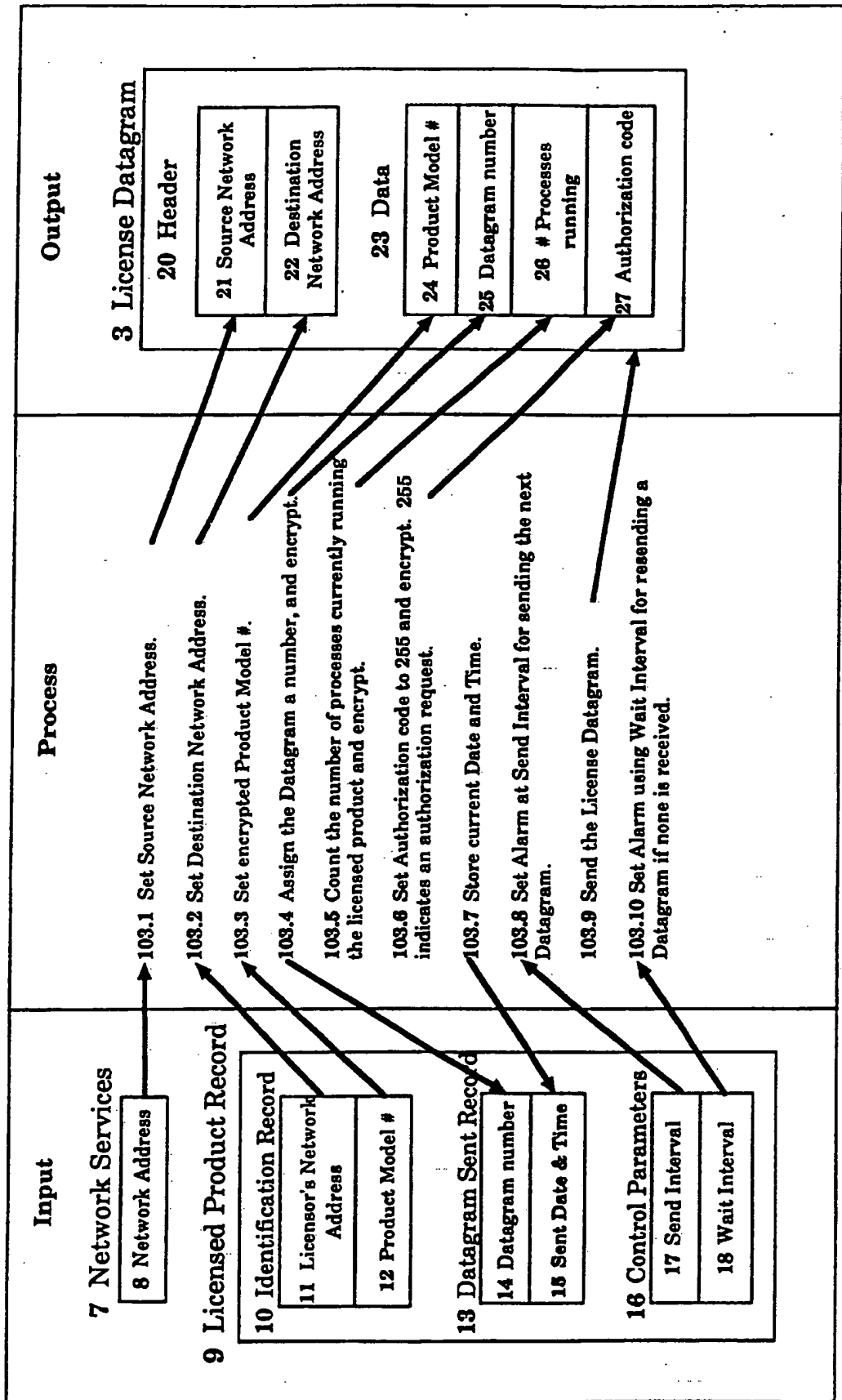


FIG. 5

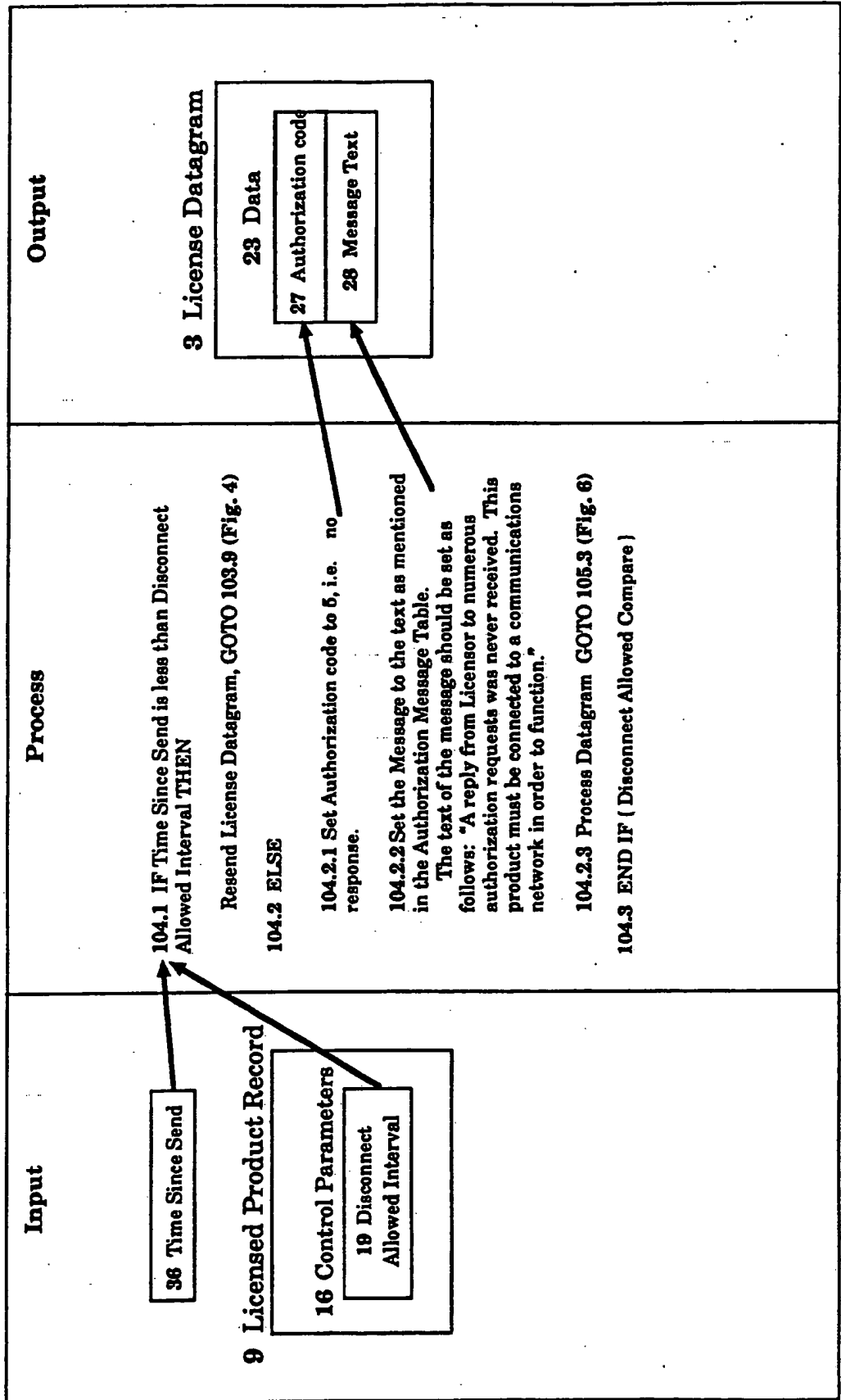


FIG. 6

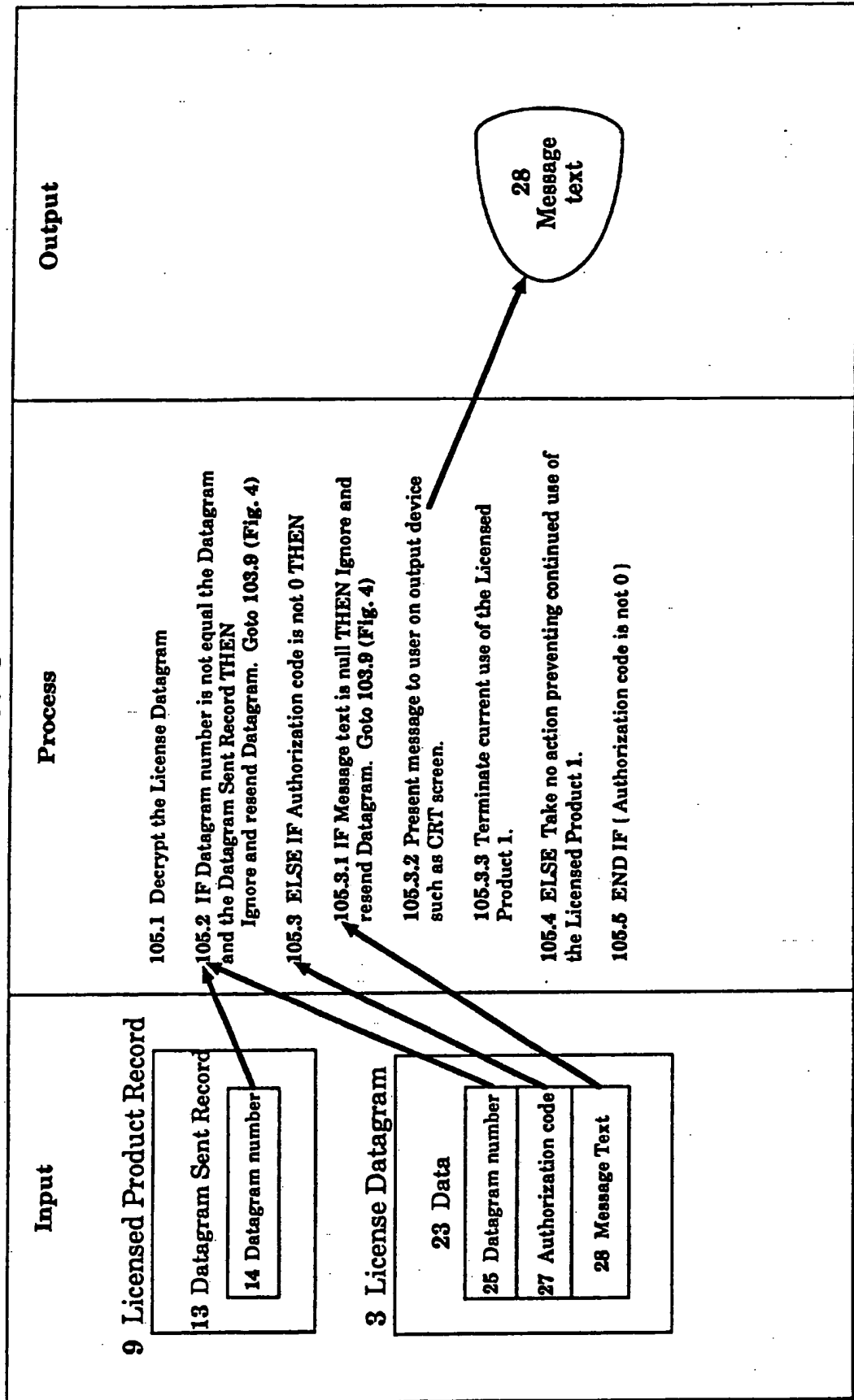


FIG. 7

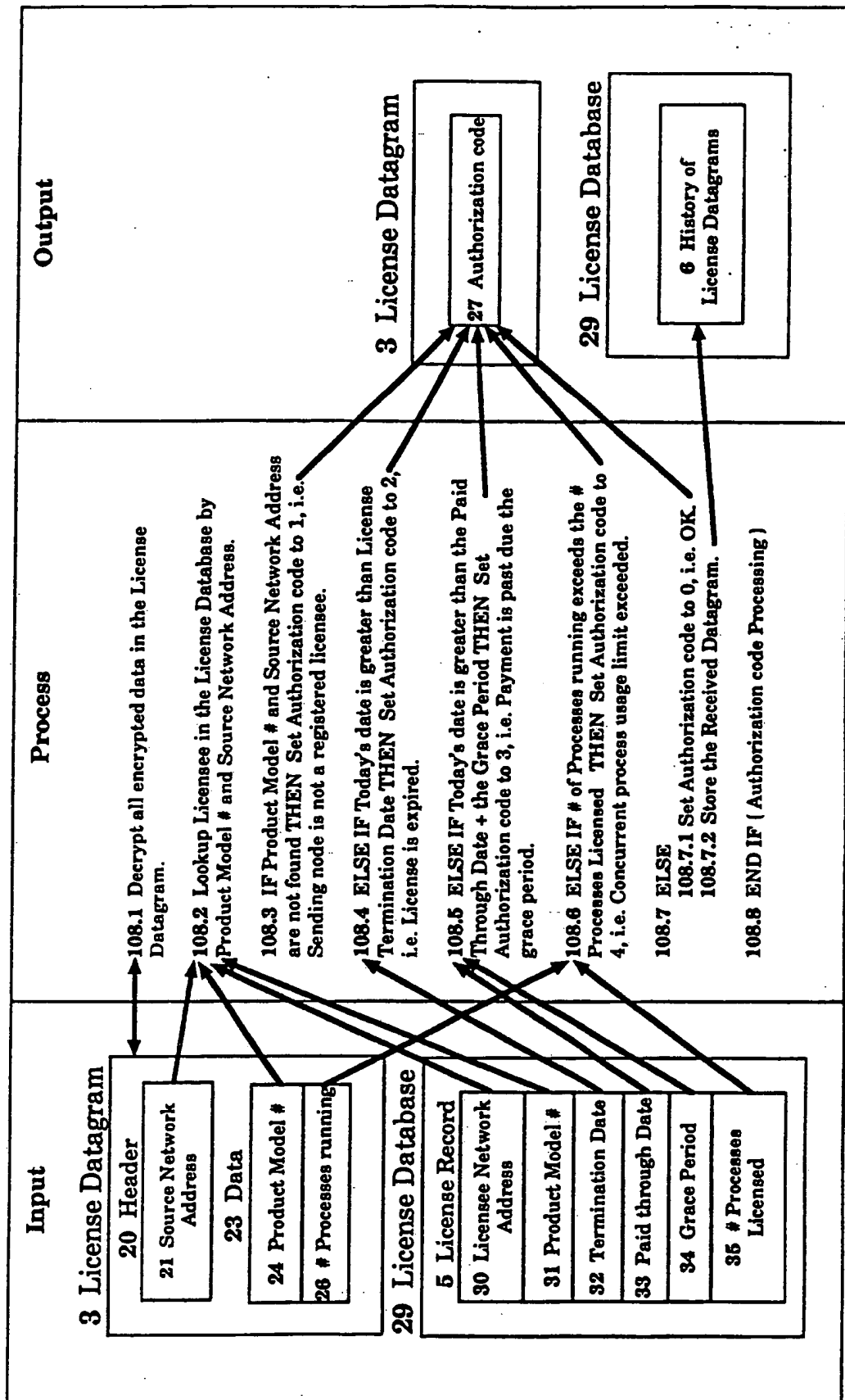
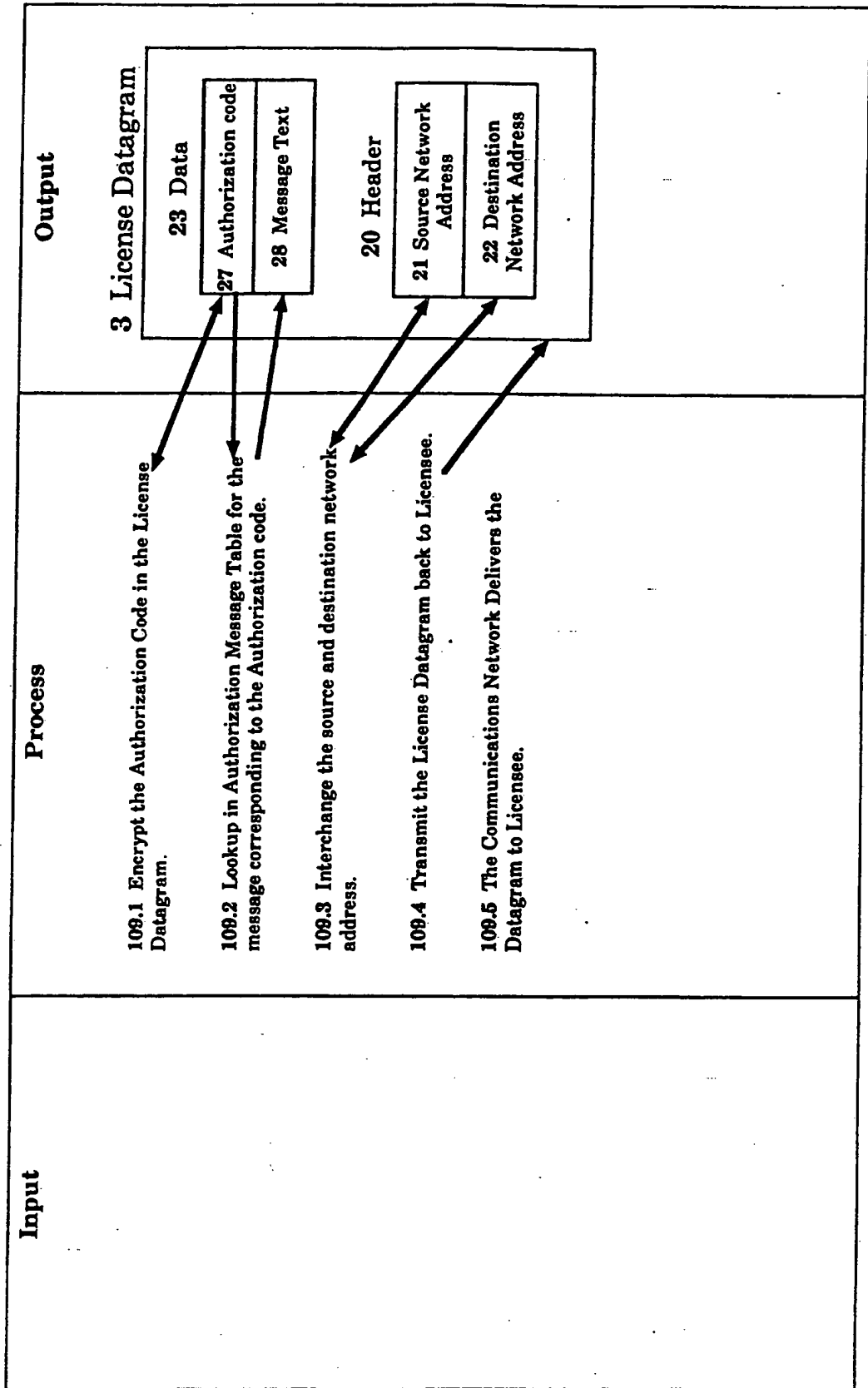


FIG. 8



INTERNATIONAL SEARCH REPORT

International application No.
PCT/US92/05387

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

B. FIELDS SEARCHED
 Minimum documentation searched (classification system followed by classification symbols)
 U.S. : 364/406; 380/25

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

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
 APS DATABASE: Software#, information, usage, monitor?, Licens?

C. DOCUMENTS CONSIDERED TO BE RELEVANT

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

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

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

Date of the actual completion of the international search 05 AUGUST 1992	Date of mailing of the international search report 04 NOV 1992
-----------------------------------------------------------------------------	--------------------------------------------------------------------------

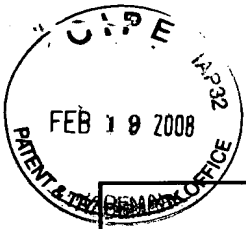
Name and mailing address of the ISA/ Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231	Authorized officer <i>Kenneth S. Kim</i> KENNETH S. KIM
Facsimile No. NOT APPLICABLE	Telephone No. (703) 308-1634

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US92/05387

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

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



TFW

TRANSMITTAL FORM <i>(to be used for all correspondence after initial filing)</i>		Application Number	10/956,121
		Filing Date	October 4, 2004
		First Named Inventor	Xin WANG, <i>et al.</i>
		Group Art Unit	3621
		Examiner Name	Thomas C. West
Total Number of Pages in This Submission		Attorney Docket Number	111325-291300

ENCLOSURES <i>(check all that apply)</i>		
<input type="checkbox"/> Fee Transmittal Form <input type="checkbox"/> Fee Attached <input type="checkbox"/> Amendment / Reply <input type="checkbox"/> After Final <input type="checkbox"/> Affidavits/declaration(s) <input type="checkbox"/> Extension of Time Request <input type="checkbox"/> Express Abandonment Request <input checked="" type="checkbox"/> Information Disclosure Statement <input type="checkbox"/> Certified Copy of Priority Document(s) <input type="checkbox"/> Response to Missing Parts/ Incomplete Application <input type="checkbox"/> Response to Missing Parts under 37 CFR 1.52 or 1.53	<input type="checkbox"/> Drawing(s) <input type="checkbox"/> Licensing-related Papers <input type="checkbox"/> Petition <input type="checkbox"/> Petition to Convert to a Provisional Application <input type="checkbox"/> Power of Attorney, Revocation Change of Correspondence Address <input type="checkbox"/> Terminal Disclaimer <input type="checkbox"/> Request for Refund <input type="checkbox"/> CD, Number of CD(s) _____	<input type="checkbox"/> After Allowance Communication to Group <input type="checkbox"/> Appeal Communication to Board of Appeals and Interferences <input type="checkbox"/> Appeal Communication to Group (<i>Appeal Notice, Brief, Reply Brief</i>) <input type="checkbox"/> Proprietary Information <input type="checkbox"/> Status Letter <input checked="" type="checkbox"/> Other Enclosure(s) <i>(please identify below)</i> : 1. PTO Form 1449 2. One Box including 112 cited references
Remarks	<input checked="" type="checkbox"/> The Director is hereby authorized to charge any additional fees required or credit any overpayments to Deposit Account No. 19-2380 for the above identified docket number.	

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT	
Firm or Individual name	Marc S. Kaufman Registration No. 35,212 Nixon Peabody LLP 401 9th Street, N.W., Suite 900 Washington, D.C. 20004-2128
Signature	/Marc S. Kaufman, Reg. # 35,212/
Date	February 19, 2008

CERTIFICATE OF MAILING OR TRANSMISSION			
I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450, or facsimile transmitted to the U.S. Patent and Trademark Office (Fax No. (571) 273-8300) on the date shown below.			
Name <i>(Print/Type)</i>			
Signature		Date	



UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:)	Confirmation No.: 8924
Xin WANG et al.)	Group Art Unit: 3621
Application No.: 10/956,121)	Examiner: West, Thomas C.
Filed: October 4, 2004)	
For: SYSTEM AND METHOD FOR)	Date: February 19, 2008
MANAGING TRANSFER OF)	
RIGHTS USING SHARED STATE)	
VARIABLES)	

INFORMATION DISCLOSURE STATEMENT

United States Patent and Trademark Office
Customer Window
Randolph Building
401 Dulany Street
Alexandria, VA 22314

Dear Sir:

In accordance with the duty of disclosure as set forth in 37 C.F.R. § 1.56, Applicants hereby submit the following information in conformance with 37 C.F.R. §§ 1.97 and 1.98. Pursuant to 37 C.F.R. § 1.98(a)(2)(ii), copies of the cited U.S. patents (*i.e.*, Reference Cite Nos. 1–101) are not enclosed. Copies of the cited Foreign patents (*i.e.*, Reference Cite Nos. 102–173) are enclosed. Copies of the cited non-patent references (*i.e.*, Reference Cite Nos. 174–213) are enclosed. The references have been cited in recent oppositions in the European Patent Office relating to cases owned by assignee.

The documents are being submitted within three (3) months of the filing of this application or entry into the national stage of this application, or before the first Office Action on the merits, whichever is later, therefore no fee or certification is required under 37 C.F.R. § 1.97(b)(4).

It is requested that the accompanying PTO/SB/08A be considered and made of record in the above-identified application. To assist the Examiner, the documents are listed on the attached form PTO/SB/08A. It is respectfully requested that an Examiner initialed copy of this form be returned to the undersigned.

The Commissioner is hereby authorized to charge any fees connected with this filing which may be required now, or credit any overpayment to Deposit Account No. 19-2380.

Respectfully submitted,
NIXON PEABODY LLP

Date: February 19, 2008

By: /Marc S. Kaufman, Reg. # 35,212/
Marc S. Kaufman
Registration No. 35,212

NIXON PEABODY LLP
CUSTOMER NO.: 22204
401 9th Street, N.W., Suite 900
Washington, DC 20004
Tel: 202-585-8000
Fax: 202-585-8080



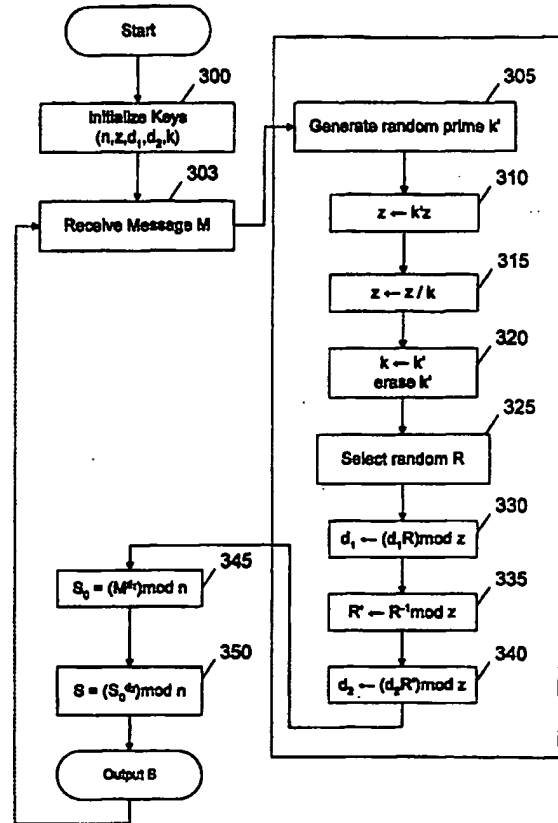
INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

<p>(51) International Patent Classification ⁶ : H04L 9/30</p>	<p>A1</p>	<p>(11) International Publication Number: WO 99/35782 (43) International Publication Date: 15 July 1999 (15.07.99)</p>
<p>(21) International Application Number: PCT/US98/27896 (22) International Filing Date: 31 December 1998 (31.12.98) (30) Priority Data: 60/070,344 2 January 1998 (02.01.98) US 60/089,529 15 June 1998 (15.06.98) US (71) Applicant: CRYPTOGRAPHY RESEARCH, INC. [US/US]; Suite 1088, 870 Market Street, San Francisco, CA 94102 (US). (72) Inventors: KOCHER, Paul, C.; 143 Fillmore Street, San Francisco, CA 94117 (US). JAFFE, Joshua, M.; 21B Bird Street, San Francisco, CA 94110 (US). (74) Agents: LAURIE, Ronald, S. et al.; Skadden, Arps, Slate, Meagher & Flom LLP, 525 University Avenue, Palo Alto, CA 94301 (US).</p>	<p>(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).</p> <p>Published With international search report. Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.</p>	

(54) Title: LEAK-RESISTANT CRYPTOGRAPHIC METHOD AND APPARATUS

(57) Abstract

The present invention provides a method and apparatus for securing cryptographic devices against attacks involving external monitoring and analysis. A "self-healing" property is introduced, enabling security to be continually re-established following partial compromises. In addition to producing useful cryptographic results, a typical leak-resistant cryptographic operation modifies or updates (330) secret key material in a manner designed to render useless any information about the secrets that may have previously leaked from the system. Exemplary leak-proof and leak-resistant implementations of the invention are shown for symmetric authentication (350), certified Diffie-Hellman (when either one or both users have certificates), RSA, ElGamal public key decryption (303).



FOR THE PURPOSES OF INFORMATION ONLY

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

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece	ML	Mali	TR	Turkey
BG	Bulgaria	HU	Hungary	MN	Mongolia	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MR	Mauritania	UA	Ukraine
BR	Brazil	IL	Israel	MW	Malawi	UG	Uganda
BY	Belarus	IS	Iceland	MX	Mexico	US	United States of America
CA	Canada	IT	Italy	NE	Niger	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NL	Netherlands	VN	Viet Nam
CG	Congo	KE	Kenya	NO	Norway	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NZ	New Zealand	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	PL	Poland		
CM	Cameroon	KR	Republic of Korea	PT	Portugal		
CN	China	KZ	Kazakstan	RO	Romania		
CU	Cuba	LC	Saint Lucia	RU	Russian Federation		
CZ	Czech Republic	LI	Liechtenstein	SD	Sudan		
DE	Germany	LK	Sri Lanka	SE	Sweden		
DK	Denmark	LR	Liberia	SG	Singapore		
EE	Estonia						

LEAK-RESISTANT CRYPTOGRAPHIC METHOD AND APPARATUS

This application claims the benefit of US Provisional Application No. 60/070,344 filed January 2, 1998, and US Provisional Application No. 60/089,529 filed June 15, 1998.

5 TECHNICAL FIELD

The method and apparatus of the present invention relate generally to cryptographic systems and, more specifically, to securing cryptographic tokens that must maintain the security of secret information in hostile environments.

BACKGROUND OF THE INVENTION

10 Most cryptosystems require secure key management. In public-key based security systems, private keys must be protected so that attackers cannot use the keys to forge digital signatures, modify data, or decrypt sensitive information. Systems employing symmetric cryptography similarly require that keys be kept secret. Well-designed cryptographic algorithms and protocols should prevent attackers who eavesdrop on communications from
15 breaking systems. However, cryptographic algorithms and protocols traditionally require that tamper-resistant hardware or other implementation-specific measures prevent attackers from accessing or finding the keys.

If the cryptosystem designer can safely assume that the key management system is completely tamper-proof and will not reveal any information relating to the keys except via
20 the messages and operations defined in the protocol, then previously known cryptographic techniques are often sufficient for good security. It is currently extremely difficult, however, to make hardware key management systems that provide good security, particularly in low-cost unshielded cryptographic devices for use in applications where attackers will have physical control over the device. For example, cryptographic tokens (such as smartcards used
25 in electronic cash and copy protection schemes) must protect their keys even in potentially hostile environments. (A token is a device that contains or manipulates cryptographic keys that need to be protected from attackers. Forms in which tokens may be manufactured include, without limitation, smartcards, specialized encryption and key management devices, secure telephones, secure picture phones, secure web servers, consumer electronics devices
30 using cryptography, secure microprocessors, and other tamper-resistant cryptographic systems.)

A variety of physical techniques for protecting cryptographic devices are known, including enclosing key management systems in physically durable enclosures, coating integrated circuits with special coatings that destroy the chip when removed, and wrapping devices with fine wires that detect tampering. However, these approaches are expensive, difficult to use in single-chip solutions (such as smartcards), and difficult to evaluate since there is no mathematical basis for their security. Physical tamper resistance techniques are also ineffective against some attacks. For example, recent work by Cryptography Research has shown that attackers can non-invasively extract secret keys using careful measurement and analysis of many devices' power consumption. Analysis of timing measurements or electromagnetic radiation can also be used to find secret keys.

Some techniques for hindering external monitoring of cryptographic secrets are known, such as using power supplies with large capacitors to mask fluctuations in power consumption, enclosing devices in well-shielded cases to prevent electromagnetic radiation, message blinding to prevent timing attacks, and buffering of inputs/outputs to prevent signals from leaking out on I/O lines. Shielding, introduction of noise, and other such countermeasures are often, however, of limited value, since skilled attackers can still find keys by amplifying signals and filtering out noise by averaging data collected from many operations. Further, in smartcards and other tamper-resistant chips, these countermeasures are often inapplicable or insufficient due to reliance on external power sources, impracticality of shielding, and other physical constraints. The use of blinding and constant-time mathematical algorithms to prevent timing attacks is also known, but does not prevent more complex attacks such as power consumption analysis (particularly if the system designer cannot perfectly predict what information will be available to an attacker, as is often the case before a device has been physically manufactured and characterized).

The present invention makes use of previously-known cryptographic primitives and operations. For example: U.S. patent 5,136,646 to Haber et al. and the pseudorandom number generator used in the RSAREF cryptographic library use repeated application of hash functions; anonymous digital cash schemes use blinding techniques; zero knowledge protocols use hash functions to mask information; and key splitting and threshold schemes store secrets in multiple parts.

SUMMARY OF THE INVENTION

The present invention introduces leak-proof and leak-resistant cryptography, mathematical approaches to tamper resistance that support many existing cryptographic primitives, are inexpensive, can be implemented on existing hardware (whether by itself or
5 via software capable of running on such hardware), and can solve problems involving secrets leaking out of cryptographic devices. Rather than assuming that physical devices will provide perfect security, leak-proof and leak-resistant cryptographic systems may be designed to remain secure even if attackers are able to gather some information about the system and its secrets. This invention describes leak-proof and leak-resistant systems that implement
10 symmetric authentication, Diffie-Hellman exponential key agreement, ElGamal public key encryption, ElGamal signatures, the Digital Signature Standard, RSA, and other algorithms.

One of the characteristic attributes of a typical leak-proof or leak-resistant cryptosystem is that it is "self-healing" such that the value of information leaked to an attacker decreases or vanishes with time. Leak-proof cryptosystems are able to withstand
15 leaks of up to L_{MAX} bits of information per transaction, where L_{MAX} is a security factor chosen by the system designer to exceed to the maximum anticipated leak rate. The more general class of leak-resistant cryptosystems includes leak-proof cryptosystems, and others that can withstand leaks but are not necessarily defined to withstand any defined maximum information leakage rate. Therefore, any leak-proof system shall also be understood to be
20 leak-resistant. The leak-resistant systems of the present invention can survive a variety of monitoring and eavesdropping attacks that would break traditional (non-leak-resistant) cryptosystems.

A typical leak-resistant cryptosystem of the present invention consists of three general parts. The initialization or key generation step produces secure keying material appropriate
25 for the scheme. The update process cryptographically modifies the secret key material in a manner designed to render useless any information about the secrets that may have previously leaked from the system, thus providing security advantages over systems of the background art. The final process performs cryptographic operations, such as producing digital signatures or decrypting messages.

30 BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 shows an exemplary leak-resistant symmetric authentication method.

Figure 2 shows an exemplary leak-resistant Diffie-Hellman exponential key exchange operation.

Figure 3 shows an exemplary leak-resistant RSA private key operation.

Figure 4 shows an exemplary leak-resistant ElGamal signing operation.

5

DETAILED DESCRIPTION OF THE INVENTION

The sections following will describe an introduction to leak-proof/leak-resistant cryptography, followed by various embodiments of the general techniques of the invention as applied to improve the security of common cryptographic protocols.

10 I. Introduction and Terminology

The leakage rate L is defined as the number of bits of useful information about a cryptosystem's secrets that are revealed per operation, where an operation is a cryptographic transaction. Although an attacker may be able to collect more than L bits worth of measurement data, by definition this data yields no more than L bits of useful information
15 about the system's secrets.

The implementer of a leak-proof system chooses a design parameter L_{MAX} , the maximum amount of leakage per operation the system may allow if it is to remain uncompromised. L_{MAX} should be chosen conservatively, and normally should significantly exceed the amount of useful information known to be leaked to attackers about the system's
20 secrets during each transaction. Designers do not necessarily need to know accurately or completely the quantity and type of information that may leak from their systems; the choice of L_{MAX} may be made using estimates and models for the system's behavior. General factors affecting the choice of L_{MAX} include the types of monitoring potentially available to attackers, the amount of error in attackers' measurements, and engineering constraints that limit L_{MAX} .
25 (Larger values of L_{MAX} increase memory and performance requirements of the device, and in some cases may increase L .) To estimate the amount of useful information an attacker could collect by monitoring a device's power consumption, for example, a designer might consider the amount of noise in the device's power usage, the power line capacitance, the useful time resolution for power consumption measurements, as well as the strength of the signals being
30 monitored. Similarly, the designer knows that timing measurements can rarely yield more than a few bits of information per operation, since timing information is normally quantized to an integral number of clock cycles. In choosing L_{MAX} , the designer should assume that

attackers will be able to combine information gleaned from multiple types of attacks. If the leakage rate is too large (as in the extreme case where L equals the key size because the entire key can be extracted during a single transaction), additional design features should be added to reduce L and reduce the value needed for L_{MAX} . Such additional measures can include
 5 known methods, such as filtering the device's power inputs, adding shielding, introducing noise into the timing or power consumption, implementing constant-time and constant execution path algorithms, and changing the device layout. Again, note that the designer of a leak-resistant system does not actually need to know what information is being revealed or how it is leaked; all he or she need do is choose an upper bound for the rate at which attackers
 10 might learn information about the keys. In contrast, the designer of a traditional system faces the much harder task of ensuring that no information about the secrets will leak out.

There are many ways information about secrets can leak from cryptosystems. For example, an attacker can use a high-speed analog-to-digital converter to record a smartcard's power consumption during a cryptographic operation. The amount of useful information that
 15 can be gained from such a measurement varies, but it would be fairly typical to gain enough information to guess each of 128 key bits correctly with a probability of 0.7. This information can reduce the amount of effort required for a brute force attack. For example, a brute force attack with one message against a key containing k bits where each bit's value is known with probability p can be completed in

$$20 \quad E(k, p) = \sum_{i=0}^k \left[\binom{k}{i} (1-p)^i p^{k-i} \left[\left(\sum_{j=0}^i \binom{k}{j} \right) - \frac{1}{2} \binom{k}{i} \right] + \frac{1}{2} \right]$$

operations. The reduction in the effort for a brute force attack is equivalent to shortening the key by $L = \log_2(E(k, 1/2) / E(k, p)) = \log_2(k - E(k, p) - 1)$ bits. (For example, in the case of $k = 128$ and $p = 0.7$, L is estimated to be about 11 bits for the first measurement. With a multiple message attack, the attacker's effort can fall to as low as $E(k, p) = \frac{1}{p^k}$.) Attackers can gain
 25 additional information about the keys by measuring additional operations; unless leak-resistance is used, finding the key becomes easy after just a few dozen operations.

When choosing L_{MAX} , a system designer should consider the signal-to-noise ratio of an attacker's measurements. For example, if the signal and noise are of roughly equivalent magnitude, the designer knows that an attacker's measurements should be incorrect about 25
 30 percent of the time (e.g., $p = 0.75$ if only one observation per key bit is possible). Many

measurement techniques, such as those involving timing, may have signal-to-noise ratios of 1:100 or worse. With such systems, L is generally quite small, but attackers who can make a large number of measurements can use averaging or other statistical techniques to recover the entire key. In extreme cases, attackers may be able to obtain all key bits with virtually perfect accuracy from a single transaction (i.e., $L = k$), necessitating the addition of shielding, noise
5 in the power consumption (or elsewhere), and other measures to reduce p and L . Of course, L_{MAX} should be chosen conservatively; in the example above where less than 4 useful bits are obtained per operation for the given attack, the designer might select $L_{\text{MAX}} = 64$ for a leak-proof design.

10 Leak-proof (and, more generally, leak-resistant) cryptosystems provide system designers with important advantages. When designing a traditional (i.e., non-leak-resistant and non-leak-proof) cryptosystem, a careful cryptosystem designer should study all possible information available to attackers if he or she is to ensure that no analytical techniques could be used to compromise the keys. In practice, many insecure systems are developed and
15 deployed because such analysis is incomplete, too difficult even to attempt, or because the cryptographers working on the system do not understand or cannot completely control the physical characteristics of the device they are designing. Unexpected manufacturing defects or process changes, alterations made to the product by attackers, or modifications made to the product in the field can also introduce problems. Even a system designed and analyzed with
20 great care can be broken if new or improved data collection and analysis techniques are found later. In contrast, with leak-proof cryptography, the system designer only needs to define an upper bound on the maximum rate at which attackers can extract information about the keys. A detailed understanding of the information available to attackers is not required, since leak-proof (and leak-resistant) cryptosystem designs allow for secret information in the device to
25 leak out in (virtually) any way, yet remain secure despite this because leaked information is only of momentary value.

In a typical leak-proof design, with each new cryptographic operation i , the attacker is assumed to be able to choose any function F_i and determine the L_{MAX} -bit result of computing F_i on the device's secrets, inputs, intermediates, and outputs over the course of the operation.
30 The attacker is even allowed to choose a new function F_i with each new operation. The system may be considered leak-proof with a security factor n and leak rate L_{MAX} if, after observing a large number of operations, an attacker cannot forge signatures, decrypt data, or

perform other sensitive operations without performing an exhaustive search to find an n -bit key or performing a comparable $O(2^n)$ operation. In addition to choosing L_{MAX} , designers also choose n , and should select a value large enough to make exhaustive search infeasible. In the sections that follow, various embodiments of the invention, as applied to improve the security of common cryptographic operations and protocols, will be described in more detail.

II. Symmetric Cryptographic Protocols

A. Symmetric Authentication

An exemplary cryptographic protocol that can be secured using the techniques of the present invention is symmetric authentication.

1. Conventional Symmetric Authentication

Assume a user wishes to authenticate herself to a server using an n -bit secret key, K , known to both the server and the user's cryptographic token, but not known to attackers. The cryptographic token should be able to resist tampering to prevent, for example, attackers from being able to extract secrets from a stolen token. If the user's token has perfect tamper resistance (i.e., $L=0$), authentication protocols of the background art can be used. Typically the server sends a unique, unpredictable challenge value R to the user's token, which computes the value $A = H(R || K)$, where " $||$ " denotes concatenation and H is a one-way cryptographic hash function such as SHA. The user sends A to the server, which independently computes A (using its copy of K) and compares its result with the received value. The user authentication succeeds only if the comparison operation indicates a match.

If the function H is secure and if K is sufficiently large to prevent brute force attacks, attackers should not be able to obtain any useful information from the (R, A) values of old authentication sessions. To ensure that attackers cannot impersonate users by replaying old values of A , the server generates values of R that are effectively (with sufficiently high probability) unique. In most cases, the server should also make R unpredictable to ensure that an attacker with temporary possession of a token cannot compute future values of A . For example, R might be a 128-bit number produced using a secure random number generator (or pseudorandom number generator) in the server. The properties of cryptographic hash functions such as H have been the subject of considerable discussion in the literature, and need not be described in detail here. Hash functions typically provide functionality modeled after a random oracle, deterministically producing a particular output from any input. Ideally, such functions should be collision-resistant, non-invertible, should not leak partial

information about the input from the output, and should not leak information about the output unless the entire input is known. Hash functions can have any output size. For example, MD5 produces 128-bit outputs and SHA produces 160-bit outputs. Hash functions may be constructed from other cryptographic primitives or other hash functions.

5 While the cryptographic security of the protocol using technology of the background art may be good, it is not leak-proof; even a one-bit leak function (with $L=1$) can reveal the key. For example, if the leak function F equals bit $(R \bmod n)$ of K , an attacker can break the system quickly since a new key bit is revealed with every transaction where $(R \bmod n)$ has a new value. Therefore, there is a need for a leak-proof/leak-resistant symmetric authentication
10 protocol.

2. Leak-Resistant Symmetric Authentication

The following is one embodiment of a leak-resistant (and, in fact, also leak-proof) symmetric authentication protocol, described in the context of a maximum leakage rate of L_{MAX} bits per transaction from the token and a security factor n , meaning that attacks of
15 complexity $O(2^n)$, such as brute-force attacks against an n -bit key, are acceptable, but there should not be significantly easier attacks. The user's token maintains a counter t , which is initialized to zero, and an $(n+2L_{MAX})$ -bit shared secret K_t , which is initialized with a secret K_0 . Note that against adversaries performing precomputation attacks based on Hellman's time/memory trade-off, larger values of n may be in order. Note also that some useful
20 protocol security features, such as user and/or server identifiers in the hash operation inputs, have been omitted for simplicity in the protocol description. It is also assumed that no leaking will occur from the server. For simplicity in the protocol description, some possible security features (such as user and/or server identifiers in the hash operation inputs) have been omitted, and it is assumed that the server is in a physically secure environment.
25 However, those skilled in the art will appreciate that the invention is not limited to such assumptions, which have been made as a matter of convenience rather than necessity.

As in the traditional protocol, the server begins the authentication process by generating a unique and unpredictable value R at step 105. For example, R might be a 128-bit output from a secure random number generator. At step 110, the server sends R to the user's
30 token. At step 112, the token receives R . At step 115, the token increments its counter t by computing $t \leftarrow t + 1$. At step 120, the token updates K_t by computing $K_t \leftarrow H_K(t \parallel K_t)$, where H_K is a cryptographic hash function that produces an $(n+2L_{MAX})$ bit output from the old value

of K_t and the (newly incremented) value of t . Note that in the replacement operations (denoted " \leftarrow "), the token deletes the old values of t and K_t , replacing them with the new values. By deleting the old K_t , the token ensures that future leak functions cannot reveal information about the old (deleted) value. At step 122, the token uses the new values of t and K_t to compute an authenticator $A = H_A(K_t \parallel t \parallel R)$. At step 125, the token sends both t and the authenticator A to the server, which receives them at step 130. At step 135, the server verifies that t is acceptable (e.g., not too large but larger than the value received in the last successful authentication). If t is invalid, the server proceeds to step 175. Otherwise, at step 140, the server initializes its loop counter i to zero and its key register K_t' to K_0 . At step 145, the server compares i with the received value of t , proceeding to step 160 if they are equal. Otherwise, at step 150, the server increments i by computing $i \leftarrow i + 1$. At step 155, the server computes $K_t' \leftarrow H_K(i \parallel K_t')$, then proceeds back to step 145. At step 160, the server computes $A' = H_A(K_t' \parallel t \parallel R)$. Finally, at step 165, the server compares A and A' , where the authentication succeeds at step 170 if they match, or fails at 175 if they do not match.

This design assumes that at the beginning of any transaction the attacker may have L_{MAX} bits of useful information about the state of the token (e.g., K_t) that were obtained using the leak function F in a previous operation. During the transaction, the attacker can gain an additional L_{MAX} bits of useful information from the token. If, at any time, any $2L_{MAX}$ (or fewer) bits of useful information about the secret are known to the attacker, there are still $(n+2L_{MAX})-2L_{MAX} = n$ or more unknown bits. These n bits of unknown information ensure that attacks will require $O(2^n)$ effort, corresponding to the desired security factor. However, the attacker should have no more than L_{MAX} bits of useful information about K_t at the end of the transaction. The property that attackers lose useful information during normal operation of the system is a characteristic of the leak-proof or leak-resistant cryptosystem. In general, this information loss is achieved when the cryptosystem performs operations that convert attackers' useful partial information about the secret into useless information. (Information is considered useless if it gives an attacker nothing better than the ability to test candidate values in an $O(2^n)$ exhaustive search or other "hard" operation. For example, if exhaustive search of X is hard and H is a good hash function, $H(X)$ is useless information to an attacker trying to find X .)

Thus, the attacker is assumed to begin with L_{MAX} bits of useful information about K_t before the token's $K_t \leftarrow H_K(t \parallel K_t)$ computation. (Initial information about anything other

than K_t , is of no value to an attacker because K_t is the only secret value in the token. The function H_K and the value of t are not assumed to be secret.) The attacker's information can be any function of K_t , produced from the previous operation's leaks.

5 **3. Security Characteristics of Leak-Proof Systems**

The following section provides a technical discussion of the security characteristics of the exemplary leak-proof system described above. The following analysis is provided as an example of how the design can be analyzed, and how a system may be designed using general assumptions about attackers' capabilities. The discussion and assumptions do not necessarily
10 apply to other embodiments of the invention and should not be construed as limiting the scope or applicability of the invention in any way.

During the course of a transaction, the leak function F might reveal up to L_{MAX} information about the system and its secrets. The design assumes that any information contained in the system may be leaked by F , provided that F does not reveal useful new
15 information about values of K_t that were deleted before the operation started, and F does not reveal useful information about values of K_t that will be computed in future operations. These constraints are completely reasonable, since real-world leaks would not reveal information about deleted or not-yet-existent data. (The only way information about future
20 K_t values could be leaked would be the bizarre case where the leak function itself included, or was somehow derived from, the function H_K .) In practice, these constraints on F are academic and of little concern, but they are relevant when constructing proofs to demonstrate the security of a leak-proof system.

If the leak occurs at the beginning of the H_K computation, it could give the attacker up to $2L_{MAX}$ bits of useful information about the input value of K_t . Because K_t contains
25 $(2L_{MAX}+n)$ bits of secret information and the attacker may have up to $2L_{MAX}$ bits of useful information about the initial value of K_t , there remain at least $(2L_{MAX}+n)-2L_{MAX} = n$ bits of information in K_t that are secret. The hash function H_K effectively mixes up these n bits to produce a secure new K_t during each transaction such that the attacker's information about the old K_t is no longer useful.

30 If the leak occurs at the end of the H_K computation, it could give an attacker up to L_{MAX} bits of information about the final value of H_K , yielding L_{MAX} bits of information about

the input to the subsequent transaction. This is not a problem, since the design assumes that attackers have up to L_{MAX} bits of information about K_I at the beginning of each transaction.

A third possibility is that the attacker's L_{MAX} bits of information might describe intermediates computed during the operation H_K . However, even if the attacker could obtain
5 L_{MAX} new bits of information about the input to H_K and also L_{MAX} bits of information about the output from H_K , the system would be secure, since the attacker would never have more than $2L_{MAX}$ bits of information about the input K_I or more than L_{MAX} bits of information about the output K_I . Provided that L_{MAX} bits of information from within H_K cannot reveal more than
10 L_{MAX} bits of information about the input, or more than L_{MAX} bits of information about the output, the system will be secure. This will be true unless H_K somehow compresses the input to form a short intermediate which is expanded to form the output. While hash functions whose internal states are smaller than their outputs should not be used, most cryptographic hash functions are fine.

A fourth possibility is that part or all of the leak could occur during the $A = H_A(K_I || t || R)$ calculation. The attacker's total "budget" for observations is L_{MAX} bits. If L_1 bits of leak occur during the H_K computation, an additional L_2 bits of information can leak during the $A = H_A(K_I || t || R)$ operation, where $L_2 \leq L_{MAX} - L_1$. If the second leak provides information about K_I , this is no different from leaking information about the result of the H_K computation; the attacker will still conclude the transaction with no more than L_{MAX} bits of information about
20 K_I because $L_1 + L_2 \leq L_{MAX}$. However, the second leak could reveal information about A . To keep A secure against leaks (to prevent, for example, an attacker from using a leak to capture A and using A before the legitimate user can), the size of A should include an extra L_{MAX} bits (to provide security even if $L_2 = L_{MAX}$). Like H_K , H_A should not leak information about deleted or future values of K_I that are not used in or produced by the given operation. As with the
25 similar assumptions on leaks from H_K , this limitation is primarily academic and of little practical concern, since real-world leak functions do not reveal information about deleted or not-yet-computed data. However, designers might be cautious when using unusual designs for H_A that are based on or derived from H_K , particularly if the operation $H_A(K_I || t || R)$ could reveal useful information about the result of computing $H_K(t || K_I)$.

30 B. Other Leak-Resistant Symmetric Schemes

The same basic technique of updating a key (K) with each transaction, such that leakage about a key during one transaction does not reveal useful information about a key in a

subsequent (or past) transaction, can be easily extended to other applications besides authentication.

1. Symmetric Data Verification

For example and without limitation, leak-resistant symmetric data verification is often
5 useful where a device needs to support symmetrically-signed code, data, content, or
parameter updates (all of which will, as a matter of convenience, be denoted as "data" herein).
In existing systems, a hash or MAC of the data is typically computed using a secret key and
the data is rejected if computed hash or MAC does not match a value received with the data.
For example, a MAC may be computed as $\text{HMAC}(K, \text{data})$, where HMAC is defined in "RFC
10 2104, HMAC: Keyed-Hashing for Message Authentication" by H. Krawczyk, M. Bellare,
and R. Canetti, 1997. Traditional (non-leak-resistant) designs are often vulnerable to attacks
including power consumption analysis of MAC functions and timing analysis of comparison
operations.

In an exemplary leak-resistant verification protocol, a verifying device (the "verifier")
15 maintains a counter t and a key K_t , which are initialized (for example at the factory) with $t \leftarrow 0$
and $K_t \leftarrow K_0$. Before the transaction, the verifier provides t to the device providing the
signed data (the "signer"), which also knows K_0 . The signer uses t to compute K_{t+1}' (the
prime indicating a quantity derived by the signer, rather than at the verifier) from K_0 (or K_t'
or any other available value of K_i'), using the relation $K_i' = H_K(i \parallel K_{i-1}')$, computes signature
20 $S' = \text{HMAC}(K_{t+1}', \text{data})$, and sends S' plus any other needed information (such as data or t)
to the verifier. The verifier confirms that the received value of t (if any) matches its value of
 t , and rejects the signature if it does not. If t matches, the verifier increments t and updates K_t
in its nonvolatile memory by computing $t \leftarrow t + 1$ and $K_t \leftarrow H_K(t \parallel K_t)$. In an alternative
embodiment, if the received value of t is larger than the internal value but the difference is not
25 unreasonably large, it may be more appropriate to accept the signature and perform multiple
updates to K_t (to catch up with the signer) instead of rejecting the signature outright. Finally,
the verifier computes $S = \text{HMAC}(K_t, \text{data})$ and verifies that $S = S'$, rejecting the signature if S
does not equal the value of S' received with the data.

2. Symmetric Encryption

30 Besides authentication and verification, leak-resistant symmetric cryptography can
also be tailored to a wide variety of applications and environments. For example, if data

encryption is desired instead of authentication, the same techniques as were disclosed above may be used to generate a key K_t used for encryption rather than verification.

3. Variations in Computational Implementation

In the foregoing, various applications were disclosed for the basic technique of
5 updating a key K_t in accordance with a counter and deleting old key values to ensure that
future leakage cannot reveal information about the now-deleted key. Those skilled in the art
will realize, however, that the exemplary techniques described above may be modified in
various ways without departing from the spirit and scope of the invention. For example, if
communications between the device and the server are unreliable (for example if the server
10 uses voice recognition or manual input to receive t and A), then small errors in the signature
may be ignored. (One skilled in the art will appreciate that many functions may be used to
determine whether a signature corresponds – sufficiently closely -- to its expected value.) In
another variation of the basic technique, the order of operations and of data values may be
adjusted, or additional steps and parameters may be added, without significantly changing the
15 invention. In another variation, to save on communication bandwidth or memory, the high
order bits or digits of t may not need to be communicated or remembered. In another
variation, as a performance optimization, devices need not recompute K_t from K_0 with each
new transaction. For example, when a transaction succeeds, the server can discard K_0 and
maintain the validated version of K_t . In another variation, if bi-directional authentication is
20 required, the protocol can include a step whereby the server can authenticate itself to the user
(or user's token) after the user's authentication is complete. In another variation, if the server
needs to be secured against leaks as well (as in the case where the role of "server" is played
by an ordinary user), it can maintain its own counter t . In each transaction, the parties agree
to use the larger of their two t values, where the device with the smaller t value performs extra
25 updates to K_t to synchronize t . In an alternate embodiment for devices that contain a clock
and a reliable power source (e.g., battery), the update operation may be performed
periodically, for example by computing $K_t \leftarrow H_K(t \parallel K_t)$ once per second. The token uses the
current K_t to compute $A = H_A(K_t \parallel t \parallel R)$ or, if the token does not have any means for
receiving R , it can output $A = H_A(K_t)$. The server can use its clock and local copy of the
30 secret to maintain its own version of K_t , which it can use to determine whether received
values of A are recent and correct. All of the foregoing show that the method and apparatus
of the present invention can be implemented using numerous variations and modifications to

the exemplary embodiments described herein, as would be understood by one skilled in the art.

III. Asymmetric Cryptographic Protocols

The foregoing illustrates various embodiments of the invention that may be used with symmetric cryptographic protocols. As will be seen below, still other techniques of the present invention may be used in connection with asymmetric cryptographic operations and protocols. While symmetric cryptosystems are sufficient for some applications, asymmetric cryptography is required for many applications. There are several ways leak resistance can be incorporated into public key cryptosystems, but it is often preferable to have as little impact as possible on the overall system architecture. Most of the exemplary designs have thus been chosen to incorporate leak resistance into widely used cryptosystems in a way that only alters the key management device, and does not affect the certification process, certificate format, public key format, or processes for using the public key.

A. Certified Diffie-Hellman

Diffie-Hellman exponential key exchange is a widely used asymmetric protocol whereby two parties who do not share a secret key can negotiate a shared secret key. Implementations of Diffie-Hellman can leak information about the secret exponents, enabling attackers to determine the secret keys produced by those implementations. Consequently, a leak-resistant implementation of Diffie-Hellman would be useful. To understand such a leak-resistant implementation, it will be useful to first review a conventional Diffie-Hellman implementation.

1. Conventional Certified Diffie-Hellman

Typical protocols in the background art for performing certified Diffie-Hellman exponential key agreement involve two communicating users (or devices) and a certifying authority (CA). The CA uses an asymmetric signature algorithm (such as DSA) to sign certificates that specify a user's public Diffie-Hellman parameters (the prime p and generator g), public key ($p^x \bmod g$, where x is the user's secret exponent), and auxiliary information (such as the user's identity, a description of privileges granted to the certificate holder, a serial number, expiration date, etc.). Certificates may be verified by anyone with the CA's public signature verification key. To obtain a certificate, user U typically generates a secret exponent (x_u), computes his or her own public key $y_u = g^{x_u} \bmod p$, presents y_u along with any required auxiliary identifying or authenticating information (e.g., a passport) to the CA,

who issues the user a certificate C_u . Depending on the system, p and g may be unique for each user, or they may be system-wide constants (as will be assumed in the following description of Diffie-Hellman using the background art).

Using techniques of the background art, Alice and Bob can use their certificates to
 5 establish a secure communication channel. They first exchange certificates (C_{Alice} and C_{Bob}). Each verifies that the other's certificate is acceptable (e.g., properly formatted, properly signed by a trusted CA, not expired, not revoked, etc.). Because this protocol will assume that p and g are constants, they also check that the certificate's p and g match the expected values. Alice extracts Bob's public key (y_{Bob}) from C_{Bob} and uses her secret exponent (x_{Alice}) to
 10 compute $z_{\text{Alice}} = (y_{\text{Bob}})^{x_{\text{Alice}}} \bmod p$. Bob uses his secret exponent and Alice's public key to compute $z_{\text{Bob}} = (y_{\text{Alice}})^{x_{\text{Bob}}} \bmod p$. If everything works correctly, $z_{\text{Alice}} = z_{\text{Bob}}$, since:

$$\begin{aligned} z_{\text{Alice}} &= (y_{\text{Bob}})^{x_{\text{Alice}}} \bmod p \\ &= (g^{x_{\text{Bob}}})^{x_{\text{Alice}}} \bmod p \\ &= (g^{x_{\text{Alice}}})^{x_{\text{Bob}}} \bmod p \\ &= (y_{\text{Alice}})^{x_{\text{Bob}}} \bmod p \\ &= z_{\text{Bob}}. \end{aligned}$$

Thus, Alice and Bob have a shared key $z = z_{\text{Alice}} = z_{\text{Bob}}$. An attacker who pretends to be
 15 Alice but does not know her secret exponent (x_{Alice}) will not be able to compute $z_{\text{Alice}} = (y_{\text{Bob}})^{x_{\text{Alice}}} \bmod p$ correctly. Alice and Bob can positively identify themselves by showing that they correctly found z . For example, each can compute and send the other the hash of z concatenated with their own certificate. Once Alice and Bob have verified each other, they can use a symmetric key derived from z to secure their communications. (For an
 20 example of a protocol in the background art that uses authenticated Diffie-Hellman, see "The SSL Protocol Version 3.0" by A. Freier, P. Karlton, and P. Kocher, March 1996.)

2. Leak-Resistant Certified Diffie-Hellman

A satisfactory leak-resistant public key cryptographic scheme should overcome the problem that, while certification requires the public key be constant, information about the
 25 corresponding private key should not leak out of the token that contains it. In the symmetric protocol described above, the design assumes that the leak function reveals no useful information about old deleted values of K , or about future values of K , that have not yet been

computed. Existing public key schemes, however, require that implementations repeatedly perform a consistent, usually deterministic, operation using the private key. For example, in the case of Diffie-Hellman, a leak-resistant token that is compatible with existing protocols and implementations should be able to perform the secret key operation $y^x \bmod p$, while
 5 ensuring that the exponent x remains secret. The radical reshuffling of the secret provided by the hash function H_K in the symmetric approach cannot be used because the device should be able to perform the same operation consistently.

The operations used by the token to perform the private key operation are modified to add leak resistance using the following variables:

10	Register	Comment
	x_1	First part of the secret key (in nonvolatile updateable memory)
	x_2	Second part of the secret key (in nonvolatile updateable memory)
	g	The generator (not secret).
15	p	The public prime, preferably a strong prime (not secret).

The prime p and generator g may be global parameters, or may be specific to individual users or groups of users (or tokens). In either case, the certificate recipient should be able to obtain p and g securely, usually as built-in constants or by extracting them from the certificate.

To generate a new secret key, the key generation device (often but not always the
 20 cryptographic token that will contain the key) first obtains or generates p and g , where p is the prime and g is a generator mod p . If p and g are not system-wide parameters, algorithms known in the background art for selecting large prime numbers and generators may be used. It is recommended that p be chosen with $\frac{p-1}{2}$ also prime, or at least that $\phi(p)$ not be smooth. (When $\frac{p-1}{2}$ is not prime, information about x_1 and x_2 modulo small factors of $\phi(p)$ may be
 25 leaked, which is why it is preferable that $\phi(p)$ not be smooth. Note that ϕ denotes Euler's totient function.) Once p and g have been chosen, the device generates two random exponents x_1 and x_2 . The lowest-order bit of x_1 and of x_2 is not considered secret, and may be set to 1. Using p , g , x_1 , and x_2 , the device can then compute its public key as $g^{x_1 x_2} \bmod p$ and submit it, along with any required identifying information or parameters needed (e.g., p and g), to the
 30 CA for certification.

Figure 2 illustrates the process followed by the token to perform private key operations. At step 205, the token obtains the input message y , its own (non-secret) prime p , and its own secret key halves (x_1 and x_2). If x_1 , x_2 , and p are stored in encrypted and/or

authenticated form, they would be decrypted or verified at this point. At this step, the token should verify that $1 < y < p-1$. At step 210, the token uses a random number generator (or pseudorandom number generator) to select a random integer b_0 , where $0 < b_0 < p$. At step 215, the token computes $b_1 = b_0^{-1} \bmod p$. The inverse computation mod p may be performed

5 using the extended Euclidean algorithm or the formula $b_1 = b_0^{\phi(p)-1} \bmod p$. At step 220, the token computes $b_2 = b_1^{r_1} \bmod p$. At this point, b_1 is no longer needed; its storage space may be used to store b_2 . Efficient algorithms for computing modular exponentiation, widely known in the art, may be used to complete step 220. Alternatively, when a fast modular exponentiator is available, the computation b_2 may be performed using the relationship

10 $b_2 = b_0^{\phi(p)-r_1} \bmod p$. At step 225, the token computes $b_3 = b_2^{r_2} \bmod p$. At this point, b_2 is no longer needed; its storage space may be used to store b_3 . At step 230, the token computes $z_0 = b_0 y \bmod p$. At this point, y and b_0 are no longer needed; their space may be used to store r_1 (computed at step 235) and z_0 . At step 235, the token uses a random number generator to select a random integer r_1 , where $0 < r_1 < \phi(p)$ and $\gcd(r_1, \phi(p)) = 1$. (If $\frac{p-1}{2}$ is known to be

15 prime, it is sufficient to verify that r_1 is odd.) At step 240, the token updates x_1 by computing $x_1 \leftarrow x_1 r_1 \bmod \phi(p)$. The old value of x_1 is deleted and replaced with the updated value. At step 245, the token computes $r_2 = (r_1^{-1}) \bmod \phi(p)$. If $\frac{p-1}{2}$ is prime, then r_2 can be found using a modular exponentiator and the Chinese Remainder Theorem. Note that r_1 is not needed after this step, so its space may be used to store r_2 . At step 250, the token updates x_2 by

20 computing $x_2 \leftarrow x_2 r_2 \bmod \phi(p)$. The old value of x_2 should be deleted and replaced with the updated value. At step 255, the token computes $z_1 = (z_0)^{r_1} \bmod p$. Note that z_0 is not needed after this step, so its space may be used to store z_1 . At step 260, the token computes $z_2 = (z_1)^{r_2} \bmod p$. Note that z_1 is not needed after this step, so its space may be used to store z_2 . At step 265, the token finds the exponential key exchange result by computing

25 $z = z_2 b_3 \bmod p$. Finally, at step 270, the token erases and frees any remaining temporary variables.

The process shown in Figure 2 correctly computes $z = y^x \bmod p$, where $x = x_1 x_2 \bmod \phi(p)$, since:

$$\begin{aligned}
z &= z_2 b_3 \bmod p \\
&= (z_1^{x_2} \bmod p) (b_2^{x_2} \bmod p) \bmod p \\
&= ((z_0^{x_1} \bmod p)^{x_2}) ((b_1^{x_1} \bmod p)^{x_2}) \bmod p \\
&= (b_0 y \bmod p)^{x_1 x_2} (b_0^{-1} \bmod p)^{x_1 x_2} \bmod p \\
&= y^{x_1 x_2} \bmod p \\
&= y^r \bmod p.
\end{aligned}$$

The invention is useful for private key owners communicating with other users (or devices) who have certificates, and also when communicating with users who do not.

If Alice has a certificate and wishes to communicate with Bob who does not have a certificate, the protocol proceeds as follows. Alice sends her certificate (C_{Alice}) to Bob, who receives it and verifies that it is acceptable. Bob extracts y_{Alice} (along with p_{Alice} and g_{Alice} , unless they are system-wide parameters) from C_{Alice} . Next, Bob generates a random exponent x_{BA} , where $0 < x_{\text{BA}} < \phi(p_{\text{Alice}})$. Bob then uses his exponent x_{BA} and Alice's parameters to calculate $y_{\text{BA}} = (g_{\text{Alice}}^{x_{\text{BA}}}) \bmod p_{\text{Alice}}$ and the session key $z = (y_{\text{Alice}}^{x_{\text{BA}}}) \bmod p_{\text{Alice}}$. Bob sends y_{BA} to Alice, who performs the operation illustrated in Figure 2 to update her internal parameters and derive z from y_{BA} . Alice then proves that she computed z correctly, for example by sending Bob $H(z \parallel C_{\text{Alice}})$. (Alice cannot authenticate Bob because he does not have a certificate. Consequently, she does not necessarily need to verify that he computed z successfully.) Finally, Alice and Bob can use z (or, more commonly, a key derived from z) to secure their communications.

If both Alice and Bob have certificates, the protocol works as follows. First, Alice and Bob exchange certificates (C_{Alice} and C_{Bob}), and each verifies that other's certificate is valid. Alice then extracts the parameters p_{Bob} , g_{Bob} , and y_{Bob} from C_{Bob} , and Bob extracts p_{Alice} , g_{Alice} , and y_{Alice} from C_{Alice} . Alice then generates a random exponent x_{AB} where $0 < x_{\text{AB}} < \phi(p_{\text{Bob}})$, computes $y_{\text{AB}} = (g_{\text{Bob}}^{x_{\text{AB}}}) \bmod p_{\text{Bob}}$, and computes $z_{\text{AB}} = (y_{\text{Bob}}^{x_{\text{AB}}}) \bmod p_{\text{Bob}}$. Bob generates a random x_{BA} where $0 < x_{\text{BA}} < \phi(p_{\text{Alice}})$, computes $y_{\text{BA}} = (g_{\text{Alice}}^{x_{\text{BA}}}) \bmod p_{\text{Alice}}$, and computes $z_{\text{BA}} = (y_{\text{Alice}}^{x_{\text{BA}}}) \bmod p_{\text{Alice}}$. Bob sends y_{BA} to Alice, and Alice sends y_{AB} to Bob. Alice and Bob each perform the operation shown in Figure 2, where each uses the prime p from their own certificate and their own secret exponent halves (x_1 and x_2). For the message y in Figure 2, Alice uses y_{BA} (received from Bob), and Bob uses y_{AB} (received from Alice). Using the process shown in Figure 2, Alice computes z . Using z and z_{AB} (computed

previously), she can find a session key K . This may be done, for example, by using a hash function H to compute $K = H(z \parallel z_{AB})$. The value of z Bob obtains using the process shown in Figure 2 should equal Alice's z_{AB} , and Bob's z_{BA} (computed previously) should equal Alice's z . If there were no errors or attacks, Bob should thus be able to find K , e.g., by computing $K = H(z_{BA} \parallel z)$. Alice and Bob now share K . Alice can prove her identity by showing that she
5 = $H(K \parallel C_{Alice})$. Bob can prove his identity by sending Alice $H(K \parallel C_{Bob})$. Alice and Bob can then secure their communications by encrypting and authenticating using K or a key derived from K .

Note that this protocol, like the others, is provided as an example only; many
10 variations and enhancements of the present invention are possible and will be evident to one skilled in the art. For example, certificates may come from a directory, more than two parties can participate in the key agreement, key escrow functionality may be added, the prime modulus p may be replaced with a composite number, etc. Note also that Alice and Bob as they are called in the protocol are not necessarily people; they would normally be computers,
15 cryptographic devices, etc.

For leak resistance to be effective, attackers should not be able to gain new useful information about the secret variables with each additional operation unless a comparable amount of old useful information is made useless. While the symmetric design is based on the assumption that leaked information will not survive the hash operation H_K , this design
20 uses multiplication operations mod $\phi(p)$ to update x_1 and x_2 . The most common variety of leaked information, statistical information about exponent bits, is not of use to attackers in this design, as the exponent update process ($x_1 \leftarrow x_1 r_1 \bmod \phi(p)$ and $x_2 \leftarrow x_2 r_2 \bmod \phi(p)$) destroys the utility of this information. The only relevant characteristic that survives the update process is that $x_1 x_2 \bmod \phi(p)$ remains constant, so the system designer should be
25 careful to ensure that the leak function does not reveal information allowing the attacker to find new useful information about $x_1 x_2 \bmod \phi(p)$.

There is a modest performance penalty, approximately a factor of four, for the leak-resistant design as described. One way to improve performance is to remove the blinding and unblinding operations, which are often unnecessary. (The blinding operations prevent
30 attackers from correlating input values of y with the numbers processed by the modular exponentiation operation.) Alternatively or additionally, it is possible to update and reuse

values of b_0 , b_3 , r_1 , and r_2 by computing $b_0 \leftarrow (b_0)^v \bmod p$, $b_3 \leftarrow (b_3)^v \bmod p$, $r_1 \leftarrow (r_1)^w \bmod \phi(p)$, and $r_2 \leftarrow (r_2)^w \bmod \phi(p)$, where v and w are fairly short random exponents. Note that the relationship $b_3 \leftarrow b_0^{-r_1 r_2} \bmod p$ remains true when b_0 and b_3 are both raised to the power v (mod p). The relationship $r_2 = (r_1^{-1}) \bmod \phi(p)$ also remains true when r_1 and r_2 are
 5 exponentiated (mod $\phi(p)$). Other parameter update operations may also be used, such as exponentiation with fixed exponents (e.g., $v = w = 3$), or multiplication with random values and their inverses, mod p and $\phi(p)$. The time per transaction with this update process is about half that of the unoptimized leak-resistant implementation, but additional storage is required and care should be taken to ensure that b_0 , b_3 , r_1 , and r_2 will not be leaked or otherwise
 10 compromised.

It should also be noted that with this particular type of certified Diffie-Hellman, the negotiated key is the same every time any given pair of users communicate. Consequently, though the blinding operation performed using b_0 and b_3 does serve to protect the exponents, the result K can be leaked in the final step or by the system after the process is complete. If
 15 storage is available, parties could keep track of the values of y they have received (or their hashes) and reject duplicates. Alternatively, to ensure that a different result is obtained from each negotiation, Alice and Bob can generate and exchange additional exponents, w_{Alice} and w_{Bob} , for example with $0 < w < 2^{128}$ (where $2^{128} \ll p$). Alice sets $y = (y_{\text{BA}})^{w_{\text{Alice}} w_{\text{Bob}}} \bmod p$ instead of just $y = y_{\text{BA}}$, and Bob sets $y = (y_{\text{AB}})^{w_{\text{Bob}} w_{\text{Alice}}} \bmod p$ instead of $y = y_{\text{AB}}$ before
 20 performing the operation shown in Figure 2.

B. Leak-Resistant RSA

Another asymmetric cryptographic protocol is RSA, which is widely used for digital signatures and public key encryption. RSA private key operations rely on secret exponents. If information about these secret exponents leaks from an implementation, its security can be
 25 compromised. Consequently, a leak-resistant implementation of RSA would be useful.

To give RSA private key operations resistance to leaks, it is possible to divide the secret exponent into two halves such that information about either half is destroyed with each operation. These are two kinds of RSA private key operations. The first, private key signing, involves signing a message with one's own private key to produce a digital signature
 30 verifiable by anyone with one's corresponding public key. RSA signing operations involve computing $S = M^d \bmod n$, where M is the message, S is the signature (verifiable using $M = S^e$

mod n), d is the secret exponent and equals $e^{-1} \bmod \phi(n)$, and n is the modulus and equals pq , where n and e are public and p and q are secret primes, and ϕ is Euler's phi function. An RSA public key consists of e and n , while an RSA private key consists of d and n (or other representations of them). For RSA to be secure, d , $\phi(n)$, p , and q should all be secret.

5 The other RSA operation is decryption, which is used to recover messages encrypted using one's public key. RSA decryption is virtually identical to signing, since the decrypted message M is recovered from the ciphertext C by computing $M = C^d \bmod n$, where the ciphertext C was produced by computing $C = M^e \bmod n$. Although the following discussion uses variable names from the RSA signing operation, the same techniques may be applied
10 similarly to decryption.

 An exemplary leak-resistant scheme for RSA implementations may be constructed as illustrated in Figure 3. At step 300, prior to the commencement of any signing or decryption operations, the device is initialized with (or creates) the public and private keys. The device contains the public modulus n and the secret key components d_1 , d_2 , and z , and k , where k is a
15 prime number of medium-size (e.g., $0 < k < 2^{128}$) chosen at random, $z = k\phi(n)$, d_1 is a random number such that $0 < d_1 < z$ and $\gcd(d_1, z) = 1$, and $d_2 = (e^{-1} \bmod \phi(n))(d_1^{-1} \bmod z) \bmod z$. In this invention, d_1 and d_2 replace the usual RSA secret exponent d . Techniques for generating the initial RSA primes (e.g., p and q) and modulus (n) are well known in the background art. At step 305, the device computes a random prime k' of medium size (e.g., $0 < k' < 2^{128}$).
20 (Algorithms for efficiently generating prime numbers are known in the art.)

 At step 303, the device (token) receives a message M to sign (or to decrypt). At step 310, the device updates z by computing $z \leftarrow k'z$. At step 315, the device updates z again by computing $z \leftarrow z / k$. (There should be no remainder from this operation, since k divides z .) At step 320, k is replaced with k' by performing $k \leftarrow k'$. Because k' will not be used in
25 subsequent operations, its storage space may be used to hold R (produced at step 325). At step 325, the device selects a random R where $0 < R < z$ and $\gcd(R, z) = 1$. At step 330, the device updates d_1 by computing $d_1 \leftarrow d_1 R \bmod z$. At step 335, the device finds the inverse of R by computing $R' \leftarrow R^{-1} \bmod z$ using, for example, the extended Euclidean algorithm. Note that R is no longer needed after this step, so its storage space may be erased and used to hold
30 R' . At step 340, the device updates d_2 by computing $d_2 \leftarrow d_2 R' \bmod z$. At step 345, the device computes $S_0 = M^{d_1} \bmod n$, where M is the input message to be signed (or the message

to be decrypted). Note that M is no longer needed after this step, so its storage space may be used for S_0 . At step 350, the device computes $S = S_0^{d_1} \bmod n$, yielding the final signature (or plaintext if decrypting a message). Leak-resistant RSA has similar security characteristics as normal RSA; standard message padding, post-processing, and key sizes may be used. Public key operations are also performed normally (e.g., $M = S^e \bmod n$).

A simpler RSA leak resistance scheme may be implemented by splitting the exponent d into two halves d_1 and d_2 such that $d_1 + d_2 = d$. This can be achieved during key generation by choosing d_1 to be a random integer where $0 \leq d_1 \leq d$, and choosing $d_2 \leftarrow d - d_1$. To perform private key operations, the device needs d_1 and d_2 , but it does not need to contain d . Prior to each private key operation, the cryptographic device identifies which of d_1 and d_2 is larger. If $d_1 > d_2$, then the device computes a random integer r where $0 \leq r \leq d_1$, adds r to d_2 (i.e., $d_2 \leftarrow d_2 + r$), and subtracts r from d_1 (i.e., $d_1 \leftarrow d_1 - r$). Otherwise, if $d_1 \leq d_2$, then the device chooses a random integer r where $0 \leq r \leq d_2$, adds r to d_1 (i.e., $d_1 \leftarrow d_1 + r$), and subtracts r from d_2 (i.e., $d_2 \leftarrow d_2 - r$). Then, to perform the private key operation on a message M , the device computes $s_1 = M^{d_1} \bmod n$, $s_2 = M^{d_2} \bmod n$, and computes the signature $S = s_1 s_2 \bmod n$. While this approach of splitting the exponent into two halves whose sum equals the exponent can also be used with Diffie-Hellman and other cryptosystems, dividing the exponent into the product of two numbers mod $\phi(p)$ is usually preferable since the assumption that information about $d_1 + d_2$ will not leak is less conservative than the assumption that information about $x_1 x_2 \bmod \phi(p)$ will not leak. In the case of RSA, updates mod $\phi(n)$ cannot be done safely, since $\phi(n)$ must be kept secret.

When the Chinese Remainder Theorem is required for performance, it is possible to use similar techniques to add leak resistance by maintaining multiples of the secret primes (p and q) that are updated every time (e.g., multiplying by the new multiple then dividing by the old multiple). These techniques also protect the exponents (d_p and d_q) as multiples of their normal values. At the end of the operation, the result S is corrected to compensate for the adjustments to d_p , d_q , p , and q .

An exemplary embodiment maintains state information consisting of the values n , B_i , B_p , k , p_k , q_k , d_{pk} , d_{qk} , p_{lm} , and f . To convert a traditional RSA CRT private key (consisting of p , q , d_p , and d_q with $p < q$) into the new representation, a random value for k is chosen, where $0 < k < 2^{64}$. The value B_i is chosen at random where $0 < B_i < n$, and R_1 and R_2 are chosen at

random where $0 < R_1 < 2^{64}$ and $0 < R_2 < 2^{64}$. (Of course, constants such as 2^{64} are chosen as example values. It is possible, but not necessary, to place constraints on random numbers, such as requiring that they be prime.) The leak-resistant private key state is then initialized by setting $n \leftarrow pq$, $B_f \leftarrow B_i^{-d} \pmod n$, $p_k \leftarrow (k)(p)$, $q_k \leftarrow (k)(q)$, $d_{pk} \leftarrow d_p + (R_1)(p) - R_1$, $d_{qk} \leftarrow d_q + (R_2)(q) - R_2$, $p_{inv} \leftarrow k(p^{-1} \pmod q)$, and $f \leftarrow 0$.

To update the system state, first a random value α may be produced where $0 < \alpha < 2^{64}$. Then compute $p_k \leftarrow ((\alpha)(p_k)) / k$, $q_k \leftarrow ((\alpha)(q_k)) / k$, $p_{inv} \leftarrow ((\alpha)(p_{inv})) / k$, $k \leftarrow \alpha$. The exponents d_{pk} and d_{qk} may be updated by computing $d_{pk} \leftarrow d_{pk} \pm (R_3 p_k - R_3 k)$ and $d_{qk} \leftarrow d_{qk} \pm (R_4 q_k - R_4 k)$, where R_3 and R_4 can be random or constant values (even 1). The blinding factors B_i and B_f may be updated by computing $B_i = B_i^2 \pmod n$ and $B_f = B_f^2 \pmod n$, by computing new blinding factors, by exponentiating with a value other than 2, etc. Update processes should be performed as often as practical, for example before or after each modular exponentiation process. Before the update begins, a failure counter f is incremented, and when the update completes f is set to zero. If f ever exceeds a threshold value indicating too many consecutive failures, the device should temporarily or permanently disable itself. Note that if the update process is interrupted, memory values should not be left in intermediate states. This can be done by using complete reliable memory updates. If the total set of variable changes is too large for a single complete update, it is possible to store α first then do each variable update reliably which keeping track of how many have been completed.

To perform a private key operation (such as decryption or signing), the input message C is received by the modular exponentiator. Next, the value is blinded by computing $C' \leftarrow (C)(B_i) \pmod n$. The blinded input message is then used to compute modified CRT intermediates by computing $m_{pk} \leftarrow (C')^{d_{pk}} \pmod p_k$ and $m_{qk} \leftarrow (C')^{d_{qk}} \pmod q_k$. Next in the exemplary embodiment, the CRT intermediates are multiplied by k , e.g. $m_{pk} \leftarrow (k)(m_{pk}) \pmod p_k$ and $m_{qk} \leftarrow (k)(m_{qk}) \pmod q_k$. The CRT difference is then computed as $m_{pqk} = (m_{pk} [+ qk] - m_{qk}) \pmod q_k$, where the addition of q_k and/or reduction $\pmod q_k$ are optional. (The addition of q_k ensures that the result is non-negative.) The blinded result can be computed as

$$M' = \frac{(m_{pk})k + p_k \left[\left(\frac{(p_{inv})(m_{pqk})}{k} \right) \pmod q_k \right]}{k^2},$$

then the final result M is computed as $M = (M')B_f \pmod n$.

As one of ordinary skill in the art will appreciate, variant forms of the invention are possible. For example, the computational processes can be re-ordered or modified without significantly changing the invention. Some portions (such as the initial and blinding steps) can be skipped. In another example, it is also possible to use multiple blinding factors (for
 5 example, instead of or in addition to the value k).

In some cases, other techniques may also be appropriate. For example, exponent vector codings may be rechosen frequently using, for example, a random number generator. Also, Montgomery arithmetic may be performed mod j where j is a value that is changed with each operation (as opposed to traditional Montgomery implementations where j is constant
 10 with $j = 2^k$). The foregoing shows that the method and apparatus of the present invention can be implemented using numerous variations and modifications to the exemplary embodiments described herein, as would be known by one skilled in the art.

C. Leak-Resistant ElGamal Public Key Encryption and Digital Signatures

Still other asymmetric cryptographic protocols that may be improved using the
 15 techniques of the invention. For example, ElGamal and related cryptosystems are widely used for digital signatures and public key encryption. If information about the secret exponents and parameters leaks from an ElGamal implementation, security can be compromised. Consequently, leak-resistant implementations of ElGamal would be useful.

The private key in the ElGamal public key encryption scheme is a randomly selected
 20 secret a where $1 \leq a \leq p-2$. The non-secret parameters are a prime p , a generator α , and $\alpha^a \bmod p$. To encrypt a message m , one selects a random k (where $1 \leq k \leq p-2$) and computes the ciphertext (γ, δ) where $\gamma = \alpha^k \bmod p$ and $\delta = m(\alpha^a \bmod p)^k \bmod p$. Decryption is performed by computing $m = \delta(\gamma^{p-1-a}) \bmod p$. (See the Handbook of Applied Cryptography by A. Menezes, P. van Oorschot, and S. Vanstone, 1997, pages 294-298, for a description
 25 of ElGamal public-key encryption).

To make the ElGamal public-key decryption process leak-resistant, the secret
 exponent $(p-1-a)$ is stored in two halves a_1 and a_2 , such that $a_1 a_2 = (p-1-a) \bmod (p-1)$.
 When generating ElGamal parameters for this leak-resistant implementation, it is
 recommended, but not required, that p be chosen with $\frac{p-1}{2}$ prime so that $(p-1)/2$ is prime. The
 30 variables a_1 and a_2 are normally chosen initially as random integers between 0 and $(p-1)/2$.

Alternatively, it is possible to generate a first, then choose a_1 and a_2 , as by selecting a_1 relatively prime to $\phi(p)$ and computing $a_2 = (a^{-1} \bmod \phi(p))(a_1^{-1} \bmod \phi(p)) \bmod \phi(p)$.

Figure 4 illustrates an exemplary leak-resistant ElGamal decryption process. At step 405, the decryption device receives an encrypted message pair (γ, δ) . At step 410, the device selects a random r_1 where $1 \leq r_1 < \phi(p)$ and $\gcd(r_1, \phi(p)) = 1$. At step 415, the device updates a_1 by computing $a_1 \leftarrow a_1 r_1 \bmod \phi(p)$, over-writing the old value of a_1 with the new value. At step 420, the device computes the inverse of r_1 by computing $r_2 = (r_1)^{-1} \bmod \phi(p)$. Because r_1 is not used after this step, its storage space may be used to hold r_2 . Note that if $\frac{p-1}{2}$ is prime, then r_2 may also be found by finding $r_2' = r_1^{(p-1)/2-2} \bmod \frac{p-1}{2}$, and using the CRT to find $r_2 \pmod{p-1}$. At step 425, the device updates a_2 by computing $a_2 \leftarrow a_2 r_2 \bmod \phi(p)$. At step 430, the device begins the private key (decryption) process by computing $m' = \gamma^{a_1} \bmod p$. At step 435, the device computes $m = \delta (m')^{a_2} \bmod p$ and returns the message m . If verification is successful, the result equals the original message because:

$$\begin{aligned} (\delta)(m')^{a_2} \bmod p &= (m(\alpha^a)^k (\gamma^{a_1} \bmod p)^{r_1})^{a_2} \bmod p \\ &= (m\alpha^{ak} (\gamma^{a_1 r_1} \bmod \phi(p))) \bmod p \\ &= (m\alpha^{ak} ((\alpha^k \bmod p)^{a_1 r_1 \bmod \phi(p)})) \bmod p \\ &= (m\alpha^{ak} (\alpha^{-ak})) \bmod p \\ &= m \end{aligned}$$

As with the ElGamal public key encryption scheme, the private key for the ElGamal digital signature scheme is a randomly-selected secret a , where $1 \leq a \leq p-2$. The public key is also similar, consisting of a prime p , a generator α , and public parameter y where $y = \alpha^a \bmod p$. To sign a message m , the private key holder chooses or precomputes a random secret integer k (where $1 \leq k \leq p-2$ and k is relatively prime to $p-1$) and its inverse, $k^{-1} \bmod \phi(p)$. Next, the signer computes the signature (r, s) , where $r = \alpha^k \bmod p$, $s = ((k^{-1} \bmod \phi(p))(H(m) - ar)) \bmod \phi(p)$, and $H(m)$ is the hash of the message. Signature verification is performed using the public key (p, α, y) by verifying that $1 \leq r < p$ and by verifying that $y^r r^s \bmod p = \alpha^{H(m)} \bmod p$.

To make the ElGamal digital signing process leak-resistant, the token containing the private key maintains three persistent variables, a_k , w , and r . Initially, $a_k = a$ (the private exponent), $w = 1$, and $r = \alpha$. When a message m is to be signed (or during the

precomputation before signing), the token generates a random number b and its inverse $b^{-1} \bmod \phi(p)$, where b is relatively prime to $\phi(p)$ and $0 < b < \phi(p)$. The token then updates a_k , w , and r by computing $a_k \leftarrow (a_k)(b^{-1}) \bmod \phi(p)$, $w \leftarrow (w)(b^{-1}) \bmod \phi(p)$, and $r \leftarrow (r^b) \bmod p$. The signature (r, s) is formed from the updated value of r and s , where

5 $s = (w(H(m) - a_k r)) \bmod \phi(p)$. Note that a_k , w , and r are not randomized prior to the first operation, but should be randomized before exposure to possible attack, since otherwise the first operation may leak more information than subsequent ones. It is thus recommended that a dummy signature or parameter update with $a_k \leftarrow (a_k)(b^{-1}) \bmod \phi(p)$, $w \leftarrow (w)(b^{-1}) \bmod \phi(p)$, and $r \leftarrow (r^b) \bmod p$ be performed immediately after key generation. Valid signatures

10 produced using the exemplary tamper-resistant ElGamal process may be checked using the normal ElGamal signature verification procedure.

It is also possible to split all or some the ElGamal variables into two halves as part of the leak resistance scheme. In such a variant, a is replaced with a_1 and a_2 , w with w_1 and w_2 , and r with r_1 and r_2 . It is also possible to reorder the operations by performing, for example,

15 the parameter updates as a precomputation step prior to receipt of the enciphered message. Other variations and modifications to the exemplary embodiments described herein will be evident to one skilled in the art.

D. Leak-Resistant DSA

Another commonly used asymmetric cryptographic protocol is the Digital Signature

20 Algorithm (DSA, also known as the Digital Signature Standard, or DSS), which is defined in "Digital Signature Standard (DSS)," Federal Information Processing Standards Publication 186, National Institute of Standards and Technology, May 19, 1994 and described in detail in the Handbook of Applied Cryptography, pages 452 to 454. DSA is widely used for digital signatures. If information about the secret key leaks from a DSA implementation, security

25 can be compromised. Consequently, leak-resistant implementations of DSA would be useful.

In non-leak-proof systems, the private key consists of a secret parameter a , and the public key consists of (p, q, α, y) , where p is a large (usually 512 to 1024 bit) prime, q is a 160-bit prime, α is a generator of the cyclic group of order $q \bmod p$, and $y = \alpha^a \bmod p$. To sign a message whose hash is $H(m)$, the signer first generates (or precomputes) a random

30 integer k and its inverse $k^{-1} \bmod q$, where $0 < k < q$. The signer then computes the signature (r, s) , where $r = (\alpha^k \bmod p) \bmod q$, and $s = (k^{-1} \bmod q)(H(m) + ar) \bmod q$.

In an exemplary embodiment of a leak-resistant DSA signing process, the token containing the private key maintains two variables in nonvolatile memory, a_k and k , which are initialized with $a_k = a$ and $k = 1$. When a message m is to be signed (or during the precomputation before signing), the token generates a random integer b and its inverse $b^{-1} \bmod q$, where $0 < b < q$. The token then updates a_k and k by computing $a_k \leftarrow (a_k b^{-1} \bmod q)(k) \bmod q$, followed by $k \leftarrow b$. The signature (r, s) is formed from the updated values of a_k and k by computing $r = \alpha^k \bmod p$ (which may be reduced mod q), and $s = [(b^{-1}H(m) \bmod q) + (a_k r) \bmod q] \bmod q$. As indicated, when computing s , $b^{-1}H(m) \bmod q$ and $(a_k r) \bmod q$ are computed first, then combined mod q . Note that a_k and k should be randomized prior to the first operation, since the first update may leak more information than subsequent updates. It is thus recommended that a dummy signature (or parameter update) be performed immediately after key generation. Valid signatures produced using the leak-resistant DSA process may be checked using the normal DSA signature verification procedure.

IV. Other Algorithms and Applications

Still other cryptographic processes can be made leak-proof or leak-resistant, or may be incorporated into leak-resistant cryptosystems. For example, cryptosystems such as those based on elliptic curves (including elliptic curve analogs of other cryptosystems), secret sharing schemes, anonymous electronic cash protocols, threshold signatures schemes, etc. be made leak resistant using the techniques of the present invention.

Implementation details of the schemes described may be adjusted without materially changing the invention, for example by re-ordering operations, inserting steps, substituting equivalent or similar operations, etc. Also, while new keys are normally generated when a new system is produced, it is often possible to add leak resistance retroactively while maintaining or converting existing private keys.

Leak-resistant designs avoid performing repeated mathematical operations using non-changing (static) secret values, since they are likely to leak out. However, in environments where it is possible to implement a simple function (such as an exclusive OR) that does not leak information, it is possible use this function to implement more complex cryptographic operations.

While the exemplary implementations assume that the leak functions can reveal any information present in the system, designers may often safely use the (weaker) assumption

that information not used in a given operation will not be leaked by that operation. Schemes using this weaker assumption may contain a large table of precomputed subkey values, from which a unique or random subset are selected and/or updated for each operation. For example, DES implementations may use indexed permutation lookup tables in which a few
5 table elements are exchanged with each operation.

While leak resistance provides many advantages, the use of leak resistance by itself cannot guarantee good security. For example, leak-resistant cryptosystems are not inherently secure against error attacks, so operations should be verified. (Changes can even be made to the cryptosystem and/or leak resistance operations to detect errors.) Similarly, leak resistance
10 by itself does not prevent attacks that extract the entire state out of a device (e.g., $L=L_{MAX}$). For example, traditional tamper resistance techniques may be required to prevent attackers from staining ROM or EEPROM memory cells and reading the contents under a microscope. Implementers should also be aware of interruption attacks, such as those that involve disconnecting the power or resetting a device during an operation, to ensure that secrets will
15 not be compromised or that a single leaky operation will not be performed repeatedly. (As a countermeasure, devices can increment a counter in nonvolatile memory prior to each operation, and reset or reduce the counter value when the operation completes successfully. If the number of interrupted operations since the last successful update exceeds a threshold value, the device can disable itself.) Other tamper resistance mechanisms and techniques,
20 such as the use of fixed-time and fixed-execution path code or implementations for critical operations, may need to be used in conjunction with leak resistance, particularly for systems with a relatively low self-healing rate (e.g., L_{MAX} is small).

Leak-resistant algorithms, protocols, and devices may be used in virtually any application requiring cryptographic security and secure key management, including without
25 limitation: smartcards, electronic cash, electronic payments, funds transfer, remote access, timestamping, certification, certificate validation, secure e-mail, secure facsimile, telecommunications security (voice and data), computer networks, radio and satellite communications, infrared communications, access control, door locks, wireless keys, biometric devices, automobile ignition locks, copy protection devices, payment systems,
30 systems for controlling the use and payment of copyrighted information, and point of sale terminals.

The foregoing shows that the method and apparatus of the present invention can be implemented using numerous variations and modifications to the exemplary embodiments described herein, as would be known by one skilled in the art. Thus, it is intended that the scope of the present invention be limited only with regard to the claims below.

WHAT IS CLAIMED IS:

- 1 1. A method for implementing RSA with the Chinese Remainder Theorem for use in a
2 cryptographic system, with resistance to leakage attacks against said cryptographic
3 system, comprising the steps of:
- 4 (a) obtaining a representation of an RSA private key corresponding to an RSA
5 public key, said private key characterized by secret factors p and q ;
 - 6 (b) storing said representation of said private key in a memory;
 - 7 (c) obtaining a message for use in an RSA cryptographic operation;
 - 8 (d) computing a first modulus, corresponding to a multiple of p , where the value
9 of said multiple of p and the value of said multiple of p divided by p are both
10 unknown to an attacker of said cryptographic system;
 - 11 (e) reducing said message modulo said first modulus;
 - 12 (f) performing modular exponentiation on the result of step (e);
 - 13 (g) computing a second modulus, corresponding to a multiple of q , where the
14 value of said multiple of q and the value of said multiple of q divided by q are
15 both unknown to an attacker of said cryptographic system;
 - 16 (h) reducing said message modulo said second modulus;
 - 17 (i) performing modular exponentiation on the result of step (h);
 - 18 (j) combining the results of said steps (e) and (h) to produce a result which, if
19 operated on with an RSA public key operation using said RSA public key,
20 yields said message; and
 - 21 (k) repeating steps (c) through (j) a plurality of times using different values for
22 said multiple of p and for said multiple of q .
- 1 2. The method of claim 1 where:
- 2 (i) said step (b) includes storing an exponent d_p of said RSA private key in said
3 memory as a plurality of parameters;
 - 4 (ii) an arithmetic function of at least one of said plurality of parameters is
5 congruent to d_p , modulo $(p-1)$;
 - 6 (iii) none of said parameters comprising said stored d_p is equal to d_p ;
 - 7 (iv) an exponent used in said step (f) is at least one of said parameters;
 - 8 (v) at least one of said parameters in said memory changes with said repetitions of
9 said steps (c) through (j).

- 1 3. The method of claim 2 where said plurality of parameters includes a first parameter
2 equal to said d_p , plus a multiple of $\phi(p)$, and also includes a second parameter equal
3 to a multiple of $\phi(p)$, where ϕ denotes Euler's totient function.
- 1 4. The method of claim 1 where the value of said multiple of p divided by p is equal to
2 the value of said multiple of q divided by q .
- 1 5. The method of claim 1 where said multiple of p and said multiple of q used in said
2 steps (c) through (j) are updated and modified in said memory after said step (b).
- 1 6. The method of claim 1 performed in a smart card.
- 1 7. The method of claim 1 where at least two of said steps are performed in an order other
2 than (a) through (k)
- 1 8. A method for implementing RSA for use in a cryptographic system, with resistance to
2 leakage attacks against said cryptographic system, comprising the steps of:
3 (a) obtaining an RSA private key corresponding to an RSA public key, said RSA
4 public key having an RSA modulus n ;
5 (b) storing said private key in a memory in a form whereby a secret parameter of
6 said key is stored as an arithmetic combination of $\phi(x)$ and a first at least one
7 key masking parameter, where
8 (i) an operand x in said $\phi(x)$ is an exact multiple of at least one factor of
9 said modulus n of said RSA public key; and
10 (ii) said first key masking parameter is unknown to an attacker of said
11 cryptosystem;
12 (iii) a representation of said first key masking parameter is stored in said
13 memory;
14 (iv) ϕ denotes Euler's totient function;
15 (c) receiving a message;
16 (d) deriving an RSA input from said message;
17 (e) performing modular exponentiation to raise said RSA input to a power
18 dependent on said secret parameter, modulo an RSA modulus stored in said
19 memory, to produce an RSA result such that said RSA result raised to the

- 20 power of the public exponent of said RSA public key, modulo the modulus of
21 said RSA public key, equals said RSA input;
- 22 (f) updating said secret parameter in said memory by:
- 23 (i) modifying said first key masking parameter to produce a new key
24 masking parameter, where said modification is performed in a manner
25 such that an attacker with partial useful information about said first key
26 masking parameter has less useful information about said new key
27 masking parameter; and
- 28 (ii) using said new key masking parameter to update said secret parameter
29 in said memory;
- 30 (g) repeating steps (d) through (f) a plurality of times, where the power used for
31 each of said modular exponentiation steps (e) is different.
- 1 9. The method of claim 8 where said operand x in said $\phi(x)$ corresponds to said RSA
2 modulus n of said RSA public key.
- 1 10. The method of claim 8 where said operand x in said $\phi(x)$ corresponds to a prime
2 factor of said RSA modulus n of said RSA public key, and where said modular
3 exponentiation of said step (e) is performed using the Chinese Remainder Theorem.
- 1 11. A method for implementing exponential key exchange for use in a cryptographic
2 system, with resistance to leakage attacks against said cryptographic system,
3 comprising the steps of:
- 4 (a) obtaining, and storing in a memory, exponential key exchange parameters g
5 and p , and a plurality of secret exponent parameters on which an arithmetic
6 relationship may be computed to produce an exponent x ;
- 7 (b) using a key update transformation to produce a plurality of updated secret
8 exponent parameters while maintaining said arithmetic relationship
9 thereamong;
- 10 (c) receiving a public value y from a party with whom said key exchange is
11 desired;
- 12 (d) using said updated secret exponent parameters to perform a cryptographic
13 computation yielding an exponential key exchange result $z = y^x \text{ mod } p$;
- 14 (e) using said result z to secure an electronic communication with said party; and
15 (f) performing said steps (b), (c), (d), and (e) in a plurality of transactions.

- 1 12. The method of claim 11 where each of said transactions involves a different said
2 party.
- 1 13. The method of claim 11 where said arithmetic relationship is such that said
2 exponential key exchange result is a product of certain of said secret exponent
3 parameters, both before and after said step (b).
- 1 14. The method of claim 11 where said key update transformation includes choosing a
2 random key update value r ; and where said step (b) includes multiplying one of said
3 secret exponent parameters by r and another of said secret exponent parameters by an
4 inverse of r , said multiplication being performed modulo $\phi(p)$, where ϕ is Euler's
5 totient function.
- 1 15. The method of claim 11 where said key update transformation includes choosing a
2 random key update value r ; and where said step (b) includes adding r to one of said
3 secret exponent parameters and subtracting r from another of said secret exponent
4 parameters.
- 1 16. The method of claim 15 where said secret exponent parameters include two values x_1
2 and x_2 such that x_1+x_2 is congruent to x , modulo $\phi(p)$, where ϕ is Euler's totient
3 function, and where said step of performing said cryptographic computation yielding
4 said exponential key exchange result includes computing $z_1 = y^{x_1} \bmod p$, $z_2 = y^{x_2}$
5 $\bmod p$, and $z = z_1 z_2 \bmod p$.
- 1 17. A cryptographic token configured to perform cryptographic operations using a secret
2 key in a secure manner, comprising:
3 (a) an interface configured to receive power from a source external to said token;
4 (b) a memory containing said secret key;
5 (c) a processor:
6 (i) configured to receive said power delivered via said interface;
7 (ii) configured to perform said processing using said secret key from said
8 memory;
9 (d) said token having a power consumption characteristic:
10 (i) that is externally measurable; and

- 11 (ii) that varies over time in a manner measurably correlated with said
12 cryptographic operations; and
- 13 (e) a source of unpredictable information usable in said cryptographic operations
14 to make determination of said secret key infeasible from external
15 measurements of said power consumption characteristic.
- 1 18. The cryptographic token of claim 17, in the form of a secure microprocessor.
- 1 19. The cryptographic token of claim 17, in the form of a smart card.
- 1 20. The cryptographic token of claim 19, wherein said cryptographic operations
2 performed by said smart card enable a holder thereof to decrypt an encrypted
3 communication received via a computer network.
- 1 21. The cryptographic token of claim 19, wherein said smart card is configured to store
2 value in an electronic cash scheme.
- 1 22. The cryptographic token of claim 21, wherein said cryptographic operations include
2 authenticating that a balance of said stored value has been decreased.
- 1 23. The cryptographic token of claim 17, wherein said cryptographic operations include
2 asymmetric private key operations.
- 1 24. The cryptographic token of claim 23 wherein said cryptographic operations include
2 exponential key agreement operations.
- 1 25. The cryptographic token of claim 23, wherein said cryptographic operations include
2 DSA signing operations.
- 1 26. The cryptographic token of claim 23, wherein said cryptographic operations include
2 ElGamal private key operations.
- 1 27. The cryptographic token of claim 23, wherein said asymmetric private key operations
2 include RSA private key operations.

- 1 28. The cryptographic token of claim 27 wherein said private key operations include
2 Chinese Remainder Theorem operations.
- 1 29. The cryptographic token of claim 17, wherein said cryptographic operations include
2 symmetric encryption operations.
- 1 30. The cryptographic token of claim 17, wherein said cryptographic operations include
2 symmetric decryption operations.
- 1 31. The cryptographic token of claim 17, wherein said cryptographic operations include
2 symmetric authentication operations using said secret key.
- 1 32. The cryptographic token of claim 17, wherein said cryptographic operations include
2 authenticating a payment.
- 1 33. The cryptographic token of claim 17, wherein said cryptographic operations include
2 securing a broadcast communications signal.
- 1 34. The cryptographic token of claim 33, wherein said cryptographic operations include
2 decrypting a satellite broadcast.
- 1 35. A method for securely managing and using a private key in a computing environment
2 where information about said private key may leak to attackers, comprising the steps
3 of:
4 (a) using a first private key, complementary to a public key, to perform first
5 asymmetric cryptographic operation;
6 (b) reading at least a portion of said first private key from a memory;
7 (c) transforming said read portion of said first private key to produce a second
8 private key:
9 (i) said second private key usable to perform a subsequent asymmetric
10 cryptographic operation in a manner that remains complementary to
11 said public key, and
12 (ii) said transformation enabling said asymmetric cryptographic operations
13 to be performed in a manner such that information leaked during said

14 first asymmetric cryptographic operation does not provide
15 incrementally useful information about said second private key;
16 (d) obtaining a datum;
17 (e) using said second private key to perform said subsequent asymmetric
18 cryptographic operation on said datum.

1 36. The method of claim 35 where said asymmetric cryptographic operation includes a
2 digital signing operation.

1 37. The method of claim 36 where said signing operation is an RSA operation.

1 38. The method of claim 36 where said signing operation is an DSA operation.

1 39. The method of claim 36 where said signing operation is an ElGamal operation.

1 40. The method of claim 35 where said asymmetric cryptographic operation includes a
2 decryption operation.

1 41. The method of claim 40 where said decryption operation is an RSA operation.

1 42. The method of claim 40 where said decryption operation is an ElGamal operation.

1 43. The method of claim 35 where at least two of said steps are performed in an order
2 different than (a), (b), (c), (d), (e).

1 44. The method of claim 35 further comprising the step, after at least said step (c), of
2 replacing said private key in said memory with said second private key.

1 45. The method of claim 35, performed in a smart card.

1 46. The method of claim 35, further comprising the steps of: prior to at least said step (c),
2 incrementing a counter stored in a nonvolatile memory and verifying that said counter
3 has not exceeded a threshold value; and after at least said step (c) has completed
4 successfully, decreasing a value of said counter.

- 1 47. A method for performing cryptographic transactions while protecting a stored
2 cryptographic key against compromise due to leakage attacks, comprising the steps
3 of:
4 (a) retrieving a stored private cryptographic key stored in a memory, said stored
5 key having been used in a previous cryptographic transaction;
6 (b) using a first cryptographic function to derive from said stored key an updated
7 key, about which useful information about said stored key obtained through
8 monitoring of leaked information is effectively uncorrelated to said updated
9 key;
10 (c) replacing said stored key in said memory with said updated key;
11 (d) using an asymmetric cryptographic function, cryptographically processing a
12 datum with said updated key; and
13 (e) sending said cryptographically processed datum to an external device having a
14 public key corresponding to said stored key.
- 1 48. The method of claim 47 where said stored key includes a first plurality of parameters,
2 and where said updated key includes a second plurality of parameters.
- 1 49. The method of claim 48 where no secret value within said first plurality of parameters
2 is included within said second plurality of parameters.
- 1 50. The method of claim 49 where said first plurality of parameters is different than said
2 second plurality of parameters, yet a predetermined relationship among said first
3 plurality of parameters is also maintained among said second plurality of parameters.
- 1 51. The method of claim 50 where said relationship among said plurality of parameters is
2 an arithmetic function involving at least two of said plurality of parameters.
- 1 52. The method of claim 51 where said arithmetic function is the sum of said parameters.
- 1 53. The method of claim 51 where said relationship includes a bitwise combination of
2 said parameters.
- 1 54. The method of claim 53 where said bitwise combination is an exclusive OR.

- 1 55. The method of claim 47 where said step (b) includes using pseudorandomness to
2 derive said updated key.
- 1 56. A method for implementing a private key operation for an asymmetric cryptographic
2 system with resistance to leakage attacks against said cryptographic system,
3 comprising the steps of:
4 (a) encoding a portion of a private key as at least two component parts, such that
5 an arithmetic function of said parts yields said portion;
6 (b) modifying said component parts to produce updated component parts, but
7 where said arithmetic function of said updated parts still yields said private
8 key portion;
9 (c) obtaining a message for use in an asymmetric private key cryptographic
10 operation;
11 (d) separately applying said component parts to said message to produce an
12 intermediate result;
13 (e) deriving a final result from said intermediate result such that said final result is
14 a valid result of applying said private key to said message; and
15 (f) repeating steps (b) through (e) a plurality of times.
- 1 57. The method of claim 56 where said private key portion includes an exponent, and
2 where said intermediate result represents the result of raising said message to the
3 power of said exponent, modulo a second key portion.
- 1 58. The method of claim 57 where said private key operation is configured for use with
2 an RSA cryptosystem.
- 1 59. The method of claim 57 where said private key operation is configured for use with
2 an ElGamal cryptosystem.
- 1 60. The method of claim 56 where said private key operation is configured for use with a
2 DSA cryptosystem.
- 1 61. The method of claim 60 where said private key is represented by secret parameters a_k
2 and k whose product, modulo a predetermined DSA prime q for said private key,
3 yields said private key portion.

- 1 62. The method of claim 56 implemented in a smart card.
- 1 63. The method of claim 56 where said private key is configured for use with an elliptic
2 curve cryptosystem.
- 1 64. A method for performing cryptographic transactions in a cryptographic token while
2 protecting a stored cryptographic key against compromise due to leakage attacks,
3 including the steps of:
4 (a) retrieving said stored key from a memory;
5 (b) cryptographically processing said key, to derive an updated key, by executing
6 a cryptographic update function that:
7 (i) prevents partial information about said stored key from revealing
8 useful information about said updated key, and
9 (ii) also prevents partial information about said updated key from
10 revealing useful information about said stored key;
11 (c) replacing said stored key in said memory with said updated key;
12 (d) performing a cryptographic operation using said updated key; and
13 (e) repeating steps (a) through (d) a plurality of times.
- 1 65. The method of claim 64 where said cryptographic update function of said step (b)
2 includes a one-way hash operation.
- 1 66. The method of claim 64 where said cryptographic operation of said step (d) is a
2 symmetric cryptographic operation; and comprising the further step of sending a
3 result of said cryptographic operation to a party capable of rederiving said updated
4 key.
- 1 67. The method of claim 64 further comprising the step, prior to said step (a), of receiving
2 from a second party a symmetric authentication code and a parameter; and said where
3 said step (b) includes iterating a cryptographic transformation a number of times
4 determined from said parameter; and where said step (d) includes performing a
5 symmetric message authentication code verification operation.

- 6 68. he method of claim 66 where said step (d) of performing said cryptographic operation
7 includes using said updated key to encrypt a datum.
- 1 69. The method of claim 66 where said updated key contains unpredictable information
2 such that said updated key is not stored in its entirety anywhere outside of said
3 cryptographic token; and where the result of said step (d) is independent of said
4 unpredictable information.
- 1 70. The method of claim 64 where said step (c) of replacing said stored key includes:
2 (i) explicitly erasing a region of said memory containing said stored key; and
3 (ii) storing said updated key in said region of memory.
- 1 71. The method of claim 64 performed within a smart card.

FIG. 1

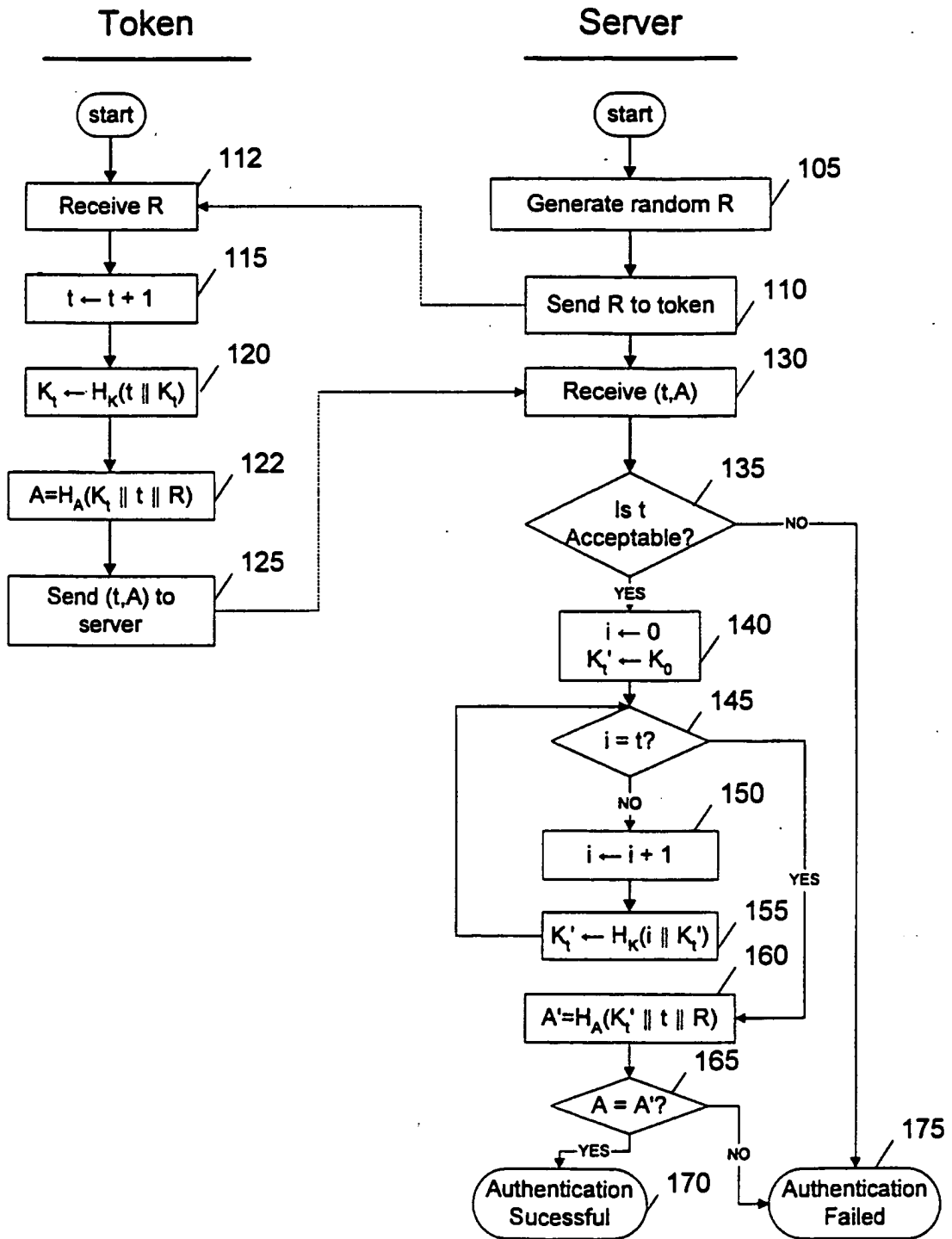


FIG. 2

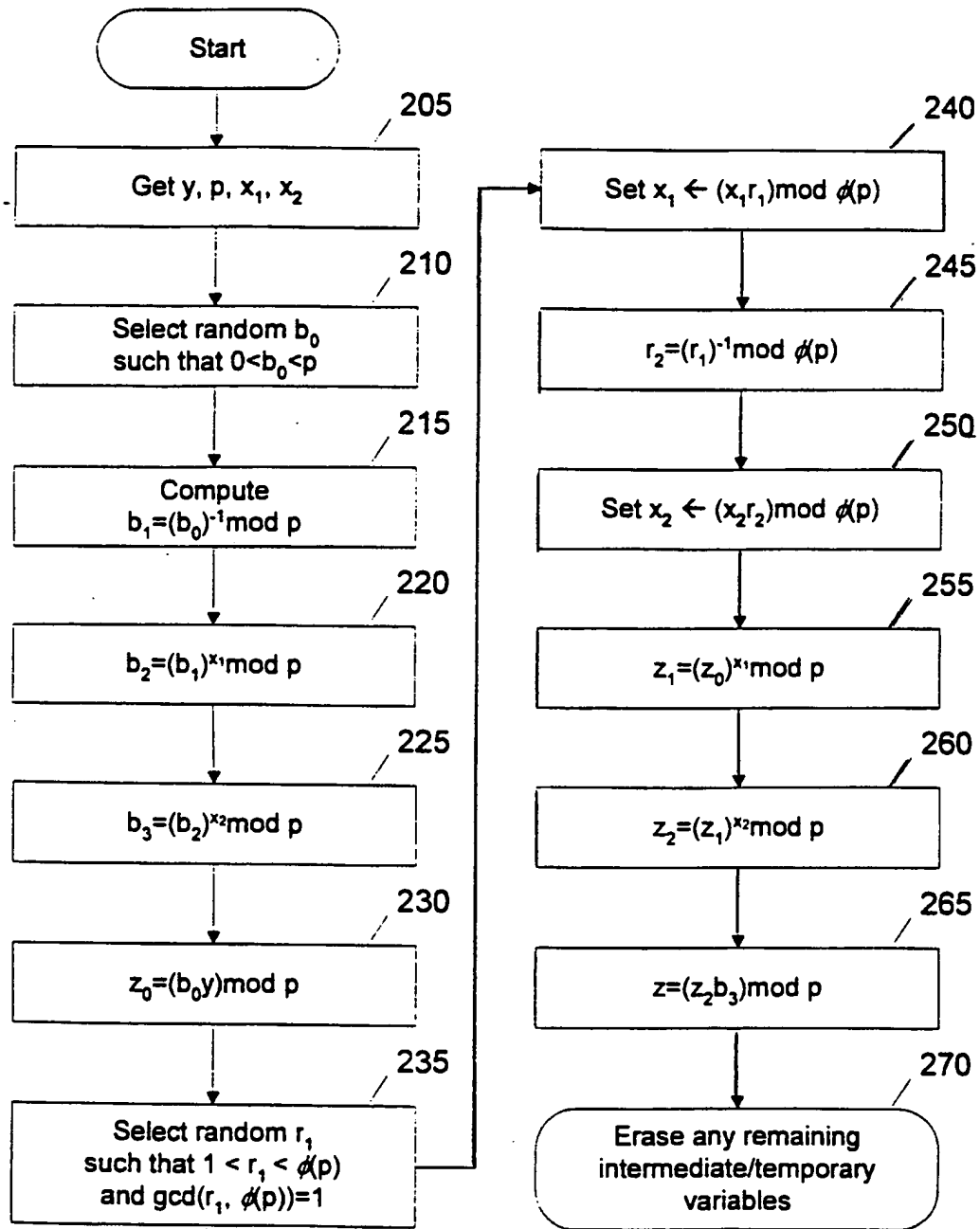


FIG. 3

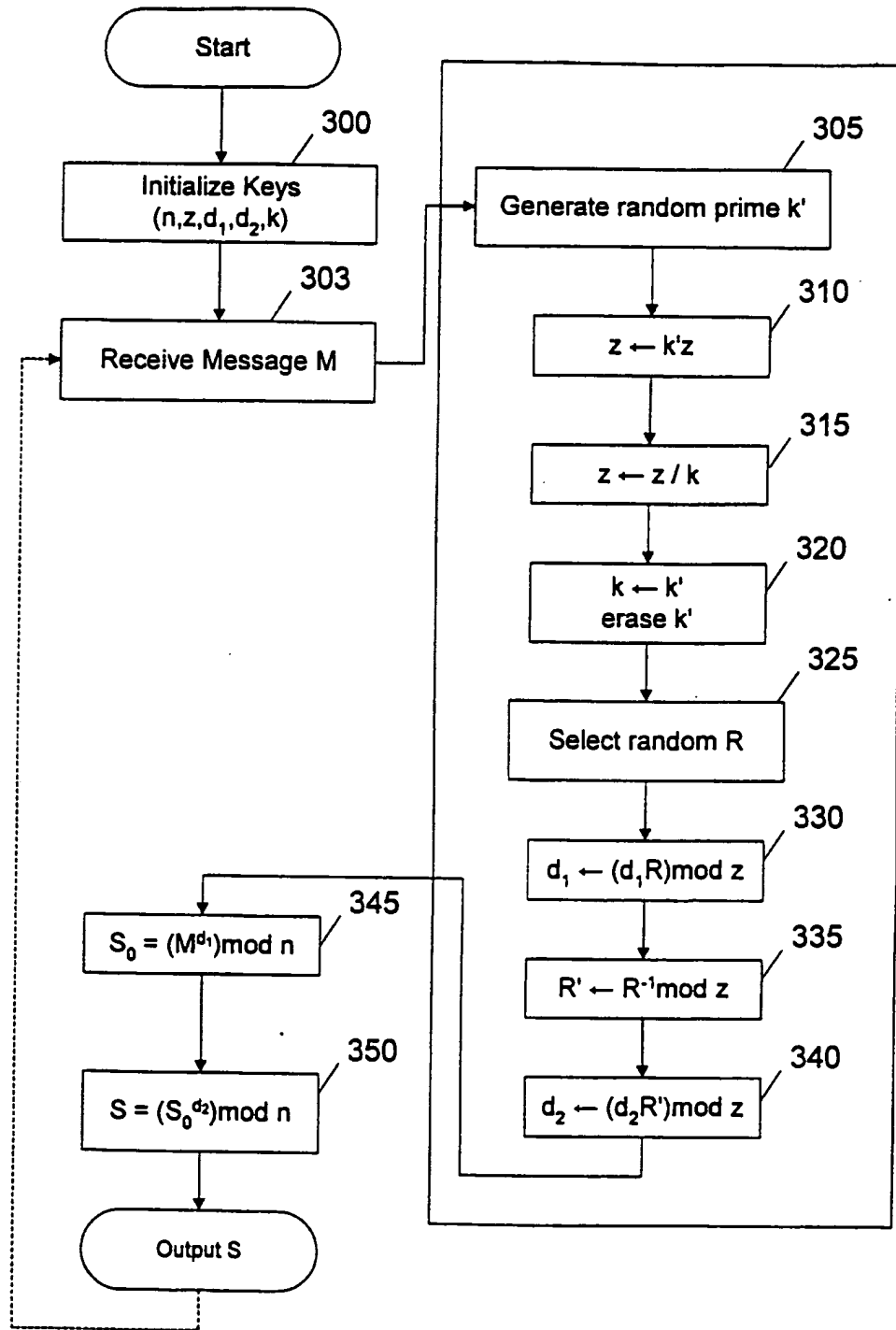
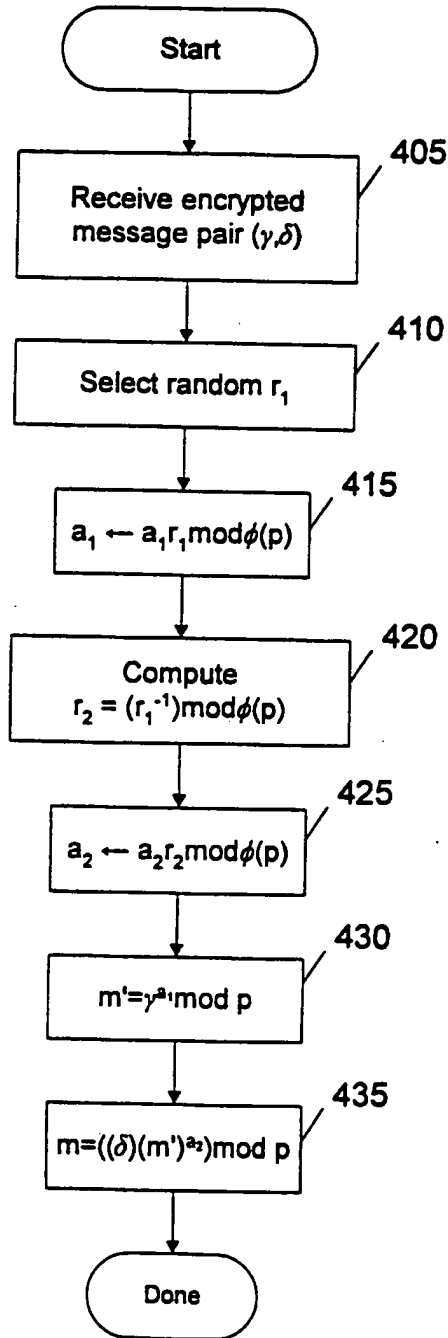


FIG. 4



INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/27896

A. CLASSIFICATION OF SUBJECT MATTER IPC(6) :HO4 L 9/30 US CL :380/30,49 According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED Minimum documentation searched (classification system followed by classification symbols) U.S. : 380/30,49 Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched Electronic data base consulted during the international search (name of data base and, where practicable, search terms used) Please See Extra Sheet.		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 4,799,258 A (DAVIES et al) 17 January 1989, abstract, col.4, lines 43-50, col.7,lines 15-33, col.8,lines 12-19	17-23,25- 45
Y	US 5,546,463 A (CAPUTO et al) 13 August 1996, abstract, col.2, lines, 60-65, col.5, lines 39-50,53-58, col.6, lines 7-12	17-23,25-45
A,P	US 5,848,159 A (COLLINS et al.) 08 December 1998, abstract, col.1, lines 56-67, col.4, lines 33-44, col.5, lines 52-67, col.6, lines 24-30	1-16,46-71
<input type="checkbox"/> Further documents are listed in the continuation of Box C.		<input type="checkbox"/> See patent family annex.
* Special categories of cited documents:	*T	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
A document defining the general state of the art which is not considered to be of particular relevance	*X*	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
E earlier document published on or after the international filing date	*Y*	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
L document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	*A*	document member of the same patent family
O document referring to an oral disclosure, use, exhibition or other means		
P document published prior to the international filing date but later than the priority date claimed		
Date of the actual completion of the international search 30 MARCH 1999	Date of mailing of the international search report 06 MAY 1999	
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-0040	Authorized officer GAIL HAYES <i>Regonia Zogger</i> Telephone No. (703) 305-9711	

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

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/27896

B. FIELDS SEARCHED

Electronic data bases consulted (Name of data base and where practicable terms used):

APS

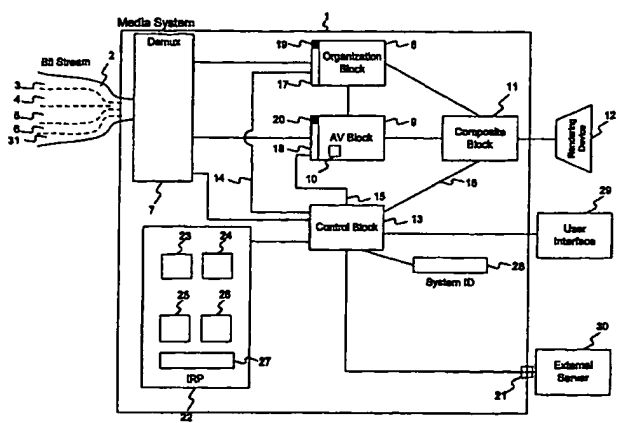
search terms: token, smart card, tamper proof, tamper resistant, leak-resistant, RSA, public key, private key, chinese remainder theorem, diffie hellman, dsa, des



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

<p>(51) International Patent Classification ⁶ : H04N 7/167, G06F 1/00</p>	<p>A1</p>	<p>(11) International Publication Number: WO 99/48296 (43) International Publication Date: 23 September 1999 (23.09.99)</p>
<p>(21) International Application Number: PCT/US99/05734 (22) International Filing Date: 16 March 1999 (16.03.99) (30) Priority Data: 60/078,053 16 March 1998 (16.03.98) US (71) Applicant: INTERTRUST TECHNOLOGIES CORPORATION [US/US]; 460 Oakmead Parkway, Sunnyvale, CA 94086 (US). (72) Inventors: SHAMOON, Talal, G.; 533 Bryant Street #5, Palo Alto, CA 94301 (US). HILL, Ralph, D.; 224 Dover Street, Los Gatos, CA 94032 (US). RADCLIFFE, Chris, D.; 3654 Farm Hill Boulevard, Redwood City, CA 94061 (US). HWA, John, P.; 503 Lower Vinters Circle, Fremont, CA 94539 (US). (74) Agents: GARRETT, Arthur, S. et al.; Finnegan, Henderson, Farabow, Garrett & Dunner, L.L.P., 1300 I Street, Washington, DC 20005-3315 (US).</p>		<p>(81) Designated States: CA, CN, JP, KR, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE). Published <i>With international search report.</i> <i>Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.</i></p>

(54) Title: METHODS AND APPARATUS FOR CONTINUOUS CONTROL AND PROTECTION OF MEDIA CONTENT



(57) Abstract

A novel method and apparatus for protection of streamed media content is disclosed. The apparatus includes control means for governance of content streams or objects, decryption means for decrypting content streams or objects under control of the control means, and feedback means for tracking actual use of content streams or objects. The control means may operate in accordance with rules received as part of the streamed content, or through a side-band channel. The rules may specify allowed uses of the content, including whether or not the content can be copied or transferred, and whether and under what circumstances received content may be "checked out" of one device and used in a second device. The rules may also include or specify budgets, and a requirement that audit information be collected and/or transmitted to an external server. The apparatus may include a media player designed to call plugins to assist in rendering content. A "trust plugin" and its use are disclosed so that a media player designed for use with unprotected content may render protected content without the necessity of requiring any changes to the media player. The streamed content may be in a number of different formats, including MPEG-4, MP3, and the RMFF format.

FOR THE PURPOSES OF INFORMATION ONLY

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

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece	ML	Mali	TR	Turkey
BG	Bulgaria	HU	Hungary	MN	Mongolia	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MR	Mauritania	UA	Ukraine
BR	Brazil	IL	Israel	MW	Malawi	UG	Uganda
BY	Belarus	IS	Iceland	MX	Mexico	US	United States of America
CA	Canada	IT	Italy	NE	Niger	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NL	Netherlands	VN	Viet Nam
CG	Congo	KE	Kenya	NO	Norway	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NZ	New Zealand	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	PL	Poland		
CM	Cameroon	KR	Republic of Korea	PT	Portugal		
CN	China	KZ	Kazakstan	RO	Romania		
CU	Cuba	LC	Saint Lucia	RU	Russian Federation		
CZ	Czech Republic	LI	Liechtenstein	SD	Sudan		
DE	Germany	LK	Sri Lanka	SE	Sweden		
DK	Denmark	LR	Liberia	SG	Singapore		
EE	Estonia						

METHODS AND APPARATUS FOR CONTINUOUS CONTROL AND PROTECTION OF MEDIA CONTENT

5 **FIELD OF THE INVENTION**

This invention relates generally to computer and/or electronic security. More particularly, this invention relates to systems and methods for protection of information in streamed format.

BACKGROUND

10 Streaming digital media consists generally of sequences of digital information received in a "stream" of packets, and designed to be displayed or rendered. Examples include streamed audio content, streamed video, etc.

Digital media streams are becoming an increasingly significant means of content delivery, and form the basis for several adopted, proposed or de facto standards. The acceptance of this format, however, has been retarded by the ease with which digital media streams can be copied and improperly disseminated, and the consequent reluctance of content owners to allow significant properties to be distributed through streaming digital means. For this reason, there is a need for a methodology by which digital media streams can be protected.

20 **SUMMARY OF THE INVENTION**

Consistent with the invention, this specification describes a new architecture for protection of information provided in streamed format. This architecture is described in the context of a generic system which resembles a system to render content encoded pursuant to the MPEG-4 specification (ISO/IEC 14496.1), though with certain modifications, and with the proviso that the described system may differ from the MPEG-4 standard in certain respects. A variety of different embodiments is described, including an MPEG-4 embodiment and a system designed to render content encoded pursuant to the MP3 specification (ISO/IEC TR 11172).

30 According to aspects of the invention, this architecture involves system design aspects and information format aspects. System design aspects include the incorporation of content protection functionality, control functionality, and feedback enabling control functionality to monitor the activities of the system. Information format aspects include the incorporation of rule/control information into information streams, and the protection of content through mechanisms such as encryption and watermarking.

- 2 -

Systems and methods consistent with the present invention perform content protection and digital rights management. A streaming media player consistent with the present invention includes a port designed to accept a digital bit stream. The digital bit stream includes content, which is encrypted at least in part, and a secure container including control information designed to control use of the content, including at least one key suitable for decryption of at least a portion of the content. The media player also includes a control arrangement including a means for opening secure containers and extracting cryptographic keys, and means for decrypting the encrypted portion of the content.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate an embodiment of the invention and, together with the description, serve to explain the advantages and principles of the invention. In the drawings,

FIG. 1 shows a generic system consistent with the present invention;

FIG. 2 shows an exemplary Header 201 consistent with the present invention;

FIG. 3 shows a general encoding format consistent with the present invention;

FIG. 4 illustrates one manner for storing a representation of a work consistent with the present invention;

FIG. 5 shows an example of a control message format;

FIG. 6 is a flow diagram illustrating one embodiment of the steps which take place using the functional blocks of FIG. 1;

FIG. 7 illustrates a form wherein the control messages may be stored in Control Block 13;

FIG. 8 shows MPEG-4 System 801 consistent with the present invention;

FIG. 9 shows an example of a message format;

FIG. 10 illustrates an IPMP table consistent with the present invention;

FIG. 11 illustrates a system consistent with the present invention;

FIG. 12 illustrates one embodiment of the DigiBox format;

FIG. 13 shows an example of a Real Networks file format (RMFF);

FIG. 14 shows an RNPF format consistent with the present invention;

FIG. 15 illustrates the flow of changes to data in the Real Networks file format in an architecture consistent with the present invention;

FIG. 16 illustrates a standard Real Networks architecture;

FIG. 17 shows an exemplary architecture in which a trust plugin operates within the overall Real Networks architecture;

SUBSTITUTE SHEET (RULE 26)

- 3 -

FIG. 18 shows a bit stream format consistent with the principles of the present invention;

FIG. 19 shows one embodiment of protection applied to the MP3 format;

FIG. 20 illustrates one embodiment of an MP3 player designed to process and render protected content;

FIG. 21 illustrates the flow of data in one embodiment in which a protected MPEG-4 file may be created consistent with the present invention;

FIG. 22 illustrates the flow of data in one embodiment in which control may be incorporated into an existing MPEG-4 stream consistent with the present invention;

FIG. 23 shows a system consistent with the principles of the present invention;

FIG. 24 shows a system consistent with the principles of the present invention;

FIG. 25 shows an example of an aggregate stream consistent with the present invention;

FIG. 26 illustrates a Header CMPO 2601 consistent with the present invention;

FIG. 27 shows exemplary Content Management Protection Objects consistent with the principles of the present invention; and

FIG. 28 shows an example of a CMPO Data Structure 2801 consistent with the present invention.

DETAILED DESCRIPTION

Reference will now be made in detail to implementations consistent with the principles of the present invention as illustrated in the accompanying drawings.

The following U.S. patents and applications, each of which is assigned to the assignee of the current application, are hereby incorporated in their entirety by reference: Ginter, et al., "Systems and Methods for Secure Transaction Management and Electronic Rights Protection," U.S. Patent Application Serial No. 08/964,333, filed on November 4, 1997 ("Ginter '333"); Ginter, et al., "Trusted Infrastructure Support Systems, Methods and Techniques for Secure electronic commerce, Electronic Transactions, Commerce Process Control Automation, Distributed Computing, and Rights Management," U.S. Patent Application Serial No. 08/699,712, filed on August 12, 1996 ("Ginter '712"); Van Wie, et al., "Steganographic Techniques for Securely Delivering Electronic Digital Rights Management Information Over Insecure Communications Channels, U.S. Patent Application Serial No. 08/689,606, filed on August 12, 1996 ("Van Wie"); Ginter, et al. "Software Tamper Resistance and Secure Communication," U.S. Patent Application Serial No. 08/706,206, filed on August 30, 1996 ("Ginter, '206"); Shear, et al, "Cryptographic Methods, Apparatus & Systems for Storage Media Electronic Rights Management in

SUBSTITUTE SHEET (RULE 26)

Closed & Connected Appliances," U.S. Patent Application Serial No. 08/848,077, filed on May 15, 1997 ("Shear"); Collberg et al, "Obfuscation Techniques for Enhancing Software Security," U.S. Patent Application Serial No. 09/095,346, filed on June 9, 1998 ("Collberg"); Shear, "Database Usage Metering and Protection System and Method," U.S. Patent No. 4,827,508, issued on May 2, 1989 ("Shear Patent").

FIG. 1 illustrates Media System 1, which is capable of accepting, decoding, and rendering streamed multimedia content. This is a generic system, though it includes elements based on the MPEG-4 specification. Media System 1 may include software modules, hardware (including integrated circuits) or a combination. In one embodiment, Media System 1 may include a Protected Processing Environment (PPE) as described in the Ginter '333 application.

In FIG. 1, Bit Stream 2 represents input information received by System 1. Bit Stream 2 may be received through a connection to an external network (e.g., an Internet connection, a cable hookup, radio transmission from a satellite broadcaster, etc.), or may be received from a portable memory device, such as a DVD player.

Bit Stream 2 is made up of a group of related streams of information, including Organization Stream 3, Audio Stream 4, Video Stream 5, Control Stream 6, and Info Stream 31. Each of these streams is encoded into the overall Bit Stream 2. Each of these represents a category of streams, so that, for example, Video Stream 5 may be made up of a number of separate Video Streams.

These streams correspond generally to streams described in the MPEG-4 format as follows:

Organization Stream 3 corresponds generally to the BIFS stream and the OD ("Object Descriptor") stream.

Audio Stream 4 and Video Stream 5 correspond generally to the Audio and Video streams.

Control Stream 6 corresponds generally to the IPMP stream.

Audio Stream 4 includes compressed (and possibly encrypted) digital audio information. This information is used to create the sound rendered and output by Media System 1. Audio Stream 1 may represent multiple audio streams. These multiple streams may act together to make up the audio output, or may represent alternative audio outputs.

Video Stream 5 includes compressed (and possibly encrypted) digital video information. This information is used to create the images and video rendered and output by Media System 1. Video Stream 5 may represent multiple video streams. These multiple streams may act together to make up the video output, or may represent alternative

video outputs.

Organization Stream 3 includes organizational information and metadata related to the work to be rendered. This information may include a tree or other organizational device which groups audio and video streams into objects. This information may also include metadata associated with the entire work, the objects, or the individual streams.

Control Stream 6 includes control information, divided generally into header information and messages. The header information includes an identifier for each discrete message. The content of the messages, which will be described further below, may include cryptographic keys and rules governing the use of content.

Info Stream 31 carries additional information associated with the content in other components of Bit Stream 2, including but not limited to graphics representing cover art, text for lyrics, coded sheet music or other notation, independent advertising content, concert information, fan club information, and so forth. Info Stream 31 can also carry system management and control information and/or components, such as updates to software or firmware in Media System 1, algorithm implementations for content-specific functions such as watermarking, etc.

Each of these streams is made up of packets of information. In one exemplary embodiment, each packet is 32 bytes in length. Since a single communications channel (e.g., a cable, a bus, an infrared or radio connection) contains packets from each of the streams, packets need to be identified as belonging to a particular stream. In a preferred embodiment, this is done by including a header which identifies a particular stream and specifies the number of following packets which are part of that stream. In another embodiment, each packet may include individual stream information.

Exemplary Header 201 is shown in FIG. 2. This header may generally be used for the Organization, Audio and Video Streams. A header for the Control Stream is described below. Header 201 includes Field 202, which includes a bit pattern identifying Header 201 as a header. Field 203 identifies the particular type of stream (e.g., Audio Stream, Organization Stream, Control Stream, etc.) Field 204 contains an Elementary Stream Identifier (ES_ID), which is used to identify the particular stream, and may be used in cases where multiple streams of a particular stream type may be encountered at the same time. Field 207 contains a time stamp, which is used by the system to synchronize the various streams, including rendering of the streams. Composite Block 11 may, for example, keep track of the elapsed time from the commencement of rendering. Time Stamp 207 may be used by Composite Block 11 to determine when each object is supposed to be rendered. Time Stamp 207 may therefore specify an elapsed time from commencement of rendering,

and Composite Block 11 may use that elapsed time to determine when to render the associated object.

Field 205 contains a Governance Indicator. Field 206 identifies the number of following packets which are part of the identified stream. In each case, the relevant information is encoded in a binary format. For example, Field 202 might include an arbitrary sequence of bits which is recognized as indicating a header, and Field 203 might include two bits, thereby allowing encoding of four different stream types.

Returning to FIG. 1, System 1 includes Demux 7, which accepts as input Bit Stream 2 and routes individual streams (sometimes referred to as Elementary Streams or "ESs") to appropriate functional blocks of the system.

Bit Stream 2 may be encoded in the format illustrated in FIG. 3. In this figure, Header 301 is encountered in the bit stream, with Packet 302 following, and so on through Packet 308.

When Demux 7 encounters Header 301, Demux 7 identifies Header 301 as a header and uses the header information to identify Packets 302-305 as organization stream packets. Demux 7 uses this information to route these packets to Organization Block 8. Demux 7 handles Header 306 in a similar manner, using the contained information to route Packets 307 and 308 to AV ("Audio Video") Block 9.

AV Block 9 includes Decompressor 10, which accepts Elementary Streams from Audio Stream 4 and Video Stream 5 and decompresses those streams. As decompressed, the stream information is placed in a format which allows it to be manipulated and output (through a video display, speakers, etc.). If multiple streams exist (e.g., two video streams each describing an aspect of a video sequence), AV Block 9 uses the ES_ID to assign each packet to the appropriate stream.

Organization Block 8 stores pointer information identifying particular audio streams and video streams contained in a particular object, as well as metadata information describing, for example, where the object is located, when it is to be displayed (e.g., the time stamp associated with the object), and its relationship to other objects (e.g., is one video object in front of or behind another video object). This organization may be maintained hierarchically, with individual streams represented at the lowest level, groupings of streams into objects at a higher level, complete scenes at a still higher level, and the entire work at the highest level.

FIG. 4 illustrates one manner in which Organization Block 8 may store a representation of a work. In this Figure, Tree 401 represents an entire audiovisual work. Branch 402 represents a high-level organization of the work. This may include, for

example, all of the video or possibly the audio and video associated with a particular scene.

Sub-Branch 403 represents a group of related video objects. Each such object may include an entire screen, or an individual entity within the screen. For example, Sub-Branch 403 may represent a background which does not change significantly from one shot to the next. If the video is moving between two points of reference (e.g., a conversation, with the camera point of view changing from one face to the other), Sub-Branch 404 could represent a second background, used in the second point of view.

Nodes 405 and 406 may represent particular video objects contained within the related group. Node 405 could, for example, represent a distant mountain range, while Node 406 represents a tree immediately behind one of the characters.

Each of the nodes specifies or contains a particular ES_ID, representing the stream containing the information used by that node. Node 405, for example, contains ES_ID 407, which identifies a particular video stream which contains compressed (and possibly encrypted) digital information representing the mountain range.

Composite Block 11 accepts input from Organization Block 8 and from AV Block 9. Composite Block 11 uses the input from Organization Block 8 to determine which specific audiovisual elements will be needed at any given time, and to determine the organization and relationship of those elements. Composite Block 11 accepts decompressed audiovisual objects from AV Block 9, and organizes those objects as specified by information from Organization Block 8. Composite Block 11 then passes the organized information to Rendering Device 12, which might be a television screen, stereo speakers, etc.

Control Block 13 stores control messages which may be received through Control Stream 6 and/or may be watermarked into or steganographically encoded in other streams, including Audio Stream 4 and Video Stream 5. One control message format is illustrated by FIG. 5, which shows Control Message 501. Control Message 501 is made up of Header 502 and Message 503. Header 502 consists of Field 508, which includes a bit pattern identifying the following information as a header; Stream Type Field 509, which identifies this as a header for the organization stream; ID Field 504, which identifies this particular control message; Pointer Field 505, which identifies those ESs which are controlled by this message; Time Stamp Field 507, which identifies the particular portion of the stream which is controlled by this control message (this may indicate that the entirety of the stream is controlled); and Length Field 506, which specifies the length (in bytes) of Message 503. Message 503 may include packets following Header 502, using the general format shown in FIG. 3. In the example shown, Control Message 501 carries the unique ID 111000,

encoded in ID Field 504. This control message controls ESs 14 and 95, as indicated by Pointer Field 505. The associated Message contains 1,024 bytes, as indicated by Length Field 506.

5 In an alternate embodiment, the association of control to content may be made in Organization Block 8, which may store a pointer to particular control messages along with the metadata associated with streams, objects, etc. This may be disadvantageous, however, in that it may be desirable to protect this association from discovery or tampering by users. Since Control Block 13 will generally have to be protected in any event, storing the association in this block may make protection of Organization Block 8 less necessary.

10 Control Block 13 implements control over System 1 through Control Lines 14, 15 and 16, which control aspects of Organization Block 8, AV Block 9 and Composite Block 11, respectively. Each of these Control Lines may allow two-way communication.

15 Control Lines 14 and 15 are shown as communicating with AV Block Stream Flow Controller 18 and with Organization Block Stream Flow Controller 17. These Stream Flow Controllers contain functionality controlled by Control Block 13. In the embodiment illustrated, the Stream Flow Controllers are shown as the first stage in a two-stage pipeline, with information being processed by the Stream Flow Controller and then passed on to the associated functional block. This allows isolation of the control functionality from the content manipulation and display functionality of the system, and allows control to be
20 added in without altering the underlying functionality of the blocks. In an alternate embodiment, the Stream Flow Controllers might be integrated directly into the associated functional blocks.

25 Stream Flow Controllers 17 and 18 contain Cryptographic Engines 19 and 20, respectively. These Cryptographic Engines operate under control of Control Block 13 to decrypt and/or cryptographically validate (e.g., perform secure hashing, message authentication code, and/or digital signature functions) the encrypted packet streams received from Demux 7. Decryption and validation may be selective or optional according to the protection requirements for the stream.

30 Cryptographic Engines 19 and 20 may be relatively complex, and may, for example, include a validation calculator that performs cryptographic hashing, message authentication code calculation, and/or other cryptographic validation processes. In addition, as is described further below, additional types of governance-related processing may also be used. In one alternative embodiment, a single Stream Flow Controller may be used for both Organization Stream 3 and Audio/Video Streams 4-5. This may reduce the
35 cost of and space used by System 1. These reductions may be significant, since System 1

may contain multiple AV Blocks, each handling a separate Audio or Video Stream in parallel. This alternative may, however, impose a latency overhead which may be unacceptable in a real-time system.

5 If the Stream Flow Controllers are concentrated in a single block, they may be incorporated directly into Demux 7, which may handle governance processing prior to routing streams to the functional blocks. Such an embodiment would allow for governed decryption or validation of the entirety of Bit Stream 2, which could occur prior to the routing of streams to individual functional blocks. Encryption of the entirety of Bit Stream 2 (as opposed to individual encryption of individual ESs) might be difficult or impossible 10 without incorporating stream controller functionality into Demux 7, since Demux 7 might otherwise have no ability to detect or read the header information necessary to route streams to functional blocks (that header information presumably being encrypted).

As is noted above, each of the individual streams contained in Bit Stream 2 may be individually encrypted. An encrypted stream may be identified by a particular indicator in 15 the header of the stream, shown in FIG. 2 as Governance Indicator 205.

When a header is passed by Demux 7 to the appropriate functional block, the stream flow controller associated with that block reads the header and determines whether the following packets are encrypted or otherwise subject to governance. If the header indicates that no governance is used, the stream flow controller passes the header and the packets 20 through to the functional blocks with no alteration. Governance Indicator 205 may be designed so that conventionally encoded content (e.g., unprotected MPEG-4 content) is recognized as having no Governance Indicator and therefore passed through for normal processing.

25 If a stream flow controller detects a set governance indicator, it passes the ES_ID associated with that stream and the time stamp associated with the current packets to Control Block 13 along Control Line 14 or 15. Control Block 13 then uses the ES_ID and time stamp information to identify which control message(s) are associated with that ES. Associated messages are then invoked and possibly processed, as may be used for governance purposes.

30 A simple governance case is illustrated by FIG. 6, which shows steps which take place using the functional blocks of FIG. 1. In Step 601, Demux 7 encounters a header, and determines that the header is part of the AV stream. In Step 602, Demux 7 passes the header to AV Stream Controller 18. In Step 603, AV Stream Controller 18 reads the header and determines that the governance indicator is set, thereby triggering further 35 processing along Path 604. In Step 605, AV Stream Controller 18 obtains the ES_ID and

time stamp from the header and transmits these to Control Block 13, along Control Line 15. In Step 606, Control Block 13 looks up the ES_ID and determines that the ES_ID is associated with a particular control message. In Step 611, Control Block 13 uses the time stamp information to choose among control messages, if there is more than one control message associated with a particular ES. In Step 607, Control Block 13 accesses the appropriate control message, and obtains a cryptographic key or keys for decryption and/or validation. In Step 608, Control Block 13 passes the cryptographic key(s) along Control Line 15 to AV Stream Controller 18. In Step 609, AV Stream Controller 18 uses the cryptographic key as an input to Cryptographic Engine 20, which decrypts and/or validates the packets following the header as those packets are received from Demux 7. In Step 610, the decrypted packets are then passed to AV Block 9, which decompresses and processes them in a conventional manner.

Time stamp information may be useful when it is desirable to change the control message applicable to a particular ES. For example, it may be useful to encode different portions of a stream with different keys, so that an attacker breaking one key (or even a number of keys) will not be able to use the content. This can be done by associating a number of control messages with the same stream, with each control message being valid for a particular period. The time stamp information would then be used to choose which control message (and key) to use at a particular time. Alternatively, one control message may be used, but with updated information being passed in through the Control Stream, the updates consisting of a new time stamp and a new key.

In an alternative embodiment, Control Block 13 may proactively send the appropriate keys to the appropriate stream flow controller by using time stamp information to determine when a key will be needed. This may reduce overall latency.

Control Line 16 from FIG. 1 comes into play once information has been passed from Organization Block 8 and AV Block 9 to Composite Block 11, and the finished work is prepared for rendering through Rendering Device 12. When Composite Block 11 sends an object to Rendering Device 11, Composite Block 11 sends a start message to Control Block 13. This message identifies the object (including any associated ES_IDs), and specifies the start time of the display (or other rendering) of that object. When an object is no longer being rendered, Composite Block 11 sends an end message to Control Block 13, specifying that rendering of the object has ended, and the time at which the ending occurred. Multiple copies of a particular object may be rendered at the same time. For this reason, start and stop messages sent by Composite Block 11 may include an assigned instance ID, which specifies which instance of an object is being rendered.

Control Block 13 may store information relating to start and stop times of particular objects, and/or may pass this information to external devices (e.g., External Server 30) through Port 21. This information allows Control Block 13 to keep track not only of which objects have been decrypted, but of which objects have actually been used. This may be used, since System 1 may decrypt, validate, and/or decompress many more objects than are actually used. Control Block 13 can also determine the length of use of objects, and can determine which objects have been used together. Information of this type may be used for sophisticated billing and auditing systems, which are described further below.

Control Line 16 may also be used to control the operation of Composite Block 11. In particular, Control Block 13 may store information specifying when rendering of a particular object is valid, and may keep track of the number of times an object has been rendered. If Control Block 13 determines that an object is being rendered illegally (i.e., in violation of rules controlling rendering), Control Block 13 may terminate operation of Composite Block 11, or may force erasure of the illegal object.

In an alternate embodiment, the level of control provided by Control Line 16 may at least in part be provided without requiring the presence of that line. Instead, Control Block 13 may store a hash of the organization information currently valid for Organization Block 8. This hash may be received through Control Stream 6, or, alternatively, may be generated by Control Block 13 based on the information contained in Organization Block 8.

Control Block 13 may periodically create a hash of the information currently resident in Organization Block 8, and compare that to the stored hash. A difference may indicate that an unauthorized alteration has been made to the information in Organization Block 8, thereby potentially allowing a user to render information in a manner violative of the rules associated with that information. In such an event, Control Block 13 may take appropriate action, including deleting the information currently resident in Organization Block 8.

If System 1 is designed so that Organization Block 8 controls the use of content by Composite Block 11, so that content cannot be rendered except as is specified by the organization information, Control Block 13 may be able to control rendering of information through verifying that the current Organization Block contents match the hash which has been received by Control Block 13, thereby eliminating at least one reason for the presence of Control Line 16.

Control Block 13 may also be responsible for securely validating the origin, integrity, authenticity, or other properties of received content, through cryptographic

validation means such as secure hashing, message authentication codes, and/or digital signatures.

System 1 may also include an Inter-Rights Point, indicated as IRP 22. IRP 22 is a protected processing environment (e.g., a PPE) in which rules/controls may be processed, and which may store sensitive information, such as cryptographic keys. IRP 22 may be incorporated within Control Block 13, or may be a separate module. As is illustrated, IRP 22 may include CPU 23 (which can be any type of processing unit), Cryptographic Engine 24, Random Number Generator 25, Real Time Clock 26, and Secure Memory 27. In particular embodiments, some of these elements may be omitted, and additional functionality may be included.

Governance Rules

Control messages stored by Control Block 13 may be very complex. FIG. 7 illustrates the form in which the control messages may be stored in Control Block 13, consisting of Array 717. Column 701 consists of the address at which the control messages are stored. Column 702 consists of the identifier for each control message. This function may be combined with that of Column 701, by using the location information of Column 701 as the identifier, or by storing the message in a location which corresponds to the identifier. Column 703 consists of the ES_IDs for each stream controlled by the control message. Column 704 consists of the message itself. Thus, the control message stored at location 1 has the ID 15, and controls stream 903.

In a simple case, the message may include a cryptographic key, used to decrypt the content associated with the stream(s) controlled by the message. This is illustrated by Cryptographic Key 705 from FIG. 7. Cryptographic keys and/or validation values may also be included to permit cryptographic validation of the integrity or origin of the stream.

In a more complex case, the message may include one or more rules designed to govern access to or use of governed content. Rules may fall into a number of categories.

Rules may require that a particular aspect of System 1, or a user of System 1, be verified prior to decryption or use of the governed content. For example, System 1 may include System ID 28, which stores a unique identifier for the system. A particular rule contained in a control message may specify that a particular stream can only be decrypted on a system in which System ID 28 contains a particular value. This is illustrated at row 2 in FIG. 7, in which the message is shown as consisting of a rule and commands. The rule may be implicit, and therefore may not be stored explicitly in the table (e.g. the table may store only the rule, the rule - specific functions (commands) invoked by the rule, or only the functions).

- 13 -

In this case, when Stream Controller 18 encounters a Header for stream 2031 containing a set governance indicator, Stream Controller 18 passes the associated ES_ID (2031) to Control Block 13. Control Block 13 then uses the ES_ID to identify Control Message 20 which governs stream 2031. Control Message 20 includes Rule 706, which includes (or invokes) Commands 707, and an Authorized System ID 708. Authorized System ID 708 may have been received by System 1, either as part of Control Message 20, or as part of another control message (e.g., Control Message 9), which Control Message 20 could then reference in order to obtain access to the Authorized System ID. Such a case might exist, for example, if a cable subscriber had pre-registered for a premium show. The cable system might recognize that registration, and authorize the user to view the show, by sending to the user an ID corresponding to the System ID.

When Rule 706 is invoked, corresponding Commands 707 access System ID 28 and obtain the system ID number. The commands then compare that number to Authorized System ID 708, specified by Rule 706. If the two numbers match, Commands 707 release Cryptographic Key 709 to Stream Controller 18, which uses Cryptographic Key 709 to decrypt the stream corresponding to ES_ID 2031. If the two numbers do not match, Commands 707 fail to release Cryptographic Key 709, so that Stream Controller 18 is unable to decrypt the stream.

In order to carry out these functions, in one embodiment, Control Block 13 includes, or has access to, a processing unit and memory. The processing unit is preferably capable of executing any of the commands which may be included or invoked by any of the rules. The memory will store the rules and association information (ID of the control message and IDs of any governed ESs).

Since the functions being carried out by Control Block 13 are sensitive, and involve governance of content which may be valuable, Control Block 13 may be partially or completely protected by a barrier which resists tampering and observation. As is described above, the processing unit, secure memory, and various other governance-related elements may be contained in IRP 22, which may be included in or separate from Control Block 13.

Control Block 13 may also carry out somewhat more complex operations. In one example, a control message may require that information from System 1 not only be accessed and compared to expected information, but stored for future use. For example, a control message might allow decryption of a Stream, but only after System ID 28 has been downloaded to and stored in Control Block 13. This would allow a control message to check the stored System ID against System ID 28 on a regular basis, or perhaps on every

SUBSTITUTE SHEET (RULE 26)

- 14 -

attempted re-viewing of a particular Stream, thereby allowing the control message to insure that the Stream is only played on a single System.

Control Block 13 may also obtain information dynamically. For example, System 1 may include User Interface 29, which can include any type of user input functionality (e.g., hardware buttons, information displayed on a video screen, etc.) A particular rule from a control message may require that the user enter information prior to allowing decryption or use of a stream. That information may, for example, be a password, which the Rule can then check against a stored password to insure that the particular user is authorized to render the stream.

Information obtained from the user might be more complicated. For example, a rule might require that the user input payment or personal information prior to allowing release of a cryptographic key. Payment information could, for example, constitute a credit card or debit card number. Personal information could include the user's name, age, address, email address, phone number, etc. Entered information could then be sent through Port 21 to External Server 30 for verification. Following receipt of a verification message from External Server 30, the Rule could then authorize release of a cryptographic key. Alternatively, Control Block 13 may be designed to operate in an "off-line" mode, storing the information pending later hookup to an external device (or network). In such a case, Control Block 13 might require that a connection be made at periodic intervals, or might limit the number of authorizations which may be obtained pending the establishment of an external connection.

In a somewhat more complex scenario, a control message may include conditional rules. One particular example is illustrated by row 4 of the table shown in FIG. 7, in which Control Message 700 is shown as controlling streams 49-53. Control Message 700 further consists of Rule 710, Commands 711 and Cryptographic Keys 712-716. There could, of course, be a number of additional cryptographic keys stored with the message.

In this case, Rule 710 specifies that a user who agrees to pay a certain amount (or provide a certain amount of information) may view Stream 49, but all other users are required to view Stream 50, or a combination of Streams 49 and 50. In this case, Stream 49 may represent a movie or television program, while Stream 50 represents advertisements. In one embodiment, different portions of Stream 49 may be decrypted with different keys so that, for example, a first portion is decrypted with Key 712, a second portion is decrypted with Key 713, a third portion is decrypted with Key 714, and so on. Rule 710 may include all keys used to decrypt the entirety of Stream 49. When the user initially attempts to access the video encoded in Stream 49, Rule 710 could put up a

SUBSTITUTE SHEET (RULE 26)

- 15 -

message asking if the user would prefer to use pay for view mode or advertising mode. If the user selects pay for view mode, Rule 710 could store (or transmit) the payment information, and pass Cryptographic Key 712 to Stream Controller 18. Stream Controller 18 could use Cryptographic Key 712 to decrypt the first stream until receipt of a header
5 indicating that a different key is needed to decrypt the following set of packets. Upon request by Stream Controller 18, Control Block 13 would then check to determine that payment had been made, and then release Cryptographic Key 713, which would be used to decrypt the following packets, and so on. Rule 710 could additionally release
10 Cryptographic Key 716, corresponding to Organization Stream 52, which corresponds to video without advertisements.

If, on the other hand, the user had chosen the advertising mode, Rule 710 could release Cryptographic Key 712 to Stream Controller 18 to allow decryption of Stream 49. Rule 710 could also authorize decryption of Stream 50 which contains the advertisements. Rule 710 could further release Cryptographic Key 715 to Organization Block 8.
15 Cryptographic Key 715 matches Organization Stream 51. Organization Stream 51 references the video from Stream 49, but also references advertisements from Stream 50. Rule 710 would refuse to release Cryptographic Key 716, which corresponds to Organization Stream 52, which corresponds to the video without advertisements.

In operation, Control Block 13 could monitor information from Composite Block
20 11 over Control Line 16. That information could include the identity of each object actually rendered, as well as a start and stop time for the rendering. Control Block 13 could use this information to determine that an advertisement had actually been rendered, prior to releasing Cryptographic Key 713 for decryption of the second portion of video from Stream 49. This feedback loop allows Control Block 13 to be certain that the
25 advertisements are not only being decrypted, but are also being displayed. This may be necessary because Composite Block 11 may be relatively unprotected, thereby allowing an unscrupulous user to remove advertisements before viewing.

A variety of additional relatively complex scenarios are possible. For example,
30 rules from Control Block 13 could customize the programming for a particular geographic location or a particular type of viewer, by using information on the location or the viewer to control conditional decryption or use. This information could be stored in System 1 or entered by the user.

In another example, shown at row 5 of Array 717, Rule 719 may specify Budget
35 718, which may include information relating to the number of uses available to the user, the amount of money the user has to spend, etc. In operation, Rule 719 may require that

SUBSTITUTE SHEET (RULE 26)

- 16 -

Budget 718 be securely stored and decremented each time a budgeted activity occurs (e.g., each time the associated work is played). Once the budget reaches zero, Rule 719 may specify that the work may no longer be played, or may display a message to the user indicating that the user may obtain additional budget by, for example, entering a credit card number or password, or contacting an external server.

In another example, a rule may control the ability of a user to copy a work to another device. The rule may, for example, specify that the user is authorized to use the governed work on more than one device, but with only one use being valid at any time. The rule may specify that an indication be securely stored regarding whether the user has "checked out" the work. If the user copies the work to another device (e.g., through Port 21), the rule may require that the work only be transmitted in encrypted form, and that the relevant control messages be transmitted along with it. The rule can further require that an indicator be securely set, and that the indicator be checked each time the user attempts to use or copy the work. If the indicator is set, the rule might require that the work not be decrypted or used, since the user only has the right to use the work on one device at a time, and the indicator establishes that the work is currently "checked out" to another device and has not been checked back in.

The receiving device may include the same type of indicator, and may allow the user to use the work only as long as the indicator is not set. If the user desires to use the work on the original device, the two devices may communicate, with the indicator being set in the second and reset in the first. This allows the work to be stored in two locations, but only used in one.

In another embodiment, the same result may be reached by copying the relevant control message from one device to the other, then erasing it from the original device. Because the control message includes keys used for decryption, this would insure that the work could only be used in one device at a time.

In one embodiment, this technique may be used to communicate digital media files (e.g., music, video, etc.) from a personal computer to a consumer electronics device without allowing the user to make multiple choices for simultaneous use. Thus, a larger, more sophisticated device (e.g., a personal computer), could download a file, then "check out" the file to a portable device lacking certain functions present in the personal computer (e.g., a hand-held music player).

Rules may also be used to specify that an initial user may transfer the file to another user, but only by giving up control over the file. Such rules could operate similarly to the

technique described above for transferring a file from one device to another, or could require that the original file be entirely erased from the original device after the transfer.

Rules in Control Block 13 may be added or updated through at least two channels. New rules may be obtained through Control Stream 6. If a control message contains an identifier corresponding to a control message already present in Control Block 13, that control message (including contained rules) may overwrite the original control message. A new rule may, for example, be identical to an existing rule, but with a new time stamp and new keys, thereby allowing decryption of a stream which had been encrypted with multiple keys. System 1 may be designed so that certain rules may not be overwritable. This may be enforced by designating certain positions in Array 717 as non-overwritable, or by providing a flag or other indicator to show that a particular rule cannot be overwritten or altered. This would allow for certain types of superdistribution models, including allowing a downstream distributor to add rules without allowing the downstream distributor to remove or alter the rules added by upstream distributors.

In addition, new rules may be encoded into Organization Stream 3, Audio Stream 4, or Video Stream 5, in the form of a watermark or steganographic encoding.

New rules may also be obtained through Port 21. Port 21 may connect to an external device (e.g., a smart card, portable memory, etc.) or may connect to an external network (e.g., External Server 30). Rules may be obtained through Port 21 either in an ad hoc manner, or as a result of requests sent by Control Block 13. For example, Control Message 14 (FIG. 7, row 6) may include a rule specifying that a new rule be downloaded from a particular URL, and used to govern Stream 1201.

Control messages, including rules, may be encoded using secure transmission formats such as DigiBoxes. A DigiBox is a secure container means for delivering a set of business rules, content description information, content decryption information and/or content validation information. One or more DigiBoxes can be placed into the headers of the media content or into data streams within the media.

FIG. 12 illustrates one embodiment of the DigiBox format and the manner in which that format is incorporated into a control message. Control Message 1201 is made up of Control Message Header 1202 and Control Message Contents 1203. As is described elsewhere, Control Message Header 1202 may include information used by Demux 7 (FIG. 1) to appropriately route the message to Control Block 13.

Control Message Contents 1203 of Control Message 1201 consists of DigiBox 1204, and may also include additional information. DigiBox 1204 consists of DigiBox Header 1205, Rules 1206 and Data 1207. Rules 1206 may include one or more rules. Data

1207 may include various types of data, including ES_ID 1208, Cryptographic Key 1209, and Validation Data 1210. Data 1207 may also include cryptographic information such as a specification of the encryption algorithm, chaining modes used with the algorithm, keys and initialization vectors used by the decryption and chaining.

5 Initialization vectors contained within Data 1207 are similar to cryptographic keys, in that they constitute input to the original encryption process and therefore are necessary for decryption. In one well-known prior art embodiment, the initialization vectors may be generated by starting with a base initialization vector (a 64 bit random number) and xor'ing in the frame number or start time for the content item.

10 Validation Data 1210 contained within Data 1207 may include cryptographic has or authentication values, cryptographic keys for calculating keyed authentication values (e.g., message authentication codes), digital signatures, and/or public key certificates used in validating digital certificates.

15 Thus, the DigiBox may incorporate the information described above as part of the control message, including the rules, the stream ID and the cryptographic keys and values.

In an alternative embodiment, DigiBox Header 1205 may be designed so that it can be read by Demux 7 and routed to Control Block 13. In such an embodiment, DigiBox 1204 would itself constitute the entirety of the control message, thus obviating the need to nest DigiBox 1204 within Control Message 1201.

20 Some or all of the contents of DigiBox 1204 will generally be encrypted. This may include Rules 1206, Data 1207, and possibly some or all of Header 1205. System 1 may be designed so that a DigiBox may only be decrypted (opened) in a protected environment such as IRP 22. In an alternate embodiment, Control Block 13 may directly incorporate the functionality of IRP 22, so that the DigiBox may be opened in Control Block 13 without the necessity of routing the DigiBox to IRP 22 for processing. In one embodiment, the

25 cryptographic key used to decrypt DigiBox 1204 may be stored in IRP 22 (or Control Block 13), so that the DigiBox can only be opened in that protected environment.

30 Rules 1206 are rules governing access to or use of DigiBox Data 1207. In one embodiment, these rules do not directly control the governed streams. Since Cryptographic Key 1209 can only be accessed and used through compliance with Rules 1206, however, Rules 1206 in fact indirectly control the governed streams, since those streams can only be decrypted through use of the key, which can only be obtained in compliance with the rules. In another embodiment, Data 1207 may include additional rules, which may be extracted from the DigiBox and stored in a table such as Array 717 of FIG. 7.

The rules governing access to or use of a DigiBox may accompany the DigiBox, (as shown in FIG. 12) or may be separately transmitted, in which event Rules 1206 would contain a pointer or reference to the rules used to access Data 1207. Upon receipt of a DigiBox, Control Block 13 may receive rules separately through Control Stream 6, or may request and receive rules through Port 21.

Pipelined Implementation

One potential drawback to the system illustrated in FIG.1 consists of the fact that the system introduces complexity and feedback into a pipelined system designed to render content in real time. The rendering pipeline generally consists of Demux 7, Organization Block 8 and AV Block 9, Composite Block 11 and Rendering Device 12. Because content is received in a streamed fashion, and must be rendered in real time, pipelined processing must occur in a highly efficient manner, under tight time constraints. A failure to process within the time available may mean that output to Rendering Device 12 may be interrupted, or that incoming Bit Stream 2 may overflow available buffers, thereby causing the loss of some portion of the incoming data.

An alternative embodiment of System 1 is designed to address these problems, although at a possible cost in the ability to use standard system components and a possible cost in overall system security. This alternative embodiment is illustrated in FIG. 11, which shows System 1101.

System 1101 is similar to System 1 from FIG. 1 in many respects. It receives Bit Stream 1102, which consists of Organization Stream 1103, Audio Stream 1104, Video Stream 1105 and Control Stream 1106. These streams are received by Demux 1107, which passes Organization Stream 1103 to Organization Block and passes Audio Stream 1104 and Video Stream 1105 to AV Block 1109. Organization Block 1108 and AV Block 1109 operate similarly to their counterparts in FIG. 1, and pass information to Composite Block 1110, which organizes the information into a coherent whole and passes it to Rendering Device 1111. Streams sent to Organization Block 1108 are decrypted and/or validated by Stream Flow Controller 1112, and streams sent to AV Block 1109 are decrypted and/or validated by Stream Flow Controller 1113.

System 1101 differs from System 1, however, in that control and feedback are distributed, and integrated directly into the processing and rendering pipeline. System 1101 thus lacks a separate control block, and also lacks a feedback path back from the Composite Block 1110.

In System 1101, control is exercised directly at Organization Block 1108 and AV Block 1109. As in System 1, cryptographic keys are received through Control Stream 1106

(in an alternative embodiment, the keys could be incorporated directly into header or other information in Organization Stream 1103 or Audio/Video Streams 1104 and 1105). Those keys are included in a data format which includes information regarding the stream type of the encrypted content and, if multiple stream types are possible, an identifier for the particular controlled stream.

When Demux 1107 encounters a key in Control Stream 1106, it reads the information relating to the stream type, and routes the key to the appropriate stream flow controller. If Demux 1107 encounters a key designated for decryption or validation of Organization Stream 1103, for example, it routes that key to Stream Flow Controller 1112.

Stream Flow Controller 1112 stores received keys in Storage Location 1114. Storage Location 1114 stores the keys and also stores an indicator of the controlled stream ID.

Stream Flow Controller 1112 includes Cryptographic Engine 1115, which uses the received keys to decrypt and/or validate encrypted and/or protected portions of Organization Stream 1103. The keys may themselves be received in an encrypted manner, in order to provide some degree of security. In such a case, Stream Flow Controller may use a variety of techniques to decrypt the key, including using stored information as a key, or as a key seed. That stored information could, for example, constitute a "meta-key" provided earlier through Bit Stream 1102 or through a separate port.

Stream Flow Controller 1113, associated with AV Block 1109, contains a corresponding Storage Location 1116 and Cryptographic Engine 1117, and operates in a manner similar to the operation described for Stream Flow Controller 1112.

This implementation avoids the latency penalty which may be inherent in the necessity for communication between stream flow controllers and a separate control block.

This alternate implementation may also eliminate the feedback channel from the composite block (FIG.1, Control Line 16). This feedback channel may be used in order to insure that the content being passed from Composite Block 11 to Rendering Device 12 is content that has been authorized for rendering. In the alternate embodiment shown in FIG.11, this feedback channel does not exist. Instead, this implementation relies on the fact that Composite Block 1110 depends upon information from Organization Block 1108 to determine the exact structure of the information being sent to Rendering Device 1111. Composite Block 1110 cannot composite information in a manner contrary to the organization dictated by Organization Block 1108.

In one embodiment, this control by Organization Block 1108 may be sufficient to obviate the need for any feedback, since Organization Block 1108 may be designed so that

it accepts information only through Stream Controller 1112, and Stream Controller 1112 may be designed so that it only decrypts or validates information under the control of rules stored in Storage Location 1114.

5 In such an embodiment, security may be further increased by incorporating Secure Memory 1118 into Organization Block 1108. Secure Memory 1118 may store a copy or hash of the organization tree validly decrypted by Stream Controller 1112, and in current use in Main Organization Block Memory 1119. Organization Block 1108 may be used to periodically compare the organization tree stored in Main Organization Block Memory 1119 to the tree stored in Secure Memory 1118. If a discrepancy is spotted, this may
10 indicate that an attacker has altered the organization tree stored in Main Organization Block 1119, thereby possibly allowing for the rendering of content in violation of applicable rules. Under such circumstances, Organization Block 1108 may be used to take protective measures, including replacing the contents of Main Organization Block Memory 1119 with the contents of Secure Memory 1118.

15 **MPEG-4 Implementation**

The generic system described above may be embodied in an MPEG-4 system, as illustrated in FIG. 8, which shows MPEG-4 System 801.

20 MPEG-4 System 801 accepts MPEG-4 Bit Stream 802 as input. MPEG-4 Bit Stream 802 includes BIFS Stream 803, OD Stream 804, Audio Stream 805, Video Stream 806 and IPMP Stream 807. These streams are passed to Demux 808, which examines header information and routes packets as appropriate, to BIFS 809, AVO 810, OD 811 or IPMP System 812.

25 IPMP System 812 receives IPMP messages through IPMP Stream 807. Those messages may include header information identifying the particular message, as well as an associated IPMP message. The IPMP message may include control information, which may include a cryptographic key, validation information, and/or may include complex governance rules, as are described above.

Stream Controllers 813, 814 and 815 act to decrypt, validate, and/or govern streams passed to BIFS 809, AVO 810 and OD 811, respectively.

30 OD 811 holds object descriptors, which contain metadata describing particular objects. This metadata includes an identifier of the particular Elementary Stream or streams which include the object, and may also include a pointer to a particular IPMP message which governs the object. Alternatively, the relationship between IPMP messages and particular objects or streams may be stored in a table or other form within IPMP
35 System 812.

IPMP System 812 may exercise control over other functional blocks through Control Lines 816, 817, 818 and 819, each of which may transmit control/governance signals from IPMP System 812 and information or requests from other functional blocks to IPMP System 812. The information requests may include an ES_ID and a time stamp,
5 which IPMP System 812 may use to determine which particular message (e.g., key) should be used and when.

In an alternative embodiment, IPMP System 812 may exercise control over Composite and Render 821 by receiving a hash of the currently valid BIFS tree (possibly through IPMP stream 807), and periodically checking the hash against the BIFS tree stored
10 in BIFS 809. Because BIFS 809 controls the manner in which Composite and Render 821 renders information, if IPMP System 812 confirms that the current BIFS tree is the same as the authorized tree received through BIFS Stream 803, IPMP System 812 can confirm that the proper content is being rendered, even without receiving feedback directly from Composite and Render 821. This may be necessary, since BIFS 809 may communicate
15 with Port 822, which may allow a user to insert information into BIFS 809, thereby creating a possibility that a user could insert an unauthorized BIFS tree and thereby gain unauthorized access to content.

When a stream controller receives encrypted or otherwise governed information, it may send the ES_ID and time stamp directly to IPMP System 812. Alternatively, it may
20 send this information to OD 811, which may reply with the ID of the IPMP message which governs that object or stream. The stream controller can then use that IPMP message ID to request decryption, validation, and/or governance from IPMP System 812. Alternatively, OD 811 can pass the IPMP ID to IPMP System 812, which can initiate contact with the appropriate stream controller.

IPMP System 812 may obtain IPMP information through two channels other than IPMP Stream 807. The first of these channels is Port 820, which may be directly
25 connected to a device or memory (e.g., a smart card, a DVD disk, etc.) or to an external network (e.g., the Internet). An IPMP message may contain a pointer to information obtainable through Port 812, such as a URL, address on a DVD disk, etc. That URL may
30 contain specific controls needed by the IPMP message, or may contain ancillary required information, such as, for example, information relating to the budget of a particular user.

IPMP System 812 may also obtain IPMP information through OD updates contained in OD Stream 804. OD Stream 804 contains metadata identifying particular
35 objects. A particular OD Message may take the format shown in FIG. 9. In this figure, OD Message 901 includes Header 902, which identifies the following packets as part of the OD

stream, and indicates the number of packets. OD Message 901 further consists of Message 903, which includes a series of Pointers 904 and associated Metadata 905. Each Pointer 904 identifies a particular Elementary Stream, and the associated metadata is applicable to that stream. Finally, OD Message 901 may contain an IPMP Pointer 906, which identifies a particular IPMP message.

In aggregate, the information contained in OD Message 901 constitutes an object descriptor, since it identifies and describes each elementary stream which makes up the object, and identifies the IPMP message which governs the object. OD Message 901 may be stored in OD 811, along with other messages, each constituting an object descriptor.

Object descriptors stored in OD 811 may be updated through OD Stream 804, which may pass through a new object descriptor corresponding to the same object. The new object descriptor then overwrites the existing object descriptor. This mechanism may be used to change the IPMP message which controls a particular object, by using a new object descriptor which is identical to the existing object descriptor, with the exception of the IPMP pointer.

OD Stream 804 can also carry IPMP_DescriptorUpdate messages. Each such message may have the same format as IPMP messages carried on the IPMP stream, including an IPMP ID and an IPMP message.

IPMP_DescriptorUpdate messages may be stored in a table or array in OD 811, or may be passed to IPMP System 812, where they may overwrite existing stored IPMP messages, or may add to the stored messages.

Since IPMP information may be separately conveyed through the OD stream or the IPMP stream, MPEG-4 System 801 may be designed so that it only accepts information through one or the other of these channels.

In another embodiment, the existence of the two channels may be used to allow multi-stage distribution, with governance added at later stages, but with no risk that later alterations may override governance added at an earlier stage.

Such a system is illustrated in FIG. 10. In this Figure, IPMP System 812 includes IPMP Table 1002, which has slots for 256 IPMP messages. This table stores the IPMP_ID implicitly, as the location at which the information is stored, shown in Column 1003. The IPMP message associated with IPMP_ID 4, for example, is stored at slot 4 of IPMP Table 1002.

Each location in IPMP Table 1002 includes Valid Indicator 1004 and Source Indicator 1005. Valid Indicator 1004 is set for a particular location when an IPMP message is stored at that location. This allows IPMP System 812 to identify slots which are

unfilled, which otherwise might be difficult, since at start-up the slots may be filled with random information. This also allows IPMP System 812 to identify messages which are no longer valid and which may be replaced. Valid Indicator 1004 may store time stamp information for the period during which the message is valid with IPMP System 812 determining validity by checking the stored time stamp information against the currently valid time.

Source Indicator 1005 is set based on whether the associated IPMP message was received from IPMP Stream 807 or from OD Stream 804.

These indicators allow IPMP System 812 to establish a hierarchy of messages, and to control the manner in which messages are added and updated. IPMP System 812 may be designed to evaluate the indicators for a particular location once a message is received corresponding to that location. If the valid indicator is set to invalid, IPMP System 812 may be designed to automatically write the IPMP message into that slot. If the valid indicator is set to valid, IPMP System 812 may then be designed to check the source indicator. If the source indicator indicates that the associated message was received through OD Stream 804, IPMP System 812 may be designed to overwrite the existing message with the new message. If, however, the source indicator indicates that the associated message was received through IPMP Stream 807, IPMP System 812 may be designed to check the source of the new message. That check may be accomplished by examining the header associated with the new message, to determine if the new message was part of OD Stream 804 or part of IPMP Stream 807. Alternatively, IPMP System 812 may derive this information by determining whether the message was received directly from Demux 808 or through OD 811.

If the new message came through IPMP Stream 807, IPMP System 812 may be designed to store the new message in Table 1002, overwriting the existing message. If the new message came through OD Stream 804, on the other hand, IPMP System 812 may be designed to reject the new message.

This message hierarchy can be used to allow for a hierarchy of control. A studio, for example, may encode a movie in MPEG-4 format. The studio may store IPMP messages in the IPMP stream. Those messages may include a requirement that IPMP System 812 require that a trailer for another movie from the same studio be displayed prior to the display of the feature movie. IPMP System 812 could be used to monitor the beginning and end of rendering of the trailer (using feedback through Control Line 819) to ensure that the entire trailer plays, and that the user does not fast-forward through it.

5 The movie studio could encrypt the various elementary streams, including the IPMP stream. The movie studio could then provide the movie to a distributor, such as a cable channel. The movie studio could provide the distributor with a key enabling the distributor to decrypt the OD stream (or could leave the OD stream unencrypted), and the ability to insert new messages in that stream. The cable channel could, for example, include a rule in the OD stream specifying that the IPMP system check to determine if a user has paid for premium viewing, decrypt the movie if premium viewing has been paid for, but insert advertisements (and require that they be rendered) if premium viewing has not been paid for).

10 The cable channel would therefore have the ability to add its own rules into the MPEG-4 Bit Stream, but with no risk that the cable channel would eliminate or alter the rules used by the movie studio (e.g., by changing the trailer from a movie being promoted by the studio to a rival movie being promoted by the cable channel). The studio's rules could specify the types of new rules which would be allowed through the OD stream, thereby providing the studio a high degree of control.

15 This same mechanism could be used to allow superdistribution of content, possibly from one user to another. A user could be provided with a programming interface enabling the insertion of messages into the OD stream. A user might, for example, insert a message requiring that a payment of \$1.00 be made to the user's account before the movie can be viewed. The user could then provide the movie to another user (or distribute it through a medium whereby copying is uncontrolled, such as the Internet), and still receive payment. Because the user's rules could not overrule the studio's rules, however, the studio could be certain that its rules would be observed. Those might include rules specifying the types of rules a user would be allowed to add (e.g., limiting the price for redistribution).

20 MPEG-4 System 801 may also be designed to include a particular type of IPMP system, which may be incompatible with IPMP systems that may be designed into other MPEG-4 systems. This may be possible because the MPEG-4 standard does not specify the format of the information contained in the IPMP stream, thereby allowing different content providers to encode information in differing manners.

25 IPMP System 812 in MPEG-4 System 801 may be designed for an environment in which differing IPMP formats exist. That system may scan the IPMP stream for headers that are compatible with IPMP System 812. All other headers (and associated packets) may be discarded. Such a mechanism would allow content providers to incorporate the same IPMP message in multiple formats, without any concern that encountering an unfamiliar format would cause an IPMP system to fail. In particular, IPMP headers can

30

35

incorporate an IPMP System Type Identifier. Those identifiers could be assigned by a central authority, to avoid the possibility that two incompatible systems might choose the same identifier.

5 IPMP System 801 might be designed to be compatible with multiple formats. In such a case, IPMP System 801 might scan headers to locate the first header containing an IPMP System Identifier compatible with IPMP System 801. IPMP System 801 could then select only headers corresponding to that IPMP System Identifier, discarding all other headers, including headers incorporating alternate IPMP System Identifiers also recognized by the IPMP system.

10 Such a design would allow a content provider to provide multiple formats, and to order them from most to least preferred, by including the most preferred format first, the second most preferred format second, and so on. Since IPMP System 801 locks onto the first compatible format it finds, this ordering in IPMP Stream 801 would insure that the IPMP system chose the format most desired by the content provider.

15 Even if different IPMP formats are used, content will probably be encoded (and encrypted) using a single algorithm, since sending multiple versions of content would impose a significant bandwidth burden. Thus, ordinarily it will be necessary for content to be encrypted using a recognized and common encryption scheme. One such scheme could use the DES algorithm in output feedback mode.

20 This method of screening IPMP headers, and locking onto a particular format may also be used to customize an MPEG-4 bit Stream for the functional capabilities of a particular MPEG-4 system. Systems capable of rendering MPEG-4 content may span a considerable range of functionality, from high-end home theaters to handheld devices. Governance options suitable for one type of system may be irrelevant to other systems.

25 For example, MPEG-4 System 801 may include a connection to the Internet through Port 820, whereas a second MPEG-4 system (for example a handheld Walkman-like device) may lack such a connection. A content provider might want to provide an option to a viewer, allowing the viewer to see content for free in return for providing information about the viewer. The content provider could insert a rule asking the user whether the user wants to view the content at a cost, or enter identification information. The rule could then send the information through a port to the Internet, to a URL specified in the rule. A site at that URL could then evaluate the user information, and download advertisements targeted to the particular user.

30 Although this might be a valuable option for a content provider, it obviously makes no sense for a device which is not necessarily connected to the Internet. It would make no

sense to present this option to the user of a non-connected device, since even if that user entered the information, the rule would have no way to provide the information to an external URL or download the advertisements. In such a case, the content provider might prefer to require that the user watch preselected ads contained in the original MPEG-4 bit stream.

Header information in the IPMP stream could be used to customize an MPEG-4 bit stream for particular devices. As with the IPMP System Type information, IPMP Header information could include MPEG-4 System Types. These could include 8 or 16-bit values, with particular features represented by bit maps. Thus, the presence of a bit at position 2, for example, could indicate that a device includes a persistent connection to the Internet.

An IPMP system could then evaluate the headers, and lock on to the first header describing functionality less than or equal to the functionality contained in the MPEG-4 device in which the IPMP system is embedded. If the header constituted a complete match for the functionality of the MPEG-4 device, the IPMP system could then cease looking. If the header constitutes less than a complete match (e.g., a header for a system which has an Internet connection, but lacks a digital output port, when the system includes both), the IPMP system can lock on to that header, but continue to scan for closer matches, locking on to a closer match if and when one is found.

The IPMP messages identified by a particular header would be those suited for the particular functionality of the MPEG-4 device, and would allow for customization of the MPEG-4 bit stream for that functionality. In the context of the example given above, the IPMP system for an MPEG-4 device containing an Internet connection would lock on to a particular header, and would download the IPMP messages characterized by that header. Those messages would prompt the user for information, would provide that information to the URL, and would authorize decryption and rendering of the movie, with the advertisements inserted at the appropriate spot.

In the case of an MPEG-4 device without an Internet connection, on the other hand, the IPMP system would lock onto a set of headers lacking the bit indicating an Internet connection, and would download the rules associated with that header. Those rules might not provide any option to the user. The rules might allow decryption of the content, but would also specify decryption of an additional ES from the MPEG-4 stream. That additional ES would contain the advertisements, and the IPMP system would require decryption and rendering of the advertisements, checking Control Line 819 to make certain that this had occurred. In the case of the system with the Internet connection, however, the rules allowing decryption and requiring rendering of the ES containing the advertisements

- 28 -

would never be loaded, since those rules would be contained within messages identified by the wrong type of header. The advertisement ES would therefore never be decrypted and would be ignored by the MPEG-4 device.

FIG. 21 illustrates one manner in which a protected MPEG-4 file may be created. In this figure, CreateBox 2101 represents a DigiBox creation utility, which accepts keys and rules. In one embodiment, CreateBox 2101 may pass these keys and rules to IRP 2102 and receive DigiBox 2103 from IRP 2102. In another embodiment, IRP 2102 may be incorporated into CreateBox 2101, which accepts keys and rules and outputs DigiBox 2103.

DigiBox 2103 contains governance rules, initialization vectors and keys. DigiBox 2103 is passed from CreateBox 2101 to Bif Encoder 2104. Bif Encoder 2104 may be conventional, with the exception that it is designed to accept and process DigiBoxes such as DigiBox 2103. Bif Encoder 2104 also accepts a .txt file containing a scene graph, and initial object descriptor commands.

Bif Encoder 2104 outputs a .bif file, containing the scene graph stream (in compressed binary form) and a .od file, containing the initial object descriptor commands, the object descriptor stream, and DigiBox 2103.

Bif Encoder 2104 passes the .bif file and the .od file to Mux 2105. Mux 2105 also accepts compressed audio and video files, as well as a .scr file that contains the stream description. Mux 2105 creates IPMP streams, descriptors and messages, encrypts the content streams, interleaves the received streams, and outputs Protected MPEG-4 Content File 2106, consisting of Initial Object Descriptor 2107 and Encrypted Content 2108. Initial Object Descriptor 2107 contains DigiBox 2103, as well as other information. Encrypted Content 2108 may include a scene graph stream (i.e., a BIFS stream), an object descriptor stream, IPMP streams, and encrypted content streams.

If DigiBox 2103 contains all keys and rules necessary to render all of the content, it may be unnecessary for Mux 2105 to create any IPMP streams. If additional keys or rules may be necessary for at least a portion of the content, Mux 2105 may incorporate those rules and keys into one or more additional DigiBoxes, and incorporate those DigiBoxes either in the IPMP stream or in the OD update stream.

FIG. 22 illustrates one manner in which control may be incorporated into an existing MPEG-4 stream. In this figure, Unprotected MPEG-4 Content File 2201 includes Initial Object Descriptor 2202 and Content 2203. The content may include a scene description stream (or BIF stream), an object descriptor stream, a video stream, an audio stream, and possibly additional content streams.

Unprotected MPEG-4 Content File 2201 is passed to Repackager 2204, which also accepts keys and rules. Repackager 2204 passes the keys and rules to IRP 2205, and receives DigiBox 2206 in return, containing keys, rules and initialization vectors. In an alternate embodiment, IRP 2205 may be incorporated directly into Repackager 2204.

5 Repackager 2204 demuxes Unprotected MPEG-4 Content File 2201. It inserts DigiBox 2206 into the Initial Object Descriptor and encrypts the various content streams. Repackager 2204 also adds the IPMP stream, if this is necessary (including if additional DigiBoxes are necessary).

10 Repackager 2204 outputs Protected MPEG-4 Content File 2207, consisting of Initial Object Descriptor 2208 (including DigiBox 2206) and Encrypted Content 2209 (consisting of various streams, including the IPMP streams, if necessary).

Real Networks Implementation

In one embodiment, the elements described above may be used in connection with information encoded in compliance with formats established by Real Networks, Inc.

15 The Real Networks file format (RMFF) is illustrated in FIG. 13. This format includes a block of headers at the beginning (Header 1301), followed by a collection of content packets (Content 1302), followed by an index used for seek and goto operations (Index 1303). Each file can contain several streams of different types. For each stream, there is a "Media Properties Header" (1304) used to describe the format of the media content (e.g., compression format) and provide stream specific information (e.g., parameters for the decompressor).

20 Real Networks streams can be protected by inserting a DigiBox into Header 1301 and encrypting the data packets contained in Content 1302. The altered format is illustrated in FIG.14, which shows Header 1401, including Media Properties Headers 1402 and 1403, which in turn contain DigiBoxes 1404 and 1405, respectively. The format also includes encrypted Content 1406 and Index 1407.

25 In one embodiment, the declared type of the data is changed from the standard Real Networks format to a new type (e.g., RNWK_Protected.) The old type is then saved. Changing the type forces the Real Networks player to load a "Trust Plugin," since this Plugin is registered as the only decoder module that can process streams of type "RNWK-Protected." The Trust Plugin opens the DigiBox, gets approval from the user, if it is needed, determines the original content type, loads a decoder plugin for the original content, and then decrypts and/or validates the content, passing it to the content decoder plugin to be decompressed and presented to the user.

30

- 30 -

In one embodiment, the specific alterations made to the Real Networks file format are the following:

- Increase the preroll time to force larger buffers on playback. In a current embodiment, an increase of 3 seconds is used. Larger buffers are needed because of the extra steps needed to decrypt the content.
- Modify each stream-specific header by changing the mime type to "RNWK-Protected", saving the old mime type in the decoder specific information and adding a content identifier and DigiBox to the decoder specific information. The DigiBox contains the key, initialization vector (IV), version information, and watermarking instructions. The key, IV and content identifier are generated automatically, or can be provided as command-line parameters. The same key, IV and content identifier are used for every stream.
- Content packets are selectively encrypted. In one embodiment, content packets whose start time in milliseconds is in the first half-second of each 5 seconds (i.e., $\text{starttime} \% 5000 < 500$) are encrypted. This encrypts approximately one-tenth of the content reducing encryption and decryption costs, and damages the content, sufficiently to prevent resale. The encryption algorithm can be DES using output-feedback mode or any similar algorithm. The initialization vector is computed for each packet by xoring the stream's IV with the packet's start time in milliseconds. Some information unique to the stream should also be xored into the IV. In one embodiment, the same IV is used for multiple packets whenever two or more streams have packets with the same start time. This usually happens for the first packet in each stream since they usually have start time 0. Other than the first packet, it is rare to have two packets have the same start time.

In one embodiment, these changes to the Real Networks file format are accomplished as is shown in FIG. 15. As is illustrated, RMFF file 1501 is formatted in the standard Real Networks RMFF format. This file is passed to Packager 1502. Also passed to Packager 1502 is Rights File 1503. Packager 1503 generates Protected RMFF File 1504, which includes various alterations as described above and as listed in FIG. 15, including the incorporation of one or more DigiBoxes in the header, encryption of the content, modification of the mime type, etc.

In one embodiment, the trust plugin described above is illustrated in FIGs. 16 and 17. FIG. 16 illustrates the standard Real Networks architecture. File 1601 (e.g., a streaming audio file in Real Networks format) is provided to Real Networks G2 Client

- 31 -

Core 1602. File 1601 may be provided to RealNetworks G2 Client Core 1602 from Server 1603, or through Direct Connection 1604.

Upon receipt of File 1601, Real Networks G2 Client Core 1602 accesses a rendering plugin appropriate to File 1601, based on information which is obtained from the header associated with File 1601. Rendering Plugins 1605 and 1606 are shown. If File 1601 is of a type which cannot be rendered by either Rendering Plugin 1605 or Rendering Plugin 1606, Real Networks G2 Client Core 1602 may attempt to access an appropriate plugin, e.g., by asking for the user's assistance or by accessing a site associated with the particular file type.

Rendering Plug-In 1605 or 1606 processes File 1601 in a conventional manner. This processing most likely includes decompression of File 1601, and may include other types of processing useful for rendering the content. Once this processing is complete (keeping in mind that the content is streamed, so that processing may be occurring on one set of packets at the same time that another set of packets is being rendered), File 1601 is passed back to Real Networks G2 Client Core 1602, which then passes the information to Rendering Device 1607. Rendering Device 1607 may, for example, be a set of stereo speakers, a television receiver, etc.

FIG. 17 illustrates the manner in which a trust plugin operates within the overall Real Networks architecture. Much of the architecture illustrated in FIG. 17 is the same as that illustrated in FIG. 16. Thus, File 1701 is provided to Real Networks G2 Client Core 1702 through Server 1703 or through Direct Connection 1704. The file is processed by Real Networks G2 Client Core 1702, using plugins, including Rendering Plugins 1705 and 1706, and is then passed to Rendering Device 1707.

FIG. 17 differs from FIG. 16 in its incorporation of Trust Plugins 1708 and 1709, and IRP 1710. When initially registered with Real Networks G2 Client Core 1702, Trust Plugins 1708 and 1709 inform Real Networks G2 Client Core 1702 that they can process content of type RNWK-Protected. Whenever Real Networks G2 Client Core 1702 encounters a stream of this type, it is then enabled to create an instance of the trust plugin to process the stream, e.g., Trust Plugin 1708. It then passes the stream to the trust plugin.

The stream passed to Trust Plugin 1708 may be in the format shown in FIG. 14. In such a case, Trust Plugin 1708 extracts DigiBox 1404 from Media Properties Header 1402. It also extracts the content id and original mime type from Media Properties Header 1402. The Trust Plugin first checks to see if any other stream with the same content identifier has been opened. If so, then DigiBox 1404 is not processed further. Instead, the key and IV from the box for this other stream are used. This avoids the time cost of opening a second

SUBSTITUTE SHEET (RULE 26)

box. Also, this ensures that a user is only asked to pay once even if there are multiple protected streams. By sharing content ids, keys, and IVs, several files can be played with the user only paying once. This is useful when SMIL is used to play several RMFF files as a single presentation.

5 In an alternate and possibly more secure embodiment, this check is not performed, and the key and IV from the current DigiBox are used even if another stream with the content identifier has already been opened.

10 If no other stream has been identified with the same content identifier, Trust Plugin 1708 passes DigiBox 1404 to IRP 1710. IRP 1710 may be a software process running on the same computer as Real Networks G2 Client Core and Trust Plugin 1708. IRP 1710 may run in a protected environment or may incorporate tamper resistance techniques designed to render IRP 1710 resistant to attack.

15 IRP 1708 may process DigiBox 1404 and extract a cryptographic key and an IV, which may then be passed to Trust Plugin 1708. Trust Plugin 1708 may then use this information to decrypt Encrypted Contents 1406.

20 Trust Plugin 1708 uses the original mime type information extracted from Media Properties Header 1402 to create an instance of the rendering plugin to be used for the content (e.g., Rendering Plugin 1705). Once this is done, Trust Plugin 1708 behaves like an ordinary rendering plugin to the Real Networks G2 Client Core 1702, in that Real Networks G2 Client Core 1702 passes streamed information to Trust Plugin 1708, which decrypts that information and passes it to Rendering Plugin 1705. From the perspective of Real Networks G2 Client Core 1702, Trust Plugin 1708 constitutes the appropriate rendering plugin, and the core is not aware that the information is being passed by Trust Plugin 1708 to a second plugin (e.g., Rendering Plugin 1705).

25 Similarly, from the point of view of Rendering Plugin 1705, Trust Plugin 1708 behaves like Real Networks G2 Client Core 1702. Thus although Rendering Plugin 1705 receives decrypted stream information from Trust Plugin 1708, Rendering Plugin 1705 operates exactly as if the information had been received directly from Real Networks G2 Client Core 1702. In this manner, content formatted for Rendering Plugin 1705 may instead be first processed by Trust Plugin 1708, without requiring any alteration to Real Networks G2 Client Core 1702 or Rendering Plugin 1705.

30 Trust Plugin 1708 may also perform other processing that may be helpful for security purposes. For example, Trust Plugin 1708 may watermark the decrypted file prior to passing it to Rendering Plugin 1705, keeping in mind that the watermark algorithm must be such that it will survive decompression of the file by Rendering Plugin 1705.

MP3 Embodiment

The techniques described above can also be applied to MP3 streaming content.

The MP-3 specification does not define a standard file format, but does define a bit stream, which is illustrated in FIG.18. In FIG. 18, MP-3 Bit Stream 1801 includes Content 1802. Content 1802 is divided into frames, shown as Frame 1803, Frame 1804 and Frame 1805. The dots between Frame 1804 and 1805 symbolize the fact that Content 1802 may include a large number of frames.

Each frame includes its own small header, shown in FIG. 18 as Headers 1806, 1807 and 1808.

Many MP3 players support a small trailer defined by the ID3 V1 specification, shown as Trailer 1809. This is a 128 byte trailer for carrying fields like artist, title and year, shown as Fields 1810, 1811 and 1812. The ID3 V1 trailer is ignored by players not designed to read such trailers, since it does not appear to be valid MP3 data.

FIG. 19 shows one embodiment of protection applied to the MP3 format. This protected format constitutes File 1908 and includes the following items:

- Unencrypted MP3 Content 1912. This is the first information encountered by a player, and will be rendered by any standard MP3 player. It can include a message to the user indicating that the content is protected and providing instructions as to how the content can be accessed (e.g., a URL for a trust plugin, instructions on payment mechanisms, etc.) Unencrypted MP3 Content 1912 may include a "teaser," consisting of an initial portion of the content (e.g., 30 seconds), which is rendered at no cost, thereby allowing a user to sample the content prior to making a decision to purchase it.

- Encrypted MP-3 Content 1901, which may include thousands of MP-3 frames. In one embodiment, the first eight frames out of every 32 frames are encrypted. Thus, one-quarter of the frames are rendered unuseable unless a player is able to decrypt them. In practice, this may render the content un-sellable or unuseable, without imposing excessive encryption or decryption costs. To further reduce encryption and decryption costs, only 32 bytes in each frame are encrypted. In a current embodiment, these are the first 32 bytes after the header and CRC information. In a different embodiment, a different 32 bytes may be encrypted in every frame. In a current embodiment, the content is encrypted with the DES using algorithm output-feedback mode. The initial IV for the file is randomly generated and then xored with the frame number to generate a unique IV for each frame.

Many alternate embodiments may exist, including encrypting more or less information, and using different encryption algorithms.

- ID3 V1 Trailer 1902, including 128 bytes.

• Content ID 1903, including 16 bytes. This is used by the player application to avoid opening DigiBoxes which it has already opened.

• DigiBox 1904, which may comprise approximately 18K bytes. It includes Key 1909, IV 1910 and Watermarking Instructions 1911. Watermarking Instructions 1911 may be used in a process of watermarking the associated content.

• Address 1905, which contains the address in the file of Content ID 1903 and consists of 4 bytes.

• Trust ID 1906, which identifies this trusted MP-3 file and consists of 16 bytes.

• ID3 V1 Trailer 1907, which is a copy of Trailer 1902.

A conventional MP3 player encountering File 1908 would be unable to render Content 1901, since at least a portion of that content is encrypted. Such a player would most likely read through to Trailer 1902 and cease processing at that point. A conventional player looking for the ID3 trailer information will seek to the end and find it.

FIG. 20 illustrates one embodiment of an MP3 player designed to process and render protected content. This figure shows MP3 Player 2001, which includes Buffer 2006 and Decompressor 2007, and renders content to Rendering Device 2008. In one embodiment, this is a modified version of a player distributed by Sonique.

Player 2001 obtains Protected MP3 File 2002 through any standard interface. Protected MP3 File 2002 may have the format illustrated in FIG. 19.

When Player 2001 is asked to play Protected MP3 File 2002, Player 2001 first calls Trust Plug-In 2003, which includes Approval Function 2009 and Decrypt Function 2005. Trust Plugin 2003 calls Approval Function 2009 to determine if Protected MP3 File 2002 is protected and whether authorization exists to play the file. Approval Function 2009 is first given a pointer to Protected MP3 File 2002. It then checks Protected MP3 File 2002 for the presence of Trust ID 1906. If Trust ID 1906 is not found, Approval Function 2009 returns an indicator that the file is not protected. Player 2001 then proceeds to render the file as a normal MP3 file.

If Trust ID 1906 is found, Approval Function 2009 checks Content ID 1903 to see if it matches the Content ID of a file that has already been opened.

If Protected MP3 File 2002 has not been previously opened, DigiBox 1904 is retrieved by Approval Function 2009, and is passed to IRP 2004, which may include software running in a protected environment, or incorporating tamper resistance. IRP 2004 attempts to open DigiBox 1904 in compliance with the rules associated with that DigiBox. One such rule may require, for example, that the user indicate assent to pay for use of the content. If DigiBox 1904 cannot be opened (e.g., the user refuses to pay) a value is

returned to Approval Function 2009 indicating that the file is protected and may not be played.

If DigiBox 1904 is opened in compliance with applicable rules, the key and IV are retrieved and passed to Decrypt Function 2005. The key and IV are stored with the content id for later re-use and Decrypt Function 2005 is initialized. This may improve overall system performance, since it reduces the number of times a DigiBox must be opened. Each such action may introduce significant latency.

On the other hand, storing this information in unprotected memory may reduce overall system security. Security may be enhanced either by not storing this information (thereby requiring that each DigiBox be opened, even if the corresponding file has already been opened through another DigiBox), or by storing this information in a protected form or in a secure location.

The stored key, IV and content id are referenced when Approval Function 2009 first checks Content ID 1903 to determine if it matches the Content ID of an already opened file. If the new Content ID matches a stored Content ID, Decrypt Function 2005 is reinitialized using the stored key and IV corresponding to the matching content id and a value indicating that this is a protected file for which play is authorized is returned to Approval Function 2009.

Once Protected MP3 File 2002 has been opened, each time Player 2001 needs a packet, Player 2001 reads it into Buffer 2006, strips off the header and CRC and passes the remaining data and a frame number to Decrypt Function 2005, which decrypts the frame if necessary, and returns it to Player 2001.

In a current embodiment, although audio content is encrypted, headers or trailers are not encrypted. This allows the Player 2001 to process information in headers or trailers without intervention from Approval Function 2009 or Decrypt Function 2005. This allows Player 2001 to place information such as playing time, artist and title into a playlist display, and initialize Decompressor 2007, without any action required from Trust Plugin 2003.

Commerce Appliance Embodiment

This section will describe an embodiment, comprising a Commerce Appliance architecture designed to allow persistent control of digital works in consumer electronics devices. Although this is described as a separate embodiment, it should be understood that the features of this embodiment may be combined with, or supplant, the features of any of the embodiments provided elsewhere in this description.

In one embodiment, this section will describe modifications to the MPEG-4 standard designed to support the association of persistent rules and controls with MPEG-4

content, as well as elements necessary for a Commerce Appliance to use such content. This is intended, however, merely as an example.

In one embodiment, shown in FIG. 23, each Commerce Appliance 2301 includes a CMPS ("Content Management and Protection System") 2302. Each CMPS is responsible for governing the use of controlled content, including decrypting the content and ensuring that the content is only used as permitted by associated rules.

Each governed digital work is associated with one or more CMPOs (Content Management Protection Object), e.g., CMPOs 2303. Each CMPO may specify rules governing the use of the digital work, and may include keys used to decrypt the work.

CMPOs may be organized in an hierarchical fashion. In one embodiment, a content aggregator (e.g., a cable channel, a web site, etc.) may specify a Channel CMPO ("CCMPO") used to associate certain global rules with all content present on that channel. Each independent work may in turn have an associated Master CMPO ("MCMPO") used to associate rules applicable to the work as a whole. Each object (or Elementary Stream, in MPEG-4) may have associated with it a CMPO containing rules governing the particular object.

In one exemplary application, Commerce Appliance 2301 may be an MPEG-4 player containing CMPS 2302. Upon receipt of a user command to play a particular work, CMPS 2302 may download a MCMPO associated with the work and obtain rules, which may include conditions required for decryption and viewing of the work. If the rules are satisfied, CMPS 2302 may use keys from the MCMPO to decrypt any Elementary Streams ("ES"), and may pass the decrypted ESs into the buffers. Composition and rendering of the MPEG-4 work may thereafter proceed according to the MPEG-4 standard, except that any storage location or bus which may contain the work in the clear must be secure, and CMPS 2302 may have the ability to govern downstream processing, as well as to obtain information regarding which AVOs were actually released for viewing.

In a variation, the process of obtaining and governing the work may include downloading a CCMPO which applies rules governing this and other works. If rules contained in the CCMPO are satisfied, CMPS 2302 may obtain a key used to decrypt the MCMPO associated with the particular work to be viewed.

In another variation, a CMPO may be associated with each ES. In this variation, the MCMPO supplies one or more keys for decryption of each CMPO, and each CMPO may in turn supply a key for decryption of the associated ES.

Commerce Appliance 2301 is a content-rendering device which includes the capability of supporting distributed, peer management of content related rights by securely

applying rules and controls to govern the use of content. Commerce Appliance 2301 may include general-purpose functions devoted to acquisition and managed rendering of content (e.g., a DVD (and/or any other optical disk format) player is able to play a DVD (and/or any other optical disk format) disk and output content to a television.) Commerce
5 Appliance 2301 may make use of any of the means for protecting and using digital content on high capacity optical disk, in one non-limiting example, a DVD disk, as described in the aforementioned Shear patent application.

Commerce Appliance 2301 also includes special-purpose functions relating to other management and protection of content functions. These special-purpose functions may be
10 supported by one or more embedded or otherwise included CMPS 2302 in the form of a single CMPS or a cooperative CMPS arrangement, and may include a user interface (e.g., User Interface 2304) designed to display control-related information to the user and/or to receive control-related information and directions from the user. Commerce Appliance 2301 may also be designed so that it is networkable with other Commerce Appliances (e.g.,
15 a set-top box connected to a DVD player and a digital television) and/or with other devices, such as a computer arrangement, which may also include one or more CMPSs.

An important form of Commerce Appliance specifically anticipates secure coupling on a periodic or continual fashion with a computer managed docking environment (e.g., a standalone computer or other computer managed device which itself may be a Commerce
20 Appliance) where the one or more CMPSs of the Commerce Appliance interoperate with the docking environment to form a single user arrangement whose performance of certain functions and/or certain content usage events is enabled by such inter-operation through, at least in part, cooperation between CMPSs and content usage management information of the Commerce Appliance and the trust environment capabilities of the docking
25 environment, (e.g., further one or more CMPSs and content usage management information, such as, for example, information provided by use of CI).

An exemplary Commerce Appliance may be designed to comply with the emerging MPEG-4 standard for the formatting, multiplexing, transmission, compositing, and rendering of video and other types of information.

30 Commerce Appliance 2301 may be any computing device, one non-limiting example of which is a Personal Computer (PC) that includes MPEG-4 software (and/or hardware) for rendering content. In accordance with the present invention, the PC may also use one or more CMPSs as described herein.

35 The commerce appliance function is not restricted to streamed channel content but may include various browser-type applications consisting of aggregated composite content

such as still imagery, text, synthetic and natural video and audio and functional content such as applets, animation models and so on. these devices include browsers, set-top boxes, etc.

Content Management and Protection System (CMPS)

5 Each commerce appliance includes one or more CMPS (e.g., CMPS 2302). The CMPS is responsible for invocation and application of rules and controls, including the use of rules and controls to govern the manner in which controlled content is used.

Particular functions of CMPS 2302 include the following:

(a) Identification and interpretation of rules.

10 CMPS 2302 must determine which rules are to be applied, and must determine how those rules are to be interpreted in light of existing state information. In one embodiment, this requires that CMPS 2302 obtain and decrypt one or more CMPOs 2303 associated with a work.

(b) Identification of content associated with particular rules.

15 CMPS 2302 must determine which content is governed by particular one or more rules. This may be accomplished by obtaining information from one or more CMPOs 2303 and/or other CI. In one embodiment, a CCMPO may identify a set of works, a MCMPO may identify a particular work and a CMPO may identify a particular ES or Audio Visual Object ("AVO").

20 (c) Decryption of content as allowed by the rules.

CMPS 2302 may be designed so that all content is routed through CMPS 2302 for decryption, prior to reinsertion into the data flow required by the relevant standard. In the case of MPEG-4, for example, the output from Demux 2305 may be fed into CMPS 2302. CMPS 2302 may then decrypt the content and, if relevant rules and controls are satisfied, feed the content into the MPEG-4 buffers. From that point, the data flow associated with the content may be as described by MPEG-4.

(d) Control of content based on rules.

30 CMPS 2302 may be used to control usage of content after the initial decryption, for example, through the use of secure event management as described in the incorporated Ginter '333 patent application. In the case of MPEG-4 systems, this may require that CMPS 2302 exercise control over hardware and/or software which performs the following functions: demuxing (performed by Demux 2305), decompression/buffering/decode into AVOs (performed by Scene Descriptor Graph 2306, AVO Decode 2307 and Object Descriptors 2308), scene rendering (performed in Composite and Render 2309).

CMPS 2302 may also be used to control use and consequences according to: (1) generational copy protection rules such as the CGMS and/or SGMS standards; (2) various Conditional Access control methods, such as those proposed and/or implemented by NDS as described in MPEG-4 document M2959, DAVIC "Copyright Control Framework" document, and in other publications; (3) a Rights Management Language, such as those proposed in the Ginter '333 patent application and/or as described by U.S. Patent No. 5,638, 443 to Stefik, et al.; (4) use policies described in accordance with AT&T's Policy Maker, as described by Blaze, Feigenbaum, and Lacy; (5) the CCI layer bits for IEEE 1394 serial bus transmission as specified by the DTDG subgroup of the DVD Copy Protection Technical Working Group and/or as implemented by the Hitachi, Intel, Matsushita, Sony and Toshiba proposed standard (hereafter "the five company proposal"); (6) controls transmitted using any secure container technology such as, for example, IBM Cryptolope; (7) any other means for specifying use rules and consequences.

(e) Monitoring use of content.

CMPS 2302 may be used to monitor content to: (i) ensure that rules are being complied with; (ii) ensure that no attempts are being made to tamper with the system or protected content; and (iii) record information used by rules, including usage information needed for payment purposes.

(f) Updating user budgets.

CMPS 2302 may be used to update user or other budgets to reflect usage.

(g) Exhaust information.

CMPS 2302 may be used to output payment and usage information ("exhaust information") to external processes, including one or more Commerce Utility Systems.

(h) Hardware identification and configuration.

(i) Obtaining new, additional, and/or augmented rules from an external process, one non-limiting example of which is a Rights and Permission Clearinghouse as described in the incorporated Shear patent application.

(j) Receiving keys, digital credentials, such as certificates, and/or administrative information, from certifying authorities, deployment managers, clearinghouses, and/or other trusted infrastructure services.

(k) Securely sending and/or receiving user and/or appliance profiling and/or attribute information.

(l) Securely identifying a user or a member of a class of users who requests content and/or CMPO and/or CMPS usage.

- 40 -

(m) Securely certifying or otherwise guaranteeing the authenticity of application code, for example certifying within CMPO 2301 and/or CMPS 2302 that application code containing rules and/or other application information, such as information written in Java code for conditional execution within a Commerce Appliance, and/or that executes at least in part outside of CMPO 2301 and/or CMPS 2302, has not been altered and/or has been delivered by a guaranteed (e.g., trusted) party.

(n) Securely processing independently delivered CI, such as described in the incorporated Ginter '333 patent application, to perform content usage control that protects the rights of plural, independent parties in a commerce value chain.

(o) Securely performing watermarking (including, for example fingerprinting) functions, for example as described in the Ginter '333 patent application and as incorporated herein, for example including interpreting watermarking information to control content usage and/or to issue an event message, wherein such event message may be reported back to a remote authority, such as, for example, a MCMPO rights clearinghouse management location.

CMPS 2302 may be used to identify and record the current hardware configuration of the Commerce Appliance and any connected devices (e.g., which loudspeakers are available, identification of attached monitors, including whether particular monitors have digital output ports, etc.) If attached devices (such as loudspeakers) also include CMPSs, the CMPSs may be used to communicate for purposes of coordination (e.g., a CMPS in a set-top box and/or loudspeaker arrangement may communicate with a CMPS in a downstream digital television or other display device to establish which CMPS will be responsible for governance or the nature of cooperative governance through a virtual rights process, said process optionally involving a rights authority server that may find, locate, provide, aggregate, distribute, and/or manage rights processes, such as described in the aforementioned Shear patent application, for employing plural CMPSs, for example, for a single user content processing and usage arrangement).

The present invention includes arrangements comprising plural Commerce Appliances and/or CMPSs in one or more user locations, non-limiting examples of which include a home, apartment, loft, office, and/or vehicle, such as a car, truck, sports utility vehicle, boat, ship, or airplane, that may communicate among themselves at least occasionally and may comprise a virtual network that operates in a logically cooperative manner, through at least in part the use of such CMPSs, to ensure optimal commercial flexibility and efficiency and the enforcement of rights of commerce value chain participants, including financial and copyright rights of providers, infrastructure rights of

SUBSTITUTE SHEET (RULE 26)

appliance providers, societal rights of government and/or societal bodies, and privacy rights of all parties, including consumers. Information related to interaction among such a network of value chain participants, including content usage auditing, content usage consequence, and CI specification, can be securely, variably reported to parties having right to such information, through, at least in part, use of such CMPSs, for example, as described in the aforementioned Ginter '712 patent application regarding the information reporting functioning of VDE nodes.

In one embodiment, shown in FIG. 24, CMPS 2401 consists of special-purpose hardware and resident software or firmware. These include the following:

(a) One or more processors or microcontrollers e.g. CPU 2402. CPU 2402 controls the overall processing of CMPS 2401, including execution of any necessary software.

(b) One or more external communications ports, e.g., Port 2403. Port 2403 communicates with External Network 2404, which may include LANs, WANs or distributed networks such as the Internet. External communications ports may also include one or more IEEE 1394 serial bus interfaces.

(c) Memory 2405. Types of memories which may be included in Memory 2405-- and examples of the information they may store -- are the following:

i. ROM 2406. ROM 2406 may include any information which is permanently stored in CMPS 2401, such as (1) CMPS Operating System 2407 and/or CMPS BIOS 2408, (2) Rules/Controls 2409 which are permanently stored in the CMPS; (3) Control Primitives 2410 which may be used to build rules or controls; (4) Keys 2411 associated with the CMPS, including a Public/Private Key Pair; (5) one or more Certificates 2412 designed to identify CMPS 2401 and/or the device, including version information; (6) Hardware Signature Information 2413 used to check for tampering (e.g., a hashed signature reflecting the expected hardware state of the device).

ii. RAM 2414. RAM 2414 may hold current state information needed by CMPS 2401, as well as information temporarily stored by CMPS 2401 for later use. Information stored in RAM 2414 may include the following: (1) Software 2415 currently executing in CPU 2402; (2) CMPOs 2416 which are currently active; (3) Content Object Identification 2417 of those content objects which are currently active (in an MPEG 4 system this would constitute, for example, an identification of active AVOs); (4) Rules 2418 which are currently active; (5) State Information 2419 regarding the current state of use of content, including an identification of any higher-order organization (in an MPEG-4 system this would constitute an identification of the scene descriptor tree and the current

- 42 -

state of composition and rendering); (6) Stored Exhaust Information 2420 relating to use and/or the user, designed for external transmission; (7) Updated Budget Information 2421; (8) Content 2422; (9) Active Content Class Information 2423; and (10) Active User Identification 2424, including identification characteristic information.

5 iii. NVRAM 2425 (e.g., flash memory). This type of memory may hold information which is persistent but changeable, including at least some: (1) Budget Information 2426; (2) User Information 2427, such as identification, credit card numbers; preferred clearinghouses and other Commerce Utility Systems; (3) User Preferences 2428, such as preferences, profiles, and/or attribute information; and (4) Appliance Information
10 2429, such as attribution and/or state information.

The types of information described above and stored in CMPS Memory 2405 may be stored in alternative of the above memory types, for example, certain budget information may be located in ROM, information regarding specific one or more clearinghouses may be stored in ROM, certain active information may be moved into NVRAM, etc.

15 Budget information may include stored budgets made up of, for example:

- (1) electronic cash;
- (2) pre-authorized uses (e.g., based on a prepayment, the user has the right to watch 12 hours of programming).
- (3) Security budgets related to patterns reflecting abnormal and/or
20 unauthorized usage, for example, as described in the incorporated Shear patent, wherein such budgets restrict and/or report certain cumulative usage conduct.
- (4) electronic credit, including credit resulting from usage events such as
25 attention to promotional material and/or the playing of multiple works from one or more classes of works (e.g., certain publisher's works) triggering a credit or cash refund event and/or a discount on future playing of one or more of such publisher's works, such as other works provided by such publisher.

30 User information may include the following types of information for one or more authorized users of the Commerce Appliance:

- (1) Name, address, telephone number, social security number or other
35 identifier
- (2) Information used to authenticate the user, which may include a user selected password and/or biometric data, such as fingerprints, retinal data, etc.
- (3) User public/private key pair

(4) User attribute and/or profiling information.

- iv. Removable Memory 2430. This may include any type of removable memory storage device, such as smart cards, floppy disks or DVD disks. If the commerce appliance is designed to play content received on removable memory devices (e.g., a DVD player), that capability may be used for purposes of the CMPS.

Memory 2405 may include a protected database, in which certain control, budget, audit, security, and/or cryptographic information is stored in secure memory, with complete information stored in an encrypted fashion in unsecure memory.

(d) Encryption/Decryption Engine 2431. CMPS 2401 must include a facility for decrypting received information, including content and CMPOs and/or other. CMPS 2401 may also include a facility for encrypting information if such information is to be transmitted outside the secure boundaries of CMPS 2401. This may include exhaust sent to clearinghouses or other external repositories; and content sent across unsecured buses for usage, such as content sent across IEEE 1394 Serial Bus 2432 to a computer central processing arrangement or to a viewing device such as a monitor, wherein a receiving CMPS may be employed to control such content's usage, including, for example, decrypting such content, as appropriate. Encryption/Decryption Engine 2431 may include a Random Number Generator 2433 used for the creation of keys or key pairs that can be used to identify and assure the uniqueness of CMPSs and support the opening of secure communication channels between such secure content control secure encryption/decryption arrangements.

(e) Secure Clock/Calendar 2434. CMPS 2401 may include Secure Clock/Calendar 2434 designed to provide absolute information regarding the date and time of day, information regarding elapsed absolute time, and/or relative timing information used to determine the elapsed time of operations performed by the system. Secure Clock/Calendar 2434 may include Battery Back Up 2435. It may further include Sync Mechanism 2436 for synchronization with outside timing information, used to recover the correct time in the event of a power loss, and/or to check for tampering.

(f) Interface 2437 to blocks used for content rendering and display. This interface is used for controlling rendering and display, based on rules, and for obtaining feedback information, which may be used for budgeting purposes or for providing information to outside servers (e.g., information on which content was actually displayed, which choices the user invoked, etc.) In the case of an MPEG-4 player such as is shown in

FIG. 23, this may include control over Commerce Appliance circuitry which handles, for example, buffering, the scene descriptor graph, AVO decode, object descriptors and composite and rendering (e.g., Control Lines 2310, 2311 and 2312).

5 Feedback Path 2313 from Composite and Render block 2309 may allow CMPS 2302 to determine whether and when content has actually been released to the viewer. For example, Composite and Render block 2309 can issue a start event to CMPS 2302 when an AVO object is released for viewing, and can issue a stop event to CMPS 2302 when the AVO object is no longer being viewed.

10 Feedback from Composite and Render block 2309 may also be used to detect tampering, by allowing CMPS 2302 to match the identification of the objects actually released for viewing with the identification of the objects authorized for release. Start and end time may also be compared with the expected elapsed time, with a mismatch possibly indicative of the occurrence of an unauthorized event.

In one embodiment, the following protocol may be used for feedback data:

15 **start <id>, T, <instance number><clock time><rendering options>**

Sent if elementary stream <id> is reachable in the SD-graph at time T , but not at time $T-1$.

end <id>, T, <instance number><clock time><rendering options>

20 T constitutes presentation time, clock time constitutes the wall clock time, including day and date information, and rendering options may include such information as QoS and rate of play (e.g., fast forward).

25 Sent if elementary stream <id> is reachable in the SD-graph at time $T-1$ but not at time T . A SD-graph stream is reachable if, during traversal of the SD-graph for display update, the renderer encounters a node that the SD-graph update stream <id> created or modified. This implies that all nodes in the tree need an update history list. This list need not be as large as the number of streams. Further, it can be labeled to indicate if the CMPS will be watching for stream, if not labeled it will not record them. An AV elementary stream is reachable if the stream's content was rendered.

30 For SD-graph update streams, the object instance number is ignored. For AV streams, the instance number can be used to disambiguate the case where the display shows two or more instances of the same data stream simultaneously. Instance numbers do not have to count up. In this case, they are simply a unique id that allows the CMPS to match a start event with an end event.

35 In a second embodiment, CMPS 2302 may include some special purpose hardware in combination with general purpose hardware which is also used for other functions of the

device. In this embodiment, care must be taken to ensure that commercially trusted CMPS functions are performed in a secure and tamper-resistant manner, despite the use of general purpose hardware. Each of the elements recited above may include dedicated CMPS functions and general purpose device functions:

5 (a) CPU/microcontroller. This may include one or more devices. If more than one device is included (e.g., a CPU and a DSP, a math coprocessor or a commerce coprocessor), these devices may be included within the same package, which may be rendered tamper-resistant, or the devices may communicate on a secure bus. The CPU may include two modes: a secure CMPS mode, and an unsecure general purpose mode. The
10 secure CMPS mode may allow addressing of secure memory locations unavailable to the processor in general purpose mode. This may be accomplished, for example, by circuitry which remaps some of the available memory space, so that, in unsecure mode, the CPU cannot address secure memory locations.

(b) External communications ports. If the device, for example, a Commerce
15 Appliance, is capable of receiving content or other information through a communications port (e.g., a cable connection, an Internet connection), this communications port can be used for CMPS purposes. In such a case, CMPS accesses to the external communications port is preferably designed to avoid or minimize interference with the use of such port for receipt of content.

20 (c) Memory. In some applications and embodiments, it is possible to operate a Commerce Appliance without NVRAM, wherein information that may be needed for CMPS operation that would employ NVRAM would be loaded into RAM, as required. ROM, RAM and NVRAM may be shared between CMPS uses and general uses. This can be accomplished in any of the following ways, or in a combination of these ways: (1)
25 Some memory space may be rendered off-limits to general purpose uses, for example by remapping; (2) the entirety of the memory may be rendered secure, so that even portions of the memory being used for non-secure purposes cannot be observed or changed except in a secure and authorized manner; (3) CMPS information may be stored in an encrypted fashion, though this requires at least some RAM to be secure, since the CMPS will require
30 direct access to unencrypted information stored in RAM.

(d) Encryption/decryption engine. Encryption and decryption functions, including key generation, may be handled by special purpose software running on a general purpose processor arrangement, particularly, for example, a floating point processor or DSP arrangement. That processor arrangement may also be used for purposes of
35 decompressing and displaying content and/or for handling watermarking/fingerprinting

insertion and/or reading. Alternatively, the device may include native encryption and decryption functions. For example, various emerging standards may require at least some degree of encryption and decryption of content designed to be passed across unsecure buses within and among devices such as DVD players, such as the “five company proposal” and other IEEE 1394 related initiatives. Circuitry designed to perform such encryption and decryption may also be usable for CMPS applications.

(e) Secure clock/calendar. The underlying device may already require at least some clock information. MPEG-4, for example, requires the use of clock information for synchronization of Elementary Streams. A secure CMPS clock can also be used for such purposes.

In a third embodiment, CMPS 2302 can be primarily software designed to run on a general purpose device which may include certain minimal security-related features. In such a case, CMPS 2302 may be received in the same channel as the content, or in a side-band channel. An I-CMPO and/or other CI may specify a particular type of CMPS, which Commerce Appliance 2301 must either have or acquire (e.g., download from a location specified by the I-CMPO), or CMPS 2302 may be included, for example, with an I-CMPO.

A software CMPS runs on the CPU of the Commerce Appliance. This approach may be inherently less secure than the use of dedicated hardware. If the Commerce Appliance includes secure hardware, the software CMPS may constitute a downloadable OS and/or BIOS which customizes the hardware for a particular type of commerce application.

In one embodiment, a software CMPS may make use of one or more software tamper resistance means that can materially “harden” software. These means include software obfuscation techniques that use algorithmic means to make it very difficult to reverse engineer some or all of a CMPS, and further make it difficult to generalize from a reverse engineering of a given one or more CMPS. Such obfuscation is preferably independent of source code and object code can be different for different CMPSs and different platforms, adding further complexity and separation of roles. Such obfuscation can be employed “independently” to both CI, such as an CMPO, as well as to some or all of the CMPS itself, thus obscuring both the processing environment and executable code for a process. The approach is also applicable for integrated software and hardware implementation CMPS implementations described above. Other tamper resistance means can also be employed, including using “hiding places” for storing certain state information in obscure and unexpected locations, such as locations in NV memory used for other purposes, and data hiding techniques such as watermarking/fingerprinting.

Association of CMPS With a Commerce Appliance

A CMPS may be permanently attached to a particular device, or may be partially or fully removable. A removable CMPS may include software which is securely loaded into a Commerce Appliance, and/or removable hardware. A removable CMPS may be
 5 personalized to one or more particular users, including user keys, budget information, preferences, etc., thereby allowing different users to use the same Commerce Appliance without commingling budgets and/or other rights, etc.

A CMPS may be designed for operation with certain types of content and/or for operation with certain types of business models. A Commerce Appliance may include
 10 more than one type of CMPS. For example, a Commerce Appliance designed to accept and display content pursuant to different standards may include one CMPS for each type of format. In addition, a Commerce Appliance may include a CMPS provided by a particular provider, designed to preferentially display certain types of content and to preferentially bill for such content through a particular channel (e.g., billing to one or more particular
 15 credit cards and/or using a particular one or more clearinghouses).

Source of Rules

The CMPS must recognize those rules which are to be applied to particular content. Such rules may be received by the CMPS from a variety of sources, depending on the particular embodiment used:

20 (a) CMPO. The rules may be included within a CMPO (e.g., CMPO 2303) and/or other CI. The CMPO and/or other CI may be incorporated within a content object or stream (as, e.g., a header on an MPEG-4 ES), and/or may be contained within a dedicated content object or stream encoded and received as per the underlying standard (e.g., an MPEG-4 CMPO ES), and/or may be received outside the normal content stream,
 25 in which event it may not be encoded as per the underlying standard (e.g., a CMPS received as an encrypted object through a sideband channel).

(b) CMPS. Rules may be permanently and/or persistently stored within a CMPS, e.g., Rules 2409. A CMPS may include default rules designed to handle certain
 30 situations, for example, where no CMPO and/or other necessary CI is received (e.g., content encoded under an earlier version of the standard which did not incorporate CMPOs, including MPEG-4 version 1). Complete rules which are stored within the CMPS may be directly or indirectly invoked by a CMPO and/or other CI. This may occur through the CI identifying particular rules through a pointer, and/or it may occur through the CI
 35 identifying itself and the general class of control it requires, with the CMPS then applying particular rules specific to that CMPS.

- 48 -

Rule "primitives" may also be stored within the CMPS (e.g., Control Primitives 2410). The CMPO and/or other CI may invoke these primitives by including a sequence of macro-type commands, each of which triggers a sequence of CMPS primitives.

5 (c) User. The user may be given the ability to create rules relating to the particular user's preferences. Such rules will generally be allowed to further restrict the use of content, but not to expand the use of content beyond that which would otherwise be allowed. Examples include: (a) rules designed to require that certain types of content (e.g., adult movies) only be accessible after entry of a password and/or only to certain CMPS users (e.g. adults, not children, as, for example, specified by parents and/or a societal body such as a government agency); (b) rules designed to require that only 10 particular users be allowed to invoke operations requiring payment beyond a certain limit and/or aggregate payment over a certain amount.

The user may be allowed to create templates of rules such as described in the 15 aforementioned Ginter '333 patent application (and incorporated herein). In addition, a CMPS arrangement, and/or a particular CMPO and/or other CI, may restrict the rules the user is allowed to specify. For example, a CI may specify that a user can copy a work, but cannot add rules to the work restricting the ability of a recipient to make additional copies (or to be able to view, but only after a payment to the first user). User supplied one or more rules may govern the use of -- including privacy restrictions related to -- payment, 20 audit, profiling, preference, and/or any other kind of information (e.g., information result as a consequence of the use of a CMPS arrangement, including, for example, use of secured content). Such user supplied one or more rules can be associated with the user and/or one or more Commerce Appliances in a user arrangement, whether or not the information is aggregated according to one or more criteria, and whether or not user and/or appliance 25 identification information is removed during aggregation and/or subsequent reporting, distribution, or any other kind of use.

The ability to allow the user to specify rules allows the CMPS to subsume (and thereby replace) V-chips, since a parent can use content rating information to specify 30 precisely what types of information each viewer will be allowed to watch (e.g., violent content can only be displayed after entry of a certain password and/or other identifier, including, for example, insertion of a removable hardware card (smart or rights card) possessed by a user).

(d) External network source. The rules may be stored on an external server. Rules may be addressed and downloaded by the CMPS if necessary (e.g., either the CMPO 35 and/or other CI and/or the CMPS contains a pointer to certain rules location(s), such as one

SUBSTITUTE SHEET (RULE 26)

or more URLs). In addition, content providers and/or clearinghouses may broadcast rules designed for general applicability. For example, a content provider might broadcast a set of rules providing a discount to any user participating in a promotional event (e.g., by providing certain user information). Such rules could be received by all connected devices, could be received by certain devices identified as of interest by the content provider (e.g., all recent viewers of a particular program, as identified by exhaust information provided by the CMPS to a clearinghouse and/or all members having certain identity characteristics such as being members of one or more classes) and/or could be posted in central locations.

Example Embodiment

In one embodiment, a set of MPEG-4 Elementary Streams may make up a work. The Elementary Streams may be encrypted and multiplexed together to form an Aggregate Stream. One or more CMPOs may be present in such stream, or may otherwise be associated with the stream. Options are as follows:

1. Content may be streamed or may be received as static data structures.
2. A Work may be made up of a single stream or data structure, or of many separately addressable streams or data structures, each of which may constitute an Object.
3. If a Work is made up of separately addressable streams or data structures, those streams or data structures may be multiplexed together into an Aggregate Stream, or may be received separately.
4. If streams or data structures are multiplexed together into an Aggregate Stream, the streams or data structures may be encrypted prior to such multiplexing. The Aggregate Stream itself may be encrypted, whether or not the underlying streams or data structures are encrypted. The following possibilities therefore exist: (a) individual streams/data structures are unencrypted (in the clear), the Aggregate Stream is unencrypted; (b) individual streams/data structures are unencrypted prior to multiplexing, the Aggregate Stream is encrypted following multiplexing; (c) individual streams/data structures are encrypted prior to multiplexing, the Aggregate Stream is not encrypted following multiplexing; or (d) individual streams/data structures are encrypted prior to multiplexing, the Aggregate Stream is encrypted following multiplexing.
5. A CMPO may be associated with a channel (CCMPO), a work (MCMPO) or an individual Object (CMPO).
6. A CMPO may be received prior to the controlled data, may be received contemporaneously with the data, or may be received after the data (in which event use of the data must wait until the CMPO has been received).
7. A CMPO may be received as part of an Aggregate Stream or separately.

8. If a CMPO is received as part of the Aggregate Stream, it may be multiplexed together with the individual streams or data structures, or may constitute a separate stream or data structure.

5 9. If a CMPO is multiplexed within the Aggregate Stream, it may be encrypted or nonencrypted. If encrypted, it may be encrypted prior to multiplexing, and/or encrypted after multiplexing, if the entire Aggregate Stream is encrypted.

10 10. If a CMPO is received as part of the Aggregate Stream, it may be (a) a part of the stream or data structure which holds the content (e.g., a header); (b) a separate stream or data structure encoded pursuant to the same format as the streams or data structures which hold the content (e.g., an MPEG-4 ES) or (c) a separate stream or data structure encoded under a different format designed for CMPOs.

15 11. If a CMPO is a part of the stream or data structure which holds the content, it may be (a) a header which is received once and then persistently maintained for control of the content; (b) a header which is received at regular intervals within the stream or data structure; or (c) data distributed throughout the stream or data structure.

These various scenarios give rise to different requirements for demultiplexing and decryption of the CMPOs. FIG. 25 illustrates the following embodiment:

20 1. Aggregate Stream 2501 is made up of multiplexed ESs (e.g., ES 2502 and 2503). A combination of such ESs makes up a single work. Aggregate Stream 2501 is generated by a cable aggregator and received by a user's set-top box as one of a number of channels.

2. CCMPOs 2504 corresponding to each channel are sent along the cable in Header 2505 at regular intervals (e.g., once per second). When the set-top box is turned on, it polls each channel, and downloads all current CCMPOs. These are stored persistently, and are changed only if a new CCMPO is received which differs from prior CCMPOs.

25 3. When the user selects a channel, the set-top box addresses the associated CCMPO. The CCMPO may specify, for example, that content in this particular channel may only be accessed by subscribers to the channel. A CMPS within the set-top box accesses a user profile persistently stored in NVRAM and determines that the user is a subscriber. The CMPS deems the CCMPO rule to have been satisfied.

30 4. The CMPS obtains an identifier for the MCMPO associated with the work (video) currently streaming on the channel and a key for the MCMPO. If works are received serially on the channel (e.g., a television channel in which one work is provided at a time), the received MCMPO identifier may include don't care bits so that it can address any MCMPO currently on the channel.

- 51 -

5 5. The CMPS begins demuxing of Aggregate Stream 2501 (this may occur in parallel with the preceding step), and obtains the MCMPO, which is encoded into an ES multiplexed within the Aggregate Stream (e.g., MCMPO 2506). Although each ES within Aggregate Stream 2501 has been encrypted, Aggregate Stream 2501 was not encrypted following multiplexing. This allows the CMPS to demultiplex Aggregate Stream 2501 without decrypting the entire Aggregate Stream.

6. The CMPS identifies the ES which constitutes the MCMPO (e.g., ES 2503). The CMPS downloads one complete instance of MCMPO 2506 into an internal buffer, and uses the key received from CCMPO 2504 to decrypt MCMPO 2506.

10 7. The CMPS determines which rules are applied by MCMPO 2506. MCMPO 2506 might, for example, include a rule stating that the user can view the associated work with advertisements at a low fee, but must pay a higher fee for viewing the work without advertisements.

15 8. The CMPS generates an options menu, and displays that menu on the screen for the user. The menu specifies the options, including the cost for each option. Additional options may be specified, including payment types.

9. The user uses a remote control pointing device to choose to view the work at a lower cost but with advertisements. The user specifies that payment can be made from an electronic cash budget stored in the CMPS.

20 10. The CMPS subtracts the specified amount from the budget persistently stored in NVRAM, and generates and encrypts a message to a server associated with the cable. The message transfers the required budget to the server, either by transferring electronic cash, or by authorizing a financial clearinghouse to transfer the amount from the user's account to the cable provider's. This message may be sent immediately, or may be buffered to be sent later (e.g., when the user connects the device to the Internet). This step may be taken in parallel with decryption of the content.)

25 11. The CMPS obtains from MCMPO 2506 a set of keys used to decrypt the Elementary Streams associated with the work (e.g., ES 2502). The CMPS also obtains identifiers for the specific ESs to be used. Since the user has indicated that advertisements are to be included, the MCMPO identifies ESs associated with the advertisements, and identifies a Scene Descriptor Graph which includes advertisements. A Scene Descriptor Graph which does not include advertisements is not identified, and is not passed through by the CMPS.

30 12. The CMPS passes the decrypted ESs to the MPEG-4 buffers. The normal process of MPEG-4 decoding, compositing and rendering then takes place. The Composite

SUBSTITUTE SHEET (RULE 26)

- 52 -

and Render block outputs Start and Stop events for each object released for viewing. The CMPS monitors this information and compares it to the expected events. In particular, the CMPS confirms that the advertisements have been released for viewing, and that each operation has occupied approximately the expected amount of time.

5 In another embodiment, a set-top box containing a CMPS (e.g., CMPS 2302 from FIG. 23) may have a cable input (e.g., carrying M4 Bit Streams 2314 and CMPOs 2303). The cable may carry multiple channels, each made up of two sub-channels, with one sub-channel carrying MPEG-4 ESs (e.g., M4 Bit Streams 2314), and the other sub-channel carrying CMPOs (e.g., CMPOs 2303). The sub-channel carrying CMPOs 2303 could be
10 routed directly to CMPS 2302, with the ES channel being routed to a decryption block (operating under control of the CMPS, e.g., CR&D 2315), and then to the MPEG-4 buffers (e.g., buffers associated with Scene Descriptor Graph 2306, AVO Decode 2307 and Object Descriptors 2308). In this case, if the ESs are not encrypted, they proceed unchanged through the decryption block and into the buffers. This may occur, for example, if the ESs
15 are being broadcast for free, with no restrictions, and/or if they are public domain information, and/or they were created prior to inclusion of CMPOs in the MPEG-4 standard.

Such an embodiment might include timing synchronization information in the CMPO sub-channel, so that CMPOs can be synchronized with the associated ESs.

20 The concept of incorporating two separate streams, one consisting of control information and connected directly to the CMPS, and the other consisting of ESs, may support a high degree of modularization, such that the formats of CMPOs, and particular types of CMPS's, may be changed without alteration to the underlying ES format. For example, it may be possible to change the CMPO format without the necessity for
25 reformatting content ESs. To take another example, it may be possible to upgrade a Commerce Appliance by including a new or different CMPS, without the necessity for any changes to any of the circuitry designed to demultiplex, composite and render the content ESs. A user might obtain a CMPS on a smart card or other removable device, and plug that device into a Commerce Appliance. This could be done to customize a Commerce
30 Appliance for a particular application or for particular content.

CMPS Interface to a CE Device

A CMPS may be designed to present a standardized interface between the general-purpose functionality of a consumer electronics device and any relevant CMPOs and/or other CI and protected content. For example, a CMPS could be designed to accept CI and
35 encrypted ESs, and output decrypted ESs into the device's buffers. In such a case, the

manufacturer of the device would be able to design the device in compliance with the specification (e.g., MPEG-4), without concern about commerce-related extensions to the standard, which extensions might differ from provider to provider. All such extensions would be handled by the CMPS.

5 **Initialization**

 1. Initialization of the CMPS.

 A CMPS may be used to identify the capabilities of the Commerce Appliance in which a CMPS is installed. A CMPS permanently associated with a particular Commerce Appliance may have such information designed-in when the CMPS is initially installed (e.g., stored in ROM 2406 shown in FIG.24). A CMPS which is
10 removable may be used to run an initialization operation in order to obtain information about the device's capabilities. Such information may be stored in a data structure stored in NVRAM 2425. Alternatively, some or all of such information may be gathered each time the device is turned on, and stored in RAM 2414.

15 For example, a DVD player may or may not contain a connection to an external server and/or process. A CMPO and/or other CI stored on a DVD (and/or any other format optical disk) inserted into a DVD (or any other format optical disk) player may include rules predicated on the possibility of outputting information to a server (e.g., content is free if user identification information is output), or may require a direct connection in order, for
20 example, to download keys used to decrypt content. In such a case, the CMPS arrangement may determine the hardware functionality which is expected by or required by the CMPO, and compare that to the hardware actually present. If the CMPS determines that the CMPO and/or other CI requires a network connection, and that the DVD player does not include such a connection, the CMPS may take a variety of steps, including: (1) if the network
25 connection is required for some options but not others, causing only those options which are possible to be displayed to the user; (2) informing the user that necessary hardware is missing; or (3) causing a graceful rejection of the disk, including informing the user of the reason for the rejection.

 To take another example, a CMPO and/or other CI may include a business model
30 which allows the user to choose among quality levels (or other forms of variations of a given work, for example, longer length and/or greater options), with a higher price being charged if the user selects a higher level of quality (e.g., music may be played at low resolution for free, but requires a payment in order to be played at a higher resolution). In such a case, the Commerce Appliance may not include loudspeakers which are capable of
35 outputting sound at the higher resolution. The CMPS arrangement preferably identifies this situation, and either eliminates the higher resolution output as an option for the user, or

informs the user that this option costs more but provides no additional benefit given the Commerce Appliance's current functionality or given the Commerce Appliance not being docked in a user arrangement that provides higher quality loudspeakers.

5 If the Commerce Appliance may be hooked up to external devices (e.g., loudspeakers, display, etc.), the CMPS will require some mechanism for identifying and registering such devices. Each device may be used to make standard ID and capability information available at all times, thereby allowing the CMPS to poll all connected devices at regular intervals, including, for example, authenticating CMPS arrangements within one or more of each such connected devices. Using another approach, all devices could be used 10 to output CMPS identification information upon power-on, with later connected devices being used to output such information upon establishment of the connection. Such identification information may take the form, for example, of authentication information provided under the "five company arrangement", such authentication methods are herein incorporated by reference.

15 As discussed earlier, a Commerce Appliance may be connected to multiple devices each containing its own CMPS arrangement (e.g., a DVD player may be connected to a digital TV) In such cases, the CMPSs must be able to initiate secure communication (e. g., using a scheme, for example, like the "five company proposal" for IEEE 1394 serial bus) and determine how the CMPSs will interact with respect to content communication 20 between CMPSs and, in certain embodiments, regarding cooperative governance of such content such as describing in the incorporated Shear patent application. In one embodiment, the first CMPS arrangement to receive content might govern the control process by downloading an initial CMPO and/or other CI, and display one or more of the rules to the user, etc. The second CMPS arrangement might recognize that it has no further 25 role to play, either as a result of a communication between the two CMPS arrangements, or as a result of changes to the content stream created by the first CMPS arrangement (which decrypted the content, and may have allowed demuxing, composition and rendering, etc.)

The relationship between upstream and downstream CMPSs arrangements may be complicated if one device handles certain aspects of MPEG-4 rendering, and the other 30 handles other aspects. For example, a DVD player might handle demuxing and buffering, transferring raw ESs to a digital TV, which then handles composition and rendering, as well as display. In such a case, there might be no back-channel from the composition and rendering block to the upstream CMPS arrangement. CMPS arrangements are preferably designed to handle stand-alone cases (a DVD (or any other optical disk) player with a 35 CMPS arrangement attached to a dumb TV with no CMPS), multiple CMPS arrangement

cases in which one CMPS arrangement handles all of the processing (a DVD (or other optical disk) player which handles everything through composition and rendering, with a video stream output to the digital TV (in one non-limiting example, via an IEEE 1349 serial bus) (that output stream would be encrypted as per the "five company proposal" for copy protection using IEEE 1394 serial bus transmission)) and/or shared processing between two or more CMPSs arrangements regarding some, or in certain cases, all, of such processing.

2. Initialization of a particular content stream.

The CMPS may be designed so that it can accept initialization information which initializes the CMPS for a particular content stream or channel. This header, which may be a CMPO and/or other CI, may contain information used by the CMPS to locate and/or interpret a particular content stream as well as CI associated with that stream. This initial header may be received through a sideband channel, or may be received as a CI ES such as a CMPO ES.

In one example, shown in FIG. 26, Header CMPO 2601 may include the following information:

(a) Stream/Object/CMPO ID 2602, which identifies the content streams/objects governed by Header CMPO 2601 and/or identification of CMPOs associated with each such content stream or object.

In one embodiment, Header CMPO 2601 identifies other CMPOs which contain rules and keys associated with particular content streams. In another embodiment, Header CMPO 2601 directly controls all content streams, by incorporating the keys and rules associated with such streams. In the latter case, no other CMPOs may be used.

In one embodiment, Header CMPO 2601 may be one or more CMPOs, CCMPOs, MCMPOs, and/or other CI.

(b) One or CMPO Keys 2603 for decrypting each identified CMPO.

(c) Work-Level Control 2604, consisting of basic control information associated with the work as a whole, and therefore potentially applicable to all of the content streams which make up the work. This basic control information may include rules governing the work as a whole, including options to be presented to the user.

(d) In one embodiment of this embodiment, a header CMPO may be updatable to contain User/Site Information 2605 regarding a particular user or site currently authorized to use certain content, as well as one or more rule sets under which the user has gained such authorization. A header CMPO associated with a work currently being viewed may be stored in RAM or NVRAM. This may include updated information. In one

embodiment, the CMPO may also store header CMPOs for certain works viewed in the past. In one embodiment, header CMPOs may be stored in non-secure memory, with information sufficient to identify and authenticate that each header CMPO had not been changed.

5 In one such header CMPO embodiment of this embodiment, the header CMPO operates as follows:

10 (a) The header CMPO is received by a CMPS arrangement. In the case of previously unreceived content which has now become available, the header CMPO may be received at an input port. In the case of content which is already available, but is not currently being used (e.g., a set-top box with 500 channels, of which either 0 or 1 are being displayed at any given time), CCMPOs for each channel may be buffered by the CMPS arrangement for possible use if the user invokes particular content (e.g., switches to a particular channel).

15 In either case, the header CMPO must include information which allows a CMPS arrangement to identify it as a header CMPO.

20 (b) The CMPS arrangement obtains business-model information held in the clear in the header CMPO. Business-model information may include, for example, a statement that content can be viewed for free if advertisements are included, or if the user authorizes Nielson-type information, user and/or audience measurement information, for example, content may be output to a server or otherwise copied once, but only at a price.

25 (c) The CMPS arrangement either accepts the business model, if the user has authorized it to accept certain types of models (e.g., the user has programmed the CMPS arrangement to always accept play with advertisements for free), rejects the business model, if the user has instructed that the particular model always be rejected, or displays the business model to the user (e.g., by presenting options on the screen).

30 (d) If a business model has been accepted, the CMPS arrangement then decrypts the remainder of the header CMPO. If the Commerce Appliance contains a live output connection to an external server (e.g., Internet connection, back-channel on a set-top box, etc.), and if latency problems are handled, decryption of these keys can be handled by communicating with the external server, each side authenticating the other, establishment of a secure channel, and receipt of a key from the server. If the Commerce Appliance is not at least occasionally connected to an external server, decryption may have to be based on one or more keys securely stored in the Commerce Appliance.

35 (e) Once a header CMPO has been decrypted, the CMPS arrangement acquires information used to identify and locate the streams containing the content, and

- 57 -

keys which are used to decrypt either the CMPOs associated with the content, or to directly decrypt the content itself.

(f) In one embodiment of this header embodiment, the header CMPO may contain a data structure for the storage of information added by the CMPS arrangement.

5 Such information may include the following:

(1) Identification of user and/or Commerce Appliance and/or CMPS arrangement. In this embodiment, such information may be stored in a header CMPO in order to provide an audit trail in the event the work (including the header CMPO) is transferred (this only works if the header CMPO is transferred in a writable form). Such
10 information may be used to allow a user to transfer the work to other Commerce Appliances owned by the user without the payment of additional cost, if such transfers are allowed by rule information associated with the header CMPO. For example, a user may have a subscription to a particular cable service, paid for in advance by the user. When a CMPS arrangement downloads a header CMPO from that cable service, the CMPS
15 arrangement may store the user's identification in the header CMPO. The CMPS arrangement may then require that the updated header CMPO be included if the content is copied or transferred. The header CMPO could include a rule stating that, once the user information has been filled in, the associated content can only be viewed by that user, and/or by Commerce Appliances associated with that user. This would allow the user to
20 make multiple copies of the work, and to display the work on multiple Commerce Appliances, but those copies could not be displayed or used by non-authorized users and/or on non-authorized Commerce Appliances. The header CMPO might also include a rule stating that the user information can only be changed by an authorized user (e.g., if user 1 transfers the work to user 2, user 2's CMPS arrangement can update the user information in
25 the header CMPO, thereby allowing user 2 to view the work, but only if user 2 is also a subscriber to the cable channel).

(2) Identification of particular rules options governing use. Rule sets included in header CMPOs may include options. In certain cases, exercise of a particular option might preclude later exercise of a different option. For example, a user
30 might be given the choice to view an unchanged work for one price, or to change a work and view the changed work for a higher price. Once the user decides to change the work and view the changed work, this choice is preferably stored in the header CMPO, since the option of viewing the original unchanged work at the lower price is no longer available. The user might have further acquired the right, or may now be presented with the option for
35 the right, to further distribute the changed work at a mark-up in cost resulting in third party

derived revenue and usage information flowing to both the user and the original work stakeholder(s).

(3) Historical usage information. The header CMPO may include information relating to the number and types of usages. For example, if the underlying work is copied, the header CMPO may be updated to reflect the fact that a copy has been made, since a rule associated with the work might allow only a single copy (e.g., for backup and/or timeshifting purposes). To take another example, a user might obtain the right to view a work one time, or for a certain number of times. The header CMPO would then be updated to reflect each such use.

Usage information may be used to determine if additional uses are authorized by rules associated with the header CMPO. Such information may also be used for audit purposes. Such information may also be provided as usage information exhaust, reported to an external server. For example, a rule may specify that a work may be viewed for free, but only if historical usage information is downloaded to a server.

Content Management Protection Objects (CMPO)

The Content Management and Protection Object ("CMPO") is a data structure which includes information used by the CMPS to govern use of certain content. A CMPO may be formatted as a data structure specified by a particular standard (e.g., an MPEG-4 ES), or may be formatted as a data structure not defined by the standard. If the CMPO is formatted as a data structure specified by the standard, it may be received in the channel utilized by the standard (e.g., as part of a composite MPEG-4 stream) or may be received through some other, side-band method. If the CMPO is formatted as a data structure not specified by the relevant standard, it is provided and decoded using some side-band method, which may include receipt through the same port as formatted content and/or may include receipt through a separate port.

Content may be controlled at virtually any level of granularity. Three exemplary levels will be discussed herein: "channel," "work," and "object."

A "channel" represents an aggregation of works. The works may be available for selection by the user (e.g., a web site, or a video library) or may be received serially (e.g., a cable television channel).

A "work" represents a single audio-visual, textual or other work, intended to be consumed (viewed, read, etc.) by a user as an integrated whole. A work may, for example, be a movie, a song, a magazine article, a multimedia product such, for example, as sophisticated videogame. A work may incorporate other works, as, for example, in a multimedia work which incorporates songs, video, text, etc. In such a case, rights may be

associated

An "object" represents a separately addressable portion of a work. An object may be, for example, an individual MPEG-4 AVO, a scene descriptor graph, an object descriptor, the soundtrack for a movie, a weapon in a videogame, or any other logically definable portion.

Content may be controlled at any of these levels (as well as intermediate levels not discussed herein). The preferred embodiment mechanism for such control is a CMPO or CMPO arrangement (which comprises one or more CMPOs, and if plural, then plural, cooperating CMPOs). CMPOs and CMPO arrangements may be organized hierarchically, with a Channel CMPO arrangement imposing rules applicable to all contained works, a MCMPO or an SGCMPO imposing rules applicable to all objects within a work, and a CMPO arrangement imposing rules applicable to a particular object.

In one embodiment, illustrated in FIG. 27, a CMPS may download CCMPO 2701. CCMPO 2701 may include one or more Rules 2702 applicable to all content in the channel, as well as one or more Keys 2703 used for decryption of one or more MCMPOs and/or SGCMPOs. MCMPO 2704 may include Rules 2705 applicable to a single work and/or works, one or more classes and/or more users and/or user classes, and may also include Keys 2706 used to decrypt CMPOs. CMPO 2707 may include Rules 2708 applicable to an individual object, as well as Key 2709 used to decrypt the object.

As long as all objects are subject to control at some level, there is no requirement that each object be individually controlled. For example, CCMPO 2701 could specify a single rule for viewing content contained in its channel (e.g., content can only be viewed by a subscriber, who is then might be free to redistribute the content with no further obligation to the content provider). In such a case, rules would not necessarily be used for MCMPOs (e.g. Rules 2705), SGCMPOs, or CMPOs (e.g., Rules 2708). In one embodiment, MCMPOs, SGCMPOs, and CMPOs could be dispensed with, and CCMPO 2701 could include all keys used to decrypt all content, or could specify a location where such keys could be located. In another embodiment, CCMPO 2701 would supply Key 2703 used to decrypt MCMPO 2704. MCMPO 2704 might include keys used to decrypt CMPOs (e.g., Keys 2706), but might include no additional Rules 2705. CMPO 2707 might include Key 2709 used to decrypt an object, but might include no additional Rules 2708. In certain embodiments, there may be no SGCMPOs.

A CMPO may be contained within a content data structure specified by a relevant standard (e.g., the CMPO may be part of a header in an MPEG-4 ES.) A CMPO may be contained within its own, dedicated data structure specified by a relevant standard (e.g., a

CMPO ES). A CMPO may be contained within a data structure not specified by any content standard (e.g., a CMPO contained within a DigiBox).

A CCMPO may include the following elements:

5 (a) ID 2710. This may take the following form: <channel ID>< CMPO type><CMPO ID><version number>. In the case of hierarchical CMPO organization (e.g., CCMPOs controlling MCMPOs controlling CMPOs), CMPO ID 2711 can include one field for each level of the hierarchy, thereby allowing CMPO ID 2711 to specify the location of any particular CMPO in the organization. ID 2710 for a CCMPO may, for example, be 123-000-000. ID 2712 for a MCMPO of a work within that channel may, for example, be 123-456-000, thereby allowing the specification of 1,000 MCMPOs as controlled by the CCMPO identified as "123." CMPO ID 2711 for a CMPO associated with an object within the particular work may, for example, be 123-456-789, thereby allowing the specification of 1,000 CMPOs as associated with each MCMPO.

10 This method of specifying CMPO IDs thereby conveys the exact location of any CMPO within a hierarchy of CMPOs. For cases in which higher levels of the hierarchy do not exist (e.g., a MCMPO with no associated CCMPO), the digits associated with that level of the hierarchy may be specified as zeroes.

15 (b) Rules 2702 applicable to all content in the channel. These may be self-contained rules, or may be pointers to rules obtainable elsewhere. Rules are optional at this level.

20 (c) Information 2713 designed for display in the event the user is unable to comply with the rules (e.g., an advertisement screen informing the user that a subscription is available at a certain cost, and including a list of content available on the channel).

25 (d) Keys 2703 for the decryption of each MCMPO controlled by this CCMPO. In one embodiment, the CCMPO includes one or more keys which decrypt all MCMPOs. In an alternate embodiment, the CCMPO includes one or more specific keys for each MCMPO.

30 (e) A specification of a CMPS Type (2714), or of hardware/software necessary or desirable to use the content associated with this channel.

The contents of a MCMPO may be similar to those of a CCMPO, except that the MCMPO may include rules applicable to a single work, and may identify CMPOs associated with each object.

The contents of each CMPO may be similar to those of the MCMPO, except that the CMPO may include rules and keys applicable to a single object.

The contents of an SGCMPPO may be similar to those of the CCMPO, except that the MCMPO may include rules applicable to only certain one or more classes of rights, certain one or more classes of works, and/or to one or more certain classes of users and/or user arrangements (e.g. CMPO arrangements and/or their devices).

5 In another embodiment, shown in FIG. 28, CMPO Data Structure 2801 may be defined as follows:

CMPO Data Structure 2801 is made up of elements. Each element includes a self-contained item of information. The CMPS parses CMPO Data Structure, one element at a time.

10 Type Element 2802 identifies the data structure as a CMPO, thereby allowing the CMPS to distinguish it from a content ES. In an exemplary embodiment, this element may include 4 bits, each of which may be set to "1" to indicate that the data structure is a CMPO.

15 The second element is CMPO Identifier 2803, which is used to identify this particular CMPO and to convey whether the CMPO is part of a hierarchical organization of CMPOs and, if so, where this CMPO fits into that organization.

20 CMPO Identifier 2803 is divided into four sub-elements, each of three bits. These are shown as sub-elements A, B, C and D. The first sub-element (2803 A) identifies the CMPO type, and indicates whether the CMPO is governed or controlled by any other CMPO:

100: this is a top-level CMPO (associated with a channel or an aggregation of works) and is not controlled by any other CMPO.

010: this is a mid-level CMPO (associated with a particular work) and is not controlled by any other CMPO.

25 110: this is a mid-level CMPO, and is controlled by a top-level CMPO.

001: this is a low-level CMPO (associated with an object within a work) and is not controlled by any other CMPO. This case will be rare, since a low-level CMPO will ordinarily be controlled by at least one higher-level CMPO.

30 011: this is a low-level CMPO, and is controlled by a mid-level CMPO, but not by a top-level CMPO.

111: this is a low-level CMPO, and is controlled by a top-level CMPO and by a mid-level CMPO.

35 The second sub-element of CMPO ID 2803 (sub-element B) identifies a top-level CMPO. In the case of a top-level CMPO, this identifier is assigned by the creator of the CMPO. In the case of a mid-level or low-level CMPO which is controlled by a top-level

CMPO, this sub-element contains the identification of the top-level CMPO which performs such control. In the case of a mid-level or low-level CMPO which is not controlled by a top-level CMPO, this sub-element contains zeroes.

5 The third sub-element of CMPO ID 2803 (sub-element C) identifies a mid-level CMPO. In the case of a top-level CMPO, this sub-element contains zeroes. In the case of a mid-level CMPO, this sub-element contains the identification of the particular CMPO. In the case of a low-level CMPO which is controlled by a mid-level CMPO, this sub-element contains the identification of the mid-level CMPO which performs such control. In the case of a low-level CMPO which is not controlled by a mid-level CMPO, this sub-element
10 contains zeroes.

The fourth sub-element of CMPO ID 2803 (sub-element D) identifies a low-level CMPO. In the case of a top-level or mid-level CMPO, this sub-element contains zeroes. In the case of a low-level CMPO, this sub-element contains the identification of the particular CMPO.

15 Following the identifier element is Size Element 2804 indicating the size of the CMPO data structure. This element contains the number of elements (or bytes) to the final element in the data structure. This element may be rewritten if alterations are made to the CMPO. The CMPS may use this size information to determine whether the element has been altered without permission, since such an alteration might result in a different size.
20 For such purposes, the CMPS may store the information contained in this element in a protected database. This information can also be used to establish that the entire CMPO has been received and is available, prior to any attempt to proceed with processing.

Following Size Element 2804 are one or more Ownership/Control Elements containing ownership and chain of control information (e.g., Ownership/Control Elements
25 2805, 2806 and 2807). In the first such element (2805), the creator of the CMPO may include a specific identifier associated with that creator. Additional participants may also be identified in following elements (e.g., 2806, 2807). For example, Element 2805 could identify the creator of the CMPO, Element 2806 could identify the publisher of the associated work and Element 2807 could identify the author of the work.

30 A specific End Element 2808 sequence (e.g., 0000) indicates the end of the chain of ownership elements. If this sequence is encountered in the first element, this indicates that no chain of ownership information is present.

Chain of ownership information can be added, if rules associated with CMPO 2801 permit such additions. If, for example, a user purchases the work associated with CMPO
35 2801, the user's identification may be added as a new element in the chain of ownership

elements (e.g., a new element following 2807, but before 2808). This may be done at the point of purchase, or may be accomplished by the CMPS once CMPO 2801 is encountered and the CMPS determines that the user has purchased the associated work. In such a case, the CMPS may obtain the user identifier from a data structure stored by the CMPS in NVRAM.

Following the ownership element chain are one or more Handling Elements (e.g., 2809, 2810) indicating chain of handling. These elements may contain the identification of any CMPS which has downloaded and decoded CMPO 2801, and/or may contain the identification of any user associated with any such CMPS. Such information may be used for audit purposes, to allow a trail of handling in the event a work is determined to have been circulated improperly. Such information may also be reported as exhaust to a clearinghouse or central server. Chain of handling information preferably remains persistent until reported. If the number of elements required for such information exceeds a specified amount (e.g., twenty separate user identifiers), a CMPS may refuse to allow any further processing of CMPO 2801 or the associated work until the CMPS has been connected to an external server and has reported the chain of handling information.

The last element in the chain of handling elements (e.g., 2811) indicates the end of this group of elements. The contents of this element may, for example, be all zeroes.

Following the chain of handling elements may be one or more Certificate Elements (e.g., 2812, 2813) containing or pointing to a digital certificate associated with this CMPO. Such a digital certificate may be used by the CMPS to authenticate the CMPO. The final element in the digital certificate chain is all zeroes (2814). If no digital certificate is present, a single element of all zeroes exists in this location.

Following the Certificate Elements may be a set of Governed Object Elements (e.g., 2815, 2816, 2817, 2818) specifying one or more content objects and/or CMPOs which may be governed by or associated with CMPO 2801. Each such governed object or CMPO is identified by a specific identifier and/or by a location where such object or CMPO may be found (e.g., these may be stored in locations 2815 and 2817). Following each such identifier may be one or more keys used to decrypt such CMPO or object (e.g., stored in locations 2816 and 2818). The set of identifiers/keys ends with a termination element made up of all zeroes (2819).

Following the set of elements specifying identifiers and/or keys may be a set of Rules Elements (e.g., 2820, 2821, 2822) specifying rules/controls and conditions associated with use of the content objects and/or CMPOs identified in the Governed Objects chain (e.g., locations 2815 and 2817). Exemplary rules are described below. Elements may

contain explicit rules or may contain pointers to rules stored elsewhere. Conditions may include particular hardware resources necessary to use associated content objects or to satisfy certain rules, or particular types of CMPS's which are necessary or preferred for use of the associated content objects.

5 Following the rules/controls and conditions elements may be a set of Information Elements 2823 containing information specified by the creator of the CMPO. Among other contents, such information may include content, or pointers to content, programming, or pointers to programming.

The CMPO ends with Final Termination Element 2824.

10 In one embodiment, the rules contained in Rules Elements 2820-2822 of CMPO 2801 may include, for example, the following operations:

(1) Play. This operation allows the user to play the content (though not to copy it) without restriction.

15 (2) Navigate. This allows the user to perform certain types of navigation functions, including fast forward/rewind, stop and search. Search may be indexed or unindexed.

20 (3) Copy. Copy may be allowed once (e.g., time-shifting, archiving), may be allowed for a specified number of times and/or may be allowed for limited period of time, or may be allowed for an unlimited period of time, so long as other rules, including relevant budgets, are not violated or exceeded. A CMPS arrangement may be designed so that a Copy operation may cause an update to an associated CMPO (e.g., including an indication that the associated content has been copied, identifying the date of copying and the site responsible for making the copy), without causing any change to any applicable content object, and in particular without requiring that associated content objects be demuxed, decrypted or decompressed. In the case of MPEG-4, for example, this may require the following multi-stage demux process:

25 (i) the CMPS arrangement receives a Copy instruction from the user, or from a header CMPO.

30 (ii) CMPO ESs associated with the MPEG-4 stream which is to be copied are separated from the content stream in a first demux stage.

 (iii) CMPOs are decrypted and updated by the CMPS arrangement. The CMPOs are then remuxed with the content ESs (which have never been demuxed from each other), and the entire stream is routed to the output port without further alteration.

35 This process allows a copy operation to take place without requiring that the content streams be demuxed and decrypted. It requires that the CMPS arrangement include

two outputs: one output connected to the digital output port (e.g., FIG. 23 line 2316, connecting to Digital Output Port 2317), and one output connected to the MPEG-4 buffers (e.g., FIG. 23, lines 2310, 2311, 2312), with a switch designed to send content to one output or the other (or to both, if content is to be viewed and copied simultaneously) (e.g., Switch 2319). Switch 2319 can be the only path to Digital Output Port 2317, thereby allowing CMPS 2302 to exercise direct control over that port, and to ensure that content is never sent to that port unless authorized by a control. If Digital Output Port 2317 is also the connector to a digital display device, CMPS 2302 will also have to authorize content to be sent to that port even if no copy operation has been authorized.

In one example embodiment, the receiving device receiving the information through Digital Output Port 2317 may have to authenticate with the sending device (e.g., CMPS 2302). Authentication may be for any characteristic of the device and/or one or more CMPSs used in conjunction with that device. Thus, for example, a sending appliance may not transmit content to a storage device lacking a compatible CMPS.

In another non-limiting example, CMPS 2302 can incorporate session encryption functionality (e.g., the "five company arrangement") which establishes a secure channel from a sending interface to one or more external device interfaces (e.g., a digital monitor), and provided that the receiving interface has authenticated with the sending interface, encrypts the content so that it can only be decrypted by one or more authenticated 1394 device interfaces. In that case, CMPS 2302 would check for a suitable IEEE 1394 serial bus interface, and would allow content to flow to Digital Output Port 2317 only if (a) an authorized Play operation has been invoked, a secure channel has been established with the device and the content has been session-encrypted, or (b) an authorized Copy or Retransmit operation has been invoked, and the content has been treated as per the above description (i.e., the CMPO has been demuxed, changed and remuxed, the content has never been decrypted or demuxed).

This is only possible if CMPOs are separately identifiable at an early demux stage, which most likely requires that they be stored in separate CMPO ESs. If the CMPOs are stored as headers in content ESs, it may be impossible to identify the CMPOs prior to a full demux and decrypt operation on the entirety of the stream.

(4) Change. The user may be authorized to change the content.

(5) Delete. This command allows the user to delete content which is stored in the memory of the Consumer Appliance. This operation operates on the entire work. If the user wishes to delete a portion of a work, the Change operation must be used.

(6) Transfer. A user may be authorized to transfer a work to a third party.

This differs from the Copy operation in that the user does not retain the content or any rights to the content. The Transfer operation may be carried out by combining a Copy operation and a Delete operation. Transfer may require alteration of the header CMPO associated with the work (e.g., adding or altering an Ownership/Control Element, such as Elements 2805-2807 of FIG. 28), so as to associate rights to the work with the third party.

These basic operations may be subject to modifications, which may include:

i. Payment. Operations may be conditioned on some type of user payment. Payment can take the form of cash payment to a provider (e.g., credit card, subtraction from a budget), or sending specified information to an external site (e.g., Nielson-type information).

ii. Quality of Service. Operations may specify particular quality of service parameters (e.g., by specifying a requested QoS in MPEG-4), including: requested level of decompression, requested/required types of display, rendering devices (e.g., higher quality loudspeakers, a particular type of game controller).

iii. Time. Operations may be conditioned such that the operation is only allowed after a particular time, or such that the price for the operation is tied to the time (e.g., real-time information at a price, delayed information at a lower price or free, e.g., allowing controlled copies but only after a particular date).

iv. Display of particular types of content. Operations may be conditioned on the user authorizing display of certain content (e.g., the play operation may be free if the user agrees to allow advertisements to be displayed).

In all of these cases, a rule may be modified by one or more other rules. A rule may specify that it can be modified by other rules or may specify that it is unmodifiable. If a rule is modifiable, it may be modified by rules sent from other sources. Those rules may be received separately by the user or may be aggregated and received together by the user.

Data types which may be used in an exemplary MPEG-4 embodiment may include the following:

a. CMP Data Stream.

The CMP-ds is a new elementary stream type that has all of the properties of an elementary stream including its own CMPO and a reference in the object descriptors. Each CMP-ds stream has a series of one or more *CMP Messages*. A *CMP_Message* has four parts:

1. **Count:** [1...n] CMPS types supported by this IP ES. Multiple CMPS systems may be supported, each identified by a unique *type*. (There may have

to be a central registry of types.)

2. **CMPS_type_identifiers:** [1...*n*] identifiers, each with an offset in the stream and a length. The offset points to the byte in the CMPO where the data for that CMPS type is found. The length is the length in bytes of this data.

3. **Data segments:** One segment for each of the *n* CMPS types encoded in a format that is proprietary to the CMPS supplier.

4. **CMP_Message_URL:** That references another CMP_Message. (This is in keeping with the standard of using URLs to point to streams.)

b. **CMPO.**

The CMPO is a data structure used to attach detailed CMP control to individual elementary streams. Each **CMPO** contains:

1. **CMPO_ID:** An identifier for the content under control. This identifier must *uniquely* identify an elementary stream.

2. **CMPO_count:** [1...*n*] CMPS types supported by this **CMPO**.

3. **CMPS_type_identifiers:** [1...*n*] identifiers, each with an offset in the stream and a length. The offset points to the byte in the CMPO where the data for that CMPS type is found. The length is the length in bytes of this data.

4. **Data segments:** *n* data segments. Each data segment is in a format that is proprietary to the CMPS supplier.

5. **CMPO_URL:** An optional URL that references an additional CMPO that adds information to the information in this CMPO. (This is a way of dynamically adding support for new CMPSs.)

c. **Feedback Event**

The feedback events come in two forms: start and end. Each feedback event contains three pieces of information:

1. **Elementary_stream_ID**
2. **Time:** in presentation time
3. **Object_instance_number**

User Interface.

Commerce Appliance 2301 may include User Interface 2304 designed to convey control-related information to the user and to receive commands and information from the user. This interface may include special purpose displays (e.g., a light which comes on if a current action requires payment), special purpose buttons (e.g., a button which accepts the payment or other terms required for display of content), and/or visual information presented on screen.

Example of Operation in an MPEG-4 Context

1. User selects a particular work or channel. The user may, for example, use a remote control device to tune a digital TV to a particular channel.

2. Selection of the channel is communicated to a CMPS arrangement, which uses the information to either download a CCMPO or to identify a previously downloaded CCMPO (e.g., if the CMPS arrangement is contained in a set-top box, the set-top box may automatically download CCMPOs for every channel potentially reachable by the box).

3. The CMPS arrangement uses the CCMPO to identify rules associated with all content found on the channel. For example, the CCMPO may specify that content may only be viewed by subscribers, and may specify that, if the user is not a subscriber, an advertisement screen should be put up inviting the user to subscribe.

4. Once rules specified by the CCMPO have been satisfied, the CCMPO specifies the location of a MCMPO associated with a particular work which is available on the channel. The channel CMPO may also supply one or more keys used for decryption of the MCMPO.

5. The CMPS arrangement downloads the MCMPO. In the case of an MPEG-4 embodiment, the MCMPO may be an Elementary Stream. This Elementary Stream must be identifiable at a relatively early stage in the MPEG-4 decoding process.

6. The CMPS arrangement decrypts the MCMPO, and determines the rules used to access and use the content. The CMPS arrangement presents the user with a set of options, including the ability to view for free with advertisements, or to view for a price without advertisements.

7. The user selects view for free with advertisements, e.g., by highlighting and selecting an option on the screen using a remote control device.

8. The CMPS arrangement acquires one or more keys from the MCMPO and uses those keys to decrypt the ESs associated with the video. The CMPS arrangement identifies two possible scene descriptor graphs, one with and one without advertisements. The CMPS arrangement passes the scene descriptor graph with advertisements through, and blocks the other scene descriptor graph.

9. The CMPS arrangement monitors the composite and render block, and checks to determine that the advertisement AVOs have actually been released for viewing. If the CMPS arrangement determines that those AVOs have not been released for viewing, it puts up an error or warning message, and terminates further decryption.

CMPS Rights Management In Provider And Distribution Chains

In addition to consumer arrangements, in other embodiments one or more CMPSs

may be used in creating, capturing, modifying, augmenting, animating, editing, excerpting, extracting, embedding, enhancing, correcting, fingerprinting, watermarking, and/or rendering digital information to associate rules with digital information and to enforce those rules throughout creation, production, distribution, display and/or performance processes.

5 In one non-limiting example, a CMPS, a non-exhaustive example of which may include a least a secure portion of a VDE node as described in the aforementioned Ginter et al., patent specification, is incorporated in video and digital cameras, audio microphones, recording, playback, editing, and/or noise reduction devices and/or any other digital device. Images, video, and/or audio, or any other relevant digital information may be captured,
10 recorded, and persistently protected using at least one CMPS and/or at least one CMPO. CMPSs may interact with compression/decompression, encryption/decryption, DSP, digital to analog, analog to digital, and communications hardware and/or software components of these devices as well.

15 In another non-exhaustive example, computer animation, special effects, digital editing, color correcting, noise reduction, and any other applications that create and/or use digital information may protect and/or manage rights associated with digital information using at least one CMPS and/or at least one CMPO.

20 Another example includes the use of CMPSs and/or CMPOs to manage digital assets in at least one digital library, asset store, film and/or audio libraries, digital vaults, and/or any other digital content storage and management means.

25 In accordance with the present applications, CMPSs and/or CMPOs may be used to manage rights in conjunction with the public display and/or performance of digital works. In one non-exhaustive example, flat panel screens, displays, monitors, TV projectors, LCD projectors, and/or any other means of displaying digital information, may incorporate at least one hardware and/or software CMPS instance that controls the use of digital works. A CMPS may allow use only in conjunction with one or more digital credentials, one example of which is a digital certificate, that warrant that use of the digital information will occur in a setting, location, and/or other context for public display and/or performance. Non-limiting
30 examples of said contexts include theaters, bars, clubs, electronic billboards, electronic displays in public areas, or TVs in airplanes, ships, trains and/or other public conveyances. These credentials may be issued by trusted third parties such as certifying authorities, non-exhaustive examples of which are disclosed in the aforementioned Ginter '712 patent application.

Additional MPEG-4 Embodiment Information

This work is based on the MPEG-4 description in the version 1 Systems Committee Draft (CD), currently the most complete description of the evolving MPEG-4 standard.

5 This section presents the structural modifications to the MPEG-4 player architecture and discusses the data lines and the concomitant functional changes. Figure 23 shows the functional components of the original MPEG-4 player. Content arrives at Player 2301 packaged into a serial stream (e.g., MPEG-4 Bit Stream 2314). It is demultiplexed via a sequence of three demultiplexing stages (e.g., Demux 2305) into elementary streams. There are three principle types of elementary streams: AV Objects (AVO), Scene
10 Descriptor Graph (SDG), and Object Descriptor (OD). These streams are fed into respective processing elements (e.g., AVO Decode 2307, Scene Descriptor Graph 2306, Object Descriptors 2308). The AVOs are the multimedia content streams such as audio, video, synthetic graphics and so on. They are processed by the player's compression/coding subsystems. The scene descriptor graph stream is used to build the
15 scene descriptor graph. This tells Composite and Render 2309 how to construct the scene and can be thought of as the "script." The object descriptors contain description information about the AVOs and the SD-graph updates.

To accommodate a CMPS (e.g., CMPS 2302) and to protect content effectively, the player structure must be modified in several ways:

- 20
- Certain data paths must be rerouted to and from the CMPS
 - Certain buffers in the SDG, AVO decode and Object descriptor modules must be secured
 - Feedback paths from the user and the composite and render units to the CMPS must be added

25 In order for CMPS 2302 to communicate with the MPEG-4 unit, and for it to effectively manage content we must specify the CMPO structure and association protocols and we must define the communication protocols over the feedback systems (from the compositor and the user.)

30 The structural modifications to the player are shown in Figure 23. The principal changes are:

- All elementary streams are now routed through CMPS 2302.
- Direct communication path between Demux 2305 and CMPS 2302.
- A required "Content Release and Decrypt" Module 2315 in CMPS 2302.

- 71 -

- The addition of a feedback loop (e.g., Line 2313) from Composite and Render 2309 to CMPS 2302.
- Bi-directional user interaction directly with the CMPS 2302, through Line 2316.

Furthermore, for M4v2P, CMP-objects are preferably associated with all elementary
5 streams. Elementary streams that the author chooses not to protect are still marked by an
“unprotected content” CMPO. The CMPOs are the primary means of attaching rules
information to the content. Content here not only refers to AVOs, but also to the scene
descriptor graph. Scene Descriptor Graph may have great value and will thus need to be
protected and managed by CMPS 2302.

10 The direct path from Demux 2305 to CMPS 2302 is used to pass a CMPS specific
header, that potentially contains business model information, that communicates business
model information at the beginning of user session. This header can be used to initiate user
identification and authentication, communicate rules and consequences, and initiate up-
front interaction with the rules (selection of quality-of-service (QoS), billing, etc.) The
15 user’s communication with CMPS 2302 is conducted through a *non-standardized* channel
(e.g., Line 2316). The CMPS designer may provide an independent API for framing these
interactions.

Feedback Path 2313 from Composite and Render block 2309 serves an important
purpose. The path is used to cross check that the system actually presented the user with a
20 given scene. Elementary streams that are processed by their respective modules may not
necessarily be presented to the user. Furthermore, there are several fraud scenarios wherein
an attacker could pay once and view multiple times. The feedback path here allows CMPS
2302 to cross check the rendering and thereby perform a more accurate accounting. This
feedback is implemented by forcing the Composite and Render block 2309 to issue a *start*
25 *event* that signals the initiation of a given object’s rendering that is complemented by a *stop*
event upon termination. The feedback signaling process may be made optional by
providing a CMP-notification flag that may be toggled to indicate whether or not CMPS
2302 should be notified. All CMPOs would be required to carry this flag.

The final modification to the structure is to require that the clear text buffers in the
30 AVO, SDG and Object Descriptor processors and in the Composite-and-Render block be
secured. This is to prevent a pirate from stealing content in these buffers. As a practical
matter, this may be difficult, since tampering with these structures may well destroy
synchronization of the streams. However, a higher state of security would come from
placing these buffers into a protected processing environment.

35 CMPS 2302 *governs* the functioning of Player 2301, consistent with the following:

- Communication mechanism between CMPS 2302 and the MPEG-4 player (via CMPOs)
- A content release and decryption subsystem
- Version authentication subsystem
- Sufficient performance so as not to interfere with the stream processing in the MPEG-4 components

5
10
CMPS 2302 may have a bi-directional side-channel that is external to the MPEG-4 player that may also be used for the exchange of CMP information. Furthermore, the CMPS designer may choose to provide a user interface API that provides the user with the ability to communicate with the content and rights management side of the stream management (e.g., through Line 2316).

15
Encrypted content is decrypted and released by CMPS 2302 as a function of the rules associated with the protected content and the results of user interaction with CMPS 2302. Unencrypted content is passed through CMPS 2302 and is governed by associated rules and user interaction with CMPS 2302. As a consequence of these rules and user interaction, CMPS 2302 may need to transact with the SDG and AVO coding modules (e.g., 2310, 2311) to change scene structure and/or the QoS grade.

20
Ultimately, the CMPS designer may choose to have CMPS 2302 generate audit trail information that may be sent to a clearinghouse authority via CMPS Side Channel Port 2318 or as encrypted content that is packaged in the MPEG-4 bit stream.

The MPEG-4 v1 Systems CD uses the term "object" loosely. In this document, "object" is used to specifically mean a data structure that flows from one or more of the data paths in Figure 23.

25
30
Using multiple SD-graph update streams, each with its own CMPO, allows an author to apply arbitrarily specific controls to the SD-graph. For example, each node in the SD-graph can be created or modified by a separate SD-graph update stream. Each of these streams will have a distinct CMPO and ID. Thus, the CMPS can release and decrypt the creation and modification of each node and receive feedback information for each node individually. The practical implications for controlling release and implementing consequences should be comparable to having a CMPO on each node of the SD-graph, without the costs of having a CMPO on each SD-graph node.

Principles consistent with the present invention may be illustrated using the following examples:

35
In the first example, there is a bilingual video with either an English or French soundtrack. The user can choose during playback to hear either the English or French. The

basic presentation costs \$1. If the French soundtrack is presented there is a \$0.50 surcharge. If the user switches back and forth between French and English, during a single viewing of the presentation, the \$0.50 surcharge will occur only once.

In this example, there will be four elementary streams:

5 The Scene Description Graph Update stream will have a CMPO. The CMPO will imply a \$1.00 fee associated with the use of the content. The scene description graph displays the video, English audio and puts up a button that allows the user to switch to French. If the user clicks that button, the English stops, the French picks up from that point and the button changes to a switch-to-English button. (Optionally, there may be a little
10 dialog at the beginning to allow the user to select the initial language. This is all easy to do in the SD graph.)

The Video Stream with the CMPO will say that it can only be released if the scene description graph update stream above is released.

The English Audio Stream will be similar to the Video stream.

15 The French Audio Stream will be similar to the Video stream but there is a \$.50 charge if it is seen in the feedback channel. (The CMPS must not count twice if the user switches between the two in a single play of the presentation.)

An important requirement is that the ID for the SD-graph update stream appears in the feedback path (e.g., Feedback Path 2313). This is so CMPS 2302 knows when the
20 presentation stops and ends so that CMPS 2302 can correctly bill for the French audio.

The rules governing the release of the video and audio streams may include some variations. The rules for these streams, for example, may state something like "if you don't see the id for the scene description graph update stream X in the feedback channel, halt
25 release of this stream." If the main presentation is not on the display, then the video should not be. This ties the video to this one presentation. Using the video in some other presentation would require access to the original video, not just this protected version of it.

In a second example, an author wants to have a presentation with a free attract sequence or "trailer". If the user clicks the correct button the system moves into the for-fee
presentation, which is organized as a set of "acts".

30 Multiple SD-graph update streams may update a scene description graph. Multiple SD-graph update streams may be open in parallel. The time stamps on the ALUs in the streams are used to synchronize and coordinate.

The trailer and each act are represented by a separate SD-graph update stream with a separate CMPO. There is likely an additional SD-graph update stream that creates a simple

root node that is invisible and silent. This node brings in the other components of the presentation as needed.

5 The foregoing description of implementations of the invention has been presented for purposes of illustration and description. It is not exhaustive and does not limit the invention to the precise form disclosed. Modifications and variations are possible in light of the above teachings or may be acquired from practicing of the invention. For example, the described implementation includes software but the present invention may be implemented as a combination of hardware and software or in hardware alone. The invention may be implemented with both object-oriented and non-object-oriented programming systems. The scope of the invention is defined by the claims and their
10 equivalents.

- 75 -

We claim:

1. A streaming media player providing content protection and digital rights management, including:

a port configured to receive a digital bit stream, the digital bit stream including:

content which is encrypted at least in part, and

a secure container including control information for controlling use of the content, including at least one key suitable for decryption of at least a portion of the content; and

a control arrangement including:

means for opening secure containers and extracting cryptographic keys, and

means for decrypting the encrypted portion of the content.

2. The player of Claim 1 in which the digital bit stream includes at least two sub-streams which have been muxed together, at least one of the sub-streams including compressed information, and

wherein the player further includes:

a demux designed to separate and route the sub-streams;

a decompression unit configured to decompress at least one of the sub-streams, the decompression unit and the demux being connected by a pathway for the transmission of information; and

a rendering unit designed to process decompressed content information for rendering.

3. The player of Claim 2, further including:

a stream controller operatively connected to the decompression unit, the stream controller including decryption functionality configured to decrypt at least a portion of a sub-stream and pass the decrypted sub-stream to the decompression unit.

4. The player of Claim 3, further including:

a path between the control arrangement and the stream controller to enable the control arrangement to pass at least one key to the stream controller for use with the stream controller's decryption functionality.

5. The player of Claim 4, further including:

a feedback path from the rendering unit to the control arrangement to allow the control arrangement to receive information from the rendering unit regarding the identification of objects which are to be rendered or have been rendered.

6. The player of Claim 1, wherein the digital bit stream is encoded in MPEG-4 format.

7. The player of Claim 1, wherein the digital bit stream is encoded in MP3 format.
8. The player of Claim 4, wherein the control arrangement contains a rule or rule set associated with governance of at least one sub-stream or object.
9. The player of Claim 8, wherein the rule or rule set is delivered from an external source.
10. The player of Claim 9, wherein the rule or rule set is delivered as part of the digital bit stream.
11. The player of Claim 8, wherein the rule or rule set specifies conditions under which the governed sub-stream or object may be decrypted.
12. The player of Claim 8, wherein the rule or rule set governs at least one aspect of access to or use of the governed sub-stream or object.
13. The player of Claim 12, wherein the governed aspect includes making copies of the governed sub-stream or object.
14. The player of Claim 12, wherein the governed aspect includes transmitting the governed sub-stream or object through a digital output port.
15. The player of Claim 14, wherein the rule or rule set specifies that the governed sub-stream or object can be transferred to a second device, but rendering of the governed sub-stream or object must be disabled in the first device prior to or during the transfer.
16. The player of Claim 15, wherein the second device includes rendering capability, lacks at least one feature present in the streaming media player, and is at least somewhat more portable than the streaming media player.
17. The player of Claim 11, wherein the control arrangement contains at least two rules governing access to or use of the same governed sub-stream or object.
18. The player of Claim 17, wherein a first of the two rules was supplied by a first entity, and the second of the two rules was supplied by a second entity.
19. The player of Claim 18, wherein the first rule controls at least one aspect of operation of the second rule.
20. The player of Claim 12, wherein the governed aspect includes use of at least one budget.
21. The player of Claim 12, wherein the governed aspect includes a requirement that audit information be provided.
22. The player of Claim 1, wherein the control arrangement includes tamper resistance.

- 77 -

23. A digital bit stream including:
content information that is compressed and at least in part encrypted; and
a secure container including
governance information for the governance of at least one aspect of
access to or use of at least a portion of the content information; and
a key for decryption of at least a portion of the encrypted content
information.

24. The digital bit stream of Claim 23, wherein the content information is
encoded in MPEG-4 format.

25. The digital bit stream of Claim 23, wherein the content information is
encoded in MP3 format.

26. A method of rendering a protected digital bit stream including:
receiving the protected digital bit stream,
passing the protected digital bit stream to a media player,
the media player reading first header information identifying a plugin used
to process the protected digital bit stream, the first header information
indicating that a first plugin is required;
the media player calling the first plugin;
the media player passing the protected digital bit stream to the first plugin;
the first plugin decrypting at least a portion of the protected digital bit stream;
the first plugin reading second header information identifying a second plugin
necessary in order to render the decrypted digital bit stream;
the first plugin calling the second plugin;
the first plugin passing the decrypted digital bit stream to the second plugin;
the second plugin processing the decrypted digital bit stream, the processing
including decompressing at least a portion of the decrypted digital bit stream;
the second plugin passing the decrypted and processed digital bit stream to the
media player; and
the media player enabling rendering of the decrypted and processed digital bit
stream,
whereby the first plugin may be used in an architecture not designed for
multiple stages of plugin processing.

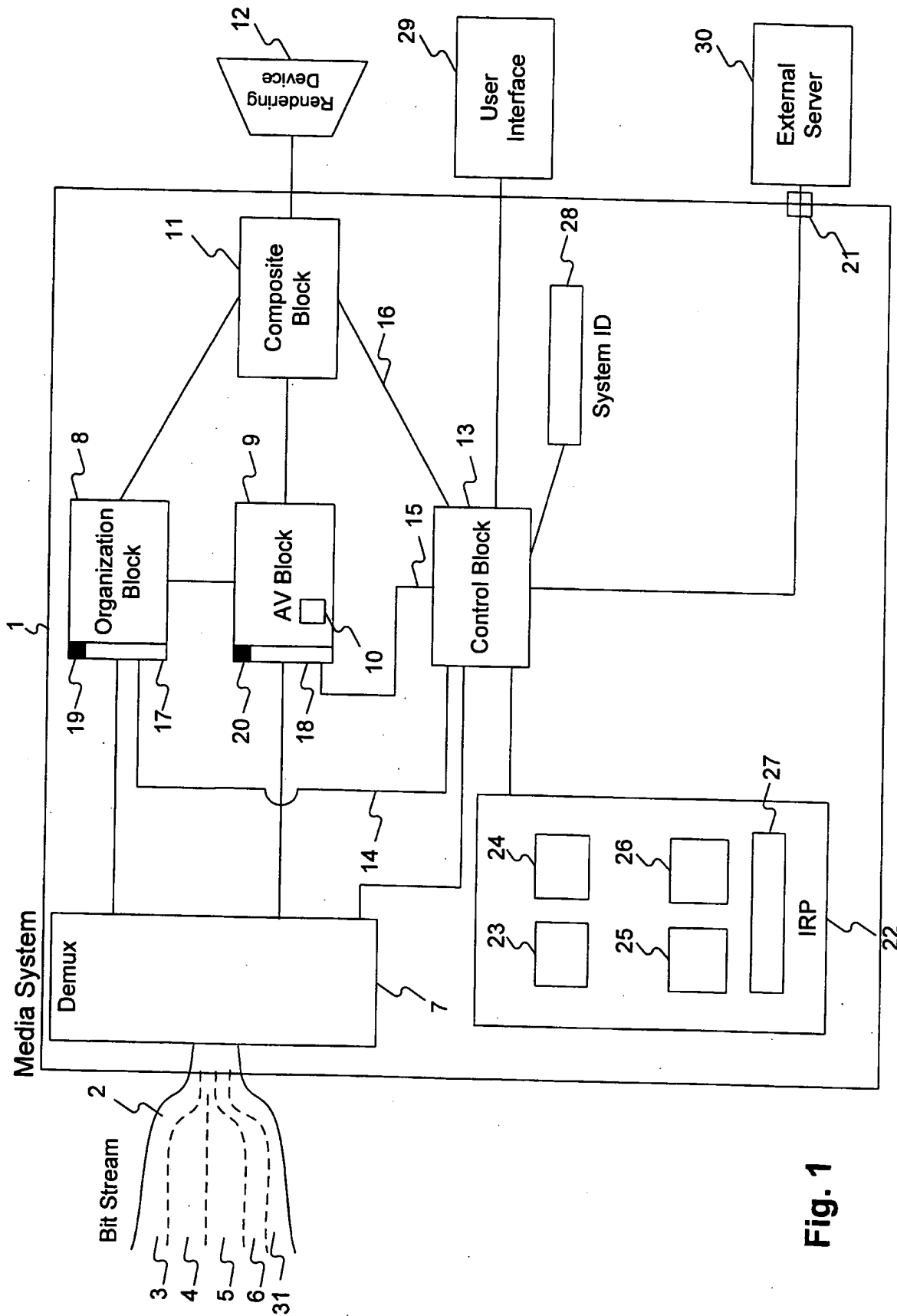


Fig. 1

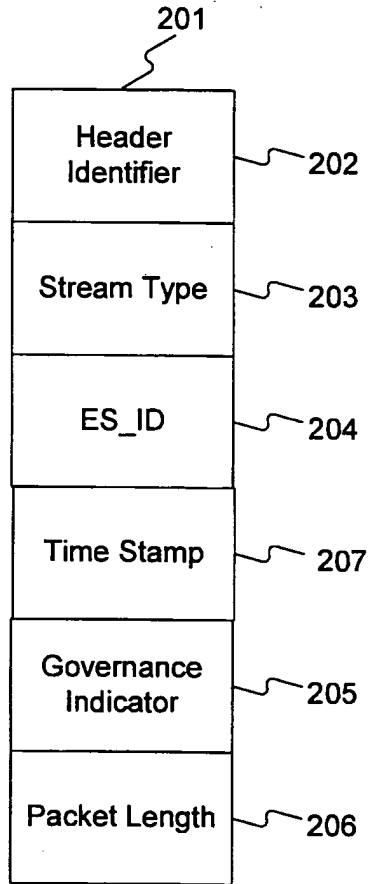


Fig. 2

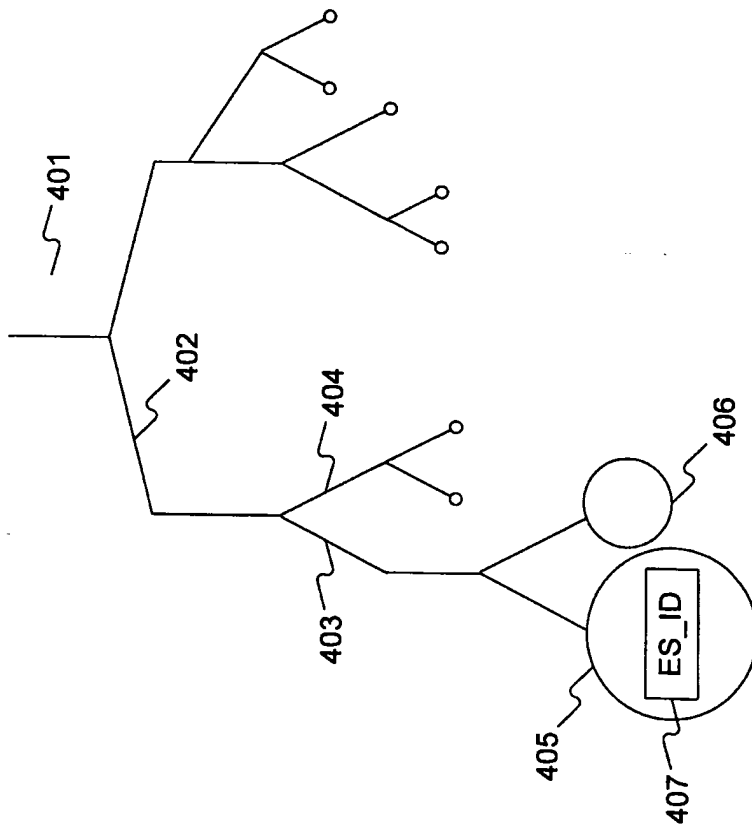


Fig. 4

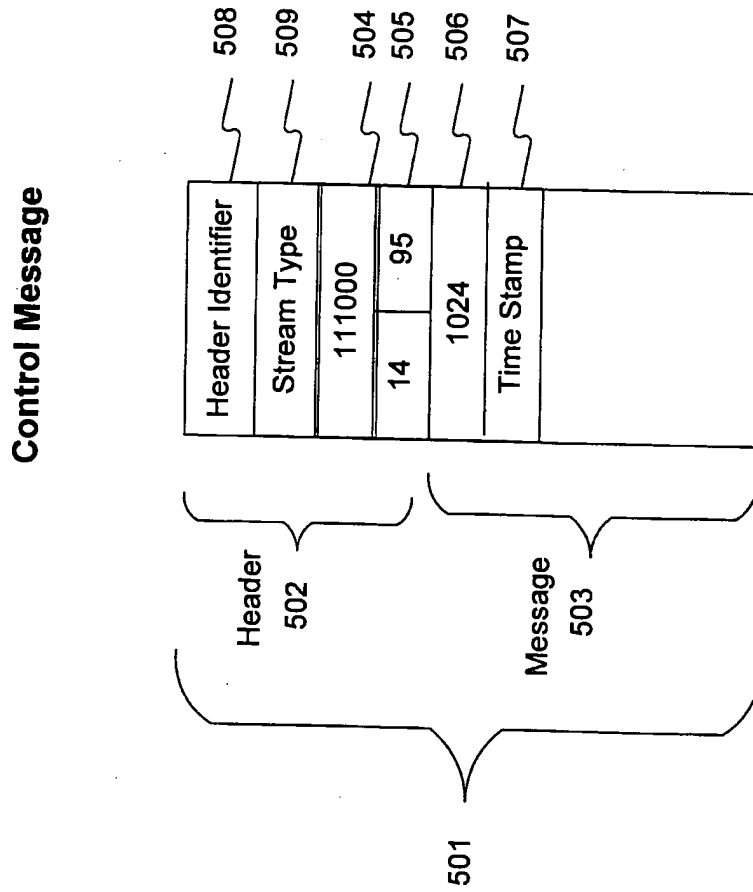


Fig. 5

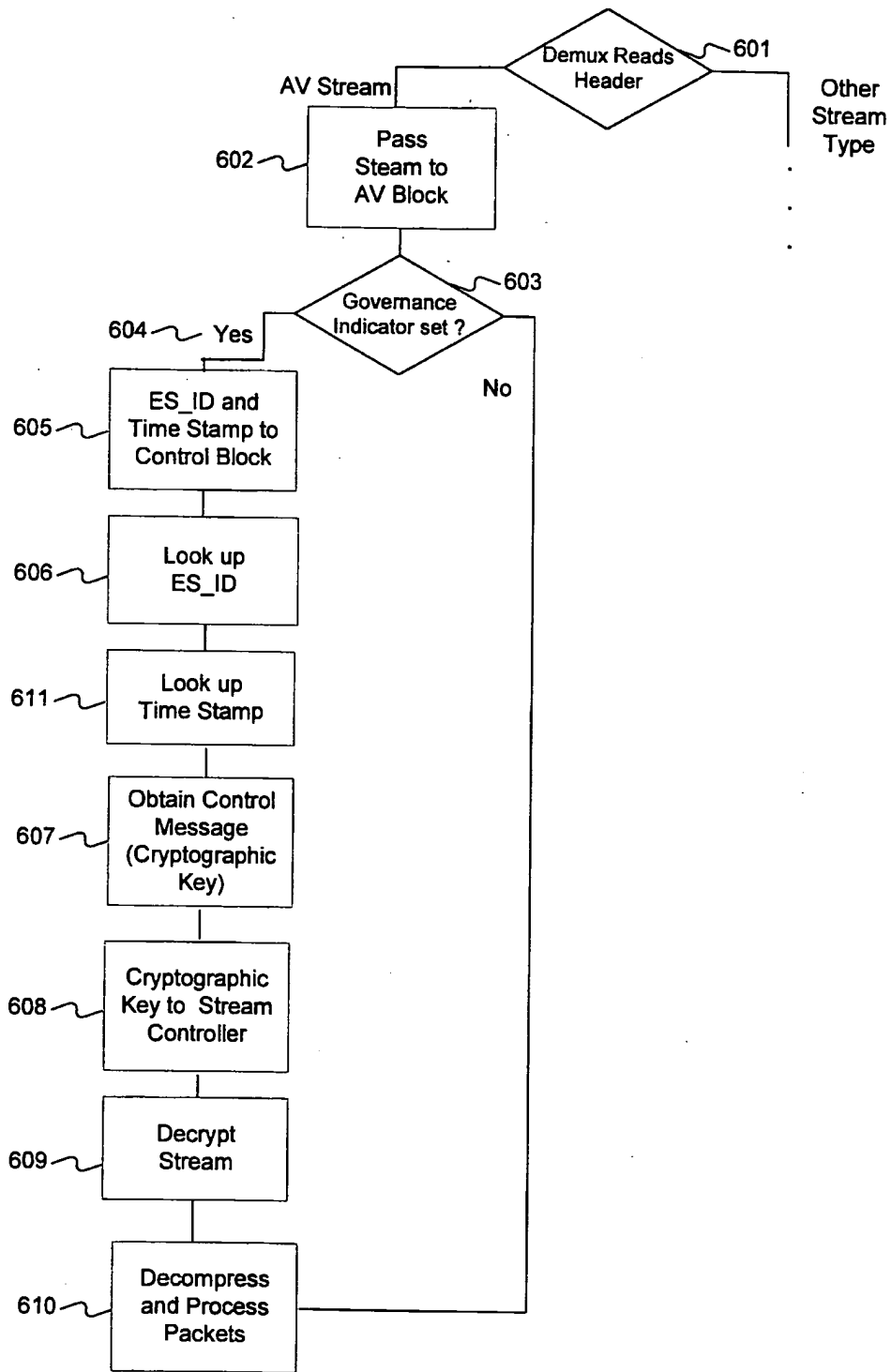


Fig. 6

717	701	702	703	704
	1	ID	Controlled Streams	Message
	2	15	903	Key 705
	3	20	2031	Rule 706
	4	9		Commands 707
	5	700	49, 50, 51, 52, 53	Authorized System ID
6	21	36	Authorized System ID	
	14	1201	Rule 710	Commands 711
			Rule 719	Key 712
			Rule 718	Budget 718
			Rule 719	Key 713
			Rule 719	Key 714
			Rule 719	Key 715
			Rule 719	Key 716
			Rule 719	Key 716

Fig. 7

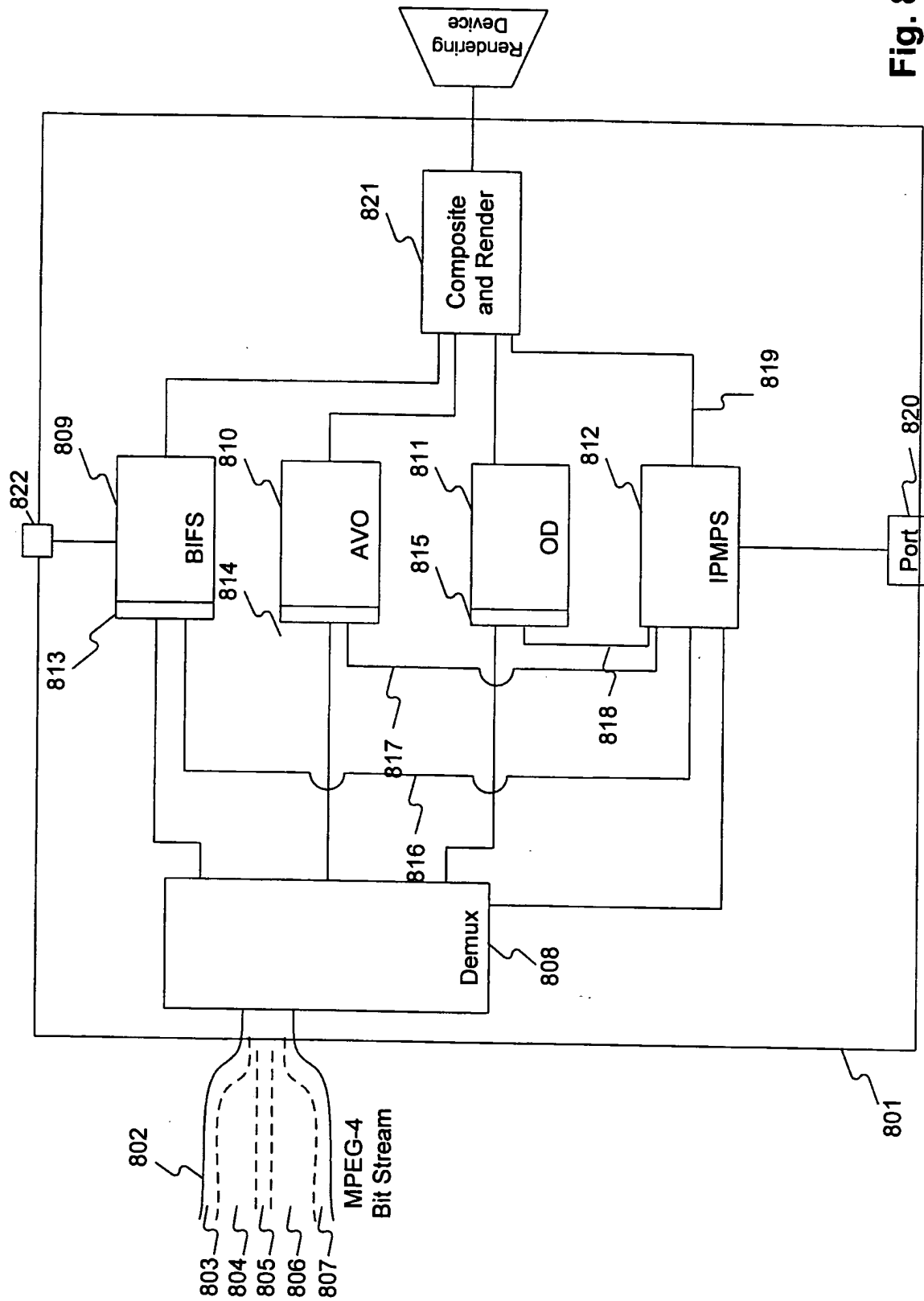


Fig. 8

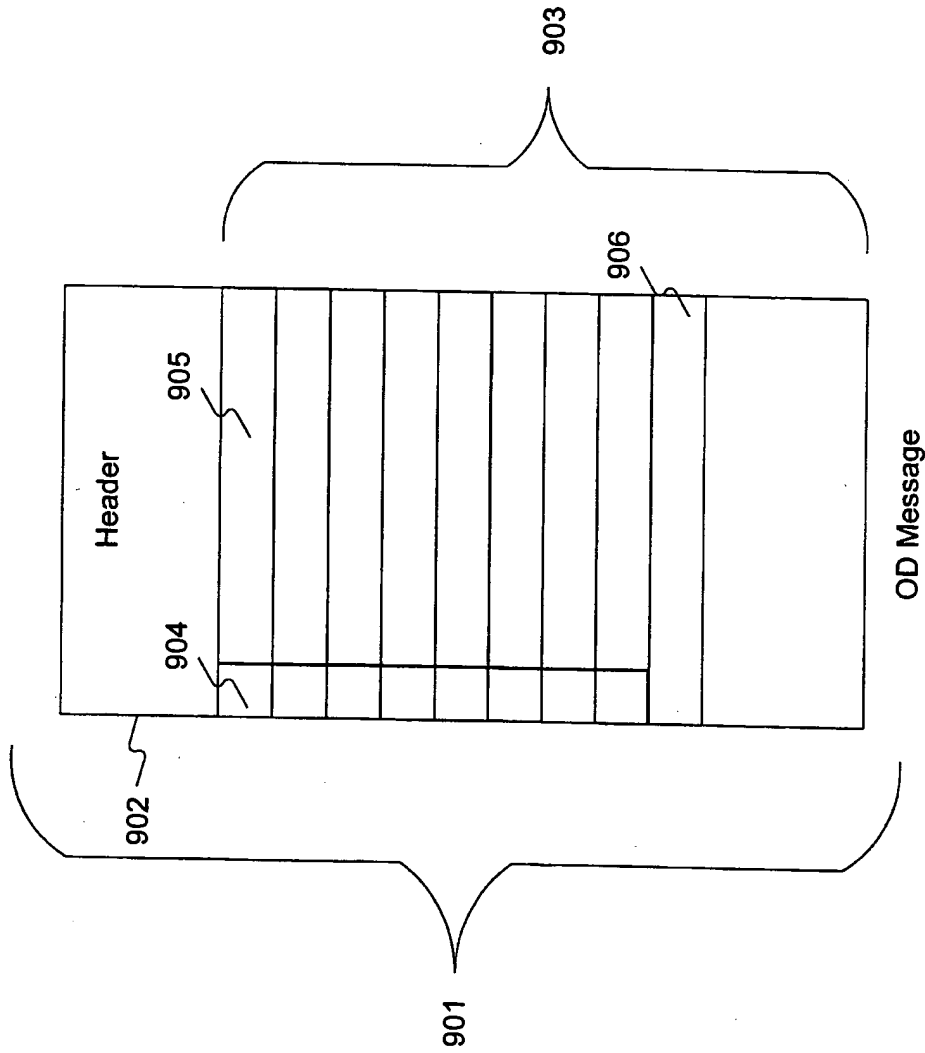


Fig. 9

IPMP Table

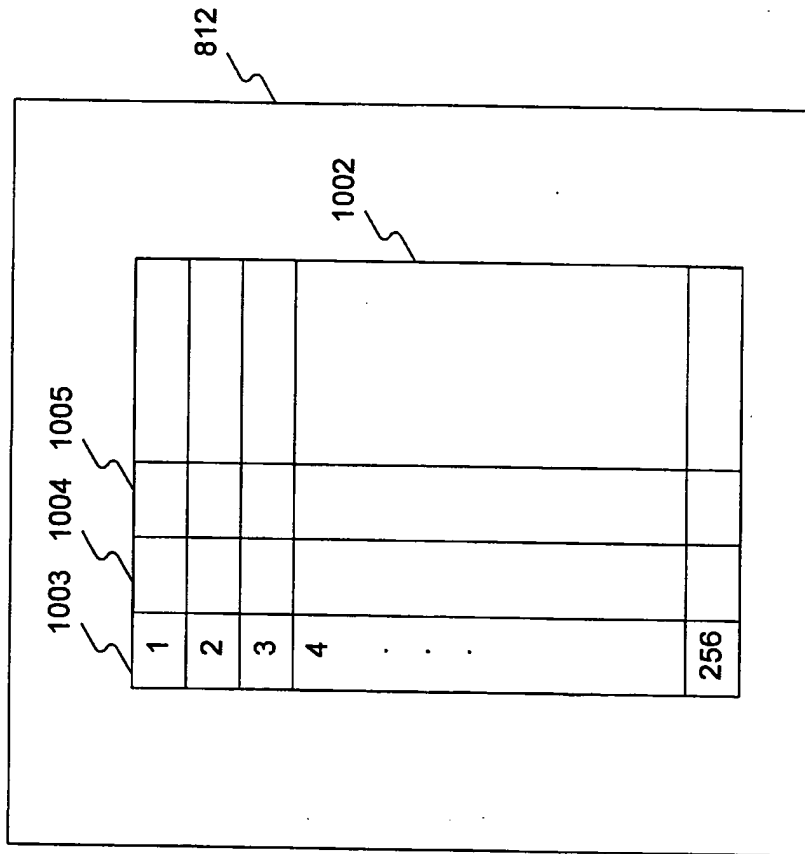


Fig. 10

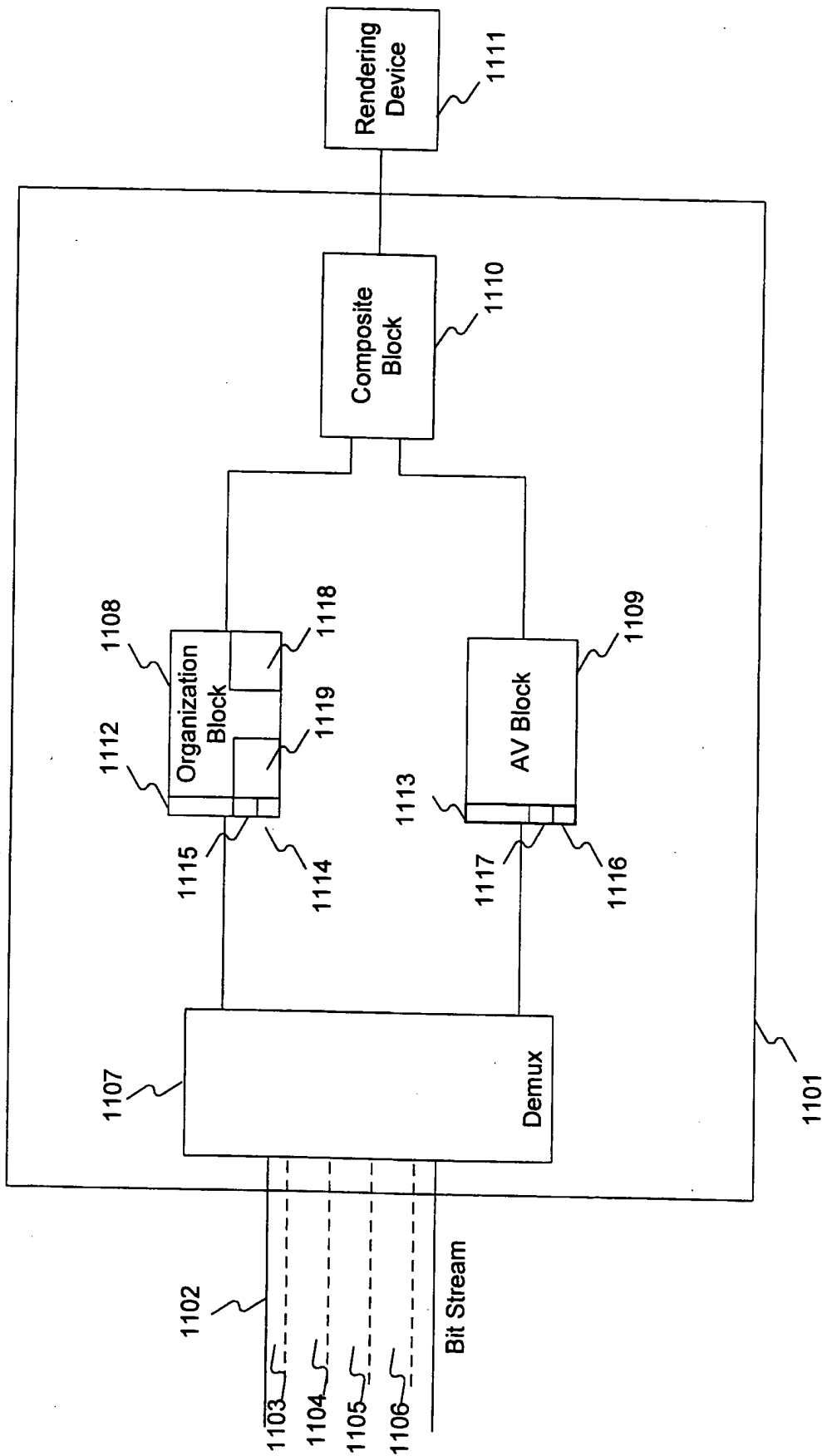


Fig. 11

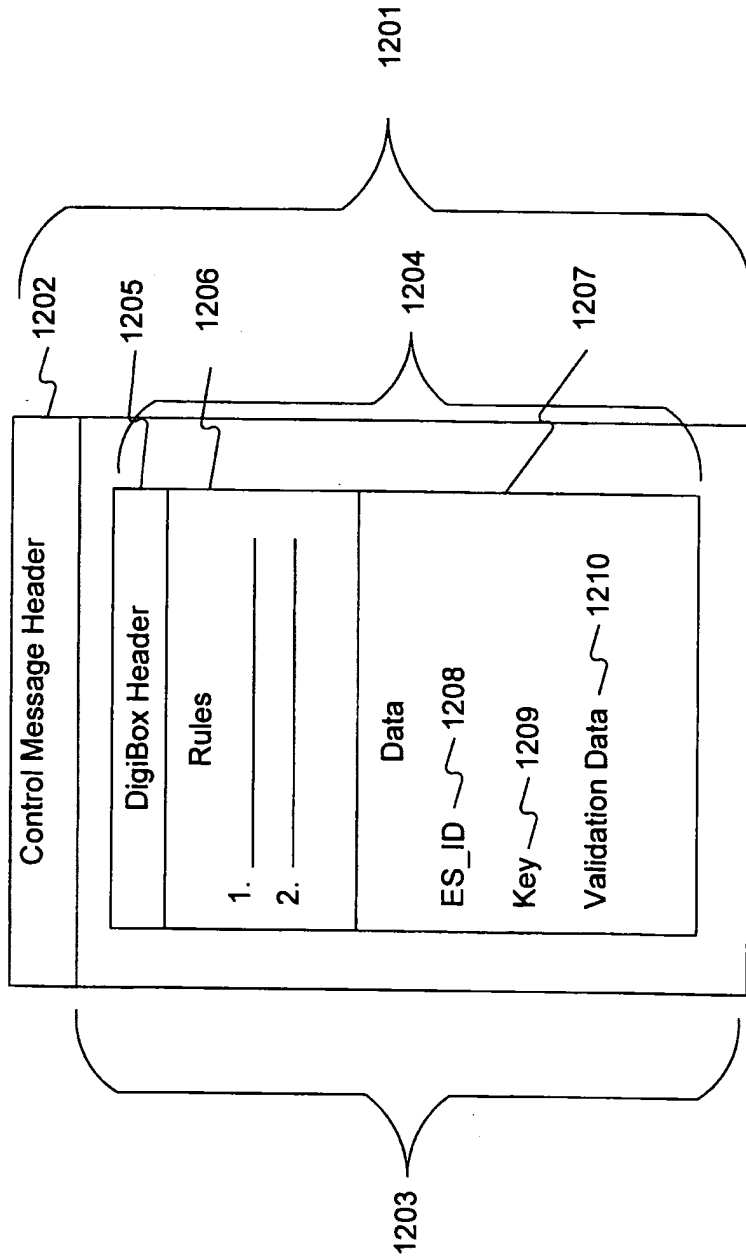


Fig. 12

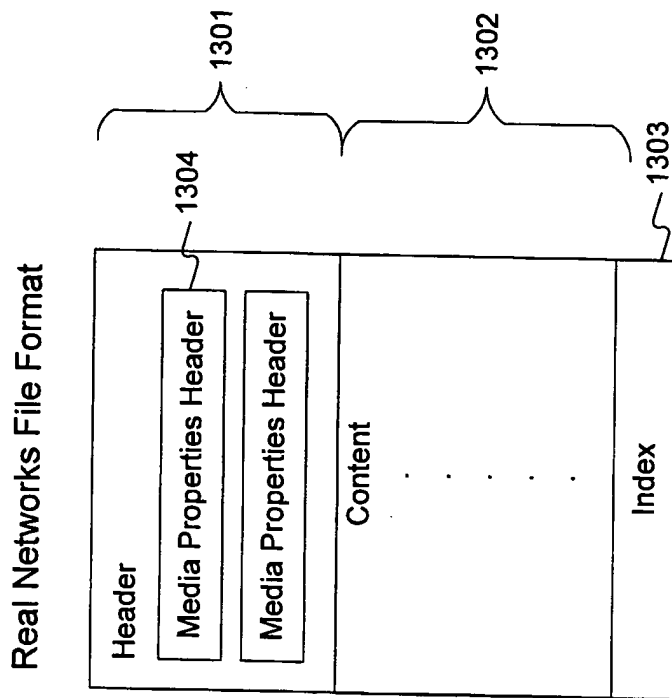


Fig. 13

Real Networks/Protected File Format

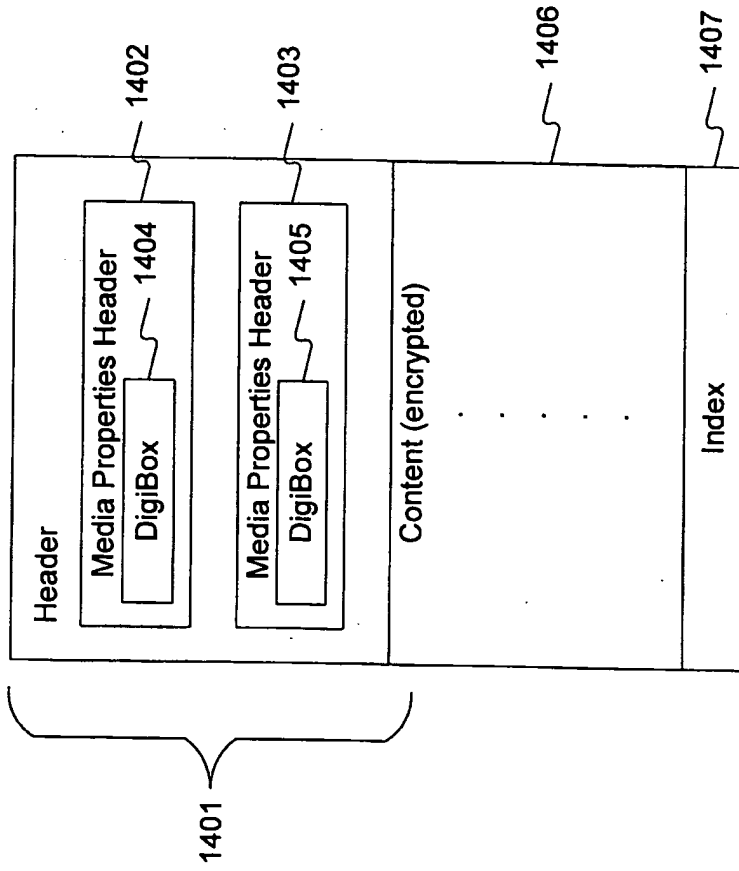


Fig. 14

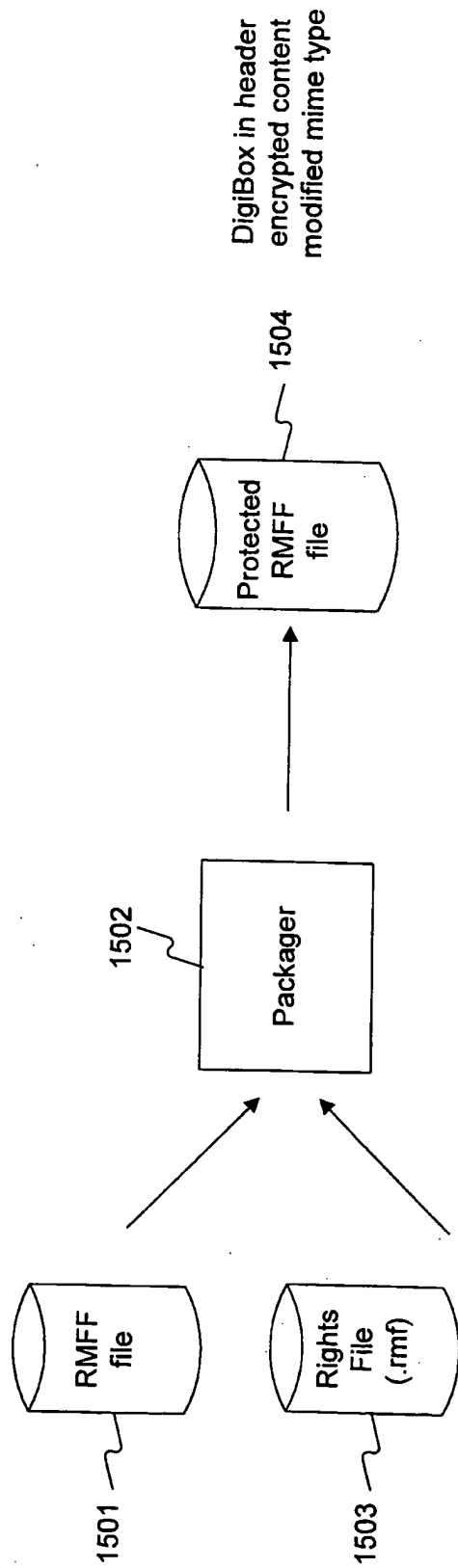


Fig. 15

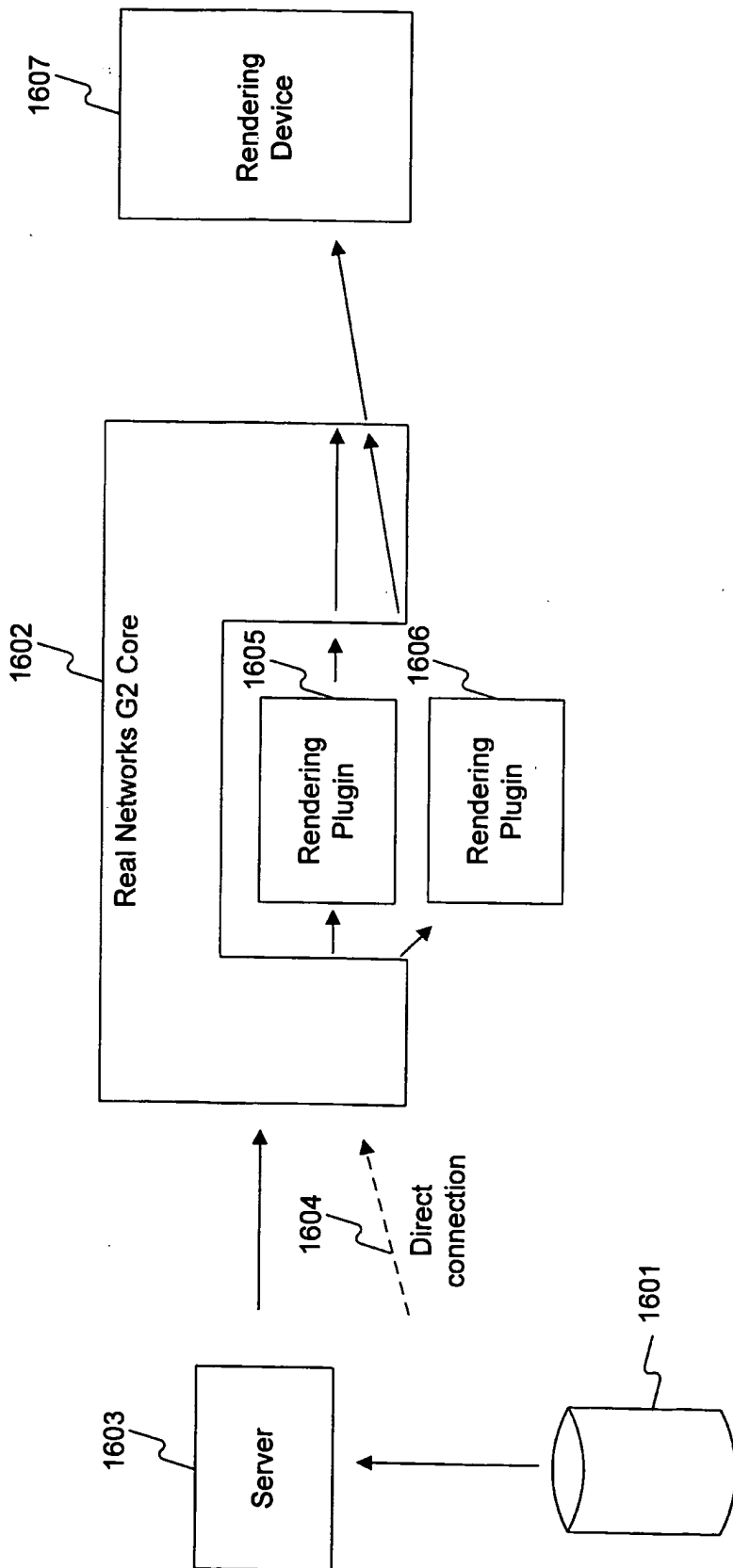


Fig. 16

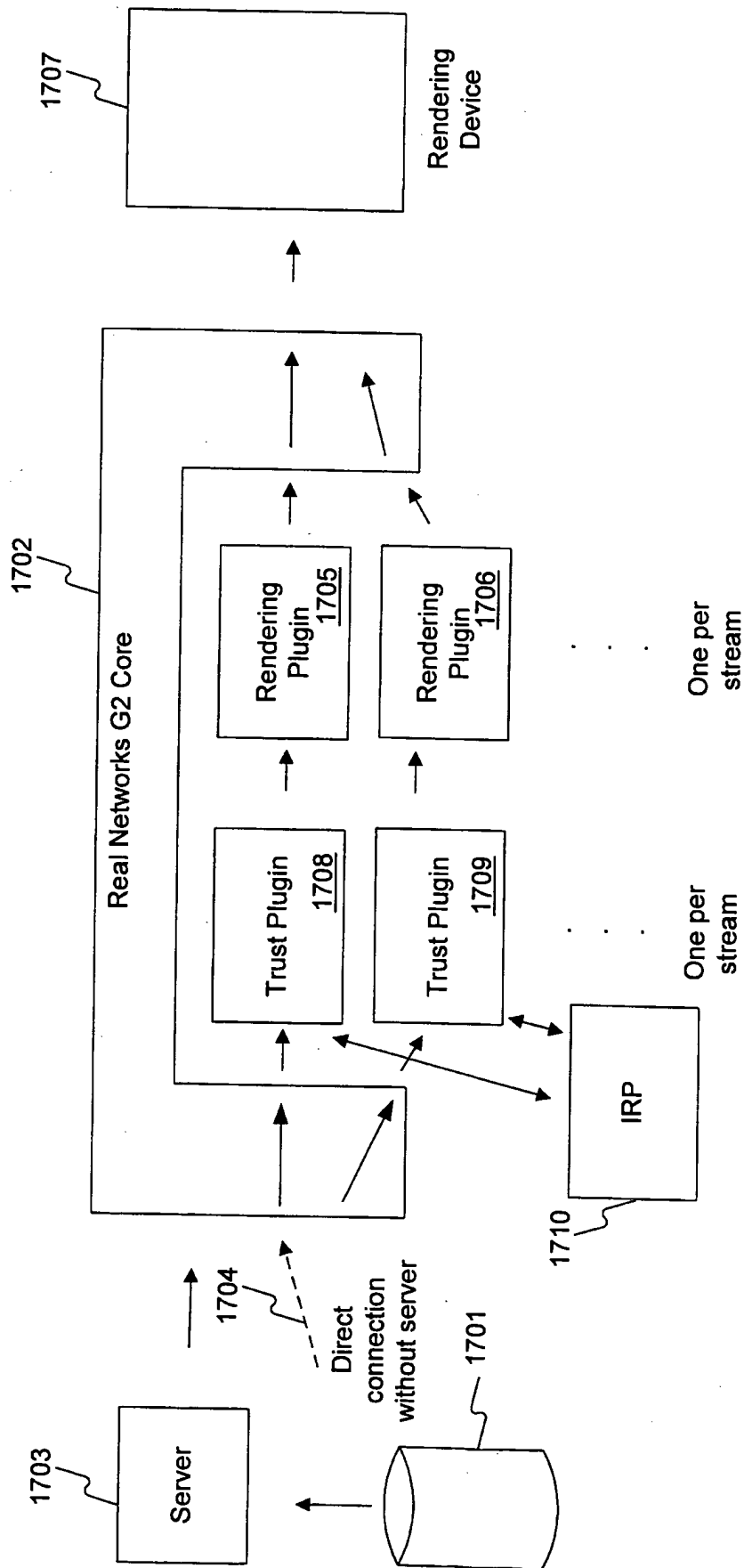


Fig. 17

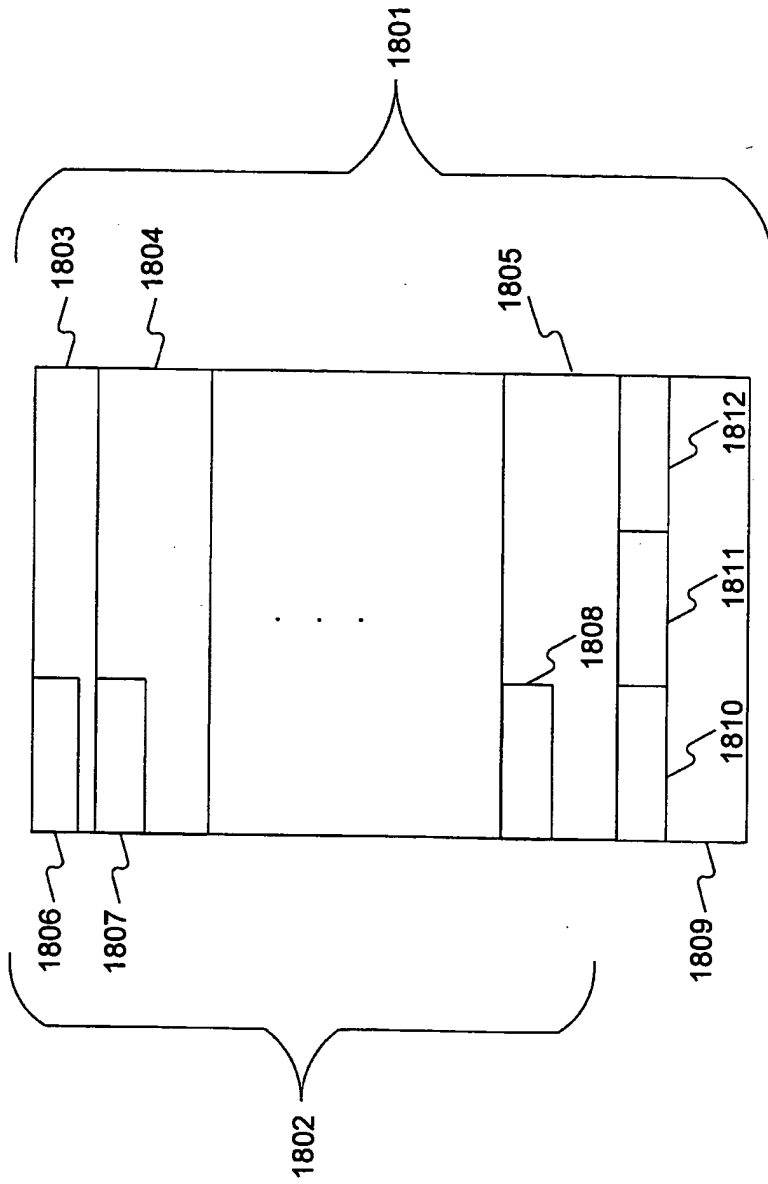
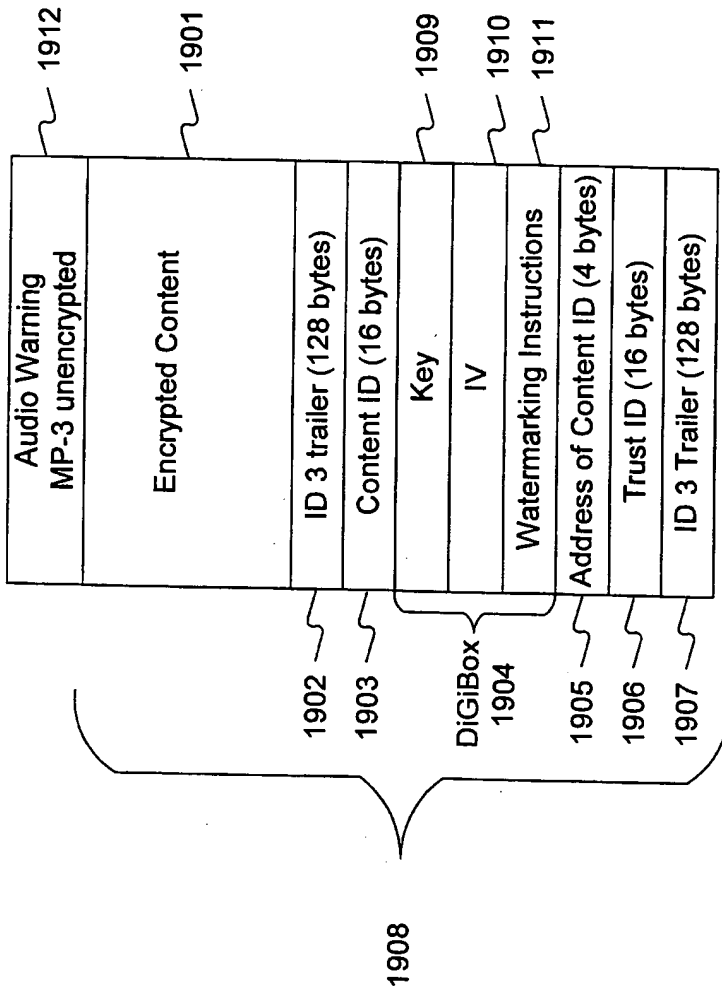


Fig. 18



Protected MP3 Format

Fig. 19

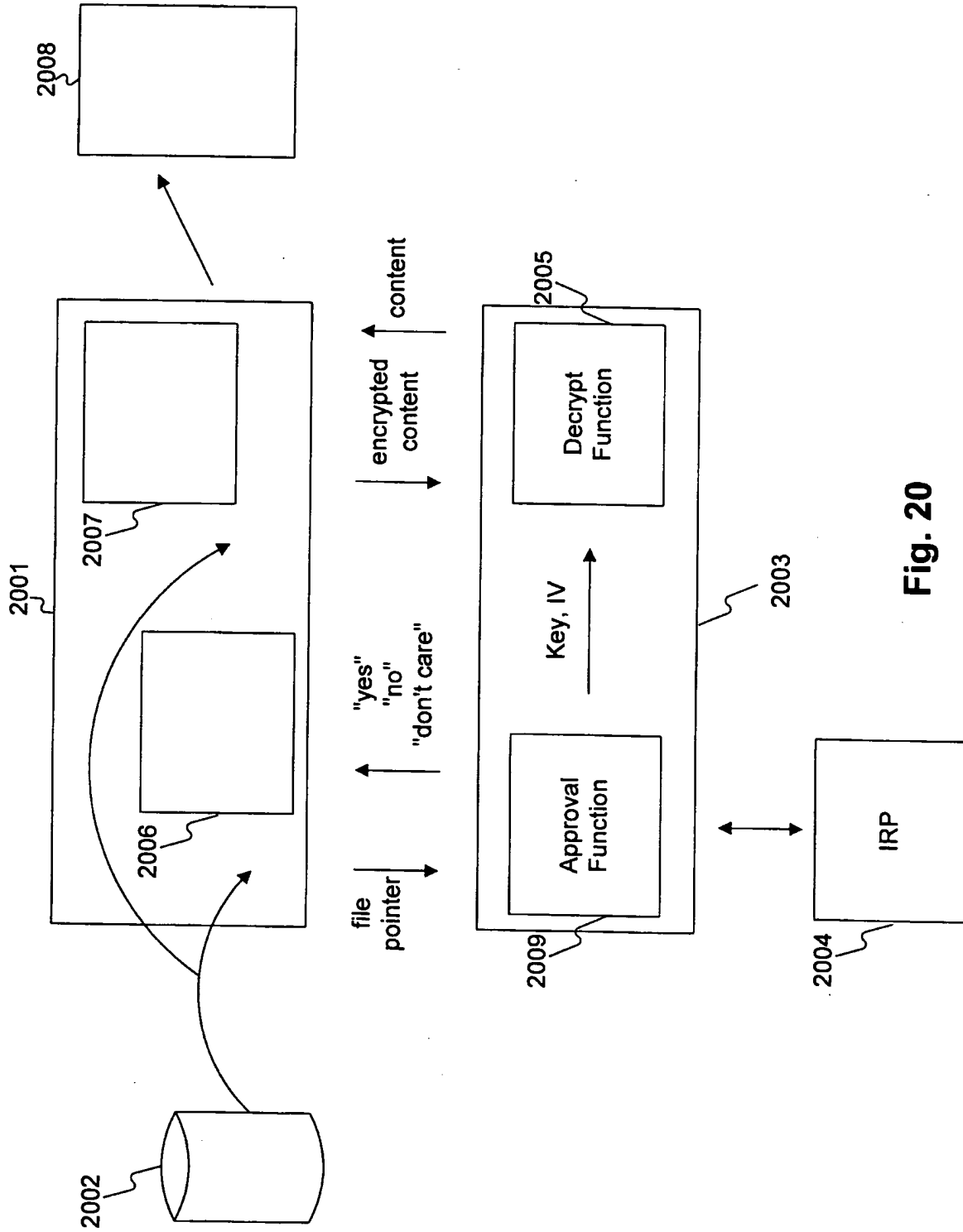


Fig. 20

Creating New MPEG-4 File

- .txt file contains:
- Scene Graph (in text)
- Initial Object Descriptor Cmds

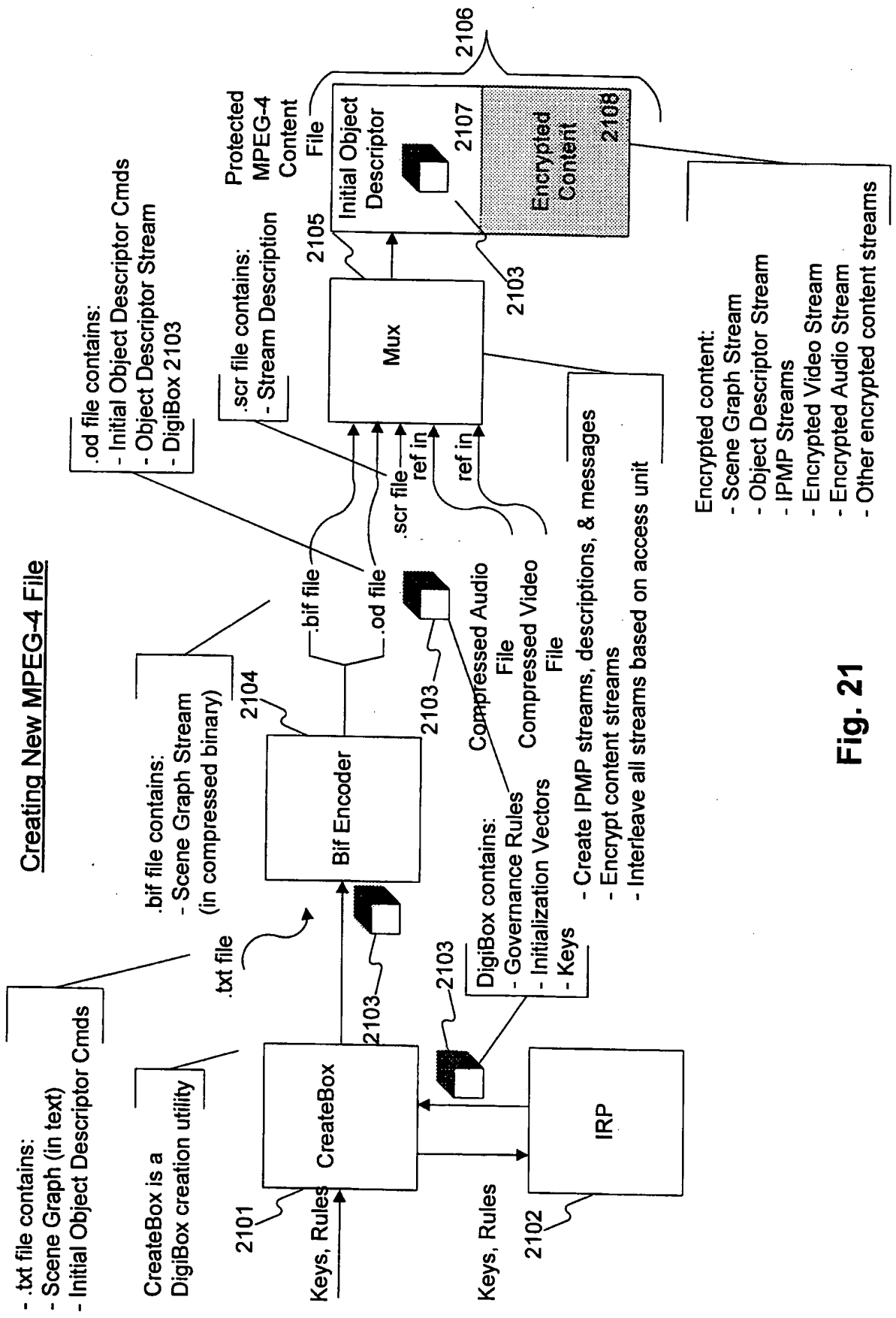
CreateBox is a DigiBox creation utility

- .bif file contains:
- Scene Graph Stream (in compressed binary)

- .od file contains:
- Initial Object Descriptor Cmds
- Object Descriptor Stream
- DigiBox 2103

- .scr file contains:
- Stream Description

Protected MPEG-4 Content File



- Encrypted content:
- Scene Graph Stream
 - Object Descriptor Stream
 - IPMP Streams
 - Encrypted Video Stream
 - Encrypted Audio Stream
 - Other encrypted content streams

Fig. 21

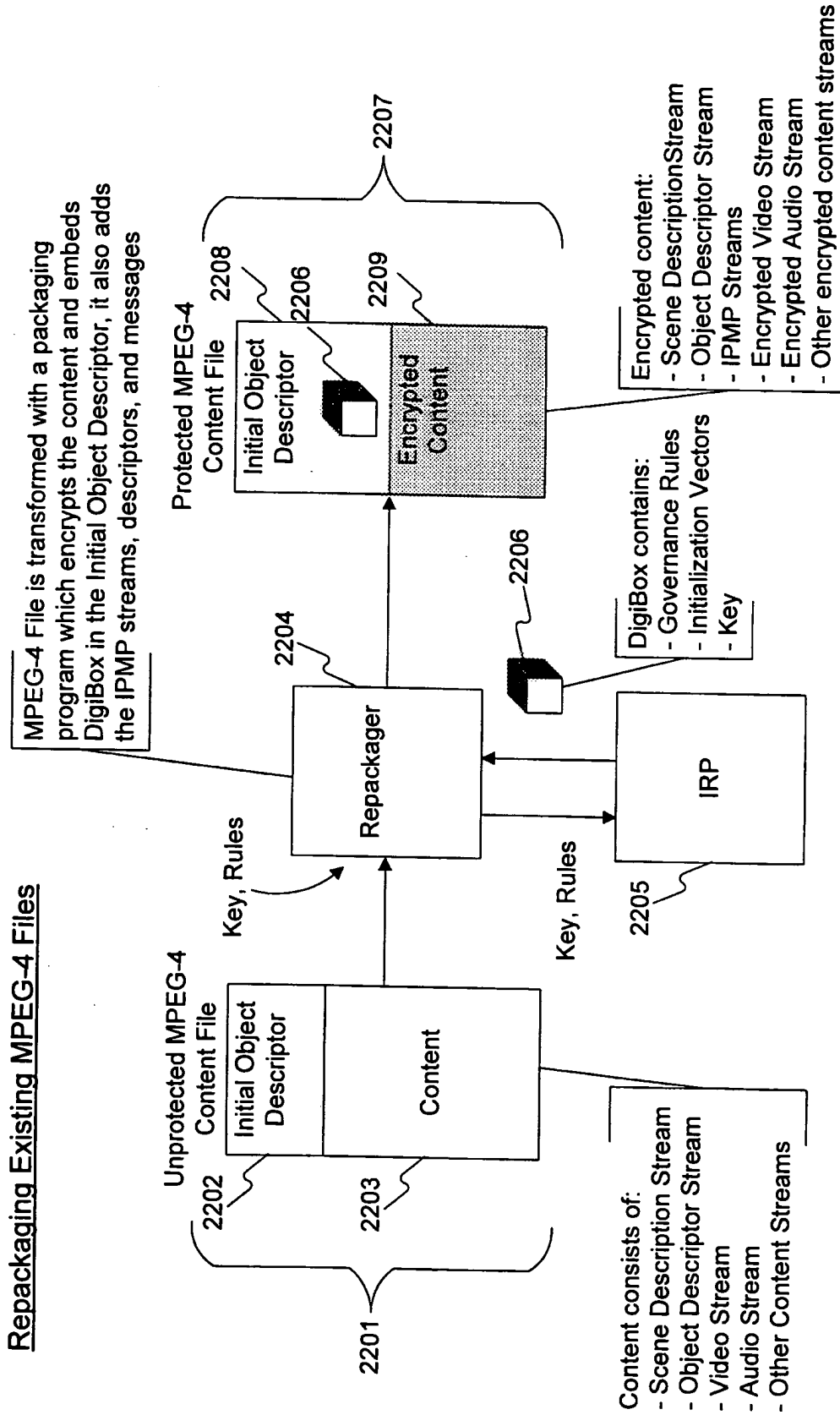


Fig. 22

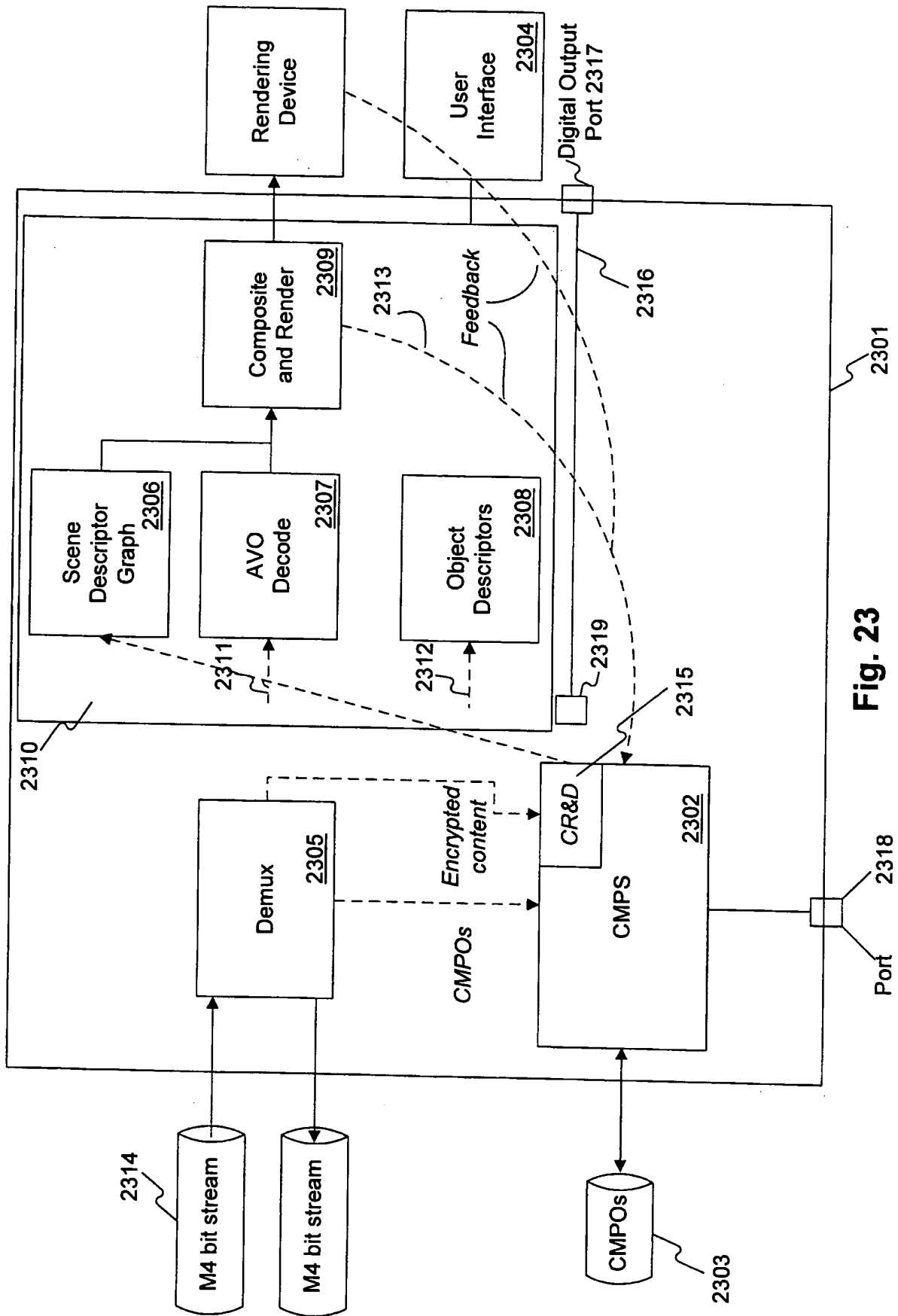


Fig. 23

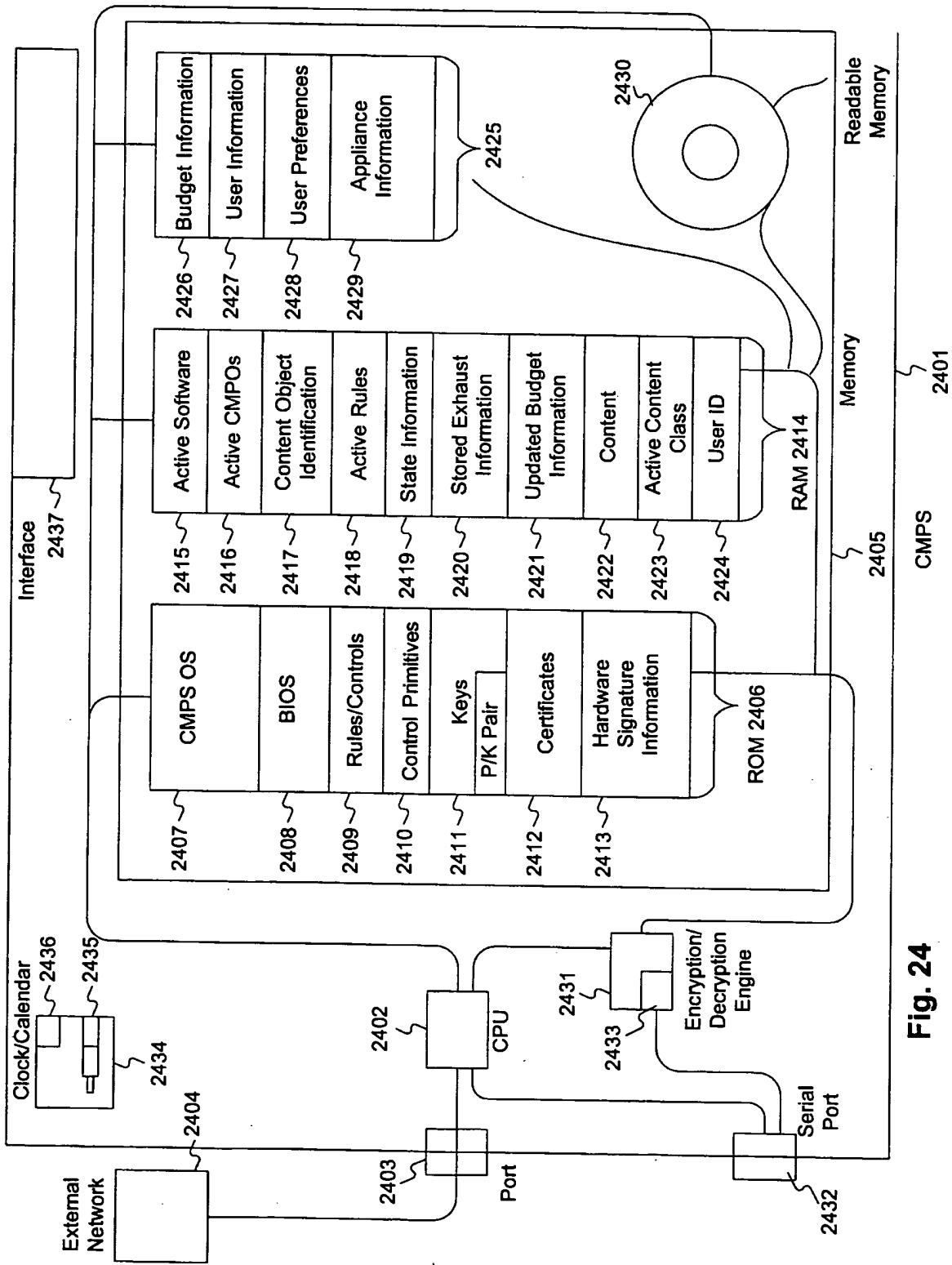


Fig. 24

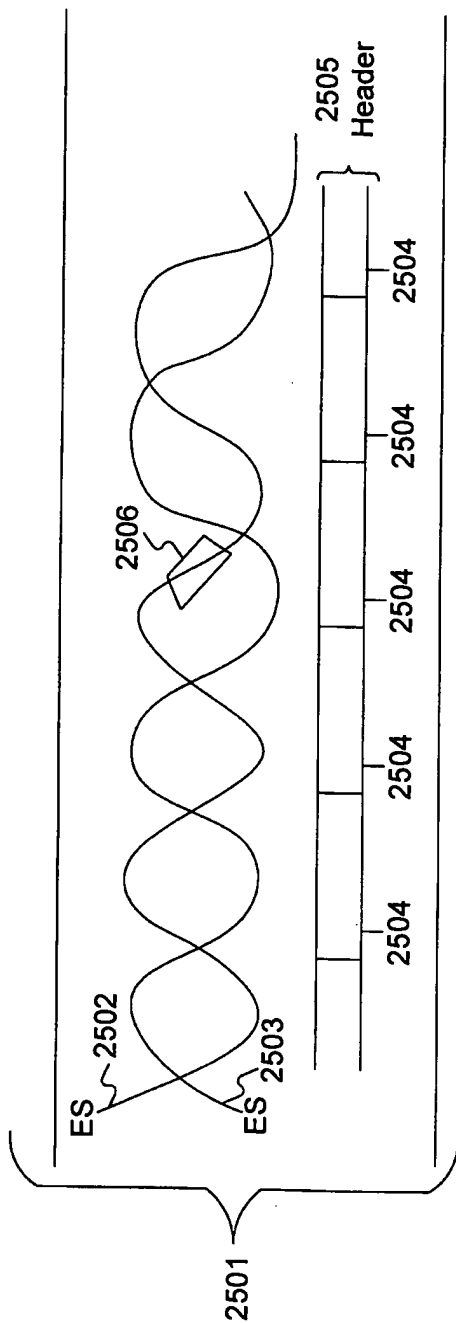


Fig. 25

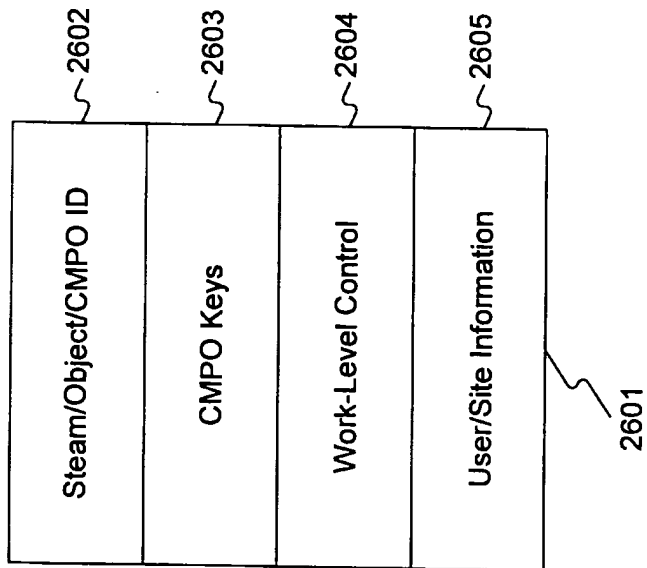


Fig. 26

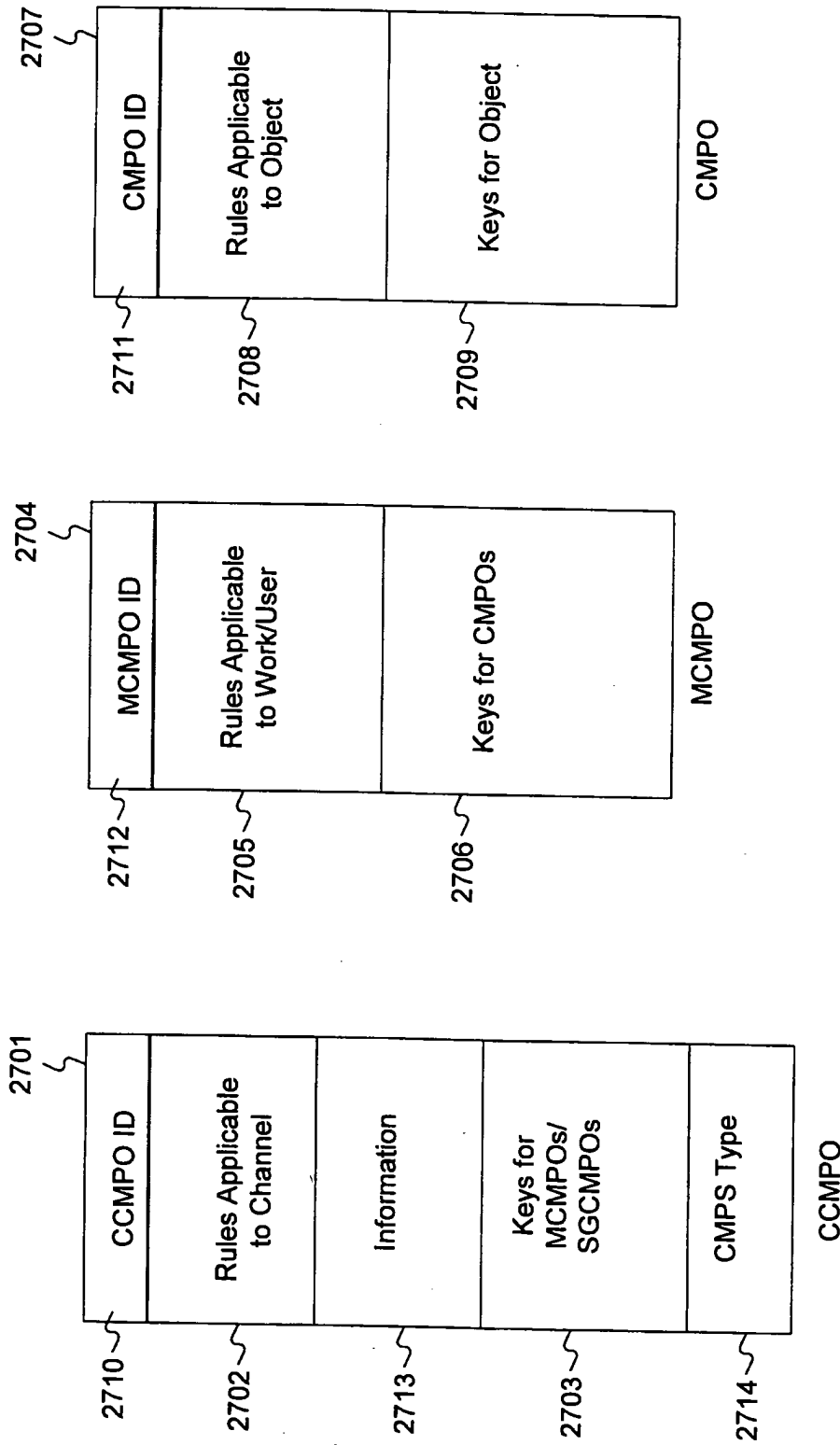
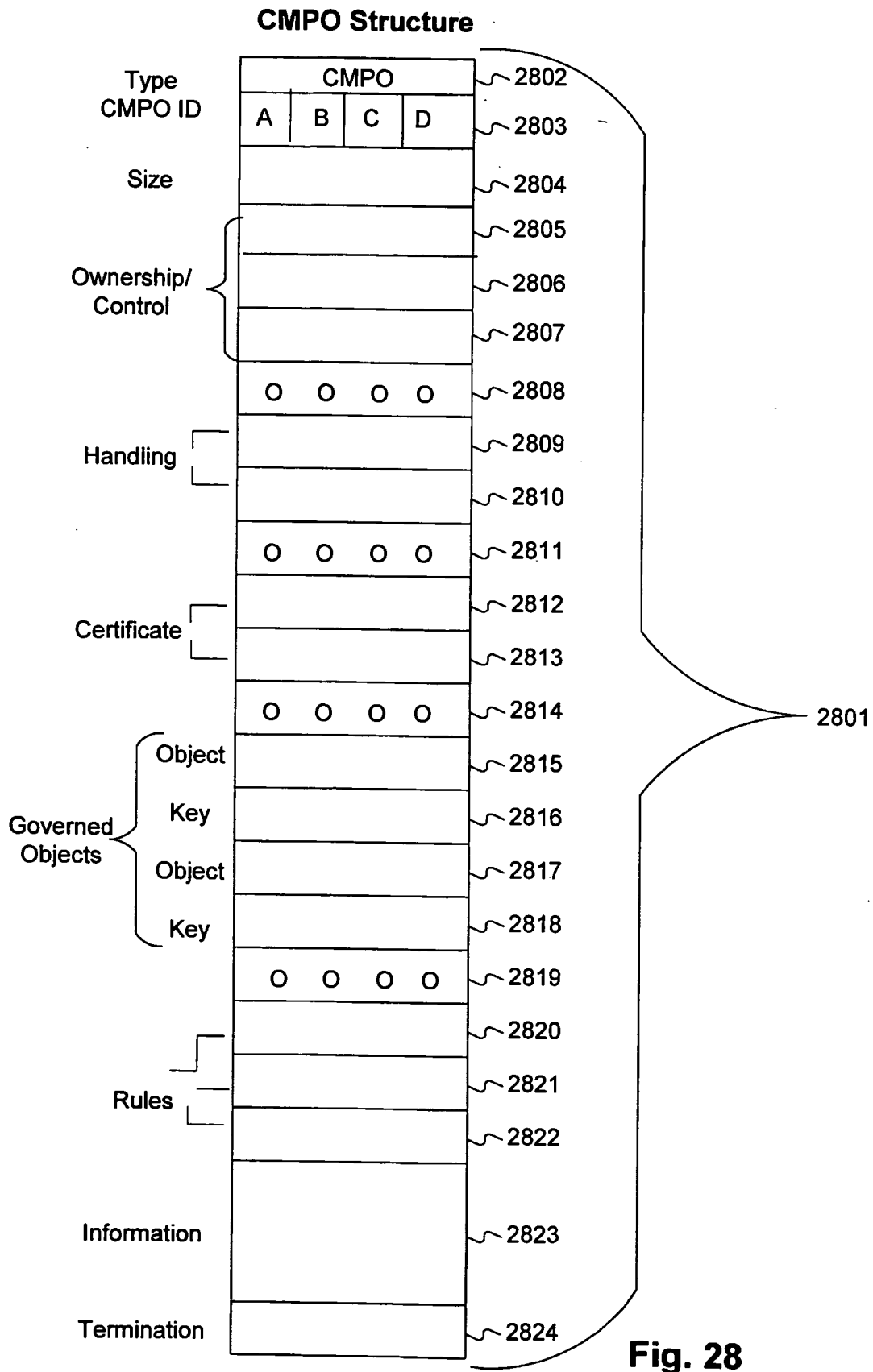


Fig. 27



INTERNATIONAL SEARCH REPORT

International Application No

PCT/US 99/05734

A. CLASSIFICATION OF SUBJECT MATTER
 IPC 6 H04N7/167 G06F1/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC 6 H04N G06F G11B

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

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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 763 936 A (LG ELECTRONICS INC) 19 March 1997	1-4, 6-14, 17-20, 22-25
A	see abstract see column 6, line 27 - column 8, line 4 see column 9, line 6 - column 11, line 43 see column 16, line 47 - column 18, line 38 see figures 4,6A,6B,7 see figures 10,16 --- -/--	5,15,16, 21,26

Further documents are listed in the continuation of box C.

Patent family members are listed in annex.

* Special categories of cited documents :

- "A" document defining the general state of the art which is not considered to be of particular relevance
- "E" earlier document but published on or after the international filing date
- "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
- "O" document referring to an oral disclosure, use, exhibition or other means
- "P" document published prior to the international filing date but later than the priority date claimed

- "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
- "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
- "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.
- "&" document member of the same patent family

Date of the actual completion of the international search

1 July 1999

Date of mailing of the international search report

09/07/1999

Name and mailing address of the ISA

European Patent Office, P.B. 5818 Patentlaan 2
 NL - 2280 HV Rijswijk
 Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,
 Fax: (+31-70) 340-3016

Authorized officer

Hampson, F

INTERNATIONAL SEARCH REPORT

Inte: onal Application No
PCT/US 99/05734

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT		
Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 714 204 A (LG ELECTRONICS INC) 29 May 1996	1-4, 6-14, 22-26
A	see abstract see page 6, line 14 - page 8, line 45 see figures 7,19A,B,20	5,15-21
A	EP 0 715 246 A (XEROX CORP) 5 June 1996 see abstract see page 3, line 46 - page 8, line 27 see figures 1-3,4A,4B	1-26
A	WO 97 25816 A (SONY CORP ;INOUE HAJIME (US); LEE CHUEN CHIEN (US); SONY ELECTRONI) 17 July 1997 see abstract see page 7, line 10 - page 10, line 7 see figures 2,3	1-26
A	EP 0 800 312 A (MATSUSHITA ELECTRIC IND CO LTD) 8 October 1997 see abstract see column 50, line 20 - column 51, line 53	1,6,7, 23-25

INTERNATIONAL SEARCH REPORT

information on patent family members

International Application No

PCT/US 99/05734

Patent document cited in search report	A	Publication date	Patent family member(s)	Publication date
EP 0763936	A	19-03-1997	CN 1150738	28-05-1997
			JP 9093561	04-04-1997
			US 5799081	25-08-1998
EP 0714204	A	29-05-1996	CN 1137723	11-12-1996
			JP 8242438	17-09-1996
			US 5757909	26-05-1998
EP 0715246	A	05-06-1996	US 5638443	10-06-1997
			JP 8263439	11-10-1996
WO 9725816	A	17-07-1997	AU 1344097	01-08-1997
			CN 1209247	24-02-1999
			EP 0882357	09-12-1998
			US 5889919	30-03-1999
EP 0800312	A	08-10-1997	WO 9714249	17-04-1997
			CN 1168054	17-12-1997
			EP 0789361	13-08-1997
			JP 10079174	24-03-1998



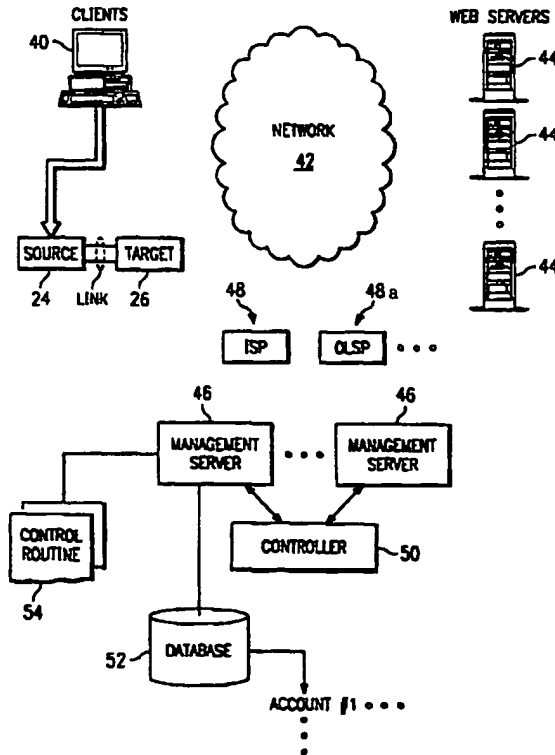
INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

<p>(51) International Patent Classification ⁶ : G06F 1/00</p>	<p>A1</p>	<p>(11) International Publication Number: WO 99/60461 (43) International Publication Date: 25 November 1999 (25.11.99)</p>
<p>(21) International Application Number: PCT/GB98/03828 (22) International Filing Date: 18 December 1998 (18.12.98) (30) Priority Data: 09/080,030 15 May 1998 (15.05.98) US (71) Applicant: INTERNATIONAL BUSINESS MACHINES CORPORATION [US/US]; New Orchard Road, Armonk, NY 10504 (US). (71) Applicant (for MC only): IBM UNITED KINGDOM LIMITED [GB/GB]; North Harbour, Portsmouth, P.O. Box 41, Hampshire PO6 3AU (GB). (72) Inventors: BERSTIS, Viktors; 5194 Cuesta Verde, Austin, TX 78746 (US). HIMMEL, Maria, Azua; 6403 Rain Creek Parkway, Austin, TX 78759 (US). (74) Agent: BOYCE, Conor; IBM United Kingdom Limited, Intellectual Property Law, Hursley Park, Winchester, Hampshire SO21 2JN (GB).</p>		<p>(81) Designated States: CN, CZ, IL, IN, JP, KR, PL, SG, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE). Published <i>With international search report.</i></p>

(54) Title: ROYALTY COLLECTION METHOD AND SYSTEM FOR USE OF COPYRIGHTED DIGITAL MATERIALS ON THE INTERNET

(57) Abstract

A method, system and computer program product to facilitate royalty collection with respect to online distribution of electronically published material over a computer network. In one embodiment, a method for managing use of a digital file (that includes content subject to copyright protection on behalf of some content provider) begins by establishing a count of a number of permitted copies of the digital file. In response to a given protocol, a copy of the digital file is then selectively transferred from a source to a target. Thus, for example, the source and target may be located on the same computer with the source being a disk storage device and the target being a rendering device (e.g., a printer, a display, a sound card or the like). The method logs an indication each time the digital file is transferred from the source to a target rendering device, and the count is decremented upon each transfer. When the count reaches a given value (e.g., zero), the file is destroyed or otherwise prevented from being transferred from the source device. The indications logged are transferred to a management server to facilitate payment of royalties to the content provider.



FOR THE PURPOSES OF INFORMATION ONLY

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

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakistan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

**ROYALTY COLLECTION METHOD AND SYSTEM FOR USE OF COPYRIGHTED
DIGITAL MATERIALS ON THE INTERNET**

BACKGROUND OF THE INVENTION

5

Technical Field

The present invention relates generally to managing collection of royalties for electronically-published material distributed over a computer network.

10

Description of the Related Art

The World Wide Web is the Internet's multimedia information retrieval system. In the Web environment, client machines effect transactions to Web servers using the Hypertext Transfer Protocol (HTTP), which is a known application protocol providing users access to files (e.g., text, graphics, images, sound, video, etc.) using a standard page description language known as Hypertext Markup Language (HTML). HTML provides basic document formatting and allows the developer to specify "links" to other servers and files. In the Internet paradigm, a network path to a server is identified by a so-called Uniform Resource Locator (URL) having a special syntax for defining a network connection. Use of an HTML-compatible browser (e.g., Netscape Navigator or Microsoft Internet Explorer) at a client machine involves specification of a link via the URL. In response, the client makes a request to the server (sometimes referred to as a "Web site") identified in the link and, in return, receives in return a document or other object formatted according to HTML.

15

20

25

30

35

40

45

One of the technical advantages of the World Wide Web is the ease with which digital content (e.g., graphics, sound, video, movies and the like) may be transmitted and distributed to many users. Indeed, copying a digital file is as easy as clicking on a computer mouse. Copyright laws afford a copyright owner the exclusive right to reproduce the copyrighted work in copies, to distribute such copies, and to publicly perform and display the work. Each time a digital file is transferred over the Internet and copied onto a user's memory, the copyright owner's exclusive reproduction right is implicated (and possibly violated). Likewise, transmission of the copyrighted work over the physical wire is tantamount to a distribution. Indeed, in an open system (e.g., a personal computer accessing the world Wide Web through an Internet Service Provider (ISP)), copies of copyrighted materials can undergo unlimited further copying and transmission without the ability of the owner to collect appropriate compensation (e.g., royalties).

Many publishers or other content providers naturally are hesitant to make their copyrighted works available over the Internet due to the ease with which these materials may be copied and widely disseminated without adequate compensation. Presently, Internet commerce remains highly
5 unregulated, and there is no central authority for managing collection and allocation of content provider royalties. Moreover, while publishers and content rights societies and organizations are attempting to address the legal and logistical issues, the art has yet to develop viable technical solutions.

10 One technique that has been proposed involves wrapping a copyrighted work in a copy protection "environment" to facilitate charging users for use of that information obtained from the Internet or World Wide Web. This approach, called COPINET, links a copyright protection mechanism with
15 a copyright management system, and it is described in *Charging, paying and copyright - information access in open networks*, Bennett et al., 19th International Online Information Meeting Proceedings, Online Information 1995 pp. 13-23 (Learned Information Europe Ltd.). Publishers in such a system can determine an appropriate level of protection while monitoring
20 use and managing the chain of rights. This approach is also said to provide protection for digital material even after delivery to the user workstation. In particular, copyright material is "wrapped" (by encryption) and "unwrapped" as a result of a specific authorization provided by a trusted subsystem. Material thus is only "visible" to the
25 environment and thus any subsequent user actions, such as "save" or "copy", result in the protected material, or material derived from it, remaining in a protected state when outside the environment.

30 Although the above-described approach provides some advantages, it does not address the problem of managing the collection of royalties and/or the allocating of such payments to content providers. Moreover, it is not an accepting solution in the context of an open PC architecture such as implemented in the public Internet. It also requires the use of a separate trusted subsystem to generate the authorizations for particular
35 content transfers, which is undesirable.

Other known techniques for managing use of content over the Internet typically involve electronic "wallets" or smart cards. Known prior art systems of this type are illustrated, for example, in U.S. Patent Nos.
40 5,590,197 and 5,613,001. These systems involve complex hardware and encryption schemes, which are expensive and difficult to implement in practice. They are not readily adaptable to provide general royalty payment schemes for Internet content usage.

Thus, there remains a need to provide improved methods and systems for collecting royalties on the Internet as a result of use of copyrighted content.

5 The present invention solves this important problem.

SUMMARY OF THE INVENTION

10 An object of this invention is to enable a pair of "certified" devices (e.g., a storage device and a rendering device) to operate within the context of a given security protocol and thereby manage copies of a digital file and associated copy control information.

15 Still another object of this invention is to enable a copyright proprietor to maintain a degree of control over copyrighted content even after that content has been fetched from a server and downloaded to a client machine, e.g., in a Web client-server environment.

20 A particular object of the present invention is to manage the number of copies of a digital file that may be made within a Web appliance having a secure disk storage and that is connectable to the Internet using a dialup network connection.

25 A still further object of this invention is to restrict a number of copies of a digital file that may be made at a given Web client machine connected to the World Wide Web.

30 It is yet another object of this invention to enable a publisher of an electronic document to control the number of copies of such document that may be made on the Internet by permitted users.

 It is a more general object of this invention to manage permissible use of copyrighted content on the Internet and World Wide Web.

35 It is still another more general object of this invention to manage collection of information to facilitate payment of appropriate compensation to content providers and publishers arising from use of their copyrighted content on the Internet.

40 Another object of this invention is to manage the charging of users for information obtained from the Internet or World Wide Web.

 A still further object of this invention is to facilitate royalty collection as a result of electronically published material distributed

online over a computer network (e.g., the public Internet, an intranet, an extranet or other network).

5 One embodiment of the invention is a method for managing copies of a digital file, which includes content subject to copyright protection, on behalf of some content provider (e.g., an author, publisher or other). It is assumed that a given usage scheme has been established with respect to the file as defined in copy control information associated with the file. Thus, for example, the copy control information may define a set of
10 payment options including, without limitation, prepayment (for "n" copies), pay-per-copy (as each copy is made), IOU (for copies made offline), or some other payment option. The copy control information may also include other data defining how the file is managed by the scheme including: a count of the number of permitted copies, a count of the
15 number of permitted pay-per-copy versions, copyright management information, payee information, an expiration date (after which copying is no longer permitted), and the like.

The present invention assumes the existence of a pair of devices, a
20 "source" and a "target", that have been or are certified to use the scheme. Typically, the "source" is a storage device while the "target" is a rendering device. An illustrative storage device may be disk storage, system memory, or the like. An illustrative rendering device may be a printer, a display, a sound card or the like. The source and target
25 devices may both be storage devices (e.g., a Web server and a client disk storage). In either case, each of the devices comprising the pair is "certified" (typically upon manufacture) to operate under a given security protocol. Under the protocol, the devices include appropriate circuitry and/or software, as the case may be, to facilitate the establishment of a
30 secure link between the storage and rendering devices. Each device requires the other to validate itself and thus prove that the device can be trusted to manage the content (namely, the digital file) sought to be protected.

35 When the technique is implemented in an "open" client-server environment, hardware devices (e.g., microcontrollers) preferably are used in the storage and rendering devices to facilitate generation of the secure link. When the technique is implemented in a "closed" Web appliance environment, the secure link may be established and managed
40 using software resident in the control routines associated with the storage and rendering devices. The secure link may be established and managed in software under such conditions because, in the Web appliance environment, it is possible to readily disable the secure link in the event of tampering with the appliance housing or other circuitry.
45 Regardless of the environment, the secure link is first established

between the "certified" storage and rendering devices. Thereafter, the digital file, together with at least part of its copy control information, is transferable between the storage and rendering devices in accordance with the particular usage and payment scheme being utilized. Thus, for example, if a prepayment scheme is implemented and an expiration date (associated therewith) has not occurred, a given number of copies of the file may be transferred between the storage and rendering devices. The prepayment funds are collected at a central location and then redistributed to the copyright proprietor or some third party.

10

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a representative system in which the present invention is implemented;

15

Figure 2 is a simplified block diagram of a source device and a target device connected by a channel over which a digital file is transferred according to the present invention;

20

Figure 3 is an illustrative example of a source device connected to a set of target rendering devices in a client computer;

Figure 4 is a block diagram of a representative copyright management system according to the present invention;

25

Figure 5 is a flowchart of a preferred method of managing a digital file according to the present invention;

Figure 6A is pictorial representation of a data processing system unit connected to a conventional television set to form a "Web" appliance;

30

Figure 6B is a pictorial representation of a front panel of the data processing system unit;

Figure 6C is a pictorial representation of a rear panel of the data processing system unit;

35

Figure 6D is a pictorial representation of a remote control unit associated with the data processing system unit; and

40

Figure 7 is a block diagram of the major components of the data processing system unit.

45

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

A representative system in which the present invention is implemented is illustrated in Figure 1. A plurality of Internet client machines 10 are connectable to a computer network Internet Service Provider (ISP) 12 via a "resource" such as a dialup telephone network 14. As is well known, the a dialup telephone network usually has a given, limited number of connections 16a-16n. ISP 12 interfaces the client machines 10 to the remainder of the network 18, which includes a plurality of Internet server machines 20. A client machine typically includes a suite of known Internet tools (e.g., Web browser 13) to access the servers of the network and thus obtain certain services. These services include one-to-one messaging (e-mail), one-to-many messaging (bulletin board), on-line chat, file transfer and browsing. Various known Internet protocols are used for these services. Thus, for example, browsing is effected using the Hypertext Transfer Protocol (HTTP), which provides users access to multimedia files using Hypertext Markup Language (HTML). The collection of servers that use HTTP comprise the World Wide Web, which is the Internet's multimedia information retrieval system.

As will be described in more detail below, the present invention may be implemented in hardware and/or in software. The software implementation is particularly useful when the client machine is an Internet or Web appliance, such as illustrated in Figures 6A-6D. In the case of the software implementation, a client machine has associated therewith a software routine 15 designed to perform one or more of the functions of the digital file copy protection method, as will be described. The software is preferably a client application (although it may be implemented with the browser as a plug-in, or with a client-side proxy, or as a standalone application). Alternatively, the agent is built into the browser, or it is implemented as a Java applet or standalone application. Thus, as used herein, in this particular embodiment, the software 15 is any application running on a client machine 10 that performs the copy protection/royalty management task(s) on behalf of the user(s) of that client according to the present invention.

The discussion which follows primarily uses the words "copying" or "copies" to describe the control of the further exercise of a copyright right for a particular work. The reader should understand that "copying" could include other types of rendering of the work for different devices. That is, "copying" in a printer would entail printing on paper or another substrate. Copying on a display is presenting an image on the screen. Copying in an audio device would be the performance of an audio portion of the work. Each of these devices both storage devices, e.g., hard disks, tapes in CDR, and rendering devices, e.g., prints, display graph, audio

player, movie player, should be equipped with the present invention so that the copies are controlled throughout the systems and networks until their final rendering place.

5 The present invention is a method for managing copies of a digital file, which includes content subject to copyright protection, on behalf of some content provider (e.g., an author, publisher or other). It is assumed that a given payment scheme has been established with respect to the file. Thus, for example, such payment schemes include, without
10 limitation, prepayment (for "n" copies), pay-per-copy (as each copy is made), IOU (for copies made offline), or some other payment option. In a prepayment option, a user prepays funds for the right to obtain copies of the digital file. In a pay-per-copy (or "pay as you go") option, the user pays for each copy of the digital file when the file is copied. In an IOU
15 scheme, the user makes copies of the digital file (e.g., while the client machine is not connected to the network) and generates an IOU (or many IOUs) that are then submitted to a clearinghouse or other payment entity when the user later goes online. Other payment schemes (such as a combination of the above options) may also be implemented.

20 The payment scheme is preferably defined in copy control information associated with the file and established by the author, publisher or some other third party. Thus, for example, the copy control information may also include a count of the number of permitted copies, a count of the
25 number of permitted pay-per-copy versions, a count of the number of copies that may be made under an IOU payment option, copyright management information identifying the author, publisher and/or other license or use restrictions, information about a bank or other financial institution that handles use payments and their reconciliation, one or more expiration
30 dates (after which copying is no longer permitted), and the like.

 The copy control information associated with a given file thus defines a usage scheme for the file because it includes information that controls how the content may be used, how such use is paid for, over what
35 period the content may be used, and other such information. A particular usage scheme (or some portion thereof) may also be implemented in the devices between which the file is transferred, although preferably such restrictions are defined by the content provider.

40 According to the present invention as illustrated in Figure 2, the present invention assumes the existence of a pair of devices, a "source" 24 and a "target" 26, that have been or are certified to use the scheme. In particular, devices that implement the inventive scheme preferably include a device certificate that is not accessible (and thus is free from
45 tampering) and stored therein. The certificate evidences that the device

is capable of understanding a given security protocol useful in carrying out the protection scheme. A representative security protocol is CSS, or the Content Scrambling System protocol, available commercially from Matsushita Corp. Thus, for example, if the source device is a disk
5 storage, the device certificate is typically stored inside a secure chip within the device control hardware. Typically, each of the devices is "certified" upon manufacture, although this is not a requirement.

As also illustrated in Figure 2, a channel 28 is established between
10 the source and target devices over which copies of a digital file (that is subject to the scheme) are communicated in a secure fashion. Thus, prior to transfer of the digital file, the channel 28 is first established between the devices to ensure that the copy restrictions (such as set forth in the copy control information) may be enforced. Typically, this
15 is accomplished by having each device (in accordance with the security protocol implemented) require the other device (of the pair) to verify that its device certificate is valid. An appropriate message exchange may be used for this purpose as defined in the protocol. Once the secure link has been established, each of the devices can be trusted to control the
20 digital file in accordance with the file's copy control information.

Typically, the "source" 24 is a storage device while the "target" 26 is a rendering device. An illustrative storage device may be disk storage, system memory, or the like. An illustrative rendering device may
25 be a printer, a display, a sound card or the like. The source and target devices may both be storage devices (e.g., a Web server and a client disk storage).

When the technique is implemented in an "open" client-server
30 environment, hardware devices (e.g., microcontrollers) are used in the storage and rendering devices to facilitate generation and management of the secure link. When less security may be tolerated, some of these functions may be implemented in software. When the technique is implemented in a "closed" Web appliance environment (Figures 6A-6D), the
35 secure link may be established in whole or in part using software resident in the control routines associated with the storage and rendering devices. The secure link may be established in software under such conditions because, in the Web appliance environment, it is possible to readily disable the secure link in the event of tampering with the appliance
40 housing or other circuitry. Regardless of the environment, the secure link is first established between the "certified" storage and rendering devices. Thereafter, the digital file, together with at least part of its copy control information, is transferable between the storage and rendering devices in accordance with the particular usage scheme defined,
45 for example, by the copy control information. Thus, for example, if a

prepayment scheme is implemented and an expiration date (associated therewith) has not occurred, a given number of copies of the file may be transferred between the storage and rendering devices.

5 Thus, as illustrated in **Figure 2** in simplified form, the digital file copy protection method and system of the present invention involves a "source" device **24** (or one or more of such devices), and a set of one or more "target" devices **26a-n** connected via the secure channel or link **28**. The physical characteristics of the channel, of course, depend on whether
10 the source and target devices are located in the same machine or are in separate machines connected via a network. In a network connection, the link may be a conventional TCP/IP connection. Channel **28** may be a physically secure channel (such as a https connection), but this is not required as the given security protocol in the certified devices
15 establishes a secure link. According to the invention, once the link is established, one or more digital files are transferred (under the control of a control routine or mechanism) between the certified devices in an predictable, auditable manner so that (a) a controlled number of file transfers can be made, and (b) the precise number of file transfers (and
20 their particular use) may be readily documented to facilitate dissemination of royalties or some such other consideration, typically to providers of such content. Generalizing, prior to transfer of a given digital file (or set of files, or file component) from the source to the target via the secure link, that transfer must first be authorized, and
25 the transfer itself is then capable of being associated with some royalty payment then due to a content provider for use of such file. The scheme thus facilitates implementation of a generalized copyright management/royalty collection and distribution scheme.

30 As previously mentioned, the source **24** and target **26** may be located on the same computer. **Figure 3** illustrates this particular connection for a disk storage subsystem **24'** and the target rendering devices, namely printer **26a'**, display **26b'** and sound card **26c'**. The illustrated computer is a Web appliance, in which case the secure link may be established (as
35 noted above) using software. Thus, in this example, each source and/or target device includes appropriate control software (part of software **15** as described above) to facilitate creation of the secure channel. Although not meant to be limiting, one convenient mechanism to create the channel involves each of the devices to generate a random number **30**, which
40 numbers are then supplied to a key generation algorithm **32** in a known manner to generate a secret of "private" key **34**. The key **34** may be generated for each digital file to be transferred over the link **28**, or a signal key may be used for a set of such files, or even for a particular browsing session. To create the secure channel, the software resident on
45 the disk storage encrypts the digital file as it leaves the source device.

The target device then decrypts the digital file using the key prior to rendering. In this way, the digital file cannot be readily intercepted as it is being transferred between these devices. As noted above, each of the source and target devices may also include secure chips or other known hardware devices to facilitate or augment such secure transfer of the digital file between the devices.

The particular mechanism for securing the channel between the source and target may be quite varied, and the present invention contemplates the use of any now known or later-developed technique, system or method for securing such communications. Thus, for example, another technique that may be used would be a public key cryptosystem.

Figure 4 is a block diagram illustrating a representative copyright royalty management system implemented according to the present invention. In this system, it is assumed that client computers 40 access the computer network 42 (e.g., the public Internet, an intranet, an extranet, or other computer network) to obtain access to Web-like documents supported on Web servers 44. One or more management servers 46 are connectable to the system via an access provider 48, and a control management server 50 may be used to facilitate scaling of the architecture if required. Control management server 50 may be controlled by a regulatory or rights agency that has responsibility for managing collection and distribution of copyright royalties.

A given management server includes a database 52 and appropriate control routines 54 for establishing a royalty account 55 for content providers. It is envisioned (although not required) that given content providers will subscribe to a royalty collection service implemented by the present invention and perhaps pay a fee (e.g., a commission or service charge) for the service provided. A given content provider thus may subscribe to the service to receive royalty payments for the use of his or her copyrighted content by users of the client machines. To this end, control routines 54 are used to establish an account for each of a set of given content providers, with each account including a representation of a given royalty value (which may be \$0 when the account is established). A control routine then adjusts the given royalty value in a given provider account in response to receipt of an indication that a given digital file associated with the given content provider has been transferred from a source 24 to a target rendering device 26 in a given client computer 40. Periodically, the content provider account is adjusted for any service or processing fees, and the remainder of the account is then distributed to the content provider. In the situation where the content provider is willing to allow his or her content (a given digital file) to be used with charges for such use paid later, a given bit may be set in the file's copy

control information indicating such preference. Other data in the copy control information may be used to set or control other content provider preferences with respect to use of the file within the context of the inventive scheme.

5

Figure 5 is a flowchart of one method of managing royalty account collection with respect to a particular digital file when a prepayment option is utilized. In this representative example, the digital file is an image (i.e. a .jpeg file) having a copyright owned by a given content proprietor or provider. Of course, the principles of the present invention are designed to be implemented collectively with many such digital files, and the following description is thus merely representative of one type of basic payment scheme. The routine assumes initially that a usage or payment account has been established for a given client computer (or a user of that computer). This is step 60 in the flowchart. It is also assumed that a royalty account has been established for the content provider at one of the management servers as previously described. This is step 62 in the flowchart. One of ordinary skill will appreciate that steps 60 and 62 need not be in any particular sequence. Step 60 typically involves the user prepaying some amount of funds into an account from which payments may be withdrawn, although this is not required.

At step 64, a count is established by a control routine for the particular digital file. Typically, this is a count of a number of permitted copies of the digital file that may be transferred from the source to one or more target devices according to the present invention. This number, as noted above, is typically identified in the file's copy control information. The count is usually a positive integer, which is then decremented (by the control routine) down to zero as permitted or authorized copies are made. Alternatively, of course, the count may begin at zero (or any other arbitrary number), which is then incremented (by the control routine) to the threshold value identified in the copy count information. As noted above, the count may be set by the copyright proprietor, by a system operator, by a Webmaster, by hardware constraints, or by any other party or entity having authority and/or ability to set the count. Under certain circumstances, e.g., where a prepaid user account is used, it may be unnecessary to use an explicit count as the number of copies transferred may simply depend on the royalty assessed per copy. Thus, the "count" as used herein may be expressed explicitly or implicitly. The digital file may be stored on the client already, or it may be available from a Web server or other storage or archive. The particular location from which the digital file is sourced initially does not matter. Step 64 assumes, however, that the image is located already at the source device. If the file is not present at the source, it may be

necessary to obtain it (although, conceptually, the "source" may be broadly construed as the original or initial location of the file).

5 At step 66, a test is done repeatedly to determine whether a request
for the image has been received. If not, the routine cycles on step 66
and waits for such a request. If the outcome of the test at step 66 is
positive, then the routine continues at step 68 by testing whether the
10 given client computer (which generated the request) is authorized to
effect the transfer. Step 68 may comprise a simple comparison of the
user's account balance and the royalty amount to be assessed. If the
user's account balance is large enough, the transfer may be allowed. Or,
step 68 may simply test whether the count has a value indicating that
15 further copies may be made. More typically, step 68 will require that the
count be non-zero (in the situation where the count is positive and
decremented to zero) and the user have sufficient funds allocated to pay
the royalty assessment for use of the image. The step 68 may also test
whether a given expiration date set in the copy count information has
past.

20 If the outcome of the test at step 68 is negative, the transfer is
not authorized, and the routine branches to step 70 to so notify the user
of the client machine. Such notification may be in the form of an error
or "access denied" message or the like. The user may be informed merely
that a preset expiration date has passed or that his or her prepaid
25 account is exhausted and requires more funds. If, however, the outcome of
the test at step 68 is positive, the digital file may be transferred to
the target. The routine then branches to step 72 to initiate the copy
transfer. Preferably, all bytes of the file must be transferred before
the transfer is considered valid. At step 74, the control routine count
30 is adjusted (e.g., decremented) and/or a given charge is allocated against
the user's account. The given charge may be equal to the royalty or use
charge, or some fixed percentage thereof (e.g., 105%) reflecting that
royalty plus some service charge). At step 76, the appropriate content
provider account is adjusted by the amount of the royalty payment (plus or
35 minus appropriate service fees or other charges).

 Neither step 74 nor step 76 need occur at the time of the file
transfer. Typically, the account adjustments will take place in batch at
a given time. Thus, for example, where the Web client is a Web appliance
40 connected to the computer network via a dialup connection, the account
information may be transferred to the management server upon establishing
a given connection (e.g. perhaps once each day). Other variations
regarding the timing of delivery of this information are, of course,
within the scope of the present invention.

45

The present invention thus provides numerous advantages. Certified source and target devices first establish a secure link between themselves. Upon transfer of the file copy between source and target, the control routine records an appropriate indication thereof in the copy count, and the central authority is notified of the transfer of the digital file. Such notification may occur upon transfer of the digital file between the source and target devices, or at some later time (e.g., upon dialup connection of the computer to the network). Royalty accounts are then managed at a central authority; to facilitate distribution of royalties to content owners/publishers. When the copy count reaches the authorized limit (as set in the copy control information), the control routine destroys the file or otherwise prevents further copying of the digital file.

Thus, in one embodiment, the user establishes a "prepaid" account from which royalty or usage payments are drawn against as files are copied/transmitted. The system detects use of the file and, preferably, allows only a certain number of copies of the file to be made before the document is destroyed or otherwise rendered inaccessible (from the client machine). The resulting copyright management infrastructure is robust, secure, scaleable and easily managed.

In one embodiment of this invention as described above, the Internet client is a data processing system or a so-called "Web appliance" such as illustrated in Figures 6A-6D and 7. Figure 6A is a pictorial representation of the data processing system as a whole. Data processing system 100 in the depicted example provides, with minimal economic costs for hardware to the user, access to the Internet. Data processing system 100 includes a data processing unit 102. Data processing unit 102 is preferably sized to fit in typical entertainment centers and provides all required functionality, which is conventionally found in personal computers, to enable a user to "browse" the Internet. Additionally, data processing unit 102 may provide other common functions such as serving as an answering machine or receiving facsimile transmissions.

Data processing unit 102 is connected to television 104 for display of graphical information. Television 104 may be any suitable television, although color televisions with an S-Video input will provide better presentations of the graphical information. Data processing unit 102 may be connected to television 104 through a standard coaxial cable connection. A remote control unit 106 allows a user to interact with and control data processing unit 102. Remote control unit 106 allows a user to interact with and control data processing unit 102. Remote control unit 106 emits infrared (IR) signals, preferably modulated at a different frequency than the normal television, stereo, and VCR infrared remote

control frequencies in order to avoid interference. Remote control unit 106 provides the functionality of a pointing device (such as a mouse, glidepoint, trackball or the like) in conventional personal computers, including the ability to move a cursor on a display and select items.

5

Figure 6B is a pictorial representation of the front panel of data processing unit 102. The front panel includes an infrared window 108 for receiving signals from remote control unit 106 and for transmitting infrared signals. Data processing unit 102 may transmit infrared signals to be reflected off objects or surfaces, allowing data processing unit 102 to automatically control television 104 and other infrared remote controlled devices. Volume control 110 permits adjustment of the sound level emanating from a speaker within data processing unit 102 or from television 104. A plurality of light-emitting diode (LED) indicators 112 provide an indication to the user of when data processing unit 102 is on, whether the user has messages, whether the modem/phone line is in use, or whether data processing unit 102 requires service.

Figure 6C is a pictorial representation of the rear panel of data processing unit 102. A three wire (ground included) insulated power cord 114 passes through the rear panel. Standard telephone jacks 116 and 118 on the rear panel provide an input to a modem from the phone line and an output to a handset (not shown). The rear panel also provides a standard computer keyboard connection 120, mouse port 122, computer monitor port 124, printer port 126, and an additional serial port 128. These connections may be employed to allow data processing unit 102 to operate in the manner of a conventional personal computer. Game port 130 on the rear panel provides a connection for a joystick or other gaming control device (glove, etc.). Infrared extension jack 132 allows a cabled infrared LED to be utilized to transmit infrared signals. Microphone jack 134 allows an external microphone to be connected to data processing unit 102.

Video connection 136, a standard coaxial cable connector, connects to the video-in terminal of television 104 or a video cassette recorder (not shown). Left and right audio jacks 138 connect to the corresponding audio-in connectors on television 104 or to a stereo (not shown). If the user has S-Video input, then S-Video connection 140 may be used to connect to television 104 to provide a better picture than the composite signal. If television 104 has no video inputs, an external channel 3/4 modulator (not shown) may be connected in-line with the antenna connection.

Figure 6D is a pictorial representation of remote control unit 106. Similar to a standard telephone keypad, remote control unit 106 includes buttons 142 for Arabic numerals 0 through 9, the asterisk or "star" symbol

45

(*), and the pound sign (#). Remote control unit also includes "TV" button 144 for selectively viewing television broadcasts and "Web" button 146 for initiating "browsing" of the Internet. Pressing "Web" button 146 will cause data processing unit 102 to initiate modem dial-up of the user's Internet service provider and display the start-up screen for an Internet browser.

A pointing device 147, which is preferably a trackpoint or "button" pointing device, is included on remote control unit 106 and allows a user to manipulate a cursor on the display of television 104. "Go" and "Back" buttons 148 and 150, respectively, allow a user to select an option or return to a previous selection. "Help" button 151 causes context-sensitive help to be displayed or otherwise provided. "Menu" button 152 causes a context-sensitive menu of options to be displayed, and "Update" button 153 will update the options displayed based on the user's input, while home button 154 allows the user to return to a default display of options. "PgUp" and "PgDn" buttons 156 and 158 allows the user to change the context of the display in display-sized blocks rather than by scrolling. The message button 160 allows the user to retrieve messages.

In addition to, or in lieu of, remote control unit 106, an infrared keyboard (not shown) with an integral pointing device may be used to control data processing unit 102. The integral pointing device is preferably a trackpoint or button type of pointing device. A wired keyboard (also not shown) may also be used through keyboard connection 120, and a wired pointing device such as a mouse or trackball may be used through mouse port 122. When a user has one or more of the remote control unit 106, infrared keyboard, wired keyboard and/or wired pointing device operable, the active device locks out all others until a prescribed period of inactivity has passed.

Referring now to Figure 7, a block diagram for the major components of data processing unit 102 is portrayed. As with conventional personal computers, data processing unit 102 includes a motherboard 202 containing a processor 204 and memory 206 connected to system bus 280. Processor 205 is preferably at least a 486 class processor operating at or above 100 MHz. Memory 206 may include cache memory and/or video RAM. Processor 205, memory 206, and system bus 208 operate in the same manner as corresponding components in a conventional data processing system.

Video/TV converter 210, located on motherboard 202 and connected to system bus 208, generates computer video signals for computer monitors, a composite television signal, and an S-Video signal. The functionality of Video/TV converter 210 may be achieved through a Trident TVG9685 video

chip in conjunction with an Analog Devices AD722 converter chip. Video/TV converter 210 may require loading of special operating system device drivers.

5 Keyboard/remote control interface unit 212 on motherboard 202 receives keyboard codes through controller 214, regardless of whether a wired keyboard/pointing device or an infrared keyboard/remote control is being employed. Infrared remote control unit 106 transmits signals which are ultimately sent to the serial port as control signals generated by
10 conventional mouse or pointing device movements. Two buttons on remote control unit 106 are interpreted identically to the two buttons on a conventional mouse, while the remainder of the buttons transmit signals corresponding to keystrokes on an infrared keyboard. Thus, remote control unit 106 has a subset of the function provided by an infrared keyboard.

15 Connectors/indicators 216 on motherboard 202 provide some of the connections and indicators on data processing unit 102 described above. Other connections are associated with and found on other components. For example, telephone jacks 116 and 118 are located on modem 222. The power
20 indicator within connectors/indicators 216 is controlled by controller 214.

External to motherboard 202 in the depicted example are power supply 218, hard drive 220, modem 222 and speaker 224. Power supply 218 is a
25 conventional power supply except that it receives a control signal from controller 214 which effects shut down of all power to motherboard 202, hard drive 220 and modem 222. Power supply 218, in response to a signal from controller 214, is capable of powering down and restarting data processing unit 102.

30 Controller 214 is preferably one or more of the 805x family controllers. Controller 214 receives and processes input from infrared remote control 106, infrared keyboard, wired keyboard, or wired mouse. When one keyboard or pointing device is used, all others are locked out
35 (ignored) until none have been active for a prescribed period. Then the first keyboard or pointing device to generate activity locks out all others. Controller 214 also directly controls all LED indicators except that indicating modem use. As part of the failure recovery system, controller 214 specifies the boot sector selection during any power off-on
40 cycle.

Hard drive 220 contains operating system and applications software for data processing unit 102, which preferably includes IBM DOS 7.0, a product of International Business Machines Corporation in Armonk, New
45 York; an operating system 221 such as Windows 3.1 (or higher), a product

of Microsoft Corporation in Redmond, Washington; and a browser 223 such as Netscape Navigator (Version 1.0 or higher), a product of Netscape Communications Corporation in Mountain View, California. Hard drive 220 may also support an SMTP mechanism to provide electronic mail, an FTP
5 mechanism to facilitate file transfers from Internet FTP sites, and other Internet protocol mechanisms, all in a known manner. Hard drive 220 is not generally accessible to the user of the Web appliance.

Modem 222 may be any suitable modem used in conventional data
10 processing systems, but is preferably a 33.6 kbps modem supporting the V.42bis, V.34, V.17 Fax, MNP 1-5, and AT command sets. Modem 222 is connected to a physical communication link 227, which, in turn, in connected or connectable to the Internet (not shown).

15 Those skilled in the art will recognize that the components depicted in Figures 6A-6D and 7 and described above may be varied for specific applications or embodiments. Such variations in which the present invention may be implemented are considered to be within the spirit and scope of the present invention.

20 According to the invention, the client machine (typically the hard drive 220) also includes a proxy 225. Preferably, the proxy is implemented in software and includes a cache 227 associated therewith. The cache may be integral to the proxy or logically associated therewith.
25 The cache preferably has a size up to several hundred megabytes, which is substantially larger than the standard cache associated with a browser such as Netscape Navigator. The client machine also includes a protocol stack 229 (e.g., a TCP/IP protocol stack) and a sockets mechanism 231, which are used to support communications in a known manner. According to
30 the invention, the proxy 225 is advantageously located on the client along with the browser. Thus, the proxy is sometimes referred to as a "client side" proxy.

Preferably, the proxy starts up when the Web appliance is booted up.
35 Connectivity between the proxy and the browser is achieved using the sockets mechanism by configuring the browser to pass the HTTP requests to the proxy. To send an HTTP GET request, the browser creates a packet (including the URL and other information) and then opens a socket using the sockets mechanism. The packet is then sent to the IP address/port
40 number to service the HTTP request. Thus, when the browser issues an HTTP GET request, it binds to the socket and sends the request. The request is then intercepted and processed by the proxy instead of being sent directly over the network, all in the manner previously described.

Although in the preferred embodiment the client machine is a Web "appliance", this is not a requirement of the present invention. Thus, a client machine 10 may be a personal computer such as a desktop or notebook computer, e.g., an IBM® or IBM-compatible machine running under the OS/2® operating system, an IBM ThinkPad® machine, or some other Intel x86 or Pentium®-based computer running Windows 95 (or the like) operating system.

A representative server platform comprises an IBM RISC System/6000 computer (a reduced instruction set of so-called RISC-based workstation) running the AIX (Advanced Interactive Executive Version 4.1 and above) Operating System 21 and Server program(s) 22. The platform 20 also includes a graphical user interface (GUI) 23 for management and administration. It may also include an application programming interface (API) 24. HTTP GET requests are transferred from the client machine to the server platform, typically via the dial-up computer network, to obtain documents or objects formatted according to HTML or some other markup language. While the above platform is useful, any other suitable hardware/operating system/server software may be used.

One of the preferred implementations of the client side or server side mechanisms of the invention is as a set of instructions (program code) in a code module resident in the random access memory of the computer. Until required by the computer, the set of instructions may be stored in another computer memory, for example, in a hard disk drive, or in a removable memory such as an optical disk (for eventual use in a CD ROM) or floppy disk (for eventual use in a floppy disk drive), or downloaded via the Internet or other computer network.

In addition, although the various methods described are conveniently implemented in a general purpose computer selectively activated or reconfigured by software, one of ordinary skill in the art would also recognize that such methods may be carried out in hardware, in firmware, or in more specialized apparatus constructed to perform the required method steps.

As used herein, "Web client" should be broadly construed to mean any computer or component thereof directly or indirectly connected or connectable in any known or later-developed manner to a computer network, such as the Internet. The term "web server" should also be broadly construed to mean a computer, computer platform, an adjunct to a computer or platform, or any component thereof. Of course, a "client" should be broadly construed to mean one who requests or gets the file, and "server" is the entity which downloads the file. Moreover, although the present invention is described in the context of the Hypertext Markup Language

(HTML), those of ordinary skill in the art will appreciate that the invention is applicable to alternative markup languages including, without limitation, SGML (Standard Generalized Markup Language) and XML (Extended Markup Language).

5

In addition, the term "Web appliance" should be broadly construed to cover the display system illustrated in Figures 6A-6D, as well as any other machine in which a browser application is associated with some television class or other display monitor. Moreover, while the preferred embodiment is illustrated in the context of a dial-up network, this is not a limitation of the present invention. There may be other "bottleneck" resources in a direct connect network that could be managed indirectly by using this approach.

10

CLAIMS

1. A method for managing use of a digital file, comprising the steps of:
- 5
- establishing a secure link between a pair of devices, each of the devices being certified to operate under a given security protocol;
- establishing a usage scheme defining one or more conditions under
- 10 which the digital file may be transferred between the pair of devices; and
- transferring one or more copies of the digital file over the secure link between the pair of devices in accordance with the established usage scheme.
- 15
2. The method as described in Claim 1 wherein the pair of devices include a storage device and a rendering device.
3. The method as described in Claim 2 wherein the storage device and
- 20 the rendering device are located in a computer.
4. The method as described in Claim 2 wherein the storage device is located in a first computer and the rendering device is located in a second computer and the secure link is established over a computer network
- 25 connecting the first and second computers.
5. The method as described in Claim 4 wherein the second computer is a personal computer and the rendering device includes circuitry for establishing the secure link.
- 30
6. The method as described in Claim 4 wherein the second computer is a Web appliance and the rendering device includes software for establishing the secure link.
7. The method as described in Claim 2 wherein the rendering device is
- 35 selected from a group of rendering devices consisting essentially of a printer, a display, and a sound card.
8. The method as described in Claim 1 further including the step of
- 40 establishing an account representing a given monetary value.
9. The method as described in Claim 8 further including the step of allocating a given charge against the given monetary value when a copy of the digital file is transferred between the pair of devices.
- 45

10. The method as described in Claim 9 further including the step of associating the given charge with a content provider account to facilitate the payment of the given consideration to the provider of the digital file.

5

11. The method as described in Claim 1 wherein the usage scheme includes a given payment method.

12. A method for managing use of digital material in a computer network, comprising the steps of:

10

establishing an account for a given client computer including a representation of a given monetary value;

15 establishing an account for a given content provider including a representation of a given royalty value;

establishing a count of a number of permitted copies of a digital file;

20

in response to a given protocol, transferring a copy of the digital file from a source to a target associated with the given client computer;

25 adjusting the given monetary value in the account of the given client computer; and

adjusting the given royalty value in the account of the given content provider.

30 13. The method as described in Claim 12 wherein the given protocol includes the steps of:

determining whether a given client computer requesting transfer of the digital file is authorized to effect the transfer;

35

if the client is authorized to effect the transfer of the digital file, determining whether the count has a given value; and

40 if the count has the given value, transferring the digital file from the source to the target.

14. The method as described in Claim 13 wherein the given value is a non-zero value.

15. The method as described in Claim 13 wherein the given protocol further includes the step of adjusting the count after a copy of the digital file has been transferred.

5 16. The method as described in Claim 15 wherein the count is decremented.

10 17. The method as described in Claim 12 wherein the source and target are located in the given client computer connected to the computer network.

15 18. The method as described in Claim 17 wherein the source is a disk storage device and the target is a device selected from a group of rendering devices consisting essentially of a printer, a display, and a sound card.

19. The method as described in Claim 12 wherein the source is located on a first computer and the target is located on a second computer connected to the first computer via the computer network.

20 20. A method for managing use of digital material in a computer network including a Web client connectable to a Web server, comprising the steps of:

25 establishing a count of a number of permitted copies of a digital file located at a source device in the Web client;

30 in response to a given protocol, transferring one or more copies of the digital file from the source device to a set of one or more target rendering devices in the Web client; and

35 for each such transfer from the source device to one of the target rendering devices, logging an indication that the digital file has been transferred to facilitate payment of a given consideration to a provider of the digital file.

21. The method as described in Claim 20 wherein the Web client is a Web appliance and the source device is a secure disk storage.

40 22. The method as described in Claim 21 wherein each target rendering device is a device selected from a group of target rendering devices consisting essentially of a printer, a display, and a sound card.

45 23. The method as described in Claim 20 wherein the Web client is connected to the Web server via a non-secure connection.

24. The method as described in Claim 23 wherein the given protocol further includes the step of establishing a secure channel between the source device and a target rendering device prior to transferring the digital file.

5

25. The method as described in Claim 24 wherein the step of establishing a secure channel includes generating a secret key shared by the source device and the target rendering device.

10

26. The method as described in Claim 25 wherein the source device encrypts the digital file with the secret key as the source device transfers the digital file to the target rendering device, and wherein the target rendering device decrypts the digital file with the secret key upon receipt.

15

27. A computer program product in computer-readable media for use in a Web client having a source device and one or more target rendering devices, the computer program product comprising:

20

means for establishing a count of a number of permitted copies of a digital file located at the source device;

25

means, responsive to a given protocol, for transferring one or more copies of the digital file from the source device to the one or more target rendering devices;

30

means, responsive to each transfer, for logging an indication that the digital file has been transferred to facilitate payment of a given consideration to a provider of the digital file; and

means responsive to the logging means for adjusting the count.

35

28. The computer program product as described in Claim 27 further including means responsive to a given occurrence for transferring the indication to a central authority.

40

29. The computer program product as described in Claim 28 wherein the given occurrence is establishing a dialup connection between the Web client and an Internet Service Provider.

45

30. A computer system connected to a computer network and including a source device and one or more target rendering devices, comprising:

a processor;

an operating system;

an application for managing use of digital material, comprising:

5 means for establishing a count of a number of permitted copies of a digital file located at the source device;

10 means, responsive to a given protocol, for transferring one or more copies of the digital file from the source device to the one or more target rendering devices;

15 means, responsive to each transfer, for logging an indication that the digital file has been transferred to facilitate payment of a given consideration to a provider of the digital file; and

means responsive to the logging means for adjusting the count.

20 31. The computer system as described in Claim 30 wherein the application further includes means for restricting transfer of the digital file when the count reaches a given value.

32. A data processing system, comprising:

25 a remote control unit; and

a base unit connectable to a monitor for providing Internet access under the control of the remote control unit, the base unit comprising:

30 a processor having an operating system;

a browser application run by the operating system;

a secure disk storage in which a digital file is stored;

35 one or more target rendering devices; and

40 means for restricting a number of copies of the digital file that may be transferred between the secure disk storage and the one or more target rendering devices.

45 33. The data processing system as described in Claim 32 wherein the restricting means includes means responsive to a given occurrence for transmitting an indication of a number of copies of the digital file that were transferred between the secure disk storage and the one or more target rendering devices during a given time interval.

34. The data processing system as described in Claim 33 wherein the given occurrence is a dialup connection of the data processing system to an Internet Service Provider.

5 35. A management server for use in managing collection and allocation of royalties among content providers, the management server connected in a computer network to an access provider servicing a plurality of Web client appliances receiving dialup access to Web content, the management server comprising:

10 means for establishing an account for each of set of given content providers, each account including a representation of a given royalty value; and

15 means for adjusting the given royalty value in the account of the given content provider in response to receipt of an indication that a given digital file associated with the given content provider has been transferred from a source to a target rendering device in a given Web client appliance.

20 36. A copy management system, comprising:

a first device and a second device, each of which is certified to operate under a given security protocol;

25 means for establishing a secure link between the first and second devices; and

30 means responsive to establishment of the secure link for managing transfer of a permitted number of copies of a digital file between the first and second devices in accordance with copy control information restrictions associated with the digital file.

1 / 5

FIG. 1

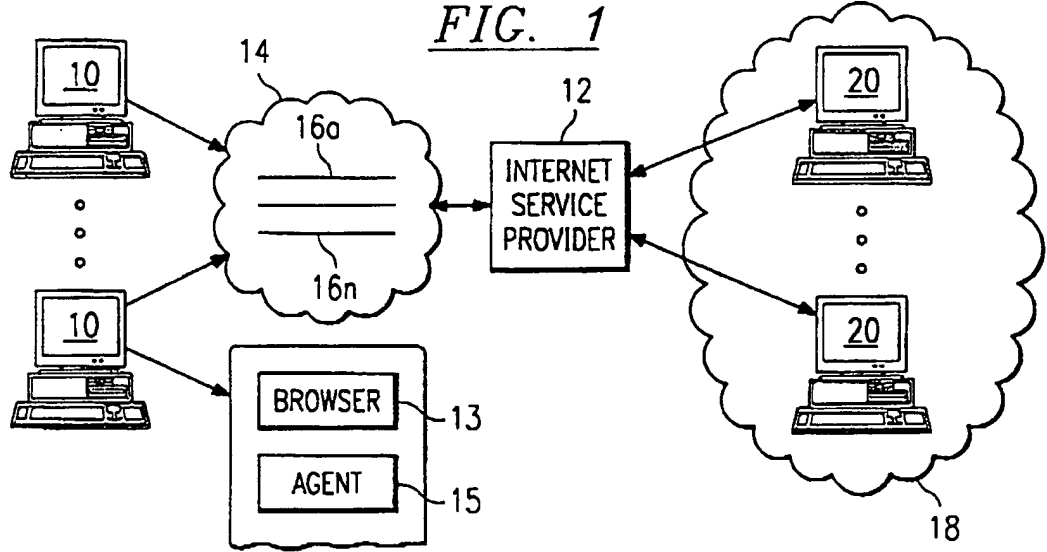


FIG. 2

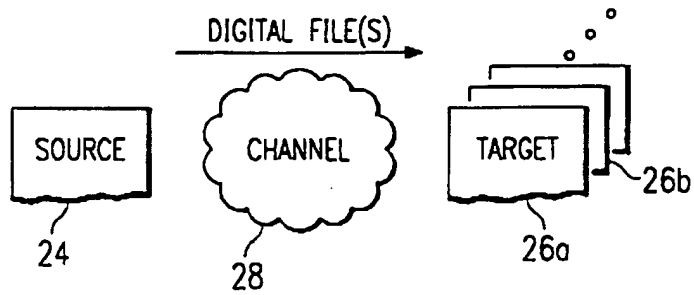


FIG. 3

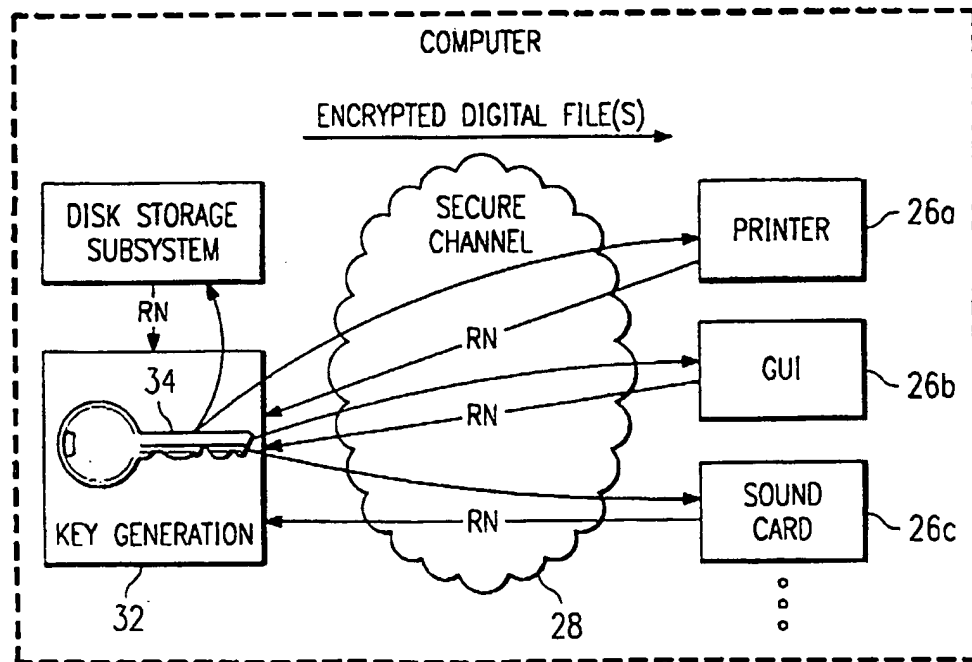
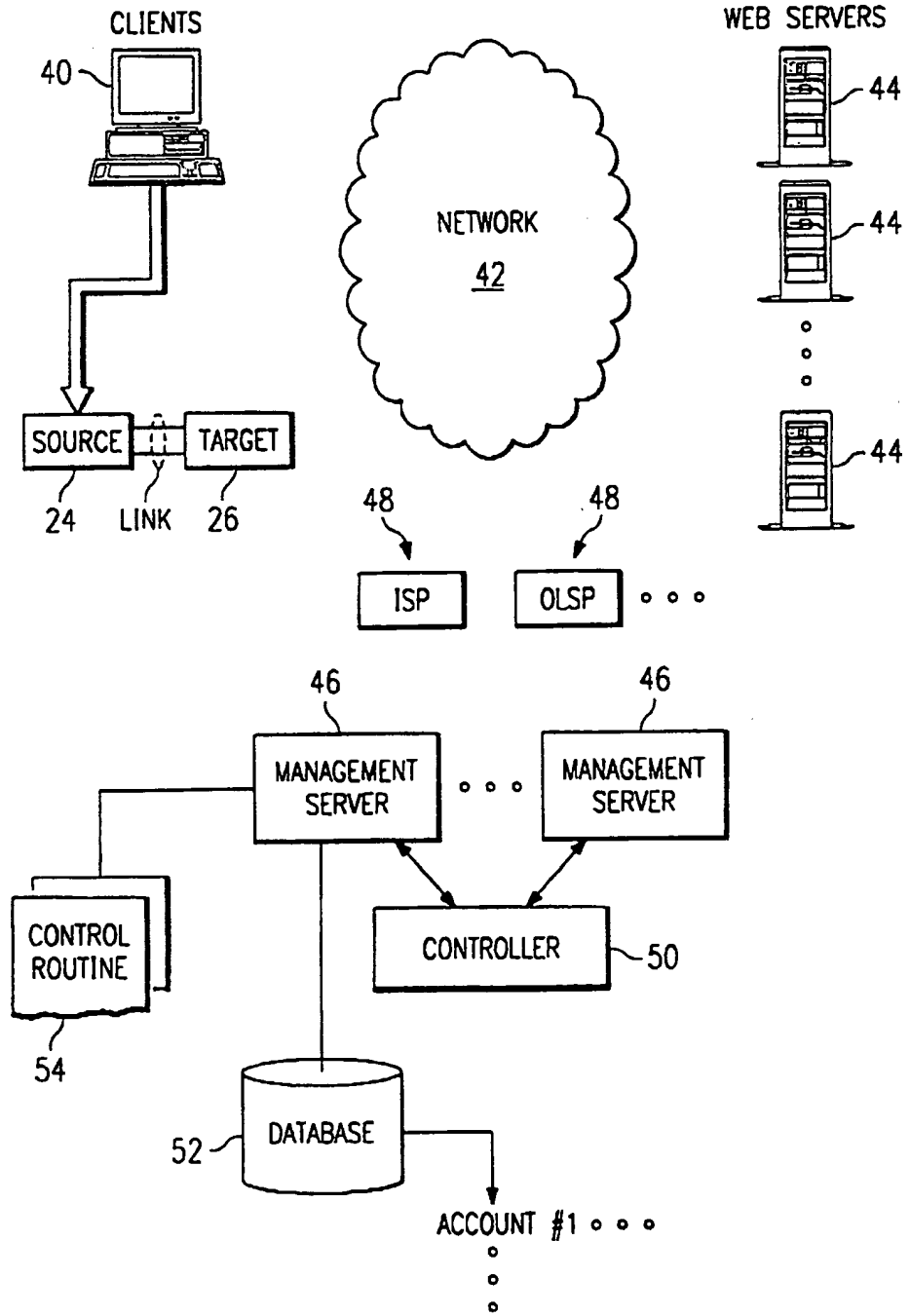


FIG. 4



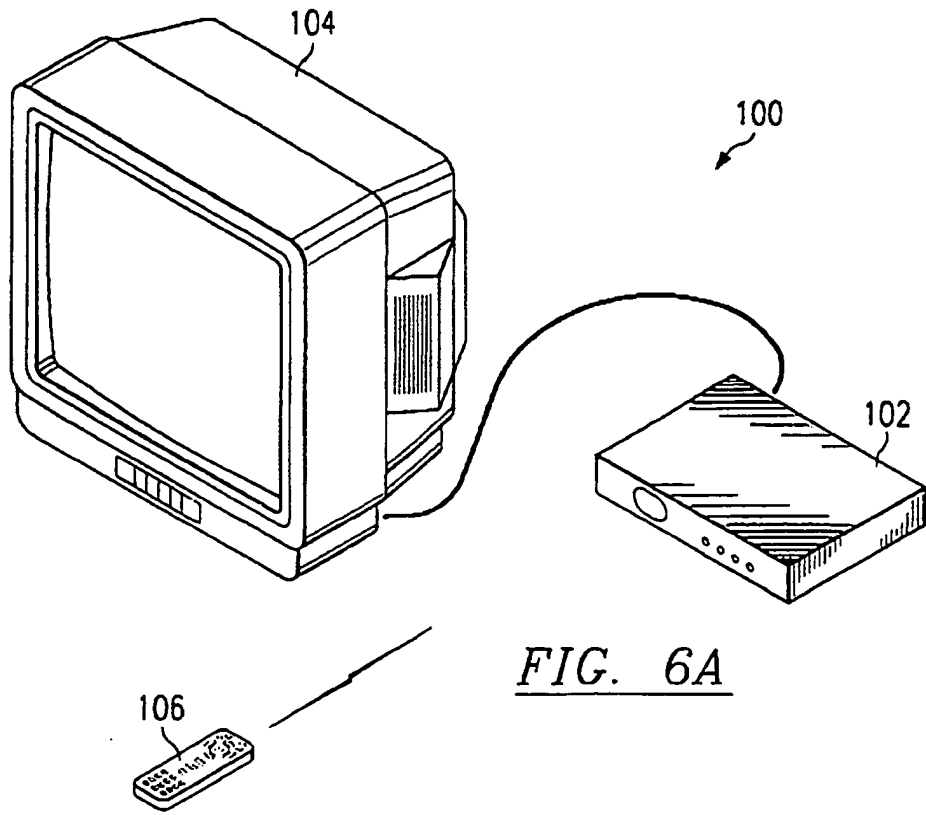
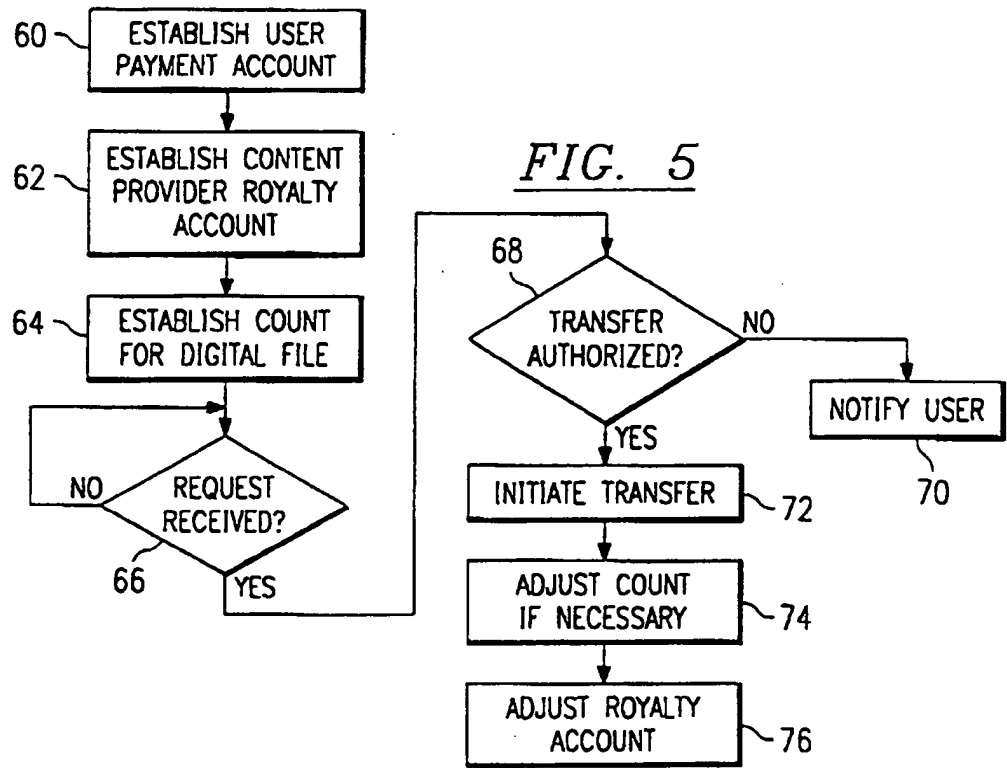


FIG. 6A

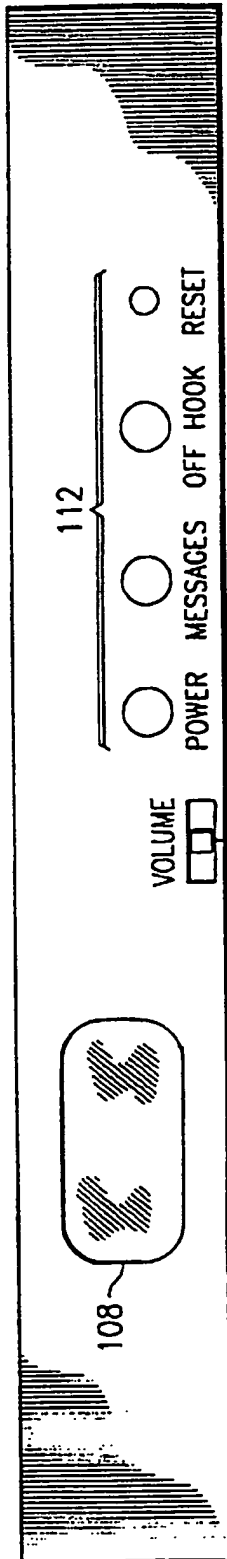


FIG. 6B

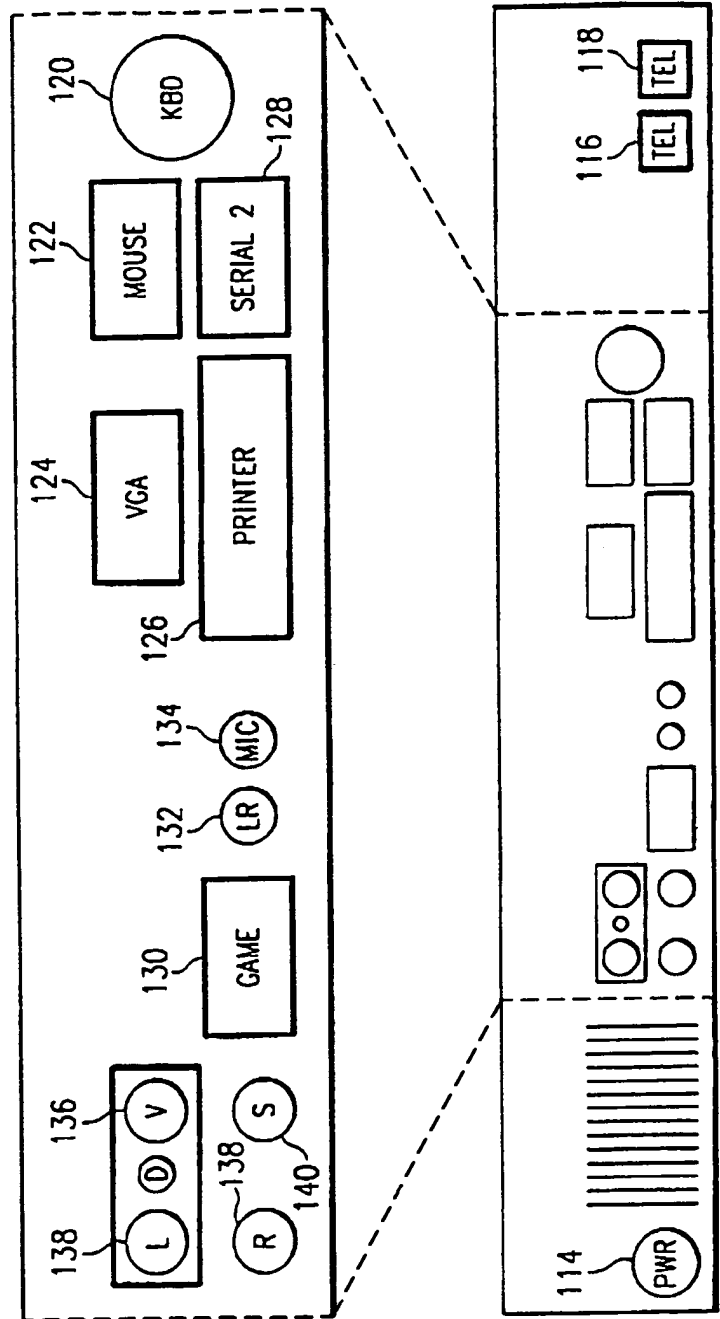


FIG. 6C

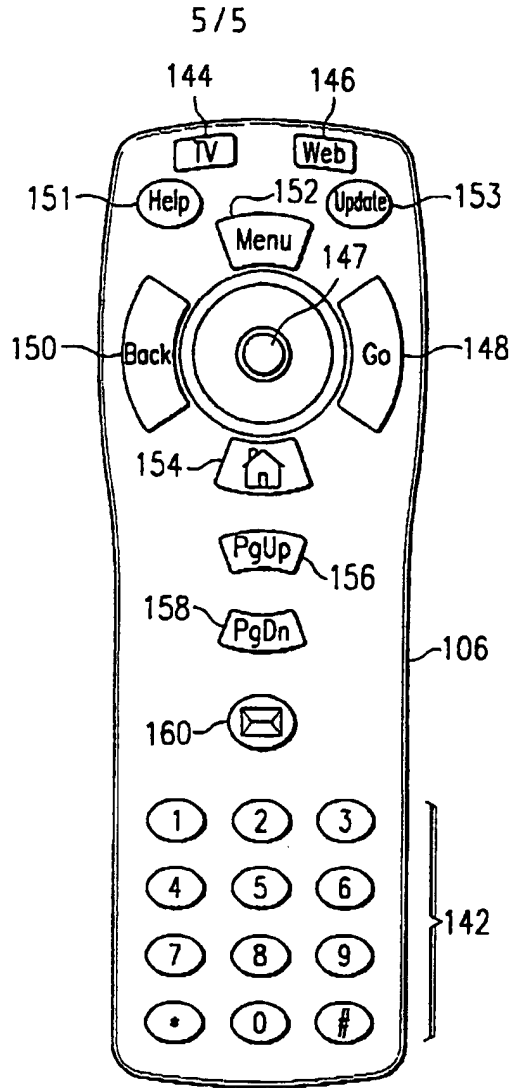
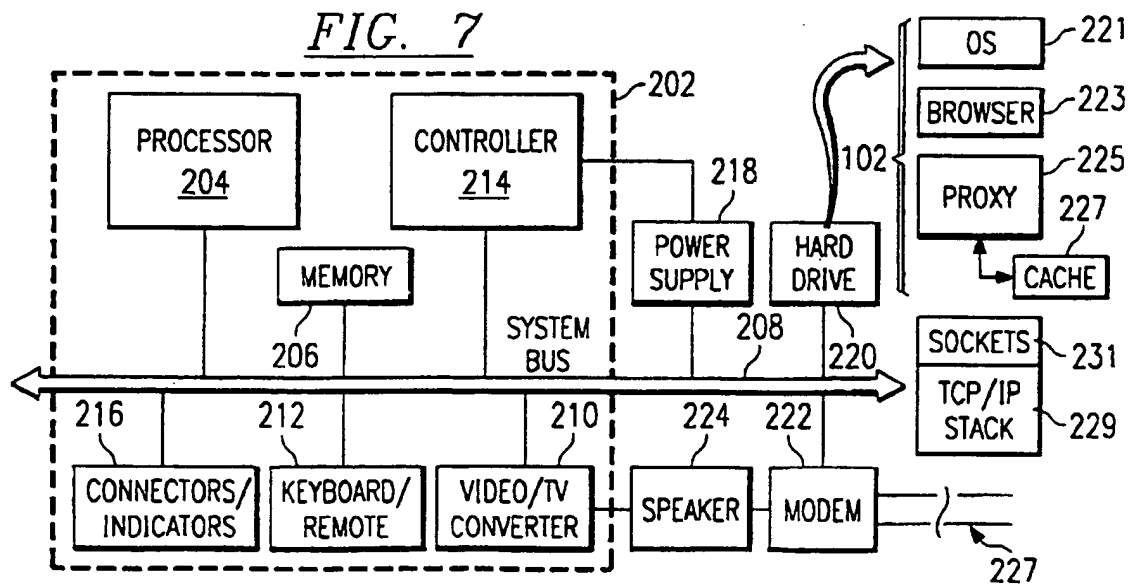


FIG. 6D



INTERNATIONAL SEARCH REPORT

Int'l. Application No
PCT/GB 98/03828

A. CLASSIFICATION OF SUBJECT MATTER IPC 6 G06F1/00				
According to International Patent Classification (IPC) or to both national classification and IPC				
B. FIELDS SEARCHED				
Minimum documentation searched (classification system followed by classification symbols) IPC 6 G06F				
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched				
Electronic data base consulted during the international search (name of data base and, where practical, search terms used)				
C. DOCUMENTS CONSIDERED TO BE RELEVANT				
Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.		
Y	US 5 532 920 A (HARTRICK THOMAS V. ET AL) 2 July 1996 see figures 1,2 see column 6, line 44 - column 7, line 15 see column 14, line 51 - column 16, line 25 ---	1-20, 23-25, 27-33, 35,36		
Y	EP 0 798 906 A (SUN MICROSYSTEMS INC) 1 October 1997 see figures 1-3 see column 4, line 21 - column 5, line 33 -----	1-20, 23-25, 27-33, 35,36		
<input type="checkbox"/> Further documents are listed in the continuation of box C.				
<input checked="" type="checkbox"/> Patent family members are listed in annex.				
* Special categories of cited documents :				
<table style="width: 100%; border: none;"> <tr> <td style="width: 50%; vertical-align: top;"> "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed </td> <td style="width: 50%; vertical-align: top;"> "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art. "&" document member of the same patent family </td> </tr> </table>			"A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed	"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art. "&" document member of the same patent family
"A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed	"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art. "&" document member of the same patent family			
Date of the actual completion of the international search <p style="text-align: center; font-size: large;">31 March 1999</p>		Date of mailing of the international search report <p style="text-align: center; font-size: large;">08/04/1999</p>		
Name and mailing address of the ISA European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016		Authorized officer <p style="text-align: center; font-size: large;">Weiss, P</p>		

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No

PCT/GB 98/03828

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5532920 A	02-07-1996	EP 0567800 A	03-11-1993
		JP 2659896 B	30-09-1997
		JP 6103286 A	15-04-1994
EP 0798906 A	01-10-1997	US 5761421 A	02-06-1998
		JP 10105529 A	24-04-1998



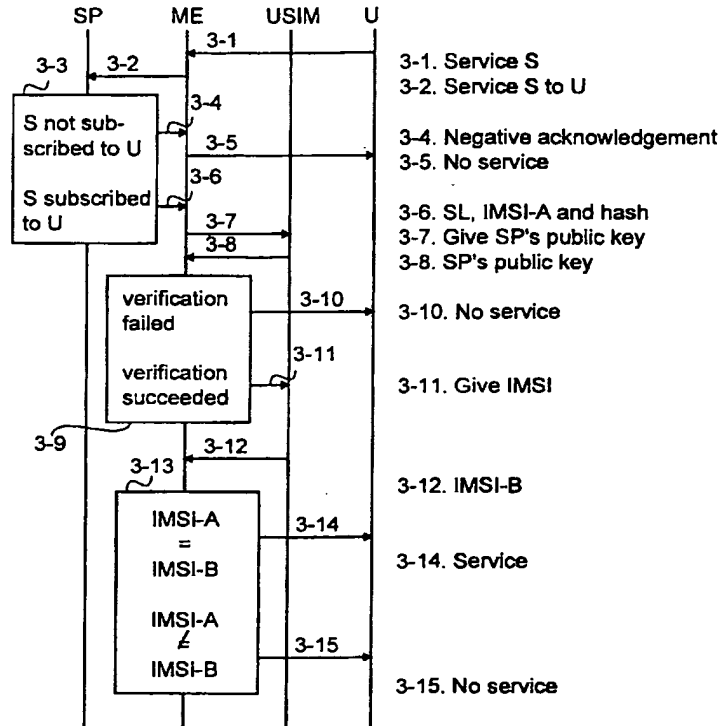
INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

<p>(51) International Patent Classification ⁶ : H04L 9/32</p>	<p>A2</p>	<p>(11) International Publication Number: WO 99/60750 (43) International Publication Date: 25 November 1999 (25.11.99)</p>
<p>(21) International Application Number: PCT/FI99/00432 (22) International Filing Date: 18 May 1999 (18.05.99) (30) Priority Data: 981132 20 May 1998 (20.05.98) FI (71) Applicant (for all designated States except US): NOKIA NETWORKS OY [FI/FI]; Keilalahdentie 4, FIN-02150 Espoo (FI). (72) Inventor; and (75) Inventor/Applicant (for US only): USKELA, Sami [FI/FI]; Puistokaari 8 B 12, FIN-00200 Helsinki (FI). (74) Agent: KOLSTER OY AB; Iso Roobertinkatu 23, P.O. Box 148, FIN-00121 Helsinki (FI).</p>	<p>(81) Designated States: AE, AL, AM, AT, AT (Utility model), AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, CZ (Utility model), DE, DE (Utility model), DK, DK (Utility model), EE, EE (Utility model), ES, FI, FI (Utility model), GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SK (Utility model), SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).</p> <p>Published <i>In English translation (filed in Finnish). Without international search report and to be republished upon receipt of that report.</i></p>	

(54) Title: PREVENTING UNAUTHORIZED USE OF SERVICE

(57) Abstract

A method, a system, a network element and an apparatus of a telecommunication system for preventing unauthorized use of a service. The method, in which a service request is received from a user and the service is generated by means of a service logic, is characterized in that to prevent unauthorized use of the service, authentication data is appended to the service logic (3-6), the user requesting the service is authenticated by means of the authentication data (3-9), and the service logic is executed (3-14) only if the authentication succeeds.



FOR THE PURPOSES OF INFORMATION ONLY

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

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece	ML	Mali	TR	Turkey
BG	Bulgaria	HU	Hungary	MN	Mongolia	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MR	Mauritania	UA	Ukraine
BR	Brazil	IL	Israel	MW	Malawi	UG	Uganda
BY	Belarus	IS	Iceland	MX	Mexico	US	United States of America
CA	Canada	IT	Italy	NE	Niger	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NL	Netherlands	VN	Viet Nam
CG	Congo	KE	Kenya	NO	Norway	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NZ	New Zealand	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	PL	Poland		
CM	Cameroon	KR	Republic of Korea	PT	Portugal		
CN	China	KZ	Kazakstan	RO	Romania		
CU	Cuba	LC	Saint Lucia	RU	Russian Federation		
CZ	Czech Republic	LI	Liechtenstein	SD	Sudan		
DE	Germany	LK	Sri Lanka	SE	Sweden		
DK	Denmark	LR	Liberia	SG	Singapore		
EE	Estonia						

PREVENTING UNAUTHORIZED USE OF SERVICE

BACKGROUND OF THE INVENTION

The invention relates to preventing unauthorized use of services and especially to preventing unauthorized use of the services in a mobile communication system.

Mobile communication systems were developed, because there was a need to allow people to move away from fixed telephone terminals without affecting their reachability. The services offered through mobile stations have developed along with the mobile communication systems. At the moment, various new forms of service are being planned for the current and particularly for the future third-generation mobile communication systems, such as Universal Mobile Telecommunication System (UMTS) and International Mobile Telecommunication 2000 (IMT-2000). UMTS is being standardized by ETSI (European Telecommunications Standards Institute), whereas ITU (International Telecommunications Union) is standardizing the IMT-2000 system. These future systems are very similar in basic features. The following will describe in greater detail the IMT-2000 system whose architecture is illustrated in Figure 1.

Like all mobile communication systems, IMT-2000 produces wireless data transmission services to mobile users. The system supports roaming, in other words, IMT-2000 users can be reached and they can make calls anywhere within the IMT-2000 system coverage area. IMT-2000 is expected to fulfil the need for a wide range of future services, such as virtual home environment (VHE). With the virtual home environment, an IMT-2000 user has access to the same services everywhere within the coverage area of the system. According to present knowledge, a flexible implementation of various services and especially supporting roaming requires the loading of certain service logics into the terminal of the user and/or the serving network. A serving network is the network through which the service provider offers his service to the end-user. A service logic is a program, partial program, script or applet related to the service. The service is generated by means of the service logic by executing at least the service logic and the functions defined in it. A service can also comprise several service logics.

A problem with the arrangement described above is that it does not in any way verify that the user really has the right to use the service. It is

especially easy to copy and make unauthorized use of services in which the service logic is loaded into the terminal and/or serving network.

BRIEF DESCRIPTION OF THE INVENTION

Thus, it is an object of the invention to develop a method and an
5 apparatus implementing the method so as to solve the above-mentioned
problem. The object of the invention is achieved by a method, a system, a
network element and an apparatus characterized by what is stated in the
independent claims. The term apparatus refers here to a network element of
the serving network, a terminal or any other corresponding service platform,
10 into which the service logic can be loaded. The preferred embodiments of the
invention are set forth in the dependent claims.

The invention is based on the idea of forming a service logic of two
parts: user authentication and the actual service logic. The data required for
user authentication is appended to the service logic, and the user is always
15 authenticated before executing the actual service logic. This provides the
advantage that an unauthorized use and copying of the service logic can be
prevented. Only the users, to whom the service is subscribed and who thus
have the right to use the service, can use it.

In a preferred embodiment of the invention, the service provider is
20 always verified before the service is executed. This improves considerably the
security of the user and a possible service platform into which the service logic
is loaded. This ensures that the service logic truly originates from the service
provider.

In a preferred embodiment of the invention, subscriber identification
25 used to individualise a user is used in user authentication. This provides the
advantage that subscriber authentication is simple, but reliable.

In a preferred embodiment of the invention, the service logic is
saved with its user and authentication data in the memory of the service
platform where it is loaded, and for a new user, only the authentication data of
30 the new user is loaded. This provides the advantage that the service logic
need not be loaded several times consecutively, which reduces the network
load.

BRIEF DESCRIPTION OF THE DRAWINGS

In the following, the invention will be described in more detail in connection with preferred embodiments and with reference to the attached drawings in which

5

Figure 1 illustrates the IMT-2000 architecture,

Figure 2 shows a flow chart of the service platform functions in a first preferred embodiment of the invention,

Figure 3 is a signalling diagram of a second preferred embodiment of the invention, and

10

Figure 4 shows the operation of a network element controlling a service of a service provider in a third preferred embodiment of the invention.

DETAILED DESCRIPTION OF THE INVENTION

The present invention can be applied to any data transmission system in which the user can receive the subscribed services in any terminal supporting service provision. In the following, the invention will be described using the IMT-2000 system as an example, without limiting the invention to this particular system, however. The specifications of mobile communication systems in general and those of the IMT-2000 and UMTS system in particular evolve rapidly. This evolution may require extra changes to the invention. Therefore, all terms and expressions should be interpreted as widely as possible and they are intended to describe and not to limit the invention. It is the function that is essential for the invention and not in which network element or apparatus it is executed.

Figure 1 shows the network architecture of the IMT-2000 system on a general level, because the system specifications are currently being defined. A more detailed network structure bears no essential significance with regard to the invention. Third-generation mobile communication systems separate a service provider SP and a network operator from each other. A service provider offers services to an end-user through a network SN of one or more network operators. This type of network SN is called a serving network. A service provider can offer services through a serving network SN of one or more network operators. In addition, a service provider may switch to another serving network during the service without the user noticing it. A service provider can also be a network operator. A serving network SN comprises an actual access network AN, one or more core networks CN, and an

interworking unit adapting interfaces IWU for each different type of core network. According to present knowledge, an access network comprises base stations BS and radio network controllers RNC controlling them (not shown in the figure). A core network can be a network according to the pan-European mobile communication system GSM (Global System for Mobile Communication). Connections to other networks ON are established through a core network CN.

In the example in Figure 1, a home location register with IMT-2000 enhancement HLRi and the service control node SCN have been located in the serving network SN. The enhanced home location register HLRi contains not only the home register data of the core network but also the subscriber and service data required by the IMT-2000 system. The service provider SP maintains this IMT-2000 data for the part of the services. The subscriber makes an order agreement with the service provider which then charges the subscriber for the use of the services. The service control node SCN is a service platform to which the service logic related to the service can be loaded and in which it can be executed. The service control node SCN can also take care of loading the service elsewhere in the network and forward service requests from the user to the service provider. In addition to this, the service control node SCN makes sure that the services of the home network are also available in the visited networks.

In third-generation mobile communication networks, subscriber and user are also separated. The subscriber grants the user access to the subscribed services by giving the user an identification card (IC Card), for instance a USIM card (User and Services and Identity Module). The user accesses the services with a mobile terminal MT which is connected through base stations BS to a serving network SN over a radio path. A mobile terminal MT comprises actual mobile equipment ME and a detachably connected identification card USIM, also called a subscriber identity module. It is a smart card which can be detached from the mobile terminal and with which the subscriber can use a card-controlled mobile terminal. The user is identified by the card in the mobile terminal and not by the terminal itself. According to present knowledge, the USIM card is a multi-functional card and supports mobile communication system applications and other applications, such as Java applications, healthcare applications, etc. The subscriber can subscribe to the services of several different service providers with the same subscriber

identity module USIM. The subscriber and the user can be one and the same person. The subscriber identity module USIM also contains an international mobile subscriber identity IMSI with which the subscriber can be explicitly identified and which can also be used to identify the user. The identifier of a
5 mobile subscriber is called subscriber identity.

The terminal selection of third-generation systems will probably be extremely versatile. The terminal can be a simplified terminal for speech only or it can be a terminal providing diverse services, which acts as a service platform and supports the loading and execution of various service logics.

10 A mobile communication system implementing the functionality of the present invention comprises not only means required for generating and loading services according to prior art, but also means for appending authentication data to a service logic and means for authenticating the user prior to executing the service logic. Here, appending also refers to embedding
15 data into the service logic. In addition, the system can comprise means for verifying the service provider and means for saving the service logic with its supplementary data into the memory and means for receiving plain authentication data. The means for appending authentication data and the possible means for appending verification data are preferably located together
20 with the means required for loading the service logic of the service provider. The other means are preferably located on the service platform, for instance in the terminal or the service control point of the network operator. The means or a part of them can also be located elsewhere, for instance in the network node of the subscriber network or in the serving support node of the core network.

25 Figure 2 shows a flow chart of the operation according to the first preferred embodiment of the invention on a service platform which can be actual mobile equipment ME or a service control node SCN, for instance. In the first preferred embodiment of the invention, an encryption technique, known per se, based on public keys is utilized in a novel and inventive
30 manner. One such encryption technique is RSA (Rivest Shamir Adleman public-key cryptographic algorithm) which can be used for both encryption and digital signature. In the first preferred embodiment of the invention, at least the secret key of the subscriber and the public key of the service provider are saved in the subscriber identity module USIM. If the subscriber has several
35 key pairs, the secret key of the pair, whose public key has been entered in the subscriber data of the user, is saved. Correspondingly, a service provider can

have several key pairs of which one, for instance, is saved in the subscriber identity module and information on the pair, whose key is saved in the identity module, is entered in the subscriber data. This ensures that the secret and public key of the same pair is used. In the first preferred embodiment of the invention, the service provider is verified by a digital signature. It is generated in the first embodiment of the invention by calculating a one-way hash (one-way hash function) from the service logic, which is then encrypted. This embodiment provides the unexpected advantage that in connection with the verification of the service provider, the fact whether the service logic has been changed, is also checked. If the service logic has been changed, the hash calculated from it also changes and the service provider verification does not succeed anymore.

With reference to Figure 2, a service request concerning a service S1 is received from a user U1 in step 200. In step 201, a check is made to see whether the service logic SL1 related to the service S1 is in the memory. If it is not, the service request is forwarded to the service provider in step 202. The service provider finds the actual service logic SL1 related to the service S1 and appends to the service logic authentication data A1 required for user authentication, which in the first preferred embodiment is the public key of the subscriber. After this, the service provider calculates the hash from the actual service logic and the authentication data and appends it as verification data V1 to the service logic and encrypts the thus created file with its own secret key. The file contains the verification data V1, the authentication file A1 and the actual service logic SL1. Alternatively, the service provider could encrypt the hash with its secret key, append the encrypted hash as the verification data V1 to the file and then encrypt the file with the public key of the user. After this, in step 203, the file, i.e. the actual service logic SL1 related to the service S1, the authentication data A1 appended to it for the user U1, and the verification data V1 calculated from them, is loaded onto the service platform. In step 204, the service logic SL1 is saved and in it, the authentication data A1 and the verification data V1 related to the user U1, and, of course, information on the user U1 to which the authentication data A1 and the verification data V1 are related. The data is stored in encrypted format in the memory. Then, in the first preferred embodiment, the public key of the service provider is requested from the subscriber identity module USIM in the terminal of the user in step 205 and received in step 206, after which the encryption of the service

logic SL1, the authentication data A1 and the verification data V1 is decrypted in step 207 using the received key. In embodiments in which the service provider only has one key pair, the information on the public key of the provider can already be on the service platform and need not be separately
5 requested. When the encryptions have been decrypted, the service provider is verified by calculating a hash from the service logic and the authentication data in step 208 and by comparing the thus calculated hash with the verification data V1 in step 209. If they are the same, the verification of the service provider succeeds, and after this, a challenge, i.e. a character string, is
10 selected in step 210. How the challenge is selected bears no significance with regard to the invention. A simple and safe solution is to use a random number generator, whereby the challenge is a random number. The selected challenge is encrypted in step 211 with the public key of the subscriber, i.e. the authentication data A1. After this, in step 212, the encrypted challenge is sent
15 to the subscriber identity module USIM in the terminal of the user U1, which decrypts the encrypted challenge into plain text with the secret key of the subscriber and sends the plain text back to the service platform. In step 213, the service platform receives the plain text and in step 214, it compares the original challenge with the plain text. If the character strings are the same,
20 user authentication succeeds and the actual service logic SL1 can be executed in step 215.

If it is detected in step 209 that the calculated hash is not the same as the verification data, the service provider verification fails or the service logic has been changed. In both cases, executing the service logic would be a
25 security risk and, therefore, it is not executed, and in step 217, all data saved for the service S1, i.e. the service logic SL1 and all appended authentication and verification data with user data, is deleted from the memory.

If it is detected in step 214 that the challenge is not the same as the plain text, authentication does not succeed and the service logic is not
30 executed, and in step 216, the authentication data A1, verification data V1 and information on the user U1 appended to the service logic SL1 of the service S1, is deleted from the memory. This way, the actual service logic SL1 need not be loaded next time, only the authentication data and verification data.

If it is detected in step 201 that the service logic SL1 related to the
35 service S1 is in the memory, a check is made in step 218 to see if authentication and verification data for the user U1 is appended to it. If this

data, too, is in the memory, operation continues from step 205 where the public key of the service provider is requested from the subscriber identity module USIM. From step 205 onward, operation continues as described above. This way, network resources are saved, because the once loaded data need not be loaded again.

If it is detected in step 218 that no authentication and verification data for the user U1 is appended to the service logic SL1 in the memory, in step 219, the authentication and verification data for the service S1 is requested from the service provider for the user U1. The authentication data A1 and the verification data V1 are received in step 220, after which they and information on the user U1 are appended to the service logic SL1 in step 221. After this, operation continues from step 205 where the public key of the service provider is requested from the subscriber identity module USIM. From step 205 onward, operation continues as described above.

The service platform can be actual mobile equipment ME or a network element of the serving network, such as the service control node SCN. The memory where the data and service logics are saved can also be a cache memory. In embodiments in which the service logic is saved in the memory, the service platform can comprise means for deleting the service logic from the memory for predefined reasons, for instance after a certain time period.

In embodiment in which the service logic is not saved in the memory, steps 200, 202, 203 and 205 to 215 are executed. The data deletion described in steps 216 and 217 is not done, but the actual service logic is left unexecuted.

The steps described above in Figure 2 are not in absolute chronological order and some of the steps can be executed simultaneously or deviating from the given order. Other functions can also be executed between the steps. Some of the steps, such as the service provider verification, can also be left out. The essential thing is to authenticate the user before the actual loaded service logic is executed.

Figure 3 shows signalling according to a second preferred embodiment of the invention. In the second preferred embodiment, the subscriber identification IMSI is used as the authentication data. It is also assumed that the service logic is loaded into the actual mobile equipment ME and not saved in its memory.

With reference to Figure 3, user U sends information in message 3-1 to mobile equipment ME through the user interface requesting service S. The mobile equipment ME sends the service request through the serving network to service provider SP in message 3-2. The service request contains information on the required service S and the user U requesting the service. In step 3-3, the service provider checks if the service S is subscribed to the user U. If the service S is not subscribed to the user, the service provider sends through the serving network a negative acknowledgement to the service request in message 3-4 to the mobile equipment ME which forwards the information in message 3-5 through the user interface to the user U.

If the service S is subscribed to the user, the service provider retrieves the subscriber identification IMSI-A of the user U and the service logic SL related to the service S and calculates a hash from them. After this, the service provider encrypts the service logic and the related data (IMSI-A, hash) with the secret key of the service provider. In message 3-6, the service provider sends the service logic SL, the identification IMSI-A and the hash to the mobile equipment ME. In the second preferred embodiment, after receiving the message 3-6, the mobile equipment ME requests the public key of the service provider from the subscriber identity module USIM in message 3-7. The subscriber identity module USIM sends it to the mobile equipment in message 3-8, after which the mobile equipment ME verifies the service provider in step 3-9. The mobile equipment decrypts the encryption of the service logic SL, the subscriber identification A1 and the hash with the received public key and calculates a hash from the combination of the service logic and the subscriber identification IMSI-A. If the calculated hash is not the same as that received in message 3-6, the verification fails. In such a case, the service logic is not executed and the mobile equipment ME sends information on the verification failure through the user interface to the user U in message 3-10, saying, for instance, that the service is not available.

If the hash calculated in step 3-9 and the received hash are the same, the verification succeeds and the mobile equipment ME requests the subscriber identification IMSI of the user U from the subscriber identity module USIM in message 3-11. The subscriber identity module USIM retrieves the subscriber identification IMSI-B from its memory and sends it to the mobile equipment ME in message 3-12. In step 3-13, the mobile equipment authenticates the user by checking if the IMSI-A received from the service

provider is the same as the IMSI-B received from the identity module. If the user passes the authentication in step 3-9 (i.e. IMSI-A is the same as IMSI-B), the mobile equipment ME executes the actual service logic SL and provides the service through the user interface to the user U in messages 3-14. If the values of the subscriber identifications IMSI differ from each other, authentication fails. In such a case, the mobile equipment does not execute the actual service logic, but informs the user U through the user interface in message 3-10 that the authentication failed, saying, for instance, that the service is not available.

10 The signalling messages described above in connection with Figure 3 are for reference only and can contain several separate messages to forward the same information. In addition, the messages can also contain other information. The messages can also be freely combined. In embodiment in which the service provider is not verified, the messages 3-7, 3-8 and 3-10 related to verification and step 3-9 are left out. Depending on the service providers, core network and mobile equipment, other network elements, to which various functionalities have been distributed, can take part in the data transmission and signalling.

20 Figure 4 shows a flow chart of a network element controlling a service of a service provider in a third preferred embodiment of the invention. In the third preferred embodiment, authentication and verification are only performed when a service logic is loaded into a visited (visiting) network or mobile equipment. The visited network is a network whose network element, into which the service is loaded, is a network element belonging to a provider other than the service provider. The third preferred embodiment utilizes both public key encryption and symmetrical encryption, such as DES (Data Encryption Standard). The latter encryption technique is used when the service logic is loaded into the mobile equipment. A common key is saved for it in both the subscriber identity module and the subscriber data of the user. In addition, the public key of the service provider is saved in the subscriber identity module for the service logics to be loaded into visited networks. Only encryption of the service logic with the secret key of the service provider prior to sending the service logic to the serving network or encryption with the common key prior to loading it in the mobile equipment is used as signature.

35 With reference to Figure 4, in step 400, a service request concerning a service S2 is received from a user U2. In step 401, a check is

made to see if the user U2 subscribes to the service S2. If the user subscribes to the service S2, a check is made in step 402 to see if the service logic SL2 related to the service U2 requires loading into the mobile equipment ME of the user. If the service logic SL2 is loaded into the mobile equipment of the user, in step 403, a common key is retrieved from the subscriber data of the user U2 for encrypting the service logic SL2 in step 404. This common key is used both as the authentication data of the user and the verification data of the service provider. Nobody else should have any information on the common key in this case. The authentication and verification are performed in connection with the decryption of the service logic. The encrypted service logic SL2 is loaded into the mobile equipment ME in step 405. The user is authenticated and the service provider is verified in the mobile equipment, for instance by sending the encrypted service logic to the subscriber identity module USIM in the mobile equipment, which decrypts the service logic using the common key in its memory and sends the plain-text service logic to the mobile equipment. When the service logic has been executed, information concerning this is received in step 406, and the subscriber is charged for the use of the service in step 407.

If it is detected in step 402 that the service logic SL2 will not be loaded into the mobile equipment, a check is made in step 408 to see if the user U2 is in the home network area. If yes, the service logic SL2 is executed in step 409, after which operation continues from step 407 in which the user is charged for the use of the service.

If it is detected in step 408 that the user is not in the home network area, in the third preferred embodiment of the invention, the service logic SL2 must be loaded into the visited network. To do this, in step 410, the public key of the user U2 is retrieved from the subscriber data for appending it as authentication data to the service logic. In step 411, the public key of the user is appended to the service logic SL2, and they are encrypted using the secret key of the service provider in step 412. The encryption also acts as the verification data. If the service provider has several key pairs of public and secret keys, the secret key of the pair whose public key has been saved in the identity module of the user is used. The encrypted service logic, to which the authentication data is appended, is loaded into the visiting network in step 413. The network element of the visiting network verifies the service provider by decrypting the service logic using the public key of the service provider and

authenticates the user, for instance in the manner described in connection with Figure 2, after which the service logic is executed. When the service logic has been executed, information concerning this is received in step 406, and the subscriber is charged for the use of the service in step 407.

5 If it is detected in step 411 that the requested service is not subscribed to the user, information is transmitted in step 414 that the service is not available to the user.

 Above, in connection with Figure 4, it was assumed that the authentication and verification succeeded. If this is not the case, the service logic is not executed and the subscriber not invoiced. The steps described above in connection with Figure 4 are not in absolute chronological order and some of the steps can be executed simultaneously or deviating from the given order. Other functions can also be executed between the steps. Some of the steps can also be left out. The essential thing is that the authentication data is
10 in some way appended to the service logic being loaded.
15

 In the above embodiments, the actual service logic has been changed to ensure that the authentication and verification are done. This has been done by adding to the service logic a part taking care of the authentication and verification, which is always executed before the service logic. In some embodiments, the service logic can only be changed to ensure the authentication. In some embodiments, there is no need to change the service logic, and the authentication data and the possible verification data are appended to the service logic as separate data, and the service platform makes sure that the authentication and the possible verification are done. In
20 these embodiments, pre-encrypted service logics can be used, which reduces the load of the network element, because encryption is done only once.
25

 It has been assumed above in connection with Figures 2, 3 and 4 that the service provider appends the authentication data to the service logic before the encryption. The authentication data can also be appended to a pre-encrypted service logic. In such a case, the serving network or mobile equipment can also be adapted to append the authentication data to the service logic, for instance by means of the user data provided in the service request. It has been presented above that the user is authenticated only after the verification. However, the order bears no significance with regard to the
30 invention. The user can be authenticated before the service provider is verified in embodiment in which the service provider is also verified. The data and/or
35

service logic also need not be encrypted unless the encryption is used for authentication and/or verification. Other alternatives for authentication, verification and possible encryption than those described above in connection with the preferred embodiments can also be used. The preferred embodiments
5 can also be combined. The essential thing is that the user is authenticated before executing the service logic at least when the service logic is loaded into the mobile equipment or visiting network. In embodiment in which the service logic is loaded into the mobile equipment, the encryption of the service logic with the public key of the subscriber can also be used as the authentication
10 data. The subscriber is authenticated when the identity module USIM decrypts the encryption with the secret key of the subscriber. For security's sake, it is advantageous that USIM never sends even to the mobile equipment the secret key saved in it, and the decryption with the secret key is always performed in USIM. Other data for authentication and possible verification than used in the
15 above examples can also be used. The requirements for the authentication data and possible verification data are adequate individualization, reliability and non-repudiation. Adequate individualization means that the data specifies the user at least by subscriber.

No hardware changes are required in the structure of the serving
20 network. It comprises processors and memory that can be utilized in functions of the invention. All changes required for implementing the invention can instead be made as additional or updated software routines in the network elements into which the service logic is loaded. An example of such a network element is the service control node. Extra memory is also needed in the
25 network element saving the loaded service logic with its supplementary data.

The structure of the service provider also requires no hardware changes. The service provider comprises processors and memory that can be utilized in functions of the invention. All changes required for implementing the invention can be made as additional or updated software routines to achieve
30 the functionality of the invention. Extra memory may be needed depending on the embodiment of the invention. It is, however, limited to a small amount sufficient for saving the extra authentication data and the possible verification data.

The structure of the mobile equipment requires no hardware
35 changes. It comprises processors and memory that can be utilized in functions of the invention. All changes required for implementing the invention can

instead be made as additional or updated software routines in the mobile equipment which is adapted to function as a service platform. If the service logic is saved in the mobile equipment, extra memory is also needed.

5 In the subscriber identity module USIM, the extra memory possibly needed for implementing the invention is limited to a small amount sufficient for saving the extra authentication data, the possible verification data and the decryption algorithms possibly needed.

10 It will be understood that the above description and the figures related to it are only presented for the purpose of illustrating the present invention. The various modifications and variations of the invention will be obvious to those skilled in the art without departing from the scope or spirit of the invention disclosed in the attached claims.

CLAIMS

1. A method for preventing unauthorized use of a service in a mobile communication system, in which method
a service request is received from a user of the service, and
5 the service is generated by means of a service logic,
characterized in that in the method
authentication data is appended to the service logic (3-6),
the user requesting the service is authenticated by means of the
10 authentication data (3-9), and
the service logic is executed only if the authentication succeeds (3-14).
2. A method as claimed in claim 1, **characterized** in that
verification data of the service provider is also appended to the
service logic,
15 the service provider is verified in connection with user
authentication (3-13), and
the service logic is executed only if the verification also succeeds.
3. A method as claimed in claim 2, **characterized** in that
a first hash calculated from the service logic is used as verification
20 data of the service logic,
the service logic is loaded onto a service platform where it is
executed to generate the service,
the service provider is verified on the service platform by calculating
a second hash from the service logic, and
25 if the first and the second hash are the same, the verification
succeeds,
if the first and the second hash differ, the verification fails.
4. A method as claimed in claim 2, **characterized** in that
the signature of the service provider is used as the verification data,
30 the service logic is signed by encrypting it with the secret key of the
service provider, and
the service provider is verified by decrypting the encryption of the
service logic with the public key of the service provider.
5. A method as claimed in any one of the above claims,
35 **characterized** in that

the secret key of the subscriber is saved in the subscriber identity module (USIM) of the user of the service,

the public key of the subscriber is used as the authentication data,
a challenge encrypted with the public key of the subscriber is sent
5 to the subscriber identity module located in the mobile equipment of the user requesting the service (207),

the challenge is decrypted into plain text with the secret key of the subscriber in the identity module,

the plain text is received from the identity module (208),
10 a check is made to see if the unencrypted challenge and the plain text correspond to each other (209), and

if they correspond, the authentication succeeds, and

if they do not correspond, the authentication fails.

6. A method as claimed in claims 1, 2, 3 or 4,
15 **characterized** in that

individual identity of the subscriber is used as the authentication data,

a service request is received from the user (3-1),
the individual subscriber identity related to the user is requested (3-
20 7),

the requested identity is received (3-8),

a check is made to see if the authentication data and the requested identity correspond to each other (3-9), and

if they correspond, the authentication succeeds, and

25 if they do not correspond, the authentication fails.

7. A method as claimed in any one of the above claims,
characterized in that

the service logic is loaded onto the service platform where it is executed to generate the service, and

30 the authentication data is appended to the service logic in connection with the loading.

8. A method as claimed in claim 7, **characterized** in that

the service logic, the authentication data appended to it, and the data indicating the user are saved on the service platform in connection with
35 the loading (204),

a service request is received from the user,

a check is made to see if the service logic related to the requested service is saved on the service platform (201), and

if not, the service logic is loaded (203),

if yes,

5 - a check is made to see if authentication data has been saved for the user requesting the service (217), and

- if yes, the user is authenticated,

- if not,

-- authentication data is requested for the user (218),

10 -- the authentication data and the data indicating the user are saved in the service logic (220), and

-- the user is authenticated.

9. A telecommunication system comprising

15 a first part (SP) to produce the service for the user by means of a service logic, and

 a second part to provide the service (SN, MT) to the user of the service,

 in which system the first part (SP) is adapted to identify the user requesting the service and to check, if the service is subscribed to the user, and if the service is subscribed to the user, to generate the service by loading the service logic into the second part (SN, MT) which is adapted to provide the service by executing the loaded service logic,

characterized in that

25 the first part (SP) is adapted to append authentication data into the service logic being loaded for user authentication, and

 the second part (SN, MT) is adapted to authenticate the user and to execute the service logic only in response to a successful authentication.

10. A system as claimed in claim 9, **characterized** in that

30 the first part (SP) is adapted to sign the service logic by encrypting it with an encryption key agreed with the second part, and

 the second part (SN, MT) is adapted to verify the first part by decrypting the encryption of the service logic with a key corresponding to the agreed key and to execute the service logic only if the verification also succeeds.

35 11. A system as claimed in claim 9 or 10, **characterized** in that

the telecommunication system is a mobile communication system (IMT-2000) comprising at least one service provider and serving network, the first part is the service provider (SP), and the second part is the serving network (SN) comprising at least one network element (SCN).

12. A system as claimed in claim 9 or 10, **characterized** in that

the telecommunication system is a mobile communication system (IMT-2000) comprising at least one service provider (SP) and mobile terminal (MT) which is connected to the service provider through a serving network (SN) and which mobile terminal (MT) comprises in addition to actual mobile equipment (ME) a subscriber identity module (USIM) which is detachably connected to the mobile equipment,

the first part is the service provider (SP), and

the second part is the actual mobile equipment (ME).

13. A network element (SP) generating a telecommunication system service for a user, which produces the service by means of a service logic and which comprises means for identifying the user requesting the service and for checking if the service is subscribed to the user and for loading the service logic into the telecommunication system if the service is subscribed to the user,

characterized in that the network element (SP) comprises means for appending the authentication data to the service logic being loaded so that the user of the service is authenticated before the service logic is executed.

14. A network element (SP) as claimed in claim 13, **characterized** in that it comprises means for signing the service logic before it is loaded into the network.

15. A network element (SP) as claimed in claim 13 or 14, **characterized** in that it comprises a processor arranged to execute software routines, and said means have been implemented as software routines.

16. An apparatus of a telecommunication system, which apparatus comprises service logic executing means for providing a service from a service provider of a telecommunication system to a user of the service,

characterized in that the apparatus (SCN, ME) comprises

separation means for separating the authentication data of a user from a loaded service logic,

authentication means responsive to the separation means for user authentication, and

5 service logic execution means are adapted to be responsive to the authentication means.

17. An apparatus (SCN, ME) as claimed in claim 16, **characterized** in that

10 it comprises verification means for service provider verification by means of verification data in the loaded service logic, and

the service logic verification means are adapted to be responsive to the authentication means.

18. An apparatus (SCN, ME) as claimed in claim 16 or 17, **characterized** in that it comprises a processor arranged to execute
15 software routines, and said means are implemented as software routines.

19. An apparatus as claimed in claim 16, 17 or 18, **characterized** in that it is a network element (SCN) of a mobile communication system, which is adapted to function as a service platform.

20. An apparatus as claimed in claim 16, 17 or 18, **characterized** in that it is the mobile equipment (ME) in a mobile communication system.

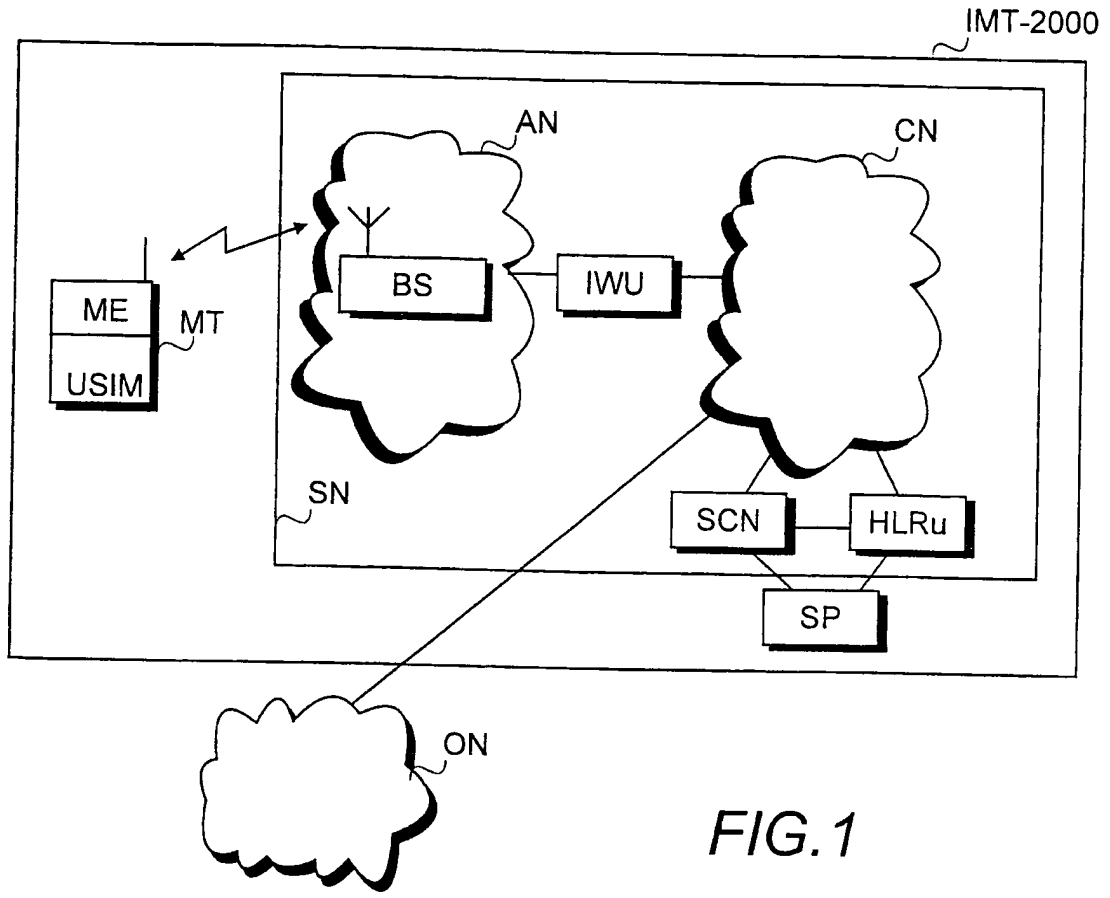


FIG. 1

2/4

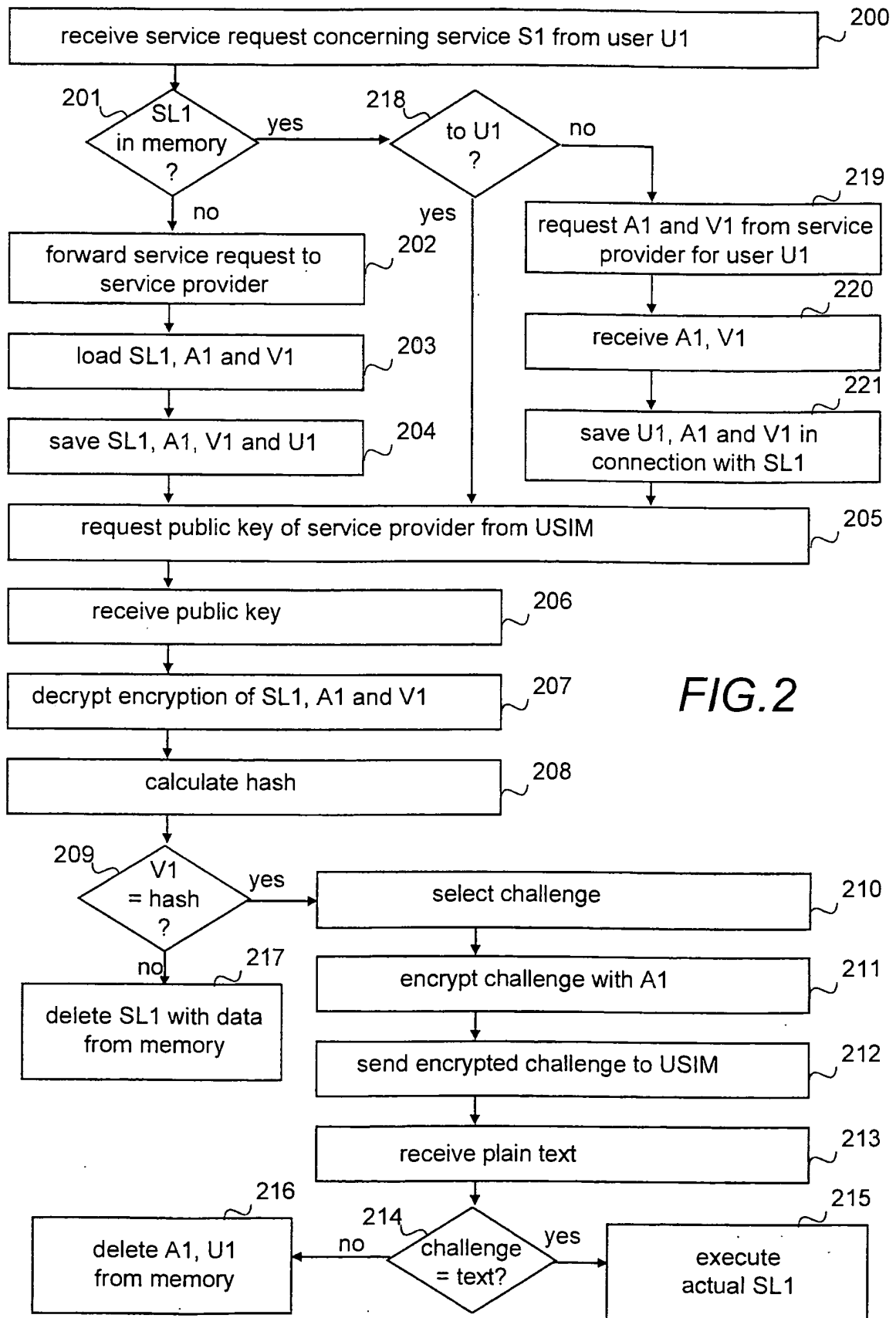


FIG. 2

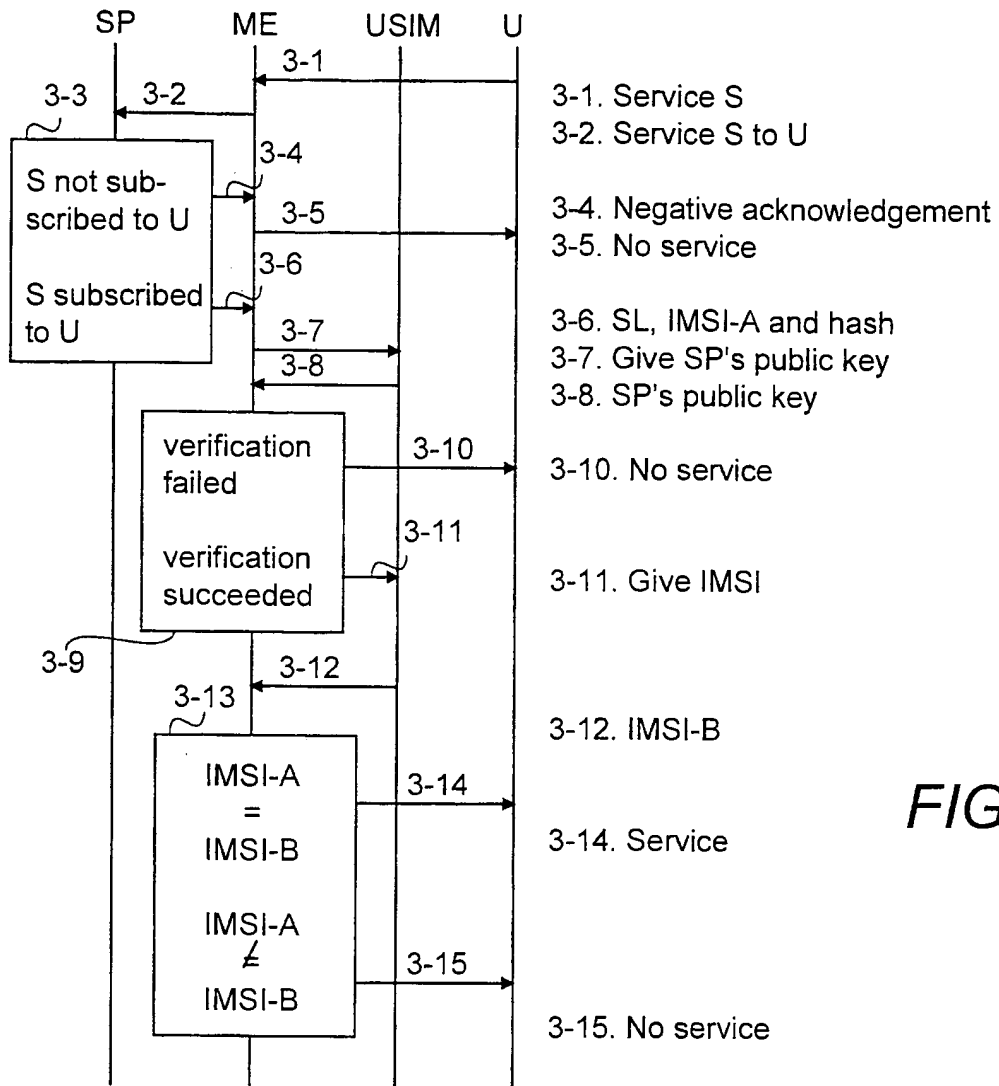


FIG.3

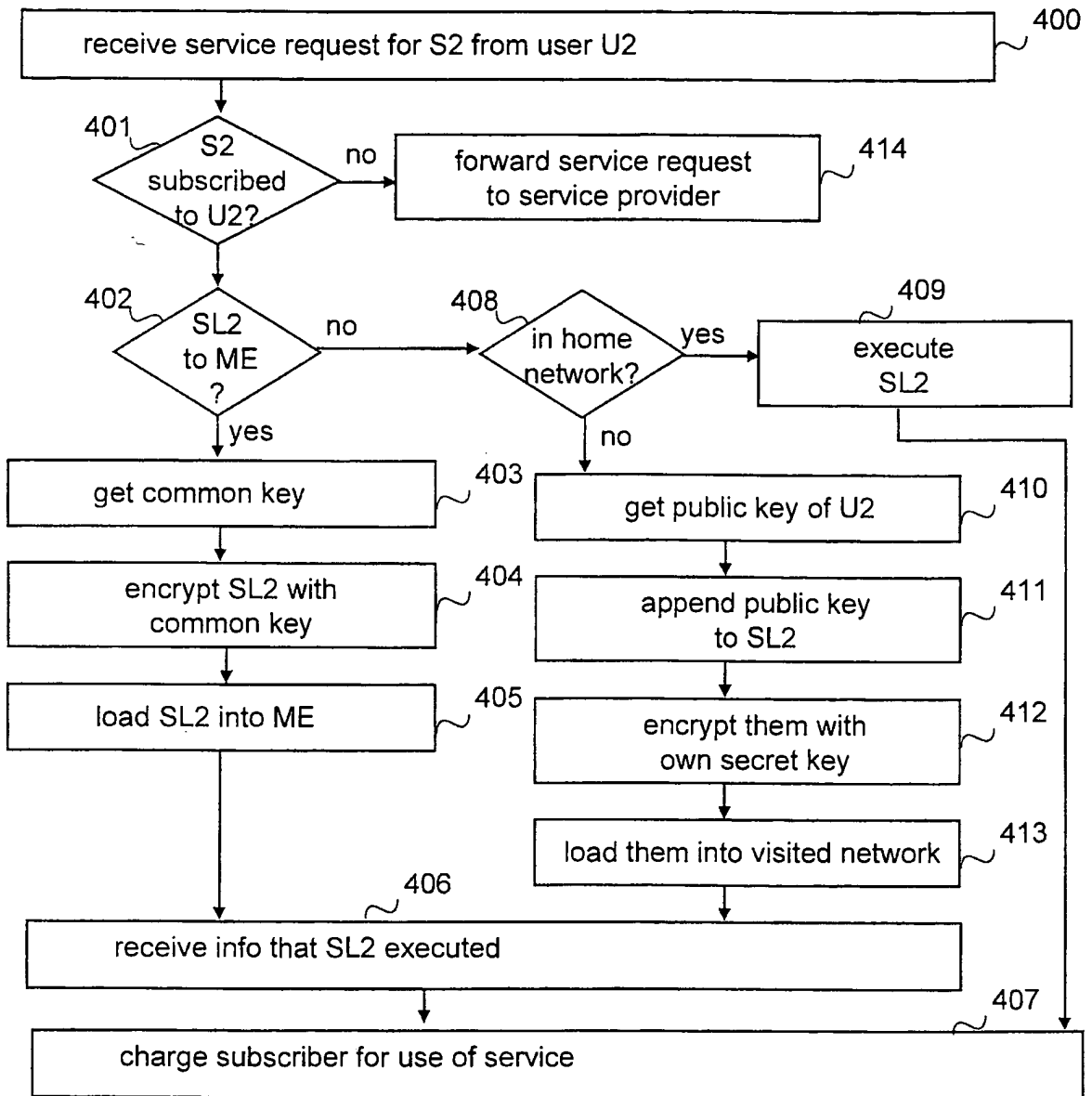


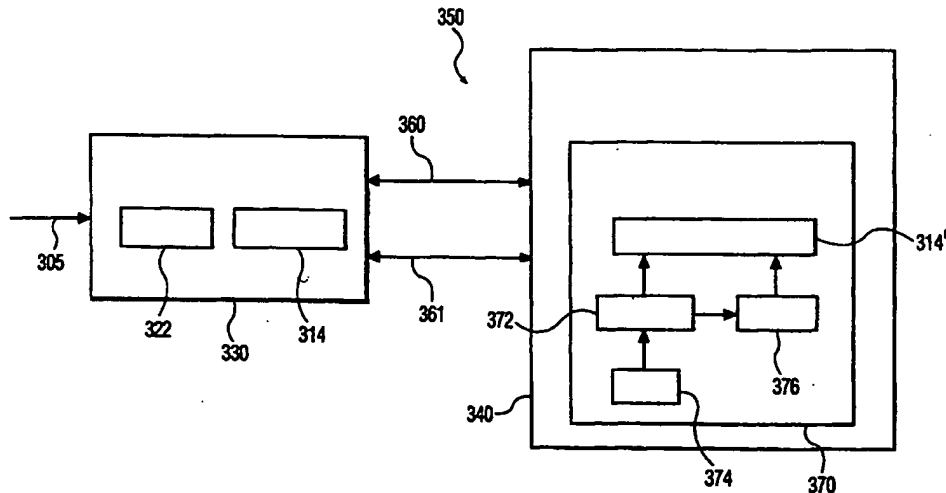
FIG.4



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

<p>(51) International Patent Classification ⁷ : H04N 7 /24</p>	<p>A2</p>	<p>(11) International Publication Number: WO 00/04727 (43) International Publication Date: 27 January 2000 (27.01.00)</p>
<p>(21) International Application Number: PCT/EP99/04704 (22) International Filing Date: 2 July 1999 (02.07.99) (30) Priority Data: 60/092,726 14 July 1998 (14.07.98) US 09/276,437 25 March 1999 (25.03.99) US (71) Applicant: KONINKLIJKE PHILIPS ELECTRONICS N.V. [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL). (72) Inventor: EPSTEIN, Michael, A.; Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL). (74) Agent: GROENENDAAL, Antonius, W., M.; Internationaal Octrooibureau B.V., Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).</p>		<p>(81) Designated States: BR, CN, JP, KR, MX, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE). Published <i>Without international search report and to be republished upon receipt of that report.</i></p>

(54) Title: USE OF A WATERMARK FOR THE PURPOSE OF COPY PROTECTION



(57) Abstract

A copyright protection system for protecting content wherein a time dependent ticket is calculated (314) at a source device (330) by combining a checkpoint with a ticket. The checkpoint is transmitted (361) from a display device (340) to the source device prior to the source device transmitting (360) watermarked content to the display device. The checkpoint is also stored (376) at the display device. Thereafter, the source device transmits, to the display device, watermarked content, the ticket, and the time dependent ticket. At the display device, the stored checkpoint is compared (314) to a current count of a local clock (374) that was utilized for producing the checkpoint. If the stored checkpoint is within a window of time of the local clock, then the stored checkpoint is combined (314') with the ticket in the same way that the checkpoint is combined with the ticket at the source device. A result of the combination is compared to the time dependent ticket and if the result equals the time dependent ticket, then the watermark and ticket may be compared in the usual way to determine the copy protection status of the copy protected content (314').

FOR THE PURPOSES OF INFORMATION ONLY

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

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece	ML	Mali	TR	Turkey
BG	Bulgaria	HU	Hungary	MN	Mongolia	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MR	Mauritania	UA	Ukraine
BR	Brazil	IL	Israel	MW	Malawi	UG	Uganda
BY	Belarus	IS	Iceland	MX	Mexico	US	United States of America
CA	Canada	IT	Italy	NE	Niger	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NL	Netherlands	VN	Viet Nam
CG	Congo	KE	Kenya	NO	Norway	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NZ	New Zealand	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	PL	Poland		
CM	Cameroon	KR	Republic of Korea	PT	Portugal		
CN	China	KZ	Kazakistan	RO	Romania		
CU	Cuba	LC	Saint Lucia	RU	Russian Federation		
CZ	Czech Republic	LI	Liechtenstein	SD	Sudan		
DE	Germany	LK	Sri Lanka	SE	Sweden		
DK	Denmark	LR	Liberia	SG	Singapore		
EE	Estonia						

Use of a Watermark for the Purpose of Copy Protection.

Field of the Invention

This invention generally relates to a system for protecting copyrighted content. Specifically, the present invention pertains to utilizing a ticket and a watermark to protect content.

5

Background of the Invention

The ability to transmit digital information securely is increasingly important. Owners of content want to be able to provide the content to authorized users without having the content utilized by unauthorized users. However, one problem with digital content is that an exact copy can be made without any degradation in the quality of the copy. Therefore, the copying of digital content is very attractive to pirating operations or attackers.

10

Both small-scale and commercial pirates are interested in defeating copy-protected content in order to produce and sell illegal copies of the content. By avoiding payments to the rightful owner of the copy-protected content, the pirates may reap large profits. Typically, the pirate may take advantage of the difference in release windows in order access high value content and distribute it.

15

For instance, in the movie industry, release windows are utilized to maximize profit from content. The essence of these release windows is to first release the content to a premium service such as a pay-per-view service or a video on demand service. Thereafter, the content may be released on a lower price service such as a home-box-office service. At this time, the content may also be available to a consumer through a purchased storage medium such as a Digital Video Disc (DVD).

20

Pirates however, frustrate the use of these release windows by pirating the content that is available through the premium service and then releasing pirated versions of the content to the public. This may cause substantial financial losses to the rightful owners of the content. Accordingly, a successful copy protection scheme should at least frustrate a pirates attempt for a sufficient period of time till the legitimate owner of the content may reap their rightful profits.

25

Beyond some level of attacker, the expense of defeating the attacker exceeds a reasonable limit whereby the device must be priced beyond what consumer is willing to pay. Thus, a copy protection solution must be cost effective but secure against a large number of attackers.

5 A cost-effective method of copy protection is discussed in detail by Jean-Paul Linnartz et al., in Philips Electronics Response to Call for Proposals Issued by the Data Hiding Subgroup Copy Protection Technical Working Group, July 1997 ("Linnartz"). Within a digital transmission, such as an MPEG transport stream, additional data may be embedded within the transport stream to set the copy protection status of content contained within the digital
10 transmission. For instance, the desired copy protection status may be "copy-once", "no-more-copy", "copy-never", and "copy-freely". Content that has a status of copy-once may be played and copied. During copying, the copy-once content is altered such that the content is in the no-more-copy state. Copy-never content is content that may only be played and may not be copied. Copy-freely content may be played and copied without restriction.

15 The additional data may take the form of a digital watermark. The watermark may be embedded directly into the content so that removal of the watermark will degrade the quality of the content. The watermark may be utilized as part of the copy protection scheme. As an example, the copy-freely state may be designated by the lack of a watermark within the content.

20 In operation, a transmission, such as a digital transmission, is sent from a source device and received by a receiving device. A source device is a device that is writing content onto a data bus, initiating a broadcast transmission, initiating a terrestrial transmission, etc. A sink device is a device that reads content from the data bus, etc.

Fig. 1 shows a typical system for the transmission of content. In Fig. 1, the
25 source device is a broadcast initiator 101 that utilizes a transmitting antenna 102 to transmit content. The sink device is a broadcast receiver, such as a set-top-box (STB) 104 that utilizes a receiving antenna 103 for receiving the transmitted content. The STB 104 is shown connected to a display device 105, a player 106, and a player/recorder 107, through a bus 108. The term bus is utilized herein to refer to any system for connecting one device to another device. The
30 bus may be a hard wired system such as a coaxial wire, an IEEE 1553 bus, etc., or the bus may be a wireless system such as an infra-red (IR) or radio frequency (RF) broadcast system. Several of the devices shown in Fig. 1 may at one time act as a source device and at another time act as a sink device. The STB 104 may be a sink for the broadcast transmission and be a

source for a transmission on the bus 108. The player/recorder 107 may be a source/sink of a transmission to/from, respectively, the bus 108.

In the copy protection scheme discussed by Linnartz, a watermark (W) is embedded within transmitted content. A ticket is transmitted along with the transmitted content. The embedded watermark and the ticket together are utilized to determine the copy protection status of the transmitted content. The watermark may be embedded into the content by at least two known methods. One method embeds the watermark (W) in the MPEG coding of the content. Another method embeds the watermark (W) in the pixel data of the content. The ticket (T) is mathematically related to the watermark (W) as discussed in more detail below.

Performing one or more one-way functions on the ticket (T) derives the watermark (W). By use of the term one-way function, what is meant is that it is computationally unfeasible to compute the inverse of the function. An example of a publicly known mathematical one-way function is a hashing function, such as secure hash algorithm one (SHA-1) or RACE Integrity Primitives Evaluation Message Digest (RIPEMD). Computing an inverse means finding which particular x_0 leads to a given y_0 with $y_0=F(x_0)$. The term unfeasible is intended to mean that the best method will take too long to be useful for a pirate. For instance, the time that is required for a pirate to compute the inverse of a hashing function is too long for the pirate to frustrate the intended release window for protected content. The most efficient method known to find such an x_0 may be to exhaustively search all possible bit combinations of x_0 and to compute and verify $F(x_0)$ for each attempt. In other cases, there may be a more efficient method than an exhaustive search to compute an inverse of a one-way function, yet these methods are still too time consuming to be feasible for the pirate.

The bit content of the ticket (T) is generated from a seed (U). The content owner provides the seed (U). From the seed (U), a physical mark (P) is created. The physical mark (P) may be embedded on a storage medium such as a Read-Only Memory (ROM) disk. Performing one or more one-way functions on the physical mark (P), produces the ticket (T). The number of functions performed on the physical mark (P) to create the ticket (T) depends on the copy protection intended for the content.

In accordance with the system, the ticket (T) changes state during every passage of a playback device (e.g., a source device) and a recording device (e.g., a sink device). As discussed above, the state modifications are mathematically irreversible and reduce the remaining copy and play rights of the content that are granted by the ticket (T). In this way,

the ticket (T) indicates the number of sequential playback and recordings that may still be performed and acts as a cryptographic counter that can be decremented but not incremented.

It should be noted that the copy protection scheme only protects content on compliant systems. A compliant system is any system that obeys the copy protection rules described above and hereinafter. A non-compliant system may be able to play and copy material irrespective of the copy protection rules. However, a compliant system should refuse to play copies of content illegally made on a non-compliant system.

In accordance with the copy protection scheme, a physical mark (P) (e.g., data) is embedded on a storage medium and is not accessible by other user equipment. The physical mark (P) data is generated at the time of manufacturing of the storage medium as described above and is attached to the storage medium in a way in which it is difficult to remove the physical mark (P) data without destroying the storage medium. The application of a one-way mathematical function, such as a hashing function, to the physical mark (P) data four times results in a watermark. Much like watermarks embedded in paper, the watermark is embedded in the medium (e.g., containing video, audio, or data) in such a way that it is infeasible to remove the watermark without destroying the material. At the same time the watermark should be imperceptible when the medium is used in the usual manner, such as when content from the medium is displayed.

A watermark by itself may indicate whether or not content stored on the storage medium is copy-once or copy-never. For instance, the absence of a watermark may indicate that the content may be copied freely. The presence of the watermark without a ticket on a storage medium may indicate copy-never content.

When the content is transmitted over a bus or other transmission medium, the physical mark (P) data is hashed twice to generate a ticket. When a compliant player receives the content, the ticket is hashed twice and matched to the watermark. In the case where the twice-hashed ticket and the watermark match, the content is played. In this way, a party may not substitute a false ticket along with the content to frustrate the copy protection scheme. In the case where there is a watermark but no ticket in the content, a compliant system will refuse to record the content.

When a compliant recorder reads the content, the watermark is checked to see if the material is copy-freely, copy-once, or copy-never. When there is no watermark, the content is copy-freely and may be copied freely as discussed above. When the content contains a watermark but no ticket, the content is copy-never and a compliant recorder will refuse to copy the content. However, a compliant player will play the content as long as the ticket

hashed two times matches the watermark. When the content is copy-once, the content contains both a watermark and a ticket, a compliant recorder will hash the ticket twice and compare the twice-hashed ticket to the watermark. In the case where the watermark matches the twice-hashed ticket, the content may be recorded along with a once-hashed ticket and the watermark, thereby creating copy-no-more content (e.g., content with a once-hashed ticket and a watermark). The physical mark will be different on a writable disc and thus, even if an illegal copy is made of copy-never content via a non-compliant recording device, a compliant player will refuse to play the content recorded on the writable disc.

It should be noted that in a broadcast system, such as a pay-per-view system, a copy-never state may be indicated by the presence of a once-hashed ticket and a watermark. Both copy-no-more stored content and copy-never broadcast content are treated by a compliant system similarly. The content containing the once-hashed ticket may be played but may not be recorded in a compliant system. In the event that a party tries to record the content with the once-hashed ticket, a compliant recorder will first twice-hash the once-hashed ticket and compare the result (e.g., a thrice-hashed ticket) with the watermark. Since the thrice-hashed ticket will not match the watermark, the compliant recorder will refuse to record the content.

A compliant player that receives the once-hashed ticket will hash the once-hashed ticket and compare the result (e.g., a twice-hashed ticket) to the watermark. Since the twice-hashed ticket matches the watermark, the compliant player will play the content.

However, a problem exists wherein a non-compliant recorder receives content containing a ticket (a twice-hashed physical mark) and a watermark. In the event that a non-compliant recorder does not alter the ticket upon receipt or recording (e.g., the non-compliant recorder makes a bit-for-bit copy), the non-compliant recorder may make multiple copies of the ticket and the watermark that will play on a compliant player and that may be recorded on a compliant recorder. The same problem can exist where a non-compliant recorder receives content containing a once-hashed ticket (a thrice-hashed physical mark) and a watermark indicating copy-no-more content. In this case, the non-compliant recorder may make multiple copies of the once-hashed ticket and the watermark that will play on the compliant player.

In a case wherein the player receives the content directly from a read only medium, such as a Compact Disc ROM (CD-ROM), a physical mark can be embedded in the physical medium of the CD-ROM that is produced by an authorized manufacturer. The player may then check the physical mark to ensure that the content is being received from an authorized medium. In this way, if a pirate makes an unauthorized copy, the physical mark

will not be present on the unauthorized copy and a compliant player will refuse to play the content. However, in the case of broadcast data for instance, wherein a player does not read content directly from the read-only medium, this method of copy protection is unavailable. Thus, for instance, a non-compliant player may deceive a compliant display device.

5 Accordingly, it is an object of the present invention to overcome the disadvantages of the prior art.

Summary of the Invention

10 This object of the present invention is achieved by a copy protection system for protecting content, such as content containing a watermark embedded therein (e.g., watermarked content). To this end, the invention provides a content protecting method, a copy protection system, a source device, and a display device as defined in the independent claims. The dependent claims define advantageous embodiments. In accordance with the present invention, a relative time dependent ticket is created at a source device preferably utilizing a display device dependent time reference (a checkpoint). In accordance with one embodiment
15 of the present invention, the checkpoint is combined with a ticket utilizing a concatenation function and a one-way function (e.g., a hashing function). The checkpoint is transmitted from the display device to the source device prior to the source device transmitting watermarked content to the display device. The checkpoint is also stored at the display device. Thereafter,
20 the source device transmits to the display device watermarked content, the ticket, and the relative time dependent ticket.

At the display device, the stored checkpoint is compared to a current relative time reference. If the difference between the stored checkpoint and the current relative time reference is acceptable, then further steps, as discussed below, may proceed. What is an
25 acceptable difference between the stored checkpoint and the current relative time reference will depend on the nature of the desired content protection. For example, in one embodiment or for one particular type of content, the difference may be short to ensure that the content is being transmitted and received in real time. In another embodiment or for another type of content, the difference may be longer to allow for storage of the content for later playback.

30 When the difference between the stored checkpoint and the current relative time reference is acceptable, the ticket is next hashed twice and compared to the watermark in the usual way. In the event that the ticket compares to the watermark ($W = H(H(T))$), the stored checkpoint is combined with the ticket in the same way that the checkpoint was combined with the ticket at the source device. A result of the combination is compared to the relative

time dependent ticket. If the result equals the relative time dependent ticket, then the display device is provided with access (e.g., enabled to display) to the watermarked content.

Preferably, the checkpoint is derived from a counter that purposely is inaccurate such that the count can be said to be unique as compared to the count from other display
5 devices. The counter is constructed with a sufficient number of bits such that the counter will not roll over to zero in the lifetime of the display device. The counter is constructed to only count up, such that the count may not be reversed and thereby, allow expired content to be displayed.

In yet another embodiment, a certificate containing the public key of the source
10 device is sent to the display device prior to the above described process. A public key known to the display device may be used to verify the certificate. Preferably, the public key used to verify the certificate is built into the display device by the manufacturer of the display device. In this embodiment, the relative time dependent ticket (the checkpoint concatenated with the
15 ticket) may be encrypted utilizing a private key of the source device. The encrypted relative time dependent ticket is then transmitted from the source device to the display device along with the watermarked content and the ticket. Thereafter, prior to the display device verifying the checkpoint, the display device decrypts the relative time dependent ticket utilizing a public key of the source device. In still yet another embodiment, the relative time dependent ticket may be signed (as is know in the art, by hashing the relative time dependent ticket and
20 encrypting that hashed result) utilizing a private key of the source device. The resulting signature is sent along with the watermarked content, the relative time dependent ticket, and ticket to the display device. Thereafter, prior to the display device verifying the checkpoint, the display device verifies the signature on the relative time dependent ticket utilizing a public key of the source device.

25

Brief Description of the Drawings

The following are descriptions of embodiments of the present invention that when taken in conjunction with the following drawings will demonstrate the above noted features and advantages, as well as further ones. It should be expressly understood that the
30 drawings are included for illustrative purposes and do not represent the scope of a present invention. The invention is best understood in conjunction with the accompanying drawings in which:

Fig. 1 shows a conventional system for the transmission of content;

Fig. 2 shows an illustrative communication network in accordance with an embodiment of the present invention; and

Fig. 3 shows details of an illustrative communication network in accordance with embodiment of the present invention wherein a source device provides content to a sink device.

Detailed Description of the Invention

Fig. 2 depicts an illustrative communication network 250 in accordance with an embodiment of the present invention. A source device 230, such as Set Top Box (STB), a Digital Video Disc (DVD), a Digital Video Cassette Recorder (DVCR), or another source of content, utilizes a transmission channel 260 to transmit content to a sink device 240. The transmission channel 260 may be a telephone network, a cable television network, a computer data network, a terrestrial broadcast system, a direct broadcast satellite network, some combination thereof, or some other suitable transmission system that is known in the art. As such, the transmission channel 260 may include RF transmitters, satellite transponders, optical fibers, coaxial cables, unshielded twisted pairs of wire, switches, in-line amplifiers, etc. The transmission channel 260 may also operate as a bi-directional transmission channel wherein signals may be transmitted from/to the source device 230, respectively, to/from the sink device 240. An additional transmission channel 261 may also be utilized between the source device 230 and the sink device 240. Typically, the transmission channel 260 is a wide-bandwidth channel that in addition to transmitting copy protection content (e.g., copy protection related messages), transmits copy protected content. The transmission channel 261 typically is a low-bandwidth channel that is utilized to transmit copy protection content.

The sink device 240 contains a memory 276 that is utilized for storing a checkpoint. The sink device 240 also contains a counter, such as a counter 272, that is utilized for generating the checkpoint. Preferably, the counter 272 should increment on a microsecond or better resolution as suitable for the application. The counter 272 should be free running. For instance, the counter 272 should count at all times that the sink device 240 is on. The bits of the counter 272 should employ non-volatile memory such as an electrically erasable programmable read-only memory (EEPROM) for the storage of the count. The counter 272 preferably is constructed to only count in one direction (e.g., up) and not in another direction (e.g., down). In a preferred embodiment, the counter 272 is driven by an inaccurate time source (e.g., inaccurate in terms of keeping time over hours, not necessarily over seconds), such as clock 274. The clock 274 is preferably unreliable so that drift with respect to time and

temperature is also non-negligible. Over time, this has the effect of randomizing the count of a counter for each sink device of a population of sink devices. In addition, the counter 272 may be driven fast for a random period of time to initialize the counter 272 to a random number at the time of manufacture. All of the above, has an effect of further randomizing the counter
5 272. The counter 272 is also configured such that it is inaccessible to a user. Accordingly, the user may not reset the counter 272.

The checkpoint, in accordance with the present invention, is transmitted to the source device 230 utilizing at least one of the transmission channels 260, 261. The source device 230 utilizes the checkpoint to change the ticket such that the watermarked content may
10 only be utilized (e.g., played) by a corresponding sink device as described in more detail below. In the event that the corresponding sink device, such as the sink device 240, receives the watermarked content, then the content may be provided to a device, such as a display device 265, for display thereon. Preferably, the display device 265 is integral to the sink device 240 such that the display device 265 is the final arbiter in determining whether the copy
15 protected content may be utilized. It should be obvious that although the device is illustratively shown as the display device 265, in fact the device may be any known device that may be suitably utilized for the copy protected content. For instance, in a case wherein the copy protected content is audio content, the device may be the device that outputs the audio signal.

In one embodiment of the present invention, the content may be provided from
20 the source device 230 in the form of a Moving Picture Experts Group (MPEG) compliant transport stream, such as an MPEG-2 compliant transport stream. However, the present invention is not limited to the protection of an MPEG-2 compliant transport stream. As a person skilled in the art would readily appreciate, the present invention may be suitably employed with any other data stream that is known in the art for transmitting content.

25 In another embodiment, the source device 230 may be a conditional access (CA) device. In this embodiment, the transmission channel 260 is a conditional access module bus.

Fig. 3 depicts details of an illustrative communication network 350 in accordance with an embodiment of the present invention. In the communication network 350,
30 a source device 330 provides content including copy protected content to a sink device 340 over a transmission channel 360. As discussed above with regard to the transmission channel 260, the transmission channel 360 may be a wide bandwidth transmission channel that may also have a bi-directional capability, such as a CA module bus.

The sink device 340 contains a copy protection status determination circuit 370 for creating/storing a checkpoint (C) and for determining the copy protection status of received content. The copy protection status determination circuit 370 contains a counter 372 and a clock 374 for creating the checkpoint (C). The counter 372 preferably contains a large number of bits (e.g., 64 bits for a clock 374 that increments on a millisecond basis). Preferably, the counter 372 should have a total count cycle time (the time required for the counter 372 to reach a top count from a bottom count) longer than a useful life of the sink device 340 (e.g., ten years). The clock 374 is preferably randomized (e.g., unreliable such that drift with respect to time and temperature is non-negligible) as discussed above with regard to the clock 274 shown in Fig. 2. The counter 372 is configured such that it is inaccessible and has no reset function even in the event of a removal of power. As such, the counter 372 may contain non-volatile storage, such as programmable read-only memory (PROM), electrically erasable PROM (EEPROM), static random access memory (static-RAM), etc. Further, the copy protection status determination circuit 370 contains a memory device 376 for storing the checkpoint (C):

In operation, the source device 330 may request the checkpoint (C) from the sink device 340 prior to transmitting copy protected content. In alternate embodiments, the sink device 340 may transmit the checkpoint (C) to the source device 330 as a portion of a request for the source device 330 to begin transmission of copy protected content to the sink device 340. The sink device 340 may utilize either of the transmission channels 360, 361 for transmission of the request for copy protected content and/or for transmission of the checkpoint (C). However, in some embodiments of the present invention, the transmission channel 360 may be unidirectional and may only be utilized for the transmission of content to the sink device 340 from the source device 330. In these embodiments, the transmission channel 361 is utilized for the transmission of the checkpoint (C) from the sink device 340 to the source device 330. The transmission channel 361 may also be utilized for transmitting a request for copy protected content from the sink device 340 to the source device 330.

In an alternate embodiment, the transmission channel 360 has bi-directional capability and may be utilized for transmissions both to and from the source device 330, and to and from the sink device 340. In this embodiment, the transmission channel 361 may not be present or it may be utilized solely for the transmission of content requiring low bandwidth. For instance, the source device 330 may utilize the transmission channel 361 to transmit to the sink device 340 a request for the transmission of the checkpoint (C).

In one particular embodiment, the source device 330 is a conditional access (CA) device 330, the transmission channel 360 is a CA module bus 360, and the sink device 340 is a display device 340. Prior to the transmission of copy protected content, the CA device 330 transmits a request for a checkpoint (C) (e.g., the current count from the free running counter 372) from the display device 340. In response to the request, the display device 340 transmits the checkpoint (C) to the CA device 330 over the CA module bus 360. In addition to sending the checkpoint (C) to the CA device 330, the display device 340 saves the checkpoint (C) in the memory 376.

The CA device 330 contains a processor 314. The processor 314 utilizes a ticket and the checkpoint (C), received from the display device 340, to create a relative time dependent ticket (TDT) as discussed in more detail below. In one embodiment, the processor 314 may simply be a fixed hardware device that is configured for performing functions, such as mathematical functions, including a concatenation function, a one-way function, such as a hashing function, etc. In alternate embodiments, the processor 314 may be a microprocessor or a reconfigurable hardware device. What is intended by the term "relative time dependent ticket (TDT)" is that due to the randomization of the counter 372 as discussed above, the checkpoint (C) is not directly related to an absolute time amongst all sink devices. The checkpoint (C) is only related to a relative time of a given sink device such as the display device 340.

In one embodiment, the copy protected content is received via an input 305 as an audio/video (A/V) signal. Preferably, in this embodiment, the A/V signal contains a watermark (W) and a ticket (T). The watermark (W) and the ticket (T) are related as discussed with regard to the prior art (e.g., $W = H(H(T))$). Preferably, the watermark (W) is embedded into the copy protected content. In this way, removal of the watermark (W) from the copy protected content will result in the copy protected content becoming largely degraded. The ticket accompanies the content and is not embedded in it.

In an alternate embodiment, the copy protected content is read from a physical medium, such as a digital video disc (DVD). In this embodiment, the DVD may contain a physical mark (P) as described above. Further, content contained on the DVD (e.g., A/V content) has a watermark (W) embedded therein (e.g., watermarked content) such that removal of the watermark (W) from the A/V content results in the A/V content becoming largely degraded. In this embodiment, the physical mark (P), the ticket (T), and the watermark (W) are related as follows:

$$T = H(H(P)) \quad (1)$$

$$W = H(H(T)) \quad (2)$$

In any event, at the CA device 330, the checkpoint (C) is combined with the ticket (T), utilizing for instance concatenation and hashing functions. Thereby, a time dependent ticket (TDT) is created as follows:

5

$$TDT = H(T.C). \quad (3)$$

The watermarked content, containing a watermark (W) embedded therein, the time dependent ticket (TDT), and the ticket (T), are then transmitted via the CA module bus 360 to the display device 340.

10

At the receiver 340, the copy protection status determination circuit 370 extracts the watermark (W) from the watermarked content. The copy protection status determination circuit 370 compares the watermark (W) and the ticket (T) in the usual way, as is known in the art (e.g., $W = H(H(T))$?).

15

In the event that the comparison does not pass (e.g., $W \neq H(H(T))$), then the content is discarded and any selected operation at the display device 340 (e.g., play, record, etc.) regarding the content is disabled. However, if the comparison does pass (e.g., $W = H(H(T))$), then the copy protection determination circuit 370 retrieves the stored checkpoint (C) from the memory 376 and combines the ticket (T) with the stored checkpoint (C), utilizing the same operation that was utilized at the source device 330 for creating the time dependent ticket (TDT). To this end, the receiver 340 comprises a processor 314' that is comparable to the processor 314 in the source device 330. For instance, concatenation and hashing functions may be utilized at the display device 340 for combining the ticket (T) with the stored checkpoint (C). A result of the combination is then compared to the time dependent ticket (TDT):

20

25

$$TDT = H(T.C)? \quad (4)$$

In the event that the result does not equal the time dependent ticket (TDT), then the content is discarded and any selected operation at the display device (e.g., play, record, etc.) regarding the content is disabled. This may happen, for instance, in a case wherein an improper display device (e.g., a display device other than the display device that requested the content) has received the content. If the result does equal the time dependent ticket (TDT), then access to the content is enabled in accordance with the access granted by the ticket.

30

In a preferred embodiment, a further step is performed prior to the display device 340 having access to the copy protected content. Specifically, the checkpoint (C) stored in the memory 376 is compared to a current count of the (running) counter 372. In the event that the stored checkpoint (C) is within an allowable window of the current count from the counter 372 (e.g., within 24 hours of the count for some applications), then the display device 340 is provided with access to the copy protected content. What is an allowable window between the stored checkpoint (C) and the current count will depend on the nature of the desired content protection. For example, in one embodiment or for one particular type of content, the allowed window (the difference between the stored checkpoint (C) and the current count) may be short to ensure that the content is being transmitted and received in real time. In another embodiment or for another type of content, the allowed window may be longer (e.g., months or years) to allow for storage of the content for later playback.

If the checkpoint (C) has expired (e.g., not within the allowed window), then the checkpoint (C) is erased and the display device 340 is not provided with access to the copy protected content. As is readily ascertained by a person of ordinary skill in the art, the comparison of the checkpoint (C) to the current count may be performed any time prior to the display device having access to the copy protected content. In a preferred embodiment, the checkpoint (C) is compared to the current count prior to the comparison of the watermark (W) to the ticket (T).

It should be clear that a trusted source should be utilized to create the recorded content or the real time transmitted content (e.g., received over the input 305). A CA device, such as the CA device 330, which is inherently designed to be tamper resistant is an example of a trusted real time source. In this case, it may be assumed that the CA device 330 decrypts the watermarked content so that prior to the watermarked contents arrival at the CA device 330, the watermarked content cannot be recorded.

In a case wherein the ticket (T) does not properly compare to the watermark (W), or some other portion of the copy protection status determination process fails, the copy protected content is discarded. In addition, when the copy protection status determination process fails, no operation regarding the copy protected content is enabled at the display device 340.

In accordance with the present invention, a checkpoint (C) from a counter of a given display device is in effect unique. Accordingly, the copy protected content transmitted by the CA device 330 may not be distributed to a display device other than the display device that sent the checkpoint (C). In addition, by comparing the checkpoint (C) to the count of the

counter 372, the copy protected content may be restricted to being played within a time, as determined by the window of time as discussed above.

In yet another embodiment, a private/public key system, as is known by a person of ordinary skill in the art, is utilized to further secure the copy protected content in accordance with the present invention. In accordance with this embodiment, the display device 5 340 has a public key that is trusted e.g., secure for example by being installed in part of the display device hardware, such as stored in the memory 376. The public key corresponds to a private key of the manufacturer of the display device 340 and is stored, for instance, in a memory 322 at the CA device 330. The private key is utilized to sign certificates of each CA 10 device manufacturer, as is known in the art.

In operation, when the CA device 330 is connected to the display device 340 via the CA module bus 360, a certificate containing the CA device 330 public key is sent to the display device 340. Once the certificate containing the public key of the CA device 330 is verified by the display device 340, as is known in the art, the public key of this CA device 330 15 is stored at the display device 340. Thereafter, the CA device 330 may digitally sign the time dependent ticket (TDT). For instance, the time dependent ticket (TDT) may be hashed and the result may be encrypted by the private key of the CA device 330 to form a signature. The signature is sent from the CA device 330 to the display device 340 together with the watermarked content, the ticket, and the time dependent ticket (TDT). At the display device 20 340, the signature is verified utilizing the public key of the CA device 330 and thereafter, the time dependent ticket (TDT) and checkpoint (C) are utilized as described above.

In yet another embodiment, the time dependent ticket (TDT) may be encrypted utilizing the private key of the CA device 330. The encrypted time dependent ticket (TDT) is then transmitted from the CA device 330 to the display device 340 along with the 25 watermarked content and the ticket (T). Thereafter, prior to the display device 340 verifying the checkpoint (C), the display device 340 decrypts the time dependent ticket (TDT) utilizing the public key of the CA device 330. Thereafter, the time dependent ticket (TDT) may be utilized as discussed above.

An illustrative protocol for use of a checkpoint and a private/public key system 30 in accordance with an embodiment of the present invention is described below. In accordance with the present invention, after a CA device is connected to a display device, the CA device sends a certificate containing the CA device public key to the display device. The display device verifies the certificate utilizing the embedded public key of the manufacturer and stores the verified public key of the CA device. In response to a request for copy protected content

from the display device, the CA device requests a checkpoint (C) from the display device. The display device sends the checkpoint (C) to the CA device and also stores a copy of the checkpoint (C) locally (e.g., at the display device). The CA device combines the checkpoint (C) with the ticket (T) utilizing concatenation and hashing functions to produce a time dependent ticket (TDT). The CA device encrypts the time dependent ticket (TDT) utilizing the CA device private key. The encrypted time dependent ticket (TDT) is then sent to the display device along with the watermarked content and the ticket (T). The display device compares the stored checkpoint (C) with the current state of a counter to determine if the checkpoint (C) is within an allowable window of time of the current state of the counter. If the stored checkpoint (C) is not within the allowable window of time of the current state of the counter, then access to the content is disabled. If the stored checkpoint (C) is within the allowable window, then the display device utilizes the public key of the CA device to decrypt the time dependent ticket (TDT). The display device combines the ticket (T) with the stored checkpoint (C) utilizing concatenation and hashing functions and compares a result to the time dependent ticket (TDT). If the result is not equal to the time dependent ticket (TDT), then access to the content is disabled. If the result is equal to the time dependent ticket (TDT), the ticket and watermark are compared in the usual way. If step 480 fails (e.g., $W \neq H(H(T))$), then in step 485, access to the content is disabled. If the ticket and the watermark do not correspond, (e.g., $W = H(H(T))$), access to the content is enabled (e.g., the content may be displayed).

The following embodiments of the invention overcome the disadvantages of the prior art. A display device is provided that is the final arbiter in deciding whether to display the protected content. In this way, the display device is the gatekeeper that disallows recordings that are made and played back on non-compliant players/recorders. A further embodiment provides a method of transmitting copy protected copy-never content that will prevent a pirate from making copies that will display on a compliant display device. A ticket is created that is unique to a particular display device so that copy protected content will only play on the particular display device. A still further embodiment creates a ticket that is inspected by the display device to decide whether the content is being transmitted in real time. A time dependent ticket is created that is checked by a display device to determine if content has expired or aged beyond an allowable window of time from a checkpoint. Another embodiment of the invention uses a relative time reference configured such that each display device has a different relative time reference.

It should be noted that the above-mentioned embodiments illustrate rather than limit the invention, and that those skilled in the art will be able to design many alternative

embodiments without departing from the scope of the appended claims. In the claims, any reference signs placed between parentheses shall not be construed as limiting the claim. The word "comprising" does not exclude the presence of other elements or steps than those listed in a claim. Another embodiment of the invention can be implemented by means of hardware
5 comprising several distinct elements, and by means of a suitably programmed computer. In a device claim enumerating several means, several of these means can be embodied by one and the same item of hardware.

CLAIMS:

1. A method of protecting content transmitted as a stream of data, the method comprising the steps of:
 - determining a checkpoint at a receiving device (240);
 - calculating, at a source device (230), a time dependent ticket utilizing the
 - 5 checkpoint, wherein a watermark, a ticket, and the checkpoint together indicate a copy protection status of the content;
 - transmitting said stream of data, said watermark, said ticket, and said time dependent ticket to said receiving device (240); and
 - comparing said time dependent ticket to a stored checkpoint at said receiving
 - 10 device (240).
2. The method of claim 1, wherein said step of calculating said time dependent identifier comprises the steps of:
 - combining said checkpoint with said ticket, and
 - 15 calculating a one-way operation on said combined checkpoint and ticket.
3. The method of claim 2, further comprising the step of selecting said one-way function to be a hashing function.
- 20 4. The method of claim 1, further comprising the step of comparing, at said receiving device (240), said ticket and said watermark to determine the copy protection status of the content if said time dependent ticket compares to said stored checkpoint.
5. The method of claim 1, wherein said checkpoint is a checkpoint from a receiver
- 25 counter (272).
6. The method of claim 5, wherein said receiver counter (272) is randomized.

7. The method of claim 5, wherein the step of comparing said time dependent ticket further comprises the step of comparing said stored checkpoint to a current count from said receiver counter (272).
- 5 8. The method of claim 1, wherein said step of calculating said time dependent ticket further comprises the step of signing said time dependent ticket with a private key of said source device (230), and wherein said step of comparing said time dependent ticket further comprises the step of verifying the signature using a public key of said source device (230).
- 10 9. A copy protection system for protecting content wherein a ticket and a watermark indicates a copy protection status of said content, the system comprising:
a source device (330) configured to calculate a time dependent ticket using a checkpoint and a one-way function, and to provide a data stream containing said content, said
15 ticket, a watermark, and said time dependent ticket; and
a display device (340) configured to produce said checkpoint, configured to receive said data stream, and configured to compare said time dependent ticket to said
checkpoint using said ticket and said one-way function.
- 20 10. The system of claim 9, wherein said display device (340) is further configured to compare said ticket to said watermark and to display said content if said time dependent ticket compares to said checkpoint.
11. The system of claim 9, wherein said display device (340) comprises a counter
25 (372) and wherein said checkpoint is a checkpoint from said counter (372).
12. The system of claim 11, wherein said display device (340) is further configured to randomize said counter (372).
- 30 13. The system of claim 11, wherein said display device (340) is further configured to compare said checkpoint to a current count from said counter (372) prior to displaying said content.

14. A source device (330) for protecting content wherein a ticket and a watermark indicate a copy protection status of the content, said source device (330) comprising:

a reader device configured to read watermarked content from a physical medium and configured to read a physical mark from said physical medium; and

5 a processor (314) configured to receive a checkpoint, configured to calculate said ticket using said physical mark and a one-way function, configured to calculate a time dependent ticket using said ticket, said checkpoint, and said one-way function, and configured to provide to a receiver (340) a data stream containing said watermarked content, said ticket, and said time dependent ticket.

10

15. A display device (340) for receiving data containing watermarked content and a ticket, wherein said ticket and watermark together indicate a copy protection status of the content, said display device comprising:

a counter (372) configured to provide a checkpoint and a current time reference;

15 and

a processor (314'), wherein if said checkpoint is contained within a time window determined by said current time reference, said (314') processor is configured to:

receive a time dependent ticket and said data,

combine said ticket with said checkpoint to produce a first result,

20 perform a one-way function on said first result to produce a second result, and compare said second result to said time dependent ticket, wherein said display device (340) is further configured to display said data if said second results compares to said time dependent ticket.

1/2

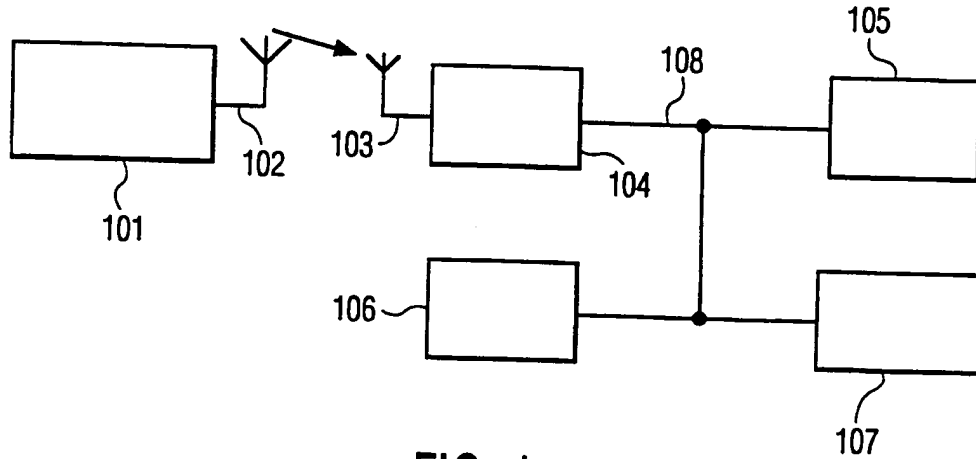


FIG. 1

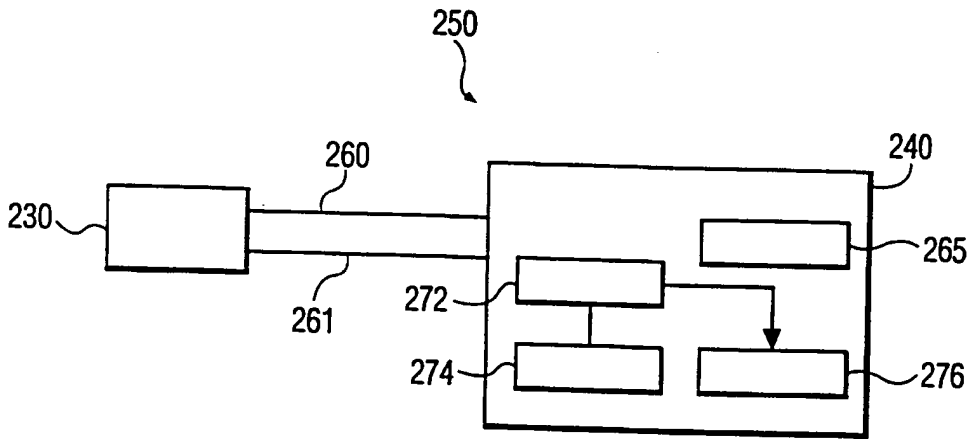


FIG. 2

2/2

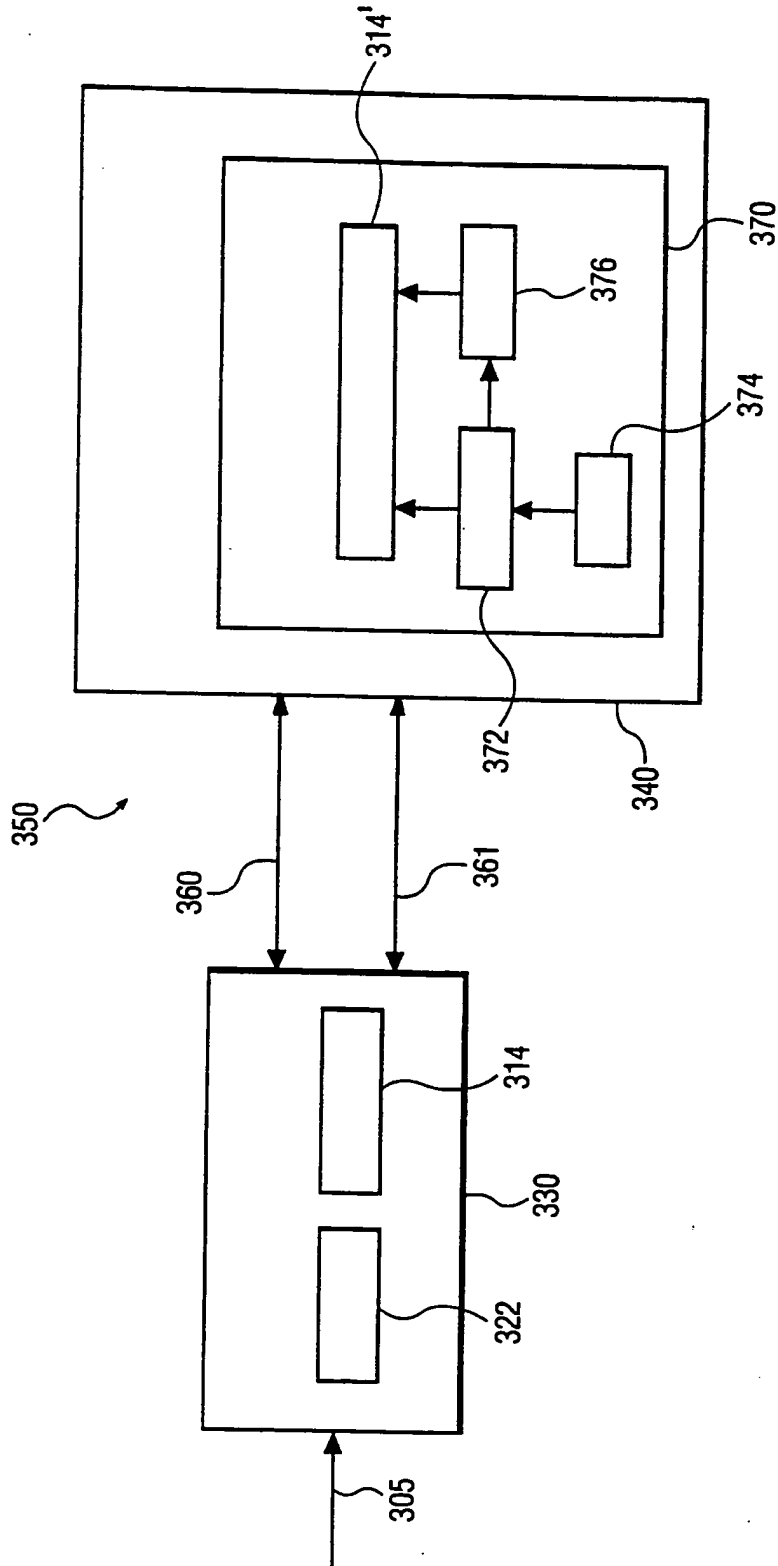


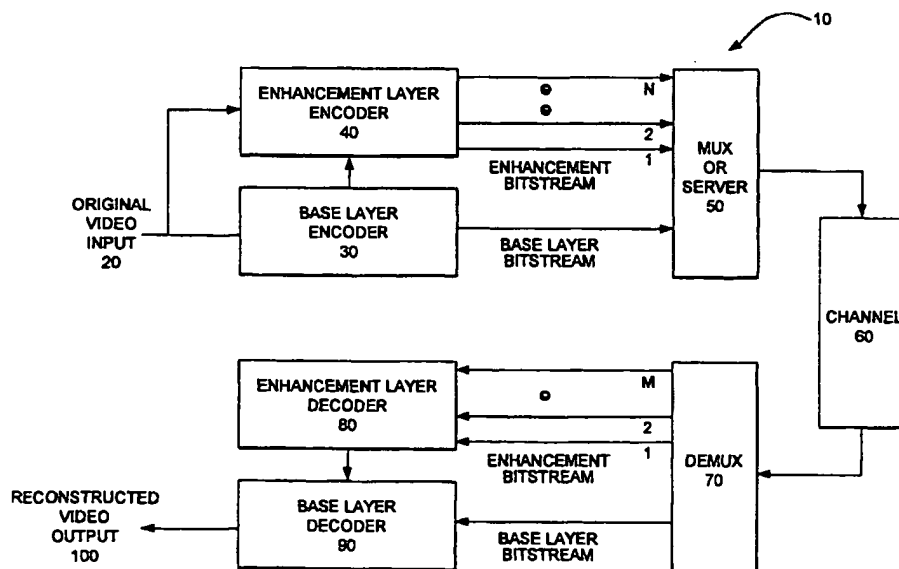
FIG. 3



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

<p>(51) International Patent Classification ⁷ : H04N 7/30</p>	<p>A2</p>	<p>(11) International Publication Number: WO 00/05898 (43) International Publication Date: 3 February 2000 (03.02.00)</p>
<p>(21) International Application Number: PCT/US99/16638 (22) International Filing Date: 21 July 1999 (21.07.99) (30) Priority Data: 60/093,860 23 July 1998 (23.07.98) US 09/169,829 11 October 1998 (11.10.98) US (63) Related by Continuation (CON) or Continuation-in-Part (CIP) to Earlier Applications US 60/093,860 (CIP) Filed on 23 July 1998 (23.07.98) US 09/169,829 (CIP) Filed on 11 October 1998 (11.10.98) (71) Applicant (for all designated States except US): OPTIVISION, INC. [US/US]; 3450 Hillview Avenue, Palo Alto, CA 94304 (US). (72) Inventor; and (75) Inventor/Applicant (for US only): LI, Weiping [US/US]; 159 California Avenue, J103, Palo Alto, CA 94306 (US). (74) Agent: DAVIS, Paul; Wilson Sonsini Goodrich & Rosati, 650 Page Mill Road, Palo Alto, CA 94304-1050 (US).</p>	<p>(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG). Published Without international search report and to be republished upon receipt of that report.</p>	

(54) Title: SCALABLE VIDEO CODING AND DECODING



(57) Abstract

A video encoding method and apparatus for adapting a video input to a bandwidth of a transmission channel of a network that includes determining the number N enhancement layer bitstreams capable of being adapted to the bandwidth of the transmission channel of a network. A base layer bitstream is encoded from the video input wherein a plurality of enhancement layer bitstreams are encoded from the video input. The enhancement layer bitstreams are based on the base layer bitstream, wherein the plurality of enhancement layer bitstreams complements the base layer bitstream and the base layer bitstream and N enhancement layer bitstreams are transmitted to the network.

FOR THE PURPOSES OF INFORMATION ONLY

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

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece	ML	Mali	TR	Turkey
BG	Bulgaria	HU	Hungary	MN	Mongolia	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MR	Mauritania	UA	Ukraine
BR	Brazil	IL	Israel	MW	Malawi	UG	Uganda
BY	Belarus	IS	Iceland	MX	Mexico	US	United States of America
CA	Canada	IT	Italy	NE	Niger	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NL	Netherlands	VN	Viet Nam
CG	Congo	KE	Kenya	NO	Norway	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NZ	New Zealand	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	PL	Poland		
CM	Cameroon	KR	Republic of Korea	PT	Portugal		
CN	China	KZ	Kazakstan	RO	Romania		
CU	Cuba	LC	Saint Lucia	RU	Russian Federation		
CZ	Czech Republic	LJ	Liechtenstein	SD	Sudan		
DE	Germany	LK	Sri Lanka	SE	Sweden		
DK	Denmark	LR	Liberia	SG	Singapore		
EE	Estonia						

SCALABLE VIDEO CODING AND DECODING

BACKGROUND OF THE INVENTION**Field of the Invention**

5 The present invention relates to a method and apparatus for the scaling of data signals the bandwidth of the transmission channel; and more particularly to a scalable video method and apparatus for coding video such that the received video is adapted to the bandwidth of the transmission channel.

Description of Related Art

10

Signal compression in the video arena has long been employed to increase the bandwidth of either the generating, transmitting, or receiving device. MPEG - an acronym for Moving Picture Experts Group - refers to the family of digital video compression standards and file formats developed by the group. For instance, the MPEG-1 video sequence is an ordered stream of bits, with special bit patterns marking the beginning and ending of a logical section.

15

MPEG achieves high compression rate by storing only the changes from one frame to another, instead of each entire frame. The video information is then encoded using a technique called DCT (Discrete Cosine Transform) which is a technique for representing a waveform data as a weighted sum of cosines. MPEG use a type of lossy compression wherein some data is removed. But the diminishment of data is generally imperceptible to the human eye. It should be noted that the DCT itself does not lose data; rather, data compression technologies that rely on DCT approximate some of the coefficients to reduce the amount of data.

20

25 The basic idea behind MPEG video compression is to remove spatial redundancy within a video frame and temporal redundancy between video frames. The DCT-based (Discrete Cosine Transform) compression is used to reduce spatial redundancy and motion compensation is used to exploit temporal redundancy. The images in a video stream usually do not change much within small time intervals.

Thus, the idea of motion-compensation is to encode a video frame based on other video frames temporally close to it.

A video stream is a sequence of video frames, each frame being a still image. A video player displays one frame after another, usually at a rate close to 30 frames per second. Macroblocks are formed, each macroblock consists of four 8 x 8 luminance blocks and two 8 x 8 chrominance blocks. Macroblocks are the units for motion-compensated compression, wherein blocks are basic unit used for DCT compression. Frames can be encoded in three types: intra-frames (I-frames), forward predicted frames (P-frames), and bi-directional predicted frames (B-frames).

An I-frame is encoded as a single image, with no reference to any past or future frames. Each 8 x 8 block is encoded independently, except that the coefficient in the upper left corner of the block, called the DC coefficient, is encoded relative to the DC coefficient of the previous block. The block is first transformed from the spatial domain into a frequency domain using the DCT (Discrete Cosine Transform), which separates the signal into independent frequency bands. Most frequency information is in the upper left corner of the resulting 8 x 8 block. After the DCT coefficients are produced the data is quantized, i.e. divided or separated. Quantization can be thought of as ignoring lower-order bits and is the only lossy part of the whole compression process other than sub-sampling.

The resulting data is then run-length encoded in a zig-zag ordering to optimize compression. The zig-zag ordering produces longer runs of 0's by taking advantage of the fact that there should be little high-frequency information (more 0's as one zig-zags from the upper left corner towards the lower right corner of the 8 x 8 block).

A P-frame is encoded relative to the past reference frame. A reference frame is a P- or I-frame. The past reference frame is the closest preceding reference frame. A P-macroblock is encoded as a 16 x 16 area of the past reference frame, plus an error term.

To specify the 16 x 16 area of the reference frame, a motion vector is included. A motion vector (0, 0) means that the 16 x 16 area is in the same position as the macroblock we are encoding. Other motion vectors are generated are relative to that position. Motion vectors may include half-pixel values, in which case pixels are averaged. The error term is encoded using the DCT, quantization, and run-length

encoding. A macroblock may also be skipped which is equivalent to a (0, 0) vector and an all-zero error term.

A B-frame is encoded relative to the past reference frame, the future reference frame, or both frames.

5 A pictorial view of the above processes and techniques in application are depicted in prior art Fig. 15, which illustrates the decoding process for a SNR scalability. Scalable video coding means coding video in such a way that the quality of a received video is adapted to the bandwidth of the transmission channel. Such a coding technique is very desirable for transmitting video over a network with a time-
10 varying bandwidth.

SNR scalability defines a mechanism to refine the DCT coefficients encoded in another (lower) layer of a scalable hierarchy. As illustrated in prior art Fig. 15, data from two bitstreams is combined after the inverse quantization processes by adding the DCT coefficients. Until the data is combined, the decoding processes of the two
15 layers are independent of each other.

The lower layer (base layer) is derived from the first bitstream and can itself be either non-scalable, or require the spatial or temporal scalability decoding process, and hence the decoding of additional bitstream, to be applied. The enhancement layer, derived from the second bitstream, contains mainly coded DCT coefficients and a small
20 overhead.

In the current MPEG-2 video coding standard, there is an SNR scalability extension that allows two levels of scalability. MPEG achieves high compression rate by storing only the changes from one frame to another, instead of each entire frame. There are at least two disadvantages of employing the MPEG-2 standard for encoding
25 video data. One disadvantage is that the scalability granularity is not fine enough, because the MPEG-2 process is an all or none method. Either the receiving device can receive all of the data from the base layer and the enhancement layer or only the data from the base layer bitstream. Therefore, the granularity is not scalable. In a network environment, more than two levels of scalability are usually needed.

30 Another disadvantage is that the enhancement layer coding in MPEG-2 is not efficient. Too many bits are needed in the enhancement layer in order to have a noticeable increase in video quality.

The present invention overcomes these disadvantages and others by providing, among other advantages, an efficient scalable video coding method with increased granularity.

5

SUMMARY OF THE INVENTION

The present invention can be characterized as a scalable video coding means and a system for encoding video data, such that quality of the final image is gradually improved as more bits are received. The improved quality and scalability are achieved by a method wherein an enhancement layer is subdivided into layers or levels of
10 bitstream layers. Each bitstream layer is capable of carrying information complementary to the base layer information, in that as each of the enhancement layer bitstreams are added to the corresponding base layer bitstreams the quality of the resulting images are improved.

15

The number N of enhancement layers is determined or limited by the network that provides the transmission channel to the destination point. While the base layer bitstream is always transmitted to the destination point, the same is not necessarily true for the enhancement layers. Each layer is given a priority coding and transmission is effectuated according to the priority coding. In the event that all of the enhancement
20 layers cannot be transmitted the lower priority coded layers will be omitted. The omission of one or more enhancement layers may be due to a multitude of reasons.

For instance, the server which provides the transmission channel to the destination point may be experiencing large demand on its resources from other users, in order to try and accommodate all of its users the server will prioritize the data and
25 only transmit the higher priority coded packets of information. The transmission channel may be the limiting factor because of the bandwidth of the channel, i.e. Internet access port, Ethernet protocol, LAN, WAN, twisted pair cable, co-axial cable, etc. or the destination device itself, i.e. modem, absence of an enhanced video card, etc. may not be able to receive the additional bandwidth made available to it. In these
30 instances only M number (M is an integer number = 0, 1, 2, . . .) of enhancement layers may be received, wherein N number (N is an integer number = 0, 1, 2, . . .) of enhancement layers were generated at the encoding stage, $M \leq N$.

To achieve these and other advantages and in accordance with the purpose of the present invention, as embodied and broadly described, the scalable video method and apparatus according to one aspect of the invention includes a video encoding method for adapting a video input to a bandwidth of a transmission channel of a network, the method includes determining the number N of enhancement layer bitstreams capable of being adapted to the bandwidth of the transmission channel of the network. Encoding a base layer bitstream from the video input is then performed and encoding N number of enhancement layer bitstreams from the video input based on the base layer bitstream, wherein the plurality of enhancement layer bitstreams complements the base layer bitstream. The base layer bitstream and the N enhancement layer bitstreams are then provided to the network.

According to another aspect of the present invention, a video decoding method for adapting a video input to a bandwidth of a transmission channel of a network includes, determining number M of enhancement layer bitstreams of said video input capable of being received from said transmission channel of said network. Decoding a base layer bitstream from received video input and decoding M number of enhancement layer bitstreams from the received video input based on the base layer bitstream, wherein the M received enhancement layer bitstreams complements the base layer bitstream. Then reconstructing the base layer bitstream and N enhancement layer bitstreams.

According to still another aspect of the present invention, a video decoding method for adapting a video input to a bandwidth of a receiving apparatus, the method includes demultiplexing a base layer bitstream and at least one of a plurality of enhancement layer bitstreams received from a network, decoding the base layer bitstream, decoding at least one of the plurality of enhancement layer bitstreams based on generated base layer bitstream, wherein the at least one of the plurality of enhancement layer bitstreams enhances the base layer bitstream. Then reconstructing a video output.

According to a further aspect of the present invention, a video encoding method for encoding enhancement layers based on a base layer bitstream encoded from a video input, the video encoding method includes, taking a difference between an

original DCT coefficient and a reference point and dividing the difference between the original DCT coefficient and the reference point into N bit-planes.

5 According to a still further aspect of the present invention, a method of coding motion vectors of a plurality of macroblocks, includes determining an average motion vector from N motion vectors for N macroblocks, utilizing the determined average motion vector as the motion vector for the N macroblocks, and encoding 1/N motion vectors in a base layer bitstream.

10 Additional features and advantages of the invention will be set forth in the description which follows, and in part will be apparent from the description, or may be learned by practice of the invention. The aspects and other advantages of the invention will be realized and attained by the structure particularly pointed out in the written description and claims hereof as well as the appended drawings.

15 It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory and are intended to provide further explanation of the invention as claimed.

BRIEF DESCRIPTION OF THE DRAWINGS

20 The accompanying drawings, which are included to provide a further understanding of the invention and are incorporated in and constitute a part of this specification, illustrate embodiments of the invention and together with the description serve to explain the principles of the invention. In the drawings:

Fig. 1 illustrates a flow diagram of the scalable video encoding method of the present invention;

25 Fig. 2A illustrates conventional probability distribution of DCT coefficient values;

Fig. 2B illustrates conventional probability distribution of DCT coefficient residues;

Fig. 3A illustrates the probability distribution of DCT coefficient values of the present invention;

30 Fig. 3B illustrates the probability distribution of DCT coefficient residues of the present invention;

Figs. 3C and 3D illustrates a method for taking a difference of a DCT coefficient of the present invention;

Fig. 5 illustrates a flow diagram for finding the maximum number of bit-planes in the DCT differences of a frame of the present invention;

5 Fig. 6 illustrates a flow diagram for generating (RUN, EOP) Symbols of the present invention;

Fig. 7 Illustrates a flow diagram for encoding enhancement layers of the present invention;

10 Fig. 8 illustrates a flow diagram for encoding (RUN, EOP) symbols and sign_enh values of one DCT block of one bit-plane;

Fig. 9 illustrates a flow diagram for encoding a sign_enh value of the present invention;

Fig. 10 illustrates a flow diagram for adding enhancement difference to a DCT coefficient of the present invention;

15 Fig. 11 illustrates a flow diagram for converting enhancement difference to a DCT coefficient of the present invention;

Fig. 12 illustrates a flow diagram for decoding enhancement layers of the present invention;

20 Fig. 13 illustrates a flow diagram for decoding (RUN, EOP) symbols and sign_enh values of one DCT block of one bit-plane;

Fig. 14 illustrates a flow diagram for decoding a sign_enh value; and

Fig. 15 illustrates a prior a conventional SNR scalability flow diagram.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

25 Reference will now be made in detail to the preferred embodiments of the present invention, examples of which are illustrated in the accompanying drawings.

Fig. 1 illustrates the scalable video diagram 10 of an embodiment of the present invention. The original video input 20 is encoded by the base layer encoder 30 in accordance with the method of represent by flow diagram 400 of Fig. 4. A DCT coefficient OC and its corresponding base layer quantized DCT coefficient QC are
30 generated and a difference determined pursuant to steps 420 and 430 of Fig. 4. The

difference information from the base layer encoder 30 is passed to the enhancement layer encoder 40 that encodes the enhancement information.

The encoding of the enhancement layer encoder is performed pursuant to methods 500 - 900 as depicted in Figs. 5 - 10, respectively and will be briefly
5 described. The bitstream from the base layer encoder 30 and the N bitstreams from the enhancement layer encoder 40 are capable of being sent to the transmission channel 60 by at least two methods.

In the first method all bitstreams are multiplexed together by multiplexor 50 with different priority identifiers, e.g., the base layer bitstream is guaranteed,
10 enhancement bitstream layer 1 provided by enhancement layer encoder 40 is given a higher priority than enhancement bitstream layer 2. The prioritization is continued until all N (wherein N is an integer from 0, 1, 2, . . .) of the bitstreams layers are prioritized. Logic in the encoding layers 30 or 40 in negotiation with the network and intermediated devices determine the number N of bitstream layers to be generated.

15 The number of bitstream layers generated is a function of the total possible bandwidth of the transmission channel 60, i.e. Ethernet, LAN, or WAN connections (this list is not intended to exhaustive but only representation of potential limiting devices and/or equipment), and the network and other intermediate devices. The number of bitstream layers M (wherein M is an integer and $M \leq N$) reaching the
20 destination point 100 can be further limited by not just the physical constraints of the intermediate devices but the congestion on the network, thereby necessitating the dropping of bitstream layers according to their priority.

In a second method the server 50 knows the transmission channel 60 condition, i.e. congestion and other physical constraints, and selectively sends the bitstreams to
25 the channel according to the priority identifiers. In either case, the destination point 100 receives the bitstream for the base layer and M bitstreams for the enhancement layer, where $M \leq N$.

The bitstreams M are sent to the base layer 90 and enhancement layer 80 decoders after being demultiplexed by demultiplexor 70. The decoded enhancement
30 information from the enhancement layer decoder is passed to the base layer decoder to composite the reconstructed video output 100. The decoding of the multiplexed

bitstreams are accomplished pursuant to the methods and algorithms depicted in flow diagrams 1100 - 1400 of Figs. 11 - 14, respectively.

The base layer encoder and decoder are capable of performing logic pursuant to the MPEG-1, MPEG-2, or MPEG-4 (Version-1) standards that are hereby
5 incorporated by reference into this disclosure.

Taking Residue with Probability Distribution Preserved

A detailed description of the probability distribution residue will now be made with reference to Figs 2A - 3B

10 In the current MPEG-2 signal-to-noise ratio (SNR) scalability extension, a residue or difference is taken between the original DCT coefficient and the quantized DCT coefficient. Fig. 2A illustrates the distribution of a residual signal as a DCT coefficient. In taking the residue small values have higher probabilities and large values have smaller probabilities. The intervals along the horizontal axis represent
15 quantization bins. The dot in the center of each interval represents the quantized DCT coefficient. Taking the residue between the original and the quantized DCT coefficient is equivalent to moving the origin to the quantization point.

Therefore, the probability distribution of the residue becomes that as shown in Figure 2B. The residue from the positive side of Fig. 2A has a higher probability of
20 being negative than positive and the residue taken from the negative side of the Fig. 2A has a higher probability of being positive than negative. The result is that the probability distribution of the residue becomes almost uniform. Thus making coding the residue more difficult.

A vastly superior method is to generate a difference between the original and
25 the lower boundary points of the quantized interval as shown in Fig. 3A and Fig. 3B. In this method, the residue is taken from the positive side of Fig. 2A remains positive and the residue from the negative side of Fig. 2A remains negative. Taking the residue is equivalent to moving the origin to the reference point as illustrated in Fig. 3A. Thus, the probability of the residue becomes as shown in Fig. 3B. This method preserves the
30 shape of the original non-uniform distribution. Although the dynamic range of the residue taken in such a manner seems to be twice of that depicted in Fig. 2B, there is no longer a need to code the sign, i.e. - or +, of the residue. The sign of the residue is

encoded in the base layer bitstream corresponding the enhancement layer, therefore this redundancy is eliminated and bits representing the sign are thus saved. Therefore, there is only a need to code the magnitude that still has a nonuniform distribution.

5 **Bit plane coding of residual DCT coefficients**

After taking residues of all the DCT coefficients in an 8 x 8 block, bit plane coding is used to code the residue. In bit-plane coding method the bit-plane coding method considers each residual DCT coefficient as a binary number of several bits instead of as a decimal integer of a certain value as in the run-level coding method. The bit-plane coding method in the present invention only replaces runlevel coding part. Therefore, all the other syntax elements remain the same.

10 An example of and description of the bit-plane coding method will now be made, wherein 64 residual DCT coefficients for an Inter-block and 63 residual DCT coefficients for an Intra-block (excluding the Intra-DC component that is coded using a separate method) are utilized for the example. The 64 (or 63) residual DCT
15 coefficients are ordered into a one-dimensional array and at least one of the residual coefficients is non-zero. The bit-plane coding method then performs the following steps.

20 The maximum value of all the residual DCT coefficients in a frame is determined and the minimum number of bits, N, needed to represent the maximum value in the binary format is also determined. N is the number of bitplanes layers for this frame and is coded in the frame header.

25 Within each 8 x 8 block is represent every one of the 64 (or 63) residual DCT coefficients with N bits in the binary format and there is formed N bit-planes or layers or levels. A bit-plane is defined as an array of 64 (or 63) bits, taken one from each residual DCT coefficient at the same significant position.

30 The most significant bit-plane is determined with at least one non-zero bit and then the number of all-zero bit-planes between the most significant bit-plane determined and the Nth one is coded. Then starting from the most significant bit plane (MSB plane), 2-D symbols are formed of two components: (a) number of consecutive O's before a I (RUN), (b) whether there are any I's left on this bit plane, i.e. End-Of-Plane (EOP). If a bit-plane after the MSB plane contains all O's, a special symbol

ALL-ZERO is formed to represent an all-zero bit-plane. Note that the MSB plane does not have the all-zero case because any all-zero bit-planes before the MSB plane have been coded in the previous steps.

5 Four 2-D VLC tables are used, wherein the table VT-C-Table-0 corresponds to the MSB plane; table VLC-Table-1 corresponds to the second MSB plane; table VLC-Table-2 corresponds to the third MSB plane; and table VLC-Table-3 corresponds to the fourth MSB and all the lower bit planes. For the ESCAPE cases, RUN is coded with 6 bits, EOP is coded with 1 bit. Escape coding is a method to code very small probability events which are not in the coding tables individually.

10 An example of the above process will now follow. For illustration purposes, we will assume that the residual values after the zigzag ordering are given as follows and N = 6: The following representation is thereby produced.

10, 0, 6, 0, 0, 3, 0, 2, 2, 0, 0, 2, 0, 0, 1, 0, ... 0, 0

15

The maximum value in this block is found to be 10 and the minimum number of bits to represent 10 in the binary format (1010) is 4. Therefore, two all-zero bit-planes before the MSB plane are coded with a code for the value 2 and the remaining 4 bit-planes are coded using the (RUN, EOP) codes. Writing every value in the binary format using 4 bits, the 4 bit-planes are formed as follows:

20

1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 (MSB-plane)

0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 (Second MSB-plane)

1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0 (Third MSB-plane)

25

0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0 (Fourth MSB-plane or LSB-plane)

Converting the bits of each bit-plane into (RUN, EOP) symbols results in the following:

30

(0, 1) (MSB-plane)

(2, 1) (Second MSB-plane)

(0, 0), (1,0), (2,0), (1,0), (0, 0), (2, 1) (Third MSB-plane)

(5, 0), (8, 1)

(Fourth MSB-plane or LSB-plane)

Therefore, there are 10 symbols to be coded using the (RUN, EOP) VLC tables. Based on their locations in the bit-planes, different VLC tables are used for the coding. The enhancement bitstream using all four bitplanes looks as follows:

5 code leading-all-zero(2)
code msb(0, 1)
code msb-1(2,1)
code-msb-2(0,0), code_msb-2(1,0), code-msb-2(2,0), code-msb-2(1,0), code-msb-2(0,0), code-msb-2(2, 1) code_msb-3(5,0), code_msb-3(8, 1).

10

In an alternative embodiment, several enhancement bitstreams may be formed from the four bit-planes, in this example from the respective sets comprising one or more of the four bit-planes.

15 Motion Vector Sharing

In this alternative embodiment of the present invention motion vector sharing is capable of being utilized when the base layer bitstream exceeds a predetermined size or more levels of scalability are needed for the enhancement layer. By lowering the number of bits required for coding the motion vectors in the base layer the bandwidth requirements of the base layer bitstream is reduced. In base layer coding, a macroblock (16 x 16 pixels for the luminance component and W pixels for each chroma-luminance components) of the current frame is compared with the previous frame within a search range. The closest match in the previous frame is used as a prediction of the current macroblock. The relative displacement of the prediction to the current macroblock, in the horizontal and vertical directions, is called a motion vector.

20

25

The difference between the current macroblock and it's prediction is coded using the DCT coding. In order for the decoder to reconstruct the current macroblock, the motion vector has to be coded in the bitstream. Since there is a fixed number of bits for coding a frame, the more bits spent on coding the motion vectors results in fewer bits for coding the motion compensated differences. Therefore, it is desirable to lower the number of bits for coding the motion vectors and leave more bits for coding the differences between the current macroblock and its prediction.

30

For each set of 2 x 2 motion vectors, the average motion vector can be determined and used for the four macroblocks. In order to not change the syntax of the base layer coding, four macroblocks are forced to have the identical motion vectors. Since only one out four motion vectors is coded in the bitstream, the amount of bits spent on motion vector coding is reduced, therefore, there are more bits available for coding the differences. The cost for pursuing such a method is that the four macroblocks, which share the same motion vector may, not get the best matched prediction individually and the motion compensated difference may have a larger dynamic range, thus necessitating more bits to code the motion vector.

For a given fixed bitrate, the savings from coding one out of four motion vectors may not compensate the increased number of bits required to code the difference with a larger dynamic range. However, for a time varying bitrate, a wider dynamic range for the enhancement layer provides more flexibility to achieve the best possible usage of the available bandwidth.

15

Coding Sign Bits

In an alternative embodiment of the present invention, if the base layer quantized DCT coefficient is non-zero, the corresponding enhancement layer difference will have the same sign as the base layer quantized DCT. Therefore, there is no need to code the sign bit in the enhancement layer.

Conversely, if the base layer quantized DCT coefficient is zero and corresponding enhancement layer difference is non-zero, a sign bit is placed into enhancement layer bitstream immediately after the MSB of the difference. An example of the above method will now follow.

25

Difference of a DCT block after ordering

- 10, 0, 6, 0, 0, 3, 0, 2, 2, 0, 0, 2, 0, 0, 1, 0, ...0, 0

Sign indications of the DCT block after ordering

- 3, 3, 3, 3, 2, 0, 3, 3, 1, 2, 2, 0, 3, 3, 1, 2, ... 2, 3

- 30
- 0: base layer quantized DCT coefficient = 0 and difference >0
 - 1: base layer quantized DCT coefficient = 0 and difference <0
 - 2: base layer quantized DCT coefficient = 0 and difference =0

- 3: base layer quantized DCT coefficient = 0.

In this example, the sign bits associated with values 10, 6, 2 don't need to be coded and the sign bits associated with 3, 2, 2, 1 are coded in the following way:

Code(All Zero)

5 code (All Zero)

code(0,1)

code(2,1)

code(0,0),code(1,0),code(2,0),0,code(1,0),code(0,0),1,code(2,1),0

code(5,0),code(8,1),1

10 For every DCT difference, there is a sign indication associated with it. There are four possible cases. In the above coding 0, 1, 2, and 3 are used to denote the four cases. If the sign indication is 2 or 3, the sign bit does not have to be coded because it is either associated with a zero difference or available from the corresponding base layer data. If the sign indication is 0 or 1 a sign bit code is required once per difference
15 value, i.e. not every bit-plane of the difference value. Therefore, a sign bit is put immediately after the most significant bit of the difference.

Optimal Reconstruction of the DCT Coefficients

20 In an alternative embodiment of the present invention, even though N enhancement bitstream layers or planes may have been generated, only M, wherein $M \leq N$ enhancement layer bits are available for reconstruction of the DCT coefficients due to the channel capacity, and other constraints such as congestion among others, the decoder 80 of Fig. 1 may receive no enhancement difference or only a partial enhancement difference. In such a case, the optimal reconstruction of the DCT
25 coefficients is capable of proceeding along the following method:

If decoded difference = 0, the reconstruction point is the same as that in base layer, otherwise, the reconstructed difference = decoded difference + $\frac{1}{4}$ * $(1 \ll \text{decoded_bit_plane})$ and the reconstruction point = reference point + reconstructed difference * $Q_{\text{enh}} + Q_{\text{enh}}/2$.

30 In the present embodiment, referring to Figs. 3C and 3D, the optimal reconstruction point is not the lower boundary of a quantization bin. The above method specifies how to obtain the optimal reconstruction point in cases where the

difference is quantized and received partially, i.e. not all of the enhancement layers generated are either transmitted or received as shown in Fig. 1. wherein $M \leq N$.

What is claimed is:

1. A video encoding method for adapting a video input to a bandwidth of a transmission channel of a network, the method comprising the steps of:
determining number N of enhancement layer bitstreams capable of being adapted to said bandwidth of said transmission channel of said network;
encoding a base layer bitstream from said video input;
encoding N number of enhancement layer bitstreams from said video input based on the base layer bitstream, wherein the N enhancement layer bitstreams complements the base layer bitstream; and
providing the base layer bitstream and N enhancement layer bitstreams to said network.
2. The video encoding method according to claim 1, wherein the determining step includes negotiating with intermediate devices on said network.
3. The video encoding method according to claim 2, wherein negotiating includes determining destination resources.
4. The video encoding method according to claim 1, wherein the step of encoding the base layer bitstreams is performed by a MPEG-1 encoding method.
5. The video encoding method according to claim 1, wherein the step of encoding the base layer bitstreams is performed by a MPEG-2 encoding method.
6. The video encoding method according to claim 1, wherein the step of encoding the base layer bitstreams is performed by a MPEG-4 encoding method.

7. The video encoding method according to claim 1, wherein the step of encoding the base layer bitstreams is performed by a Discrete Cosine Transform (DCT) method.
8. The video encoding method according to claim 7, wherein after encoding the base layer bitstreams by a Discrete Cosine Transform (DCT) method a DCT coefficient is quantized.
9. The video encoding method according to claim 1, wherein the enhancement layer bitstreams are based on the difference of an original base layer DCT coefficient and a corresponding base layer quantized DCT coefficient.
10. The video encoding method according to claim 1, wherein the base layer bitstream and the N enhancement layer provide to the network are multiplexed.
11. A video decoding method for adapting a video input to a bandwidth of a transmission channel of a network, the method comprising the steps of:
 - determining number M of enhancement layer bitstreams of said video input capable of being received from said transmission channel of said network;
 - decoding a base layer bitstream from received video input;
 - decoding M number of enhancement layer bitstreams from the received video input based on the base layer bitstream, wherein the M received enhancement layer bitstreams complements the base layer bitstream;
 - and
 - reconstructing the base layer bitstream and N enhancement layer bitstreams.
12. The video decoding method according to claim 11, wherein the determining step includes negotiating with intermediate devices on said network.

13. The video decoding method according to claim 12, wherein negotiating includes determining destination resources.
14. The video decoding method according to claim 11, wherein the step of decoding the base layer bitstreams is performed by a MPEG-1 decoding method.
15. The video decoding method according to claim 11, wherein the step of decoding the base layer bitstreams is performed by a MPEG-2 decoding method.
16. The video decoding method according to claim 11, wherein the step of decoding the base layer bitstreams is performed by a MPEG-4 decoding method.
17. The video decoding method according to claim 11, wherein the step of decoding the base layer bitstreams is performed by a Discrete Cosine Transform (DCT) method.
18. The video decoding method according to claim 17, wherein after decoding the base layer bitstreams by a Discrete Cosine Transform (DCT) method a DCT coefficient is unquantized.
19. The video decoding method according to claim 11, wherein coding of the enhancement layer bitstreams are based on the difference of an original base layer DCT coefficient and a corresponding base layer quantized DCT coefficient.
20. The video decoding method according to claim 11, wherein the base layer bitstream and the M enhancement layers to be reconstructed are de-multiplexed.

21. A video decoding method for adapting a video input to a bandwidth of a receiving apparatus, the method comprising the steps of:
demultiplexing a base layer bitstream and at least one of a plurality of enhancement layer bitstreams received from a network;
decoding the base layer bitstream;
decoding at least one of the plurality of enhancement layer bitstreams based on generated base layer bitstream, wherein the at least one of the plurality of enhancement layer bitstreams enhances the base layer bitstream; and
reconstructing a video output.
22. A video encoding method for encoding enhancement layers based on a base layer bitstream encoded from a video input, the video encoding method comprising the steps of:
taking a difference between an original DCT coefficient and a reference point;
and
dividing the difference between the original DCT coefficient and the reference point into N bit-planes.
23. The video encoding method according to claim 22, wherein RUN and EOP symbols represents the N bit-planes of a DCT block.
24. The video encoding method according to claim 23, wherein the RUN and EOP symbols are encoded.
25. The video encoding method according to claim 24, wherein a sign bit is encoded if the DCT difference is equal to zero or the sign of the DCT difference is the same as the corresponding base layer bitstream data.

26. A video decoding method for reconstructing DCT coefficients M enhancement layers of N enhancement layers have been received, wherein $M \leq N$, comprising:
- means for taking a reconstruction difference as a decoded difference and a portion of a decoded bit-plane;
 - means for taking a reconstruction point as a reference point and a reconstructed difference; and
- determining an optimal reconstruction point.
27. A method of coding motion vectors of a plurality of macroblocks, the method comprising the steps of:
- determining an average motion vector from N motion vectors for N macroblocks;
 - utilizing the determined average motion vector as the motion vector for the N macroblocks; and
 - encoding $1/N$ motion vectors in a base layer bitstream.

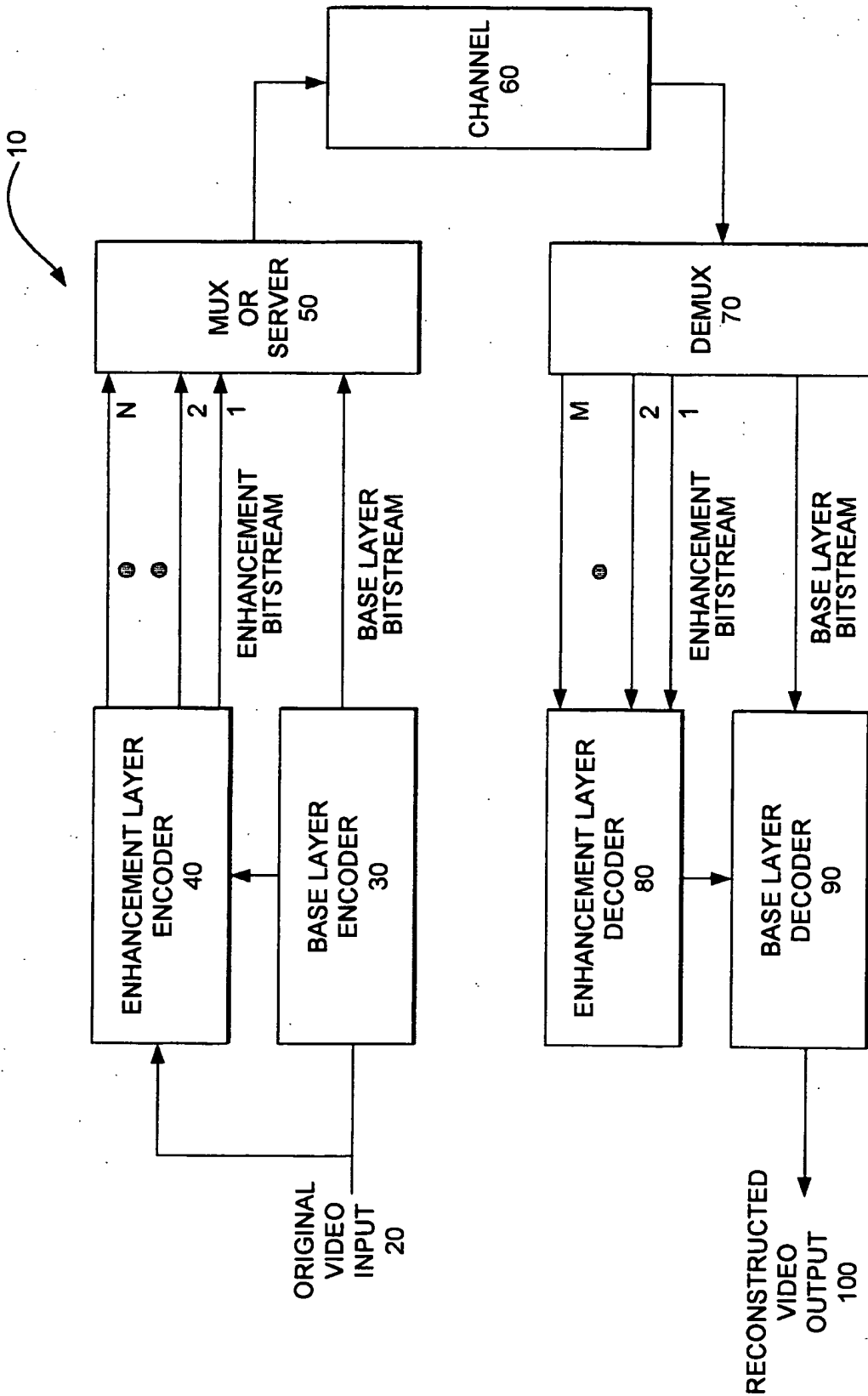


FIG. 1