(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
10 December 2009 (10.12.2009)

PCT

(10) International Publication Number
WO 2009/148957 A2

(51) International Patent Classification:
G06F 21/00 (2006.01) *G06F 21/24* (2006.01)

(21) International Application Number:
PCT/US2009/045670

(22) International Filing Date:
29 May 2009 (29.05.2009)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/133,354 4 June 2008 (04.06.2008) US

(71) Applicant *(for all designated States except US)*: **MICROSOFT CORPORATION** [US/US]; One Microsoft Way, Redmond, WA 98052-6399 (US).

(72) Inventors: **DUBHASHI, Kedarnath, A.**; C/o Microsoft Corporation International Patents, One Microsoft Way, Redmond, WA 98052-6399 (US). **BARDE, Sumedh, N.**; c/o Microsoft Corporation International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **FARAG, Hany**; c/o Microsoft Corporation International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

(54) **Title**: TRANSLATING DRM SYSTEM REQUIREMENTS

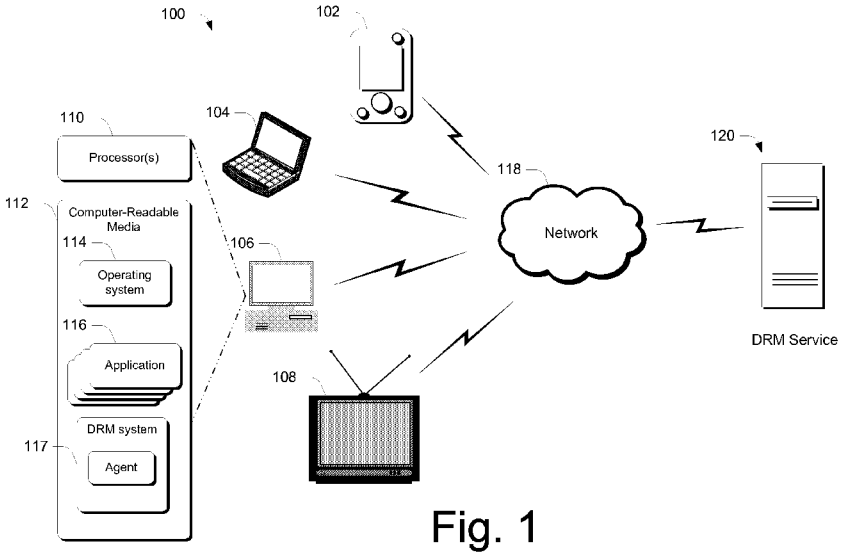

Fig. 1

(57) **Abstract**: Various embodiments provide a mapping layer to translate DRM system requirements from one DRM system, such as a source system, to another DRM system, such as a target system. In at least some embodiments, DRM system requirement translation is performed using a signed data structure that maps DRM system requirements from one DRM system to one or more other DRM systems. By mapping DRM system requirements from one system to another, licenses associated with DRM-protected content and associated content can be safely transferred between systems.

WO 2009/148957 A2

Translating DRM System Requirements

Background

[0001] Digital rights management (DRM) interoperability can be an important issue when DRM-protected content and licenses are moved from one DRM system to another. Specifically, if DRM protections from one system to another are not compatible in terms of the protections they apply, or if DRM system compatibility is unknown, then content and associated licenses may not be able to be easily transferred to and/or consumed by another DRM system. This can be particularly problematic for an individual user who wishes to transfer licenses and content from one of their devices to another.

Summary

[0002] This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter.

[0003] In one or more embodiments, a mapping layer is provided to translate DRM system requirements from one DRM system, such as a source DRM system, to another DRM system, such as a target DRM system. In at least some embodiments, DRM system requirement translation is performed using a signed data structure that maps DRM system requirements from one DRM system to one or more other DRM systems.

[0004] By mapping DRM system requirements from one system to another, licenses associated with DRM-protected content and associated content can be safely transferred between systems.

Brief Description of the Drawings

[0005] The same numbers are used throughout the drawings to reference like features.

[0006] Fig. 1 illustrates an operating environment in accordance with one or more embodiments.

[0007] Fig. 2 illustrates an example data structure in accordance with one or more embodiments.

[0008] Fig. 3 illustrates a system in accordance with one or more embodiments.

[0009] Fig. 4 is a flow diagram that describes steps in a method in accordance with one or more embodiments.

[0010] Fig. 5 is a block diagram of an example system that can be utilized to implement one or more embodiments.

Detailed Description

Overview

[0011] In one or more embodiments, a mapping layer is provided to translate DRM system requirements from one DRM system, such as a source DRM system, to another DRM system, such as a target DRM system. In at least some embodiments, DRM system requirement translation is performed using a signed data structure that maps DRM system requirements from one DRM system to one or more other DRM systems.

[0012] By mapping DRM system requirements from one system to another, licenses associated with DRM-protected content and associated content can be safely transferred between systems.

[0013] In the discussion that follows, a section entitled "Operating Environment" describes but one operating environment that can be utilized to practice the inventive principles described herein in accordance with one or more embodiments. Following this, a section entitled "Example Data Structure" describes an example data structure in accordance with one or more embodiments. Next, a section entitled " Using a Data Structure to Map DRM System Requirements" is provided and describes an example system that can be utilized to map DRM system requirements in accordance with one or more embodiments. Following this, a section entitled "Example Method" describes an example method in accordance with one or more embodiments. Last, a section entitled "Example System" describes an example system that can be utilized to implement one or more embodiments.

Operating Environment

[0014] Fig. 1 illustrates an operating environment in accordance with one or more embodiments, generally at 100. Operating environment 100 includes multiple different computing devices, examples of which are shown at 102, 104, 106, and 108. Individual computing devices or components associated therewith can typically include one or more processors 110, one or more computer-readable media 112, an operating system 114 and one or more applications 116 that reside on the computer-readable media and which are executable by the processor(s). Such applications can include, by way of example and not limitation, a media playing

application or any other type of application that can enable distributed content to be consumed by a user. The computing devices can also include a digital rights management (DRM) system 117 that includes a DRM agent. The DRM agent is typically responsible for enforcing policy that is described in a particular license associated with content that can be received and consumed by the computing device.

[0015] The computer-readable media can include, by way of example and not limitation, all forms of volatile and non-volatile memory and/or storage media that are typically associated with a computing device. Such media can include ROM, RAM, flash memory, hard disk, removable media and the like.

[0016] In addition, in at least some embodiments, environment 100 includes a network 118, such as a local network or the Internet. The network can be utilized to receive DRM-protected content and associated licenses.

[0017] Operating environment 100 also includes, in at least some embodiments, an entity that provides a data structure that can be utilized to map DRM system requirements. In this particular example, such entity takes the form of a trusted DRM service or server 120 that can generate and provide a data structure that can be utilized by the various computing devices to translate DRM system requirements from one DRM system to another, as will become apparent below.

[0018] The computing devices can be embodied as any suitable computing device such as, by way of example and not limitation, a desktop computer (such as computing device 106), a portable computer (such as computing device 104), a handheld computer such as a personal digital assistant (such as computing device 102), a television that may or may not include a set-top box (such as computing

device 108), and the like. One example of a computing device is shown and described below in relation to Fig. 5.

[0019] Having discussed the general notion of an example operating environment in which various embodiments can operate, consider now a discussion of how DRM system requirements can be mapped or translated between different DRM systems.

Example Data Structure

[0020] Fig. 2 illustrates an example data structure in accordance with one or more embodiments generally at 200. In the illustrated example, data structure 200 is referred to as a Revocation Information Structure or "RevInfo Structure". Revocation information version refers to a number included in DRM policy that identifies a particular version of revocation data. Revocation data refers to version numbers, certificate revocation lists, system renewability messages or other data utilized to execute revocation, as will be appreciated by the skilled artisan. Data structure 200 includes, in one or more embodiments, a revocation information version 202 and associations between various DRM systems and corresponding system version numbers. For example, in the illustrated example, association 204 defines an association between "DRM system 1" and "Version 1.0", association 206 defines an association between "DRM system 2" and "Version 3.0", and association 208 defines an association between "DRM system 3" and "Version 2.0". So, in the above example, "DRM system 1" might correspond to Microsoft's PlayReady DRM system, "DRM system 2" might correspond to RealNetworks' Helix DRM system and so on.

[0021] In this example, for a particular revocation information version number 202, the version numbers described in the associations 204, 206 and 208 are seen to comply with or otherwise satisfy system requirements associated with the revocation information version number. Hence, the structure 200 provides a mapping layer that can translate DRM system requirements from one DRM system to another, such as a DRM system that might be present on a source DRM system and a DRM system that might be present on a target transferee, such as a different target DRM system.

[0022] In the illustrated and described embodiment, data structure 200 can be signed and issued to a client computing device by a DRM service, such as DRM service 120 in Fig. 1. Once issued, the client computing device can utilize data structure 200 to ensure that computing devices or target transferees to which DRM-protected content and licenses may be transferred have DRM systems that are compliant with requirements established by the DRM service.

[0023] Having considered an example data structure, consider now a discussion of how the data structure can be employed in accordance with one or more embodiments.

Using a Data Structure to Map DRM System Requirements

[0024] Fig. 3 illustrates a system in accordance with one or more embodiments generally at 300. System 300 includes, in this example, a DRM service 302, a first client computing device 304 having a first DRM system, designated "DRM System 1", and a target transferee in the form of a second client computing device 306 having a second different DRM system, designated "DRM System 2".

[0025] In operation, in at least some embodiments, DRM service 302 generates, signs and issues a RevInfo structure 308. Structure 308 includes a RIV number and

various associations between DRM systems and system version numbers as described above. In the illustrated and described embodiment, client computing device 304 receives RevInfo structure 308. The RevInfo structure 308 is processed by a DRM agent executing on client computing device 304. Part of this processing can include validating the signature on the RevInfo structure. If the signature validates, then the DRM agent can assume that the various associations described in the RevInfo structure are valid.

[0026] Assume now that client computing device 304 receives a license 310 associated with DRM-protected content. In this example, license 310 can include various policies associated with the DRM-protected content, a RIV number for DRM system 1, and a content key. When the DRM agent receives license 310, the DRM agent evaluates the license by ascertaining the license's RIV number and checking the license's RIV number against the RIV number contained in RevInfo structure 308. If the license contains a valid RIV number, and DRM System 1 has a version that is the same as or newer than one described in an association appearing in RevInfo structure 308, client computing device 304 can use the license to consume the associated DRM-protected content.

[0027] Assume now that a user of client computing device 304 wishes to transfer the DRM-protected content to client computing device 306. In this case, the DRM agent executing on client computing device 304 queries the DRM system on client computing device 306 to ascertain its version number. If client computing device 306's DRM version number is contained in an association described in RevInfo structure 308 (or, in at least some embodiments, newer than the one described in the RevInfo structure), then the license can be transferred to client computing device

306. If, on the other hand, the DRM version number of the DRM system executing on client device 306 does not match or otherwise favorably compare with a version number described in RevInfo structure 308, the license is not transferred.

[0028] In this manner, DRM licenses can be transferred from one system to another in situations where the systems are executing different DRM systems. License transfer is made possible, in this example, through the use of the signed RevInfo structure 308 that was received by client computing device 304.

[0029] Having considered an example use of a data structure in accordance with one or more embodiments, consider now a method that can be implemented using the systems described above.

Example Method

[0030] Fig. 4 is a flow diagram that describes steps in a method in accordance with one or more embodiments. The method can be implemented in connection with any suitable hardware, software, firmware, or combination thereof. In at least some embodiments, aspects of the method can be implemented by a suitably-configured DRM service. As such, those aspects are designated "DRM Service". Likewise, aspects of the method can be implemented by a suitably-configured client computing device. As such, those aspects are designated "Client Computing Device".

[0031] Step 400 generates an RevInfo structure. Examples of a suitable RevInfo structure are provided above. Step 402 signs the RevInfo structure. This step can be accomplished by signing the structure with a private key associated with the DRM service. Step 404 sends the RevInfo structure to a client computing device.

[0032] Step 406 receives the RevInfo structure. After receiving the RevInfo structure, the client computing device or, more accurately, a suitably-configured DRM agent executing on the client computing device can verify the authenticity of the RevInfo structure. This can be done, for example, by using a public key associated with the private key that was used to sign the RevInfo structure. Step 408 receives a request to transfer a license associated with DRM-protected content. This step can be performed in any suitable way. For example, a user operating the client computing device may wish to transfer DRM-protected content to another computing device or to another application executing on the current client computing device.

[0033] Step 410 ascertains a version number of a DRM system on a target transferee. In the illustrated and described embodiment, a target transferee can include by way of example and not limitation, another computing device or an application executing on the current computing device. Step 412 ascertains whether the version number of the target transferee is equal to or newer than a version number contained in the RevInfo structure. The step can be performed by comparing the version number of the target transferee with a version number contained in an association or mapping described in the RevInfo structure. If the version number of the target transferee is equal to or newer than a version number in the RevInfo structure, then step 414 transfers the license to the target transferee. If, on the other hand, the version number of the target transferee is not equal to or newer than a version number in the RevInfo structure, step 416 does not transfer the license.

[0034] Accordingly, licenses can be transferred between systems that employ different DRM systems. In at least some embodiments, the license generated for the target transferee states its version number as mapped using the data structure for the source DRM system. By using mappings that are described in the RevInfo structure, DRM system requirements can be confirmed to meet with requirements promulgated by the DRM service.

[0035] Having discussed the notion of translating DRM system requirements from one DRM system to another, consider now a discussion of an example system that can be utilized to implement one or more embodiments.

Example System

[0036] Fig. 5 illustrates an example computing device 500 that can implement the various embodiments described above. Computing device 500 can be, for example, various computing devices or servers, such as those illustrated in Fig. 1 or any other suitable computing device.

[0037] Computing device 500 includes one or more processors or processing units 502, one or more memory and/or storage components 504, one or more input/output (I/O) devices 506, and a bus 508 that allows the various components and devices to communicate with one another. Bus 508 represents one or more of any of several types of bus structures, including a memory bus or memory controller, a peripheral bus, an accelerated graphics port, and a processor or local bus using any of a variety of bus architectures. Bus 508 can include wired and/or wireless buses.

[0038] Memory/storage component 504 represents one or more computer storage media. Component 504 can include volatile media (such as random access memory (RAM)) and/or nonvolatile media (such as read only memory (ROM), Flash memory, optical disks, magnetic disks, and so forth). Component 504 can include fixed media (e.g., RAM, ROM, a fixed hard drive, etc.) as well as removable media (e.g., a Flash memory drive, a removable hard drive, an optical disk, and so forth).

[0039] One or more input/output devices 506 allow a user to enter commands and information to computing device 500, and also allow information to be presented to the user and/or other components or devices. Examples of input devices include a keyboard, a cursor control device (e.g., a mouse), a microphone, a scanner, and so forth. Examples of output devices include a display device (e.g., a monitor or projector), speakers, a printer, a network card, and so forth.

[0040] Various techniques may be described herein in the general context of software or program modules. Generally, software includes routines, programs, objects, components, data structures, and so forth that perform particular tasks or implement particular abstract data types. An implementation of these modules and techniques may be stored on or transmitted across some form of computer readable media. Computer readable media can be any available medium or media that can be accessed by a computing device. By way of example, and not limitation, computer readable media may comprise "computer storage media".

[0041] "Computer storage media" include volatile and non-volatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules, or other data. Computer storage media include, but are not limited to,

RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by a computer.

Conclusion

[0042] In one or more embodiments, a mapping layer is provided to translate DRM system requirements from one DRM system, such as a source system, to another DRM system, such as a target system. In at least some embodiments, DRM system requirement translation is performed using a signed data structure that maps DRM system requirements from one DRM system to one or more other DRM systems.

[0043] By mapping DRM system requirements from one system to another, licenses associated with DRM-protected content and associated content can be safely transferred between systems.

[0044] Although the subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that the subject matter defined in the appended claims is not necessarily limited to the specific features or acts described above. Rather, the specific features and acts described above are disclosed as example forms of implementing the claims.

Claims

What is claimed is:

1. A computer-implemented method comprising:

generating (400) a data structure that contains associations between different DRM systems and corresponding system version numbers; and

sending (404) the data structure to a client computing device.

2. The method of claim 1 further comprising prior to said sending, signing the data structure.

3. The method of claim 1 further comprising prior to said sending, signing the data structure, wherein said signing is performed by using a private key associated with a trusted DRM service.

4. The method of claim 1, wherein the data structure includes a revocation information version.

5. The method of claim 1, wherein the data structure is configured to enable licenses to be transferred between different computing devices executing different DRM systems.

6. The method of claim 1, wherein the data structure includes a revocation information version, and further comprising prior to said sending, signing the data structure.

7. A computer-implemented method comprising:

receiving (406) a data structure that contains associations between different DRM systems and corresponding system version numbers; and

using (414) the data structure to transfer a license from one computing device to a target transferee,

wherein transferring the license is based, at least in part, on a comparison of a version number associated with the target transferee and a version number contained in an association in the data structure.

8. The method of claim 7 further comprising, prior to said using, receiving a request to transfer the license, wherein the license is associated with DRM-protected content.

9. The method of claim 7 further comprising after receiving said data structure, verifying the data structure's authenticity.

10. The method of claim 7 further comprising after receiving said data structure, verifying the data structure's authenticity using a public key associated with a DRM service from which the data structure was received.

11. The method of claim 7 further comprising, after said receiving, ascertaining a version number of a DRM system on the target transferee.

12. The method of claim 7, wherein said receiving is performed by a first computing device executing a first DRM system, and said target transferee comprises a second different computing device executing a second different DRM system.

13. The method of claim 7 further comprising:

after said receiving, verifying the data structure's authenticity; and

receiving a request to transfer the license, wherein the license is associated with DRM-protected content.

14. The method of claim 7 further comprising:

after said receiving, verifying the data structure's authenticity;

receiving a request to transfer the license, wherein the license is associated with DRM-protected content; and

ascertaining a version number of a DRM system on the target transferee.

15. The method of claim 14, wherein said receiving a data structure is performed by a first computing device executing a first DRM system, and said target transferee comprises a second different computing device executing a second different DRM system, and wherein the license for the target transferee includes a version number for the second different DRM system.

16. One or more computer-readable storage media (112) embodying a data structure (200) containing data that can enable licenses to be transferred between systems executing different DRM systems.

17. The one or more computer-readable storage media of claim 16, wherein said data comprises a revocation information version.

18. The one or more computer-readable storage media of claim 16, wherein said data comprises associations between various DRM systems and system version numbers.

19. The one or more computer-readable storage media of claim 16, wherein said data comprises a revocation information version and associations between various DRM systems and system version numbers.

20. The one or more computer-readable storage media of claim 16, wherein said data structure comprises a signed data structure.

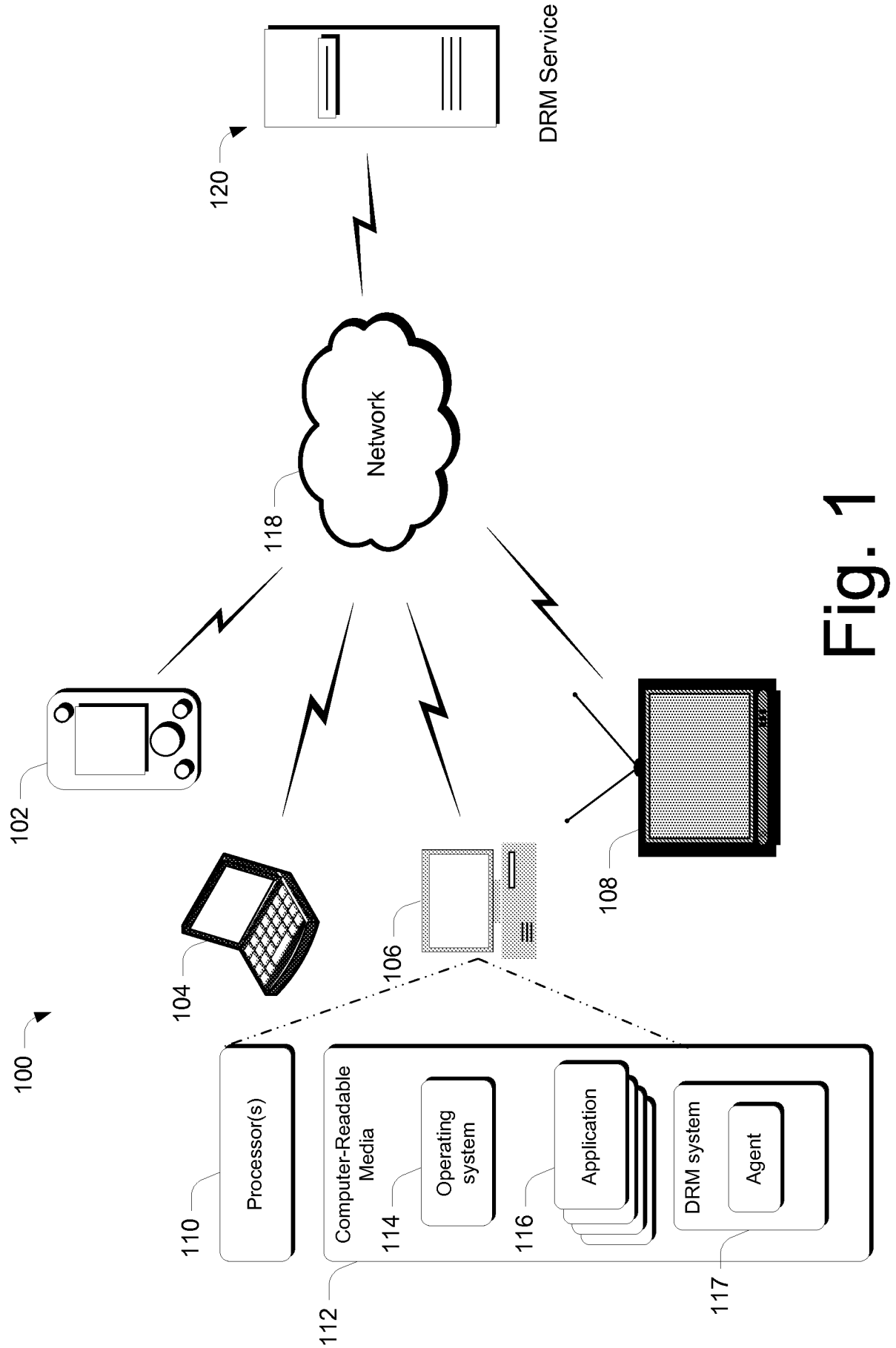
DRM Service
120
Network
118
102
104
106
108
100
Processor(s)
110
Computer-Readable Media
112
Operating system
114
Application
116
DRM system
Agent
117

Fig. 1

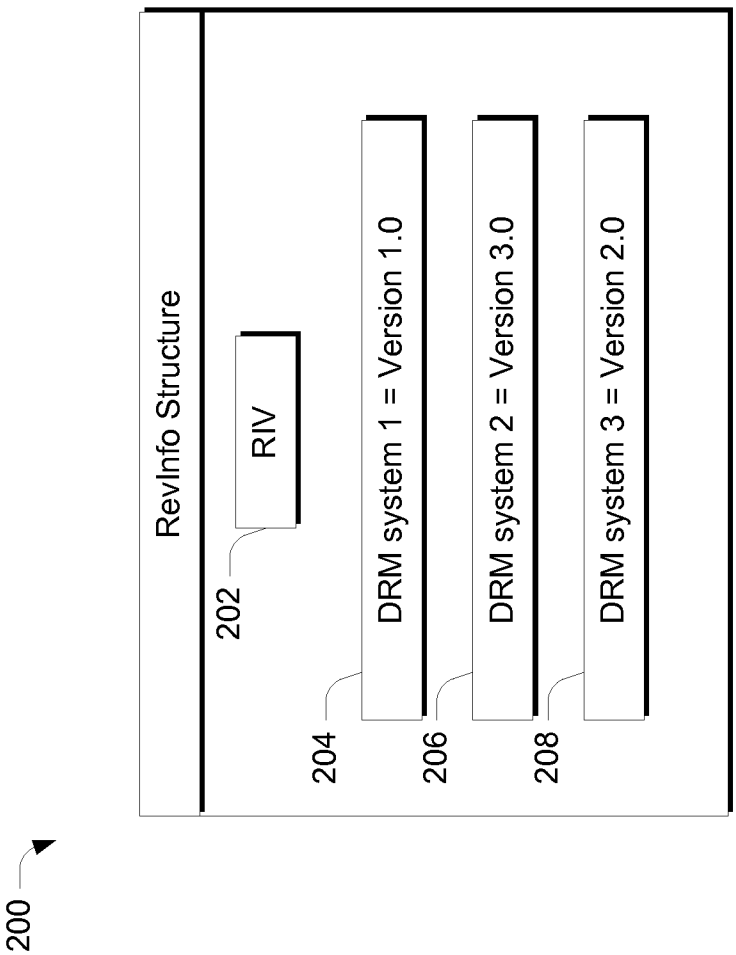
200
RevInfo Structure
RIV
202
DRM system 1 = Version 1.0
204
DRM system 2 = Version 3.0
206
DRM system 3 = Version 2.0
208

Fig. 2

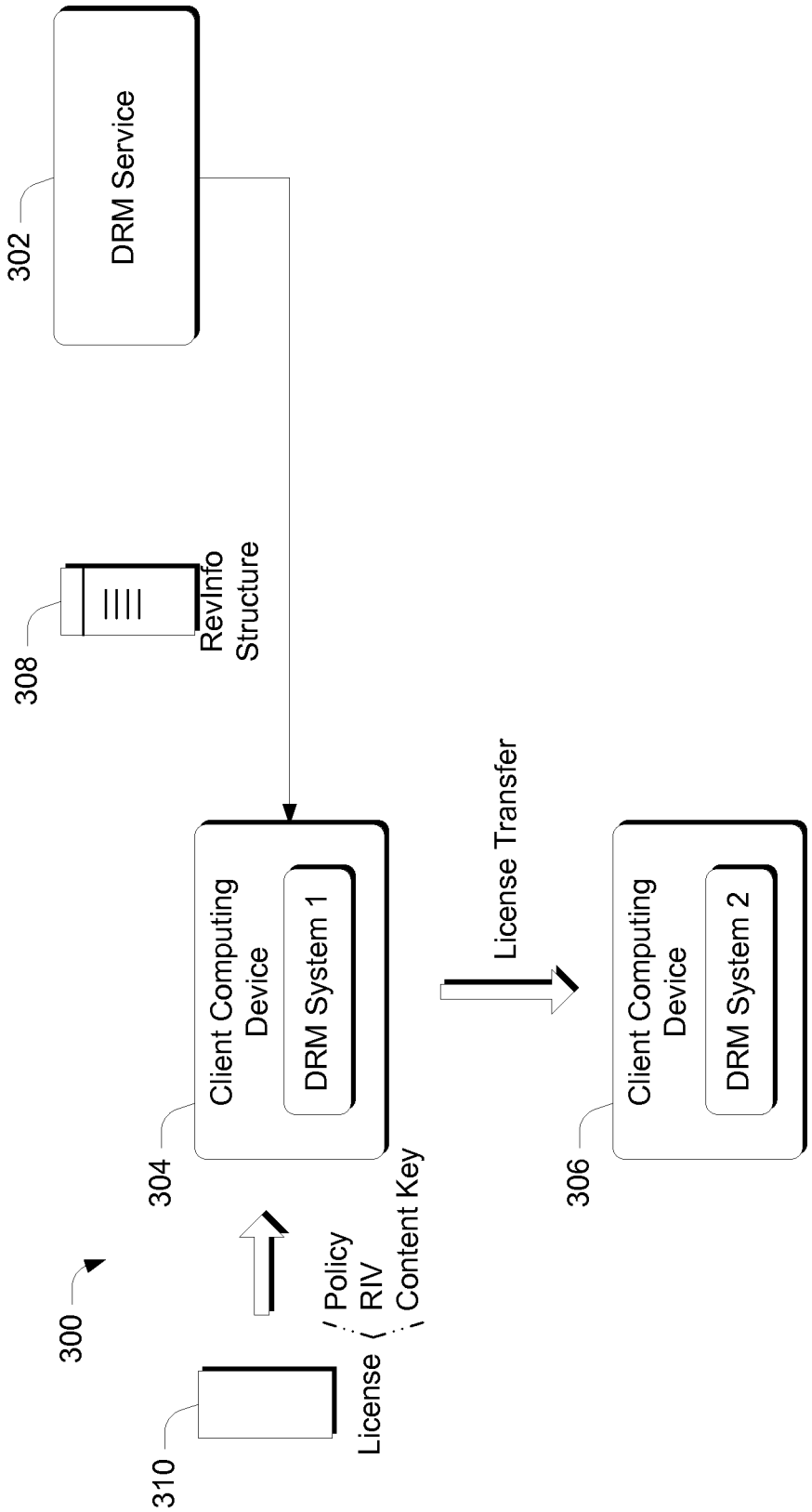
DRM Service
302
RevInfo Structure
308
Client Computing Device
DRM System 1
304
License Transfer
Client Computing Device
DRM System 2
306
300
Policy
RIV
Content Key
License
310

Fig. 3

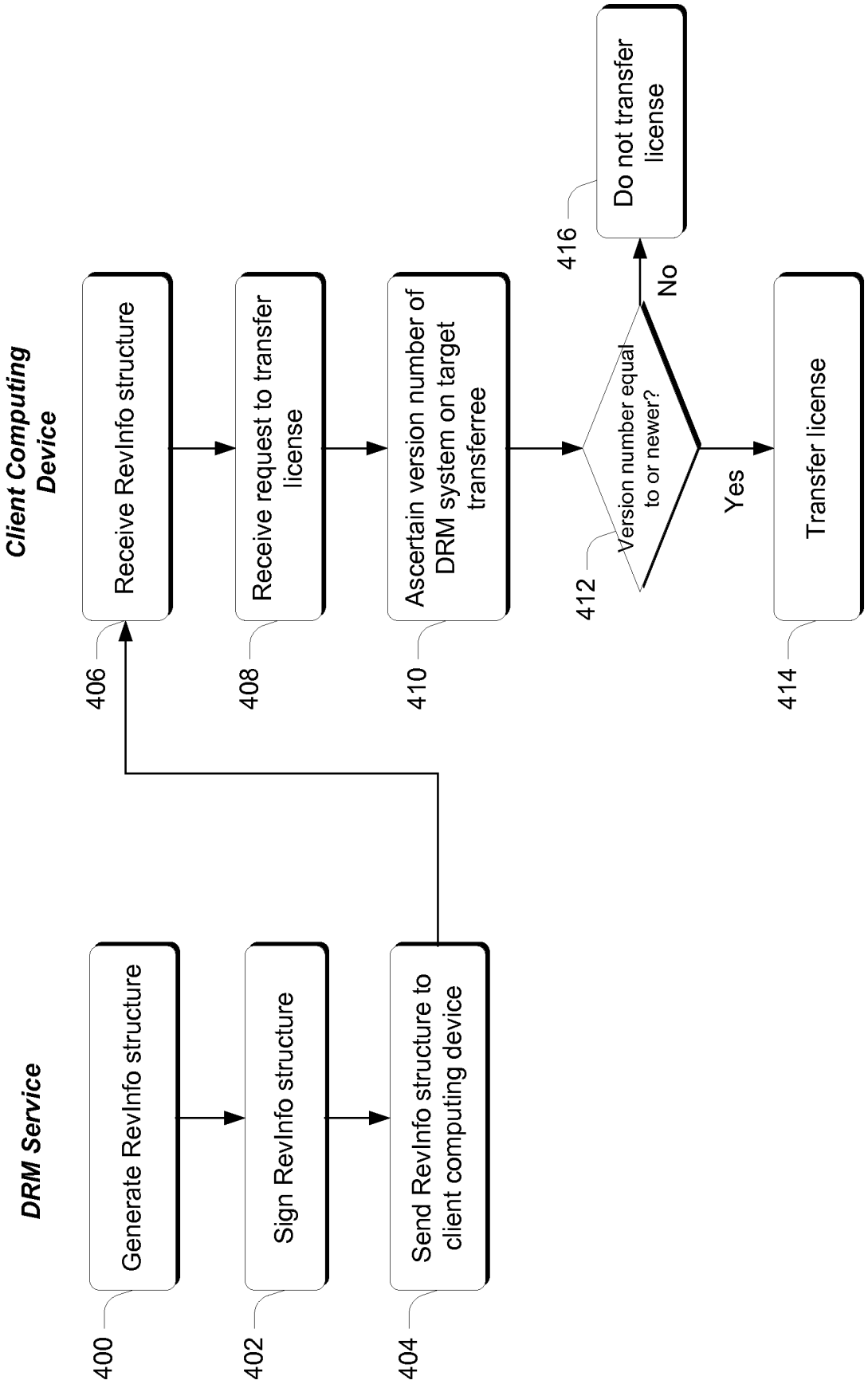
DRM Service
Client Computing Device
Generate RevInfo structure
Sign RevInfo structure
Send RevInfo structure to client computing device
Receive RevInfo structure
Receive request to transfer license
Ascertain version number of DRM system on target transferree
Version number equal to or newer?
Yes
No
Transfer license
Do not transfer license
400
402
404
406
408
410
412
414
416

Fig. 4

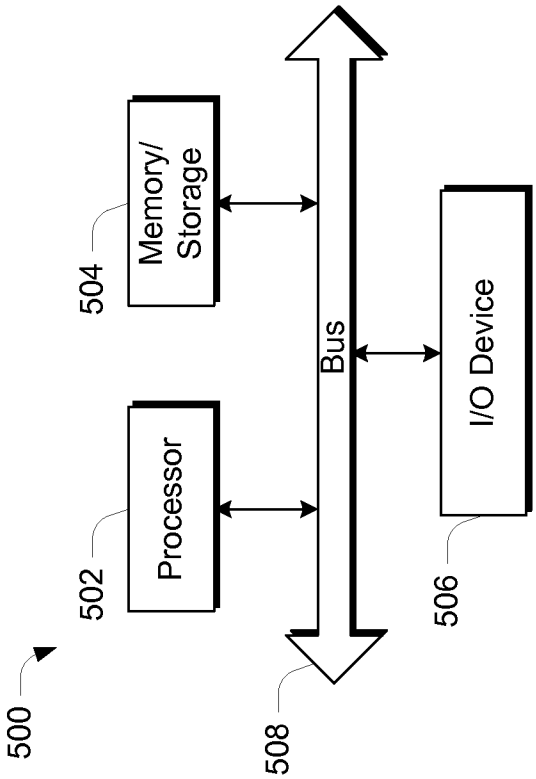

Fig. 5