



(51) International Patent Classification:
G06F 17/30 (2006.01)

(21) International Application Number:
PCT/US2014/047596

(22) International Filing Date:
22 July 2014 (22.07.2014)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
2575/CHE/2014 23 May 2014 (23.05.2014) IN

(71) Applicant: **HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.** [US/US]; 11445 Compaq Center Drive W., Houston, Texas 77070 (US).

(72) Inventors: **KARUPPUSAMY, Ramesh Kannan**; Sy. No.192, Whitefield Road, Mahadevapura Post, Bangalore, Karnataka 560048 (IN). **KANNAN, Rajkumar**; Sy. No.192, Whitefield Road, Mahadevapura Post, Bangalore, Karnataka 560048 (IN).

(74) Agents: **ORTEGA, Arthur S.** et al.; Hewlett-Packard Company, Intellectual Property Administration, 3404 E.

Harmony Road, Mail Stop 35, Fort Collins, Colorado 80528 (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, BN, BR, BW, BY, BZ, CA, CH, CL, 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, IR, IS, JP, KE, KG, 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, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, 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, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, 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, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: USING LOCATION ADDRESSED STORAGE AS CONTENT ADDRESSED STORAGE

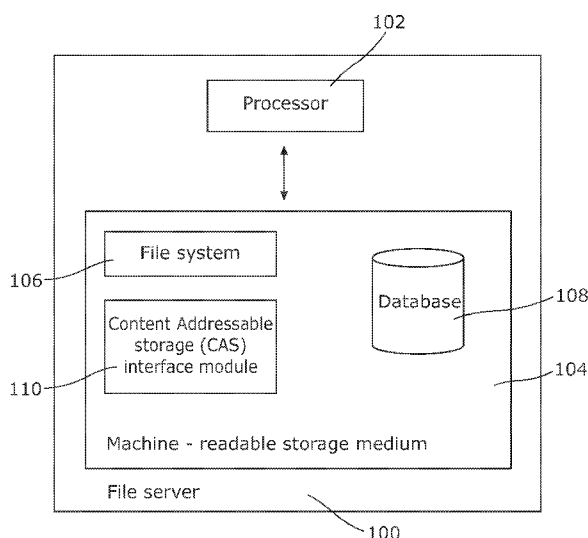


Fig. 1

(57) Abstract: Some examples describe a method for using location addressed storage as content addressed storage (CAS). A checksum of a file may be generated during trans-
ition of the file to a retained state. The generated checksum, which may represent a content address of the file, may be stored in a database. The database may be queried with the content address of the file to retrieve a location address of the file corresponding to the content address of the file. The location address of the file is used to provide access to the file in the file system.

**Declarations under Rule 4.17:**

- *as to the identity of the inventor (Rule 4.17(i))*
- *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:

- *with international search report (Art. 21(3))*

USING LOCATION ADDRESSED STORAGE AS CONTENT ADDRESSED STORAGE

Background

[001] Storage systems are inevitable in modern day computing. Whether it is a general purpose computing device or a large data center of an enterprise, storage systems have become a key part of any computing experience. Over the years, storage system technologies have evolved from storing just a few bytes of data to multi petabytes of data. In terms of nomenclature, storage systems may be divided into various types. According to one classification, storage systems may be divided into two types on the basis of access to data stored in a storage device: location addressed storage system and content addressable storage (CAS) system.

Brief Description of the Drawings

[002] For a better understanding of the solution, embodiments will now be described, purely by way of example, with reference to the accompanying drawings, in which:

[003] FIG. 1 is a block diagram of an example computing device that facilitates using location addressed storage as content addressable storage (CAS);

[004] FIG. 2 is a block diagram of an example computing environment that facilitates using location addressed storage as content addressable storage (CAS);

[005] FIG. 3 is a flowchart of an example method for using location addressed storage as content addressable storage (CAS);

[006] FIG. 4 is a flowchart of an example method for using location addressed storage as content addressable storage (CAS); and

[007] FIG. 5 is a block diagram of an example system that facilitates using location addressed storage as content addressable storage (CAS).

Detailed Description of the Invention

[008] As mentioned earlier, according to a classification, storage systems may be of two types: location addressed storage system and content addressable storage system (CAS). In location-addressed storage system, each file is retrieved based on the location where it is stored. On the other hand, in a content addressable storage system (CAS), physical location of a stored object (for example, a file) is immaterial. A storage object may be accessed from its location via a unique identifier called a content address. A content address may be used to locate, retrieve, and validate a storage object.

[009] A CAS system creates a unique global ID (content address) for each storage object and may store one or more copies of data on different storage devices. A CAS system may retrieve data from one of the stored locations by processing the unique global ID supplied by a client system. In an example, a content address may be based on a cryptographic digest of the data in a stored object.

[0010] A CAS system may be used for long-term retention of content, for example, for compliance or regulatory purposes, or for archiving a large number of records that may not change over time. A CAS system ensures consistency of stored data i.e. whatever is saved in a CAS storage system is exactly what comes out.

[0011] A retention enabled file system allows users to apply retention settings on a file such that a file may be retained in the system for a period set by a user. The retention feature may allow users to retain files up to a hundred years or more. When a file is retained it can neither be modified nor be deleted. Even after retention period expires the file can't be modified but may become eligible for deletion. This state of the file is called WORM (Write Once Read Many).

[0012] Location addressed storage system is more commonly used as compared to a CAS system since the latter may require special hardware or software (file system) resources. A CAS system, therefore, may require additional investments from a user. Presently, there's no available storage solution that provides flexibility to use location addressed storage hardware and file system as content addressable storage. A traditional storage system does not provide access to data using both content addressable and location addressable capabilities.

[0013] The present disclosure describes a mechanism for using a location addressed storage system as content addressable storage system (CAS). In an example, the present disclosure may describe generating a checksum of a file during transition of a file to a retained state in a file system. The generated checksum, which may represent a content address of the file, may be stored in a database. The database may be queried with the content address of the file to retrieve a location address of the file corresponding to the content address of the file. The location address of the file is then used to provide access to the file in the file system.

[0014] Proposed disclosure provides flexibility to use location addressed storage hardware and file system as content addressable storage. It obviates the requirement of using a dedicated storage appliance to address the need for content addressable storage (CAS). Proposed disclosure stands out from such solutions by making existing location addressed storage as content addressable storage.

[0015] FIG. 1 is a block diagram of an example computing device 100 that facilitates using location addressed storage as content addressable storage (CAS). Computing device 100 may be a server, a desktop computer, a notebook computer, a tablet computer, and the like. In an example, computing device 100 may be a file server 100. File server 100 may include a processor 102 and a machine-readable storage medium 104.

[0016] Processor 102 may be any type of Central Processing Unit (CPU), microprocessor, or processing logic that interprets and executes machine-readable instructions stored in machine-readable storage medium 104.

[0017] Machine-readable storage medium 104 may be a random access memory (RAM) or another type of dynamic storage device that may store information and machine-readable instructions that may be executed by processor 102. For example, machine-readable storage medium 104 may be Synchronous DRAM (SDRAM), Double Data Rate (DDR), Rambus DRAM (RDRAM), Rambus RAM, etc. or storage memory media such as a floppy disk, a hard disk, a CD-ROM, a DVD, a pen drive, and the like. In an example, machine-readable storage medium 104 may be a non-transitory machine-readable medium.

[0018] In an example, machine-readable storage medium 104 may store a file system 106, a database 108 and a content addressable storage (CAS) interface module 110. The term “module” may refer to a software component (machine readable instructions), a hardware component or a combination thereof. A module may include, by way of example, components, such as software components, processes, tasks, co-routines, functions, attributes, procedures, drivers, firmware, data, databases, data structures, Application Specific Integrated Circuits (ASIC) and other computing devices. A module may reside on a volatile or non-volatile storage medium (e.g. 104) and configured to interact with a processor (e.g. 102) of a computing device (e.g. 100).

[0019] In general, file system 106 may be used for storage and retrieval of data from a storage device. Typically, each piece of data is called a “file” (or file object). File system 106 may be a local file system or a scale-out file system such as a shared file system or a network file system. Examples of a shared file system may include a Network Attached Storage (NAS) file system or a cluster file system. Examples of a network file system may include a

distributed file system or a distributed parallel file system. File system 106 may allow a user to apply retention settings on a file such that the file is retained in the system for a period set by the user. When a file transitions from a normal state to a retained state (i.e. upon application of retention settings), a checksum (hash) of the file may be generated using a hash algorithm and stored in database (example, 108). The generated checksum represents a content address of the file. Thus, a file (or file object) may be location addressable as long as the content of a file changes (i.e. no retention settings are applied to the file). A file (or file object) may become content addressable after file content is marked as retained (i.e. retention settings are applied to the file). Some non-limiting examples of hash algorithms that may be used for generating a checksum of a file (or duplicate of a file) may include SHA, SHA-1, MD2, MD4, and MD5. In an instance, file system 106 may generate a notification event during transition of a file from a normal state to a retained state. A retained file may neither be modified nor deleted for a specified period depending upon the applied retention settings.

[0020] Database 108 may be a repository that stores an organized collection of data. In an example, database 108 may store a checksum of a file which, in an example, represents a content address of the file. A checksum of a file may be generated during transition of the file to a retained state i.e. once retention settings are applied to the file. Apart from the generated checksum, the database 108 may also store other attributes of a file such as, but not limited to, a unique ID of the file, file path, etc. In an example, database 108 may be queried using content address of a file (or file object) when an application (for instance, a CAS client) requests access to the file (or file object).

[0021] In an example, database 108 may be a distributed database that provides high query rates and high-throughput updates using a batching process. Database 108 may use a pipelined architecture that provides

access to update batches at various points through processing. In an instance, database 108 may be based on a batched update model, which decouples update processing from read-only queries (i.e. query processing task). In this model, the updates may be batched and processed in the background, and do not interfere with the foreground query workload. Database 108 may allow different stages of the updates in the pipeline to be queried independently. Queries that could use slightly out-of-date data may use only the final output of the pipeline, which may correspond to the completely ingested and indexed data. Queries that require even fresher results may access data at any stage in the pipeline. Database 108 may be a metadata database that stores metadata related to unstructured data. Examples of unstructured data may include documents, audio, video, images, files, body of an e-mail message, Web page, or word-processor document. In an example, database 108 may be integrated into file system 106.

[0022] Content addressable storage (CAS) interface module 110 may include instructions to query database 108 with the content address of a file. Upon querying the database 108, CAS interface module 110 may retrieve a location address of the file corresponding to the content address of the file, and use the location address of the file to provide access to the file in a file system (example, 106). In an example, an application stored on a client system, which may be communicatively coupled to computing device 100, may request CAS interface module 110 to lookup a file (or file object) in a file system (example, 106). In such case, said application may provide the content address of the file (or file object) which the application wants to access to CAS interface module 110. CAS interface module 110 then may query a database (example, 108) with the content address of the file to retrieve a location address (path) of the file for the given content address. As mentioned earlier, a database may store various attributes of a file such as, but not limited to, a unique ID of the file, file path, etc. Thus, a content

address (hash) of a file (or file object) may be mapped to a location address (path) of said file (or file object).

[0023] The location address of the file may be used by CAS interface module 110 to search for the file (or file object) in a file system (example, 106). In an example, in response to said search or file lookup, the file system (example, 106) may provide a handle for the file (or file object) to CAS interface module 110. CAS interface module 110 may in turn provide the file handle to the application that had originally requested access to the file (or file object). In such manner, using the file handle, a client application may read the file (or file object) present in a file system (example, 106). Thus, a mapping between content address (hash) and location address (path) of a file (or file object) is used to provide access to the file (or file object).

[0024] FIG. 2 is a block diagram of an example computing environment 200 that facilitates using location addressed storage as content addressable storage (CAS). Computing environment 200 may include client systems 202, 204, and 206, a file server 208, and a storage device 210. The number of client systems 202, 204, and 206, file server 208, and storage device 210 shown in FIG. 1 is for the purpose of illustration only and their number may vary in other implementations. In an example, computing environment 200 may represent a scale-out file system.

[0025] Client systems 202, 204, and 206 may each be a computing device such as a desktop computer, a notebook computer, a tablet computer, a mobile phone, personal digital assistant (PDA), a server, and the like. In an example, client systems 202, 204, and 206 may host one or more applications 224, 226, and 228 that may use a file system (example, 106) on file server (example, 208) for data storage and retrieval. In an instance, application 224, 226, and 228 may be a content addressable storage (CAS) client application that may request access to a file (or file object) in a file system (example, 106). Application 224, 226, and 228 may request access

to a file (or file object) in a file system (example, 106) by sharing the content address of a file with CAS interface module (example, 110) which in turn may provide such access by querying a database (example, 108) and retrieving a location address of the file corresponding to the content address of the file.

[0026] Client systems 202, 204, and 206, may communicate with file server 208 via a computer network 212. Computer network 212 may be a wireless or wired network. Computer network 212 may include, for example, a Local Area Network (LAN), a Wireless Local Area Network (WAN), a Metropolitan Area Network (MAN), a Storage Area Network (SAN), a Campus Area Network (CAN), or the like. Further, computer network 212 may be a public network (for example, the Internet) or a private network (for example, an intranet).

[0027] File server 208 may include a non-transitory machine-readable storage medium 214 that may store machine executable instructions. In an example, file server 208 may be similar to file server 100 described earlier. Accordingly, components of file server 208 that are similarly named and illustrated in file server 100 may be considered similar. For the sake of brevity, components or reference numerals of FIG. 2 having a same or similarly described function in FIG. 1 are not being described in connection with FIG. 2. Said components or reference numerals may be considered alike.

[0028] In an example, machine-readable storage medium 214 may store a file system 106, a database 108, a content addressable storage (CAS) interface module 110, a hash generator module 216, a journal writer 218, and a journal scanner 220.

[0029] A hash generator module 216 may include instructions to generate a checksum of a file when the file transitions from a normal state to a retained

state. In an instance, when a file transitions to a retained state, a notification event may be generated by file system 106. This notification event acts as a cue for hash generator module 216 to generate a checksum of a file that transitions to a retained state. The generated checksum may be sent to a journal writer 218 (present in the file system kernel module) which may include instructions to generate a journal for the checksum generation.

[0030] Journal scanner 220 may include instructions to process a journal generated by journal writer 218. Upon processing of a journal for checksum generation, journal scanner 220 may insert the generated checksum into database 108. Journal scanner 220 may also insert various file attributes such as, but not limited to, a unique ID of the file, file path, etc. in database 108.

[0031] Storage device 210 may be used to store and retrieve data stored by file system 106. Some non-limiting examples of storage device 210 may include a Direct Attached Storage (DAS) device, a Network Attached Storage (NAS) device, a tape drive, a magnetic tape drive, or a combination of these devices. Storage device 210 may be directly coupled to file server 106 or may communicate with file server 106 via a computer network 222. Such a computer network 222 may be similar to the computer network 212 described above. In an example, computer network 222 may be a Storage Area Network (SAN).

[0032] FIG. 3 is a flowchart of an example method 300 for using location addressed storage as content addressable storage (CAS). The method 300, which is described below, may at least partially be executed on a computing device 100 of FIG. 1 or file server 208 of FIG. 2. However, other computing devices may be used as well. At block 302, a checksum of a file may be generated during transition of the file from a normal state to a retained state in a file system (example, 106). At block 304, the generated checksum, which may represent a content address of the file, may be stored in a

database (example, 108). At block 306, database (example, 108) may be queried with the content address of the file (i.e. checksum of a file). In an example, the content address of a file is received from an application requesting access to the file in a file system (example, 106). At block 308, a location address of the file corresponding to the content address of the file may be retrieved from the database (example, 108). At block 310, the location address of the file may be used to provide access to the file in the file system (example, 106). In an example, using the location address of a file to provide access to the file in the file system may include performing a search for the file in the file system (example, 106) using the location address of the file, and retrieving a file handle of the file from the file system (example, 106). In an example, said file handle may be shared with a content addressable storage (CAS) client application requesting access to the file.

[0033] FIG. 4 is a flowchart of an example method 400 for using location addressed storage as content addressable storage (CAS). The method 400, which is described below, may at least partially be executed on a computing device 100 of FIG. 1 or file server 208 of FIG. 2. However, other computing devices may be used as well. At block 402, an event may be generated by a file system during transition of file from a normal state to a retained state. At block 404, upon generation of the event, a hash generator module (example, 216) may generate a checksum of the file transitioned to the retained state. In an example, said checksum of a file represents a content address of the file. At block 406, a journal writer (example, 218) may generate a journal for the checksum generation. At block 408, a journal scanner (example, 220) may process the journal and store the generated checksum in a database (example, 108). Journal scanner (example, 220) may store various attributes of the file in the database (example, 108) as well. At block 410, database (example, 108) may be queried with the content address of a file (i.e. checksum of a file) stored in the database. In an example, the content address of a file is received from an application requesting access to the file in a file system (example, 106). At block 412, a location address of the file

corresponding to the content address of the file may be retrieved from the database (example, 108). At block 414, the location address of the file may be used to provide access to the file in the file system (example, 106). In an example, using the location address of the file to provide access to the file in the file system may include performing a search for the file in the file system (example, 106) using the location address of the file, and retrieving a file handle of the file from the file system (example, 106). In an example, said file handle may be shared with a content addressable storage (CAS) client application requesting access to the file.

[0034] FIG. 5 is a block diagram of an example system 500 that facilitates for using location addressed storage as content addressable storage (CAS). System 500 includes a processor 502 and a machine-readable storage medium 504 communicatively coupled through a system bus. In an example, system 500 may be analogous to computing device 100 of FIG. 1 or file server 208 of FIG. 2. Processor 502 may be any type of Central Processing Unit (CPU), microprocessor, or processing logic that interprets and executes machine-readable instructions stored in machine-readable storage medium 504. Machine-readable storage medium 504 may be a random access memory (RAM) or another type of dynamic storage device that may store information and machine-readable instructions that may be executed by processor 502. For example, machine-readable storage medium 504 may be Synchronous DRAM (SDRAM), Double Data Rate (DDR), Rambus DRAM (RDRAM), Rambus RAM, etc. or a storage memory media such as a floppy disk, a hard disk, a CD-ROM, a DVD, a pen drive, and the like. In an example, machine-readable storage medium 504 may be a non-transitory machine-readable medium. Machine-readable storage medium 504 may store instructions 506, 508, 510, 512, and 514. In an example, instructions 506 may be executed by processor 502 to generate a checksum of a file during transition of the file from a normal state to a retained state. Instructions 508 may be executed by processor 502 to store the generated checksum in a database embedded within the file system, wherein the

checksum may represent a content address of the file in the database. Instructions 510 may be executed by processor 502 to query the database with the content address of the file. Instructions 512 may be executed by processor 502 to retrieve a location address of the file corresponding to the content address of the file. Instructions 514 may be executed by processor 502 to use the location address of the file to provide access to the file in the file system. In an example, instructions 514 may include instructions to perform a search for the file in the file system using the location address of the file, and instructions to retrieve a file handle of the file from the file system.

[0035] For the purpose of simplicity of explanation, the example methods of FIGS. 3 and 4 are shown as executing serially, however it is to be understood and appreciated that the present and other examples are not limited by the illustrated order. The example systems of FIGS. 1, 2 and 5, and methods of FIGS. 3 and 4 may be implemented in the form of a computer program product including computer-executable instructions, such as program code, which may be run on any suitable computing device in conjunction with a suitable operating system (for example, Microsoft Windows, Linux, UNIX, and the like). Embodiments within the scope of the present solution may also include program products comprising non-transitory computer-readable media for carrying or having computer-executable instructions or data structures stored thereon. Such computer-readable media can be any available media that can be accessed by a general purpose or special purpose computer. By way of example, such computer-readable media can comprise RAM, ROM, EPROM, EEPROM, CD-ROM, magnetic disk storage or other storage devices, or any other medium which can be used to carry or store desired program code in the form of computer-executable instructions and which can be accessed by a general purpose or special purpose computer. The computer readable instructions can also be accessed from memory and executed by a processor.

[0036] It may be noted that the above-described examples of the present solution is for the purpose of illustration only. Although the solution has been described in conjunction with a specific embodiment thereof, numerous modifications may be possible without materially departing from the teachings and advantages of the subject matter described herein. Other substitutions, modifications and changes may be made without departing from the spirit of the present solution. All of the features disclosed in this specification (including any accompanying claims, abstract and drawings), and/or all of the steps of any method or process so disclosed, may be combined in any combination, except combinations where at least some of such features and/or steps are mutually exclusive.

Claims:

1. A method of using location addressed storage as content addressable storage (CAS), comprising:

generating a checksum of a file during transition of a file to a retained state in a file system;

storing the generated checksum in a database, wherein the checksum represents a content address of the file in the database;

querying the database with the content address of the file;

retrieving a location address of the file corresponding to the content address of the file; and

using the location address of the file to provide access to the file in the file system.

2. The method of claim 1, wherein using the location address of the file to provide access to the file in the file system includes:

performing a search for the file in the file system using the location address of the file;

retrieving a file handle of the file from the file system; and

sharing the file handle of the file with an application requesting access to the file.

3. The method of claim 1, wherein the database is integrated into the file system.

4. The method of claim 1, wherein the content address of the file is received from an application requesting access to the file in the file system.

5. The method of claim 1, wherein the access to the file is provided to a content addressable storage (CAS) client application requesting said access to the file.

6. A system, comprising:

a database to store a hash of a file, wherein the hash of the file is generated during transition of the file to a retained state in a scale-out file system, and wherein the hash represents a content address of the file in the database; and

a content addressable storage (CAS) interface module to:

query the database with the content address of the file;

retrieve a location address of the file corresponding to the content address of the file; and

use the location address of the file to provide access to the file in the scale-out file system.

7. The system of claim 6, wherein the access to the file is provided to an application present on a communicatively coupled client system.

8. The system of claim 6, wherein the CAS interface module is to perform a file lookup in the file system using the location address

9. The system of claim 6, wherein the CAS interface module to:

perform a search for the file in the file system using the location address of the file;

retrieve a file handle of the file from the file system; and

share the file handle of the file with an application requesting access to the file.

10. The system of claim 6, further comprising a hash generator module to generate the checksum of the file during transition of the file to the retained state in the file system.

11. The system of claim 6, further comprising a journal scanner to store the checksum of the file in the database.

12. A non-transitory machine-readable storage medium comprising instructions executable by a processor to:

generate a checksum of a file during transition of a file to a retained state in a file system;

store the generated checksum in a database embedded within the file system, wherein the checksum represents a content address of the file in the database;

query the database with the content address of the file;

retrieve a location address of the file corresponding to the content address of the file; and

use the location address of the file to provide access to the file in the file system.

13. The storage medium of claim 12, wherein the instructions to use the location address of the file to provide access to the file in the file system comprise instructions to:

perform a search for the file in the file system using the location address of the file; and

retrieve a file handle of the file from the file system.

14. The storage medium of claim 12, wherein the file system is a Network Attached Storage (NAS) file system.

15. The storage medium of claim 12, wherein the database is to allow pipelining of updates and independent querying of the pipelined updates.

1/5

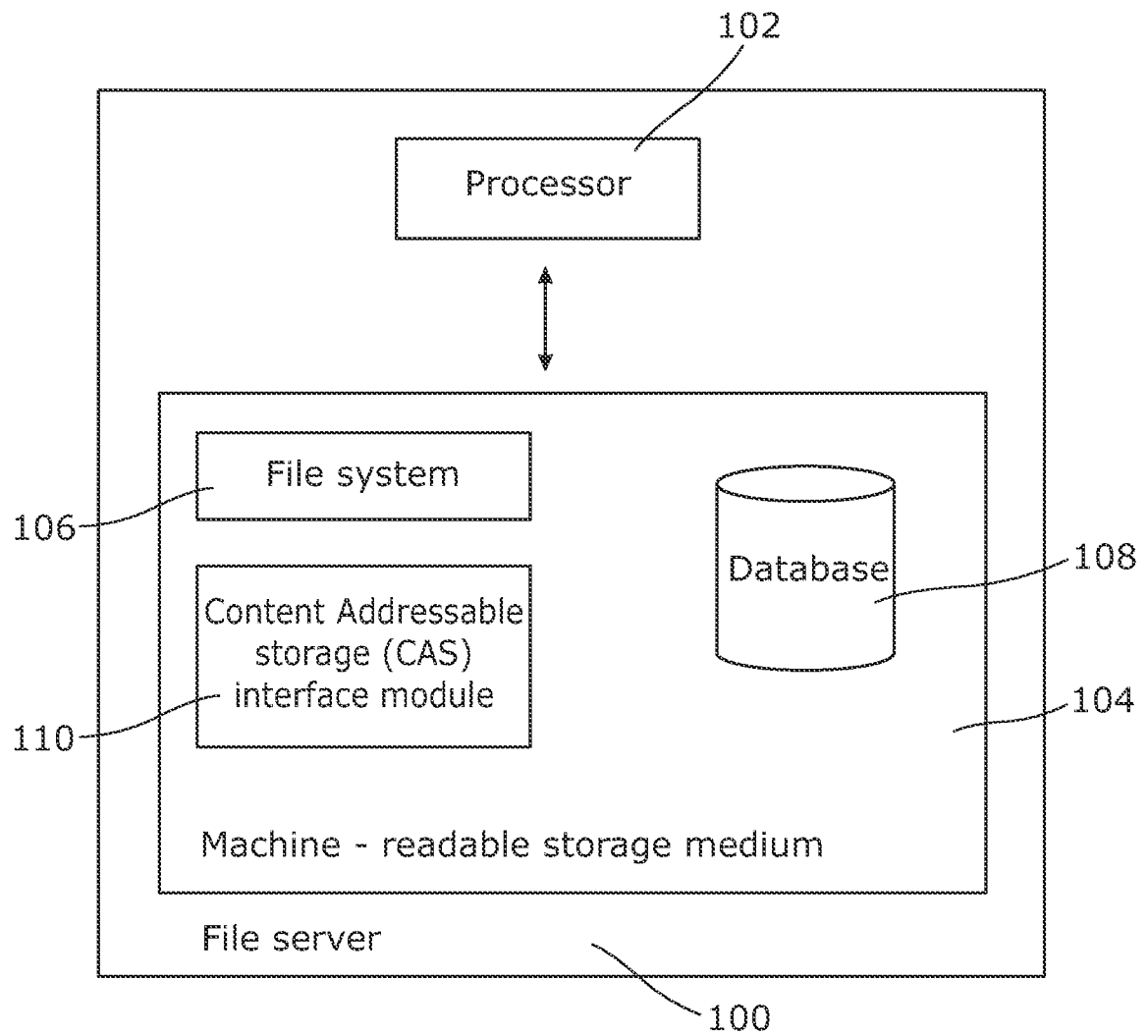


Fig. 1

2/5

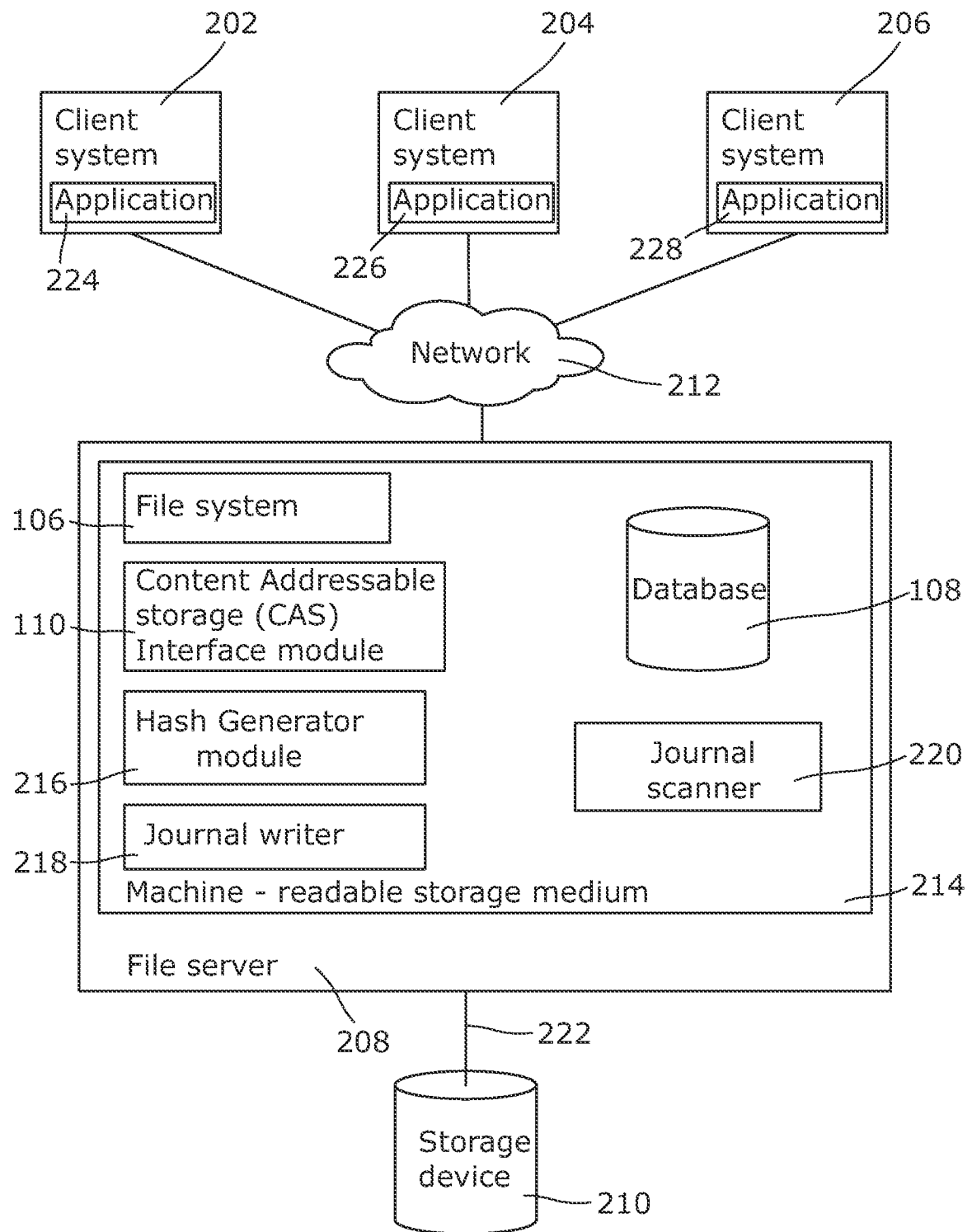
200

Fig. 2

3/5

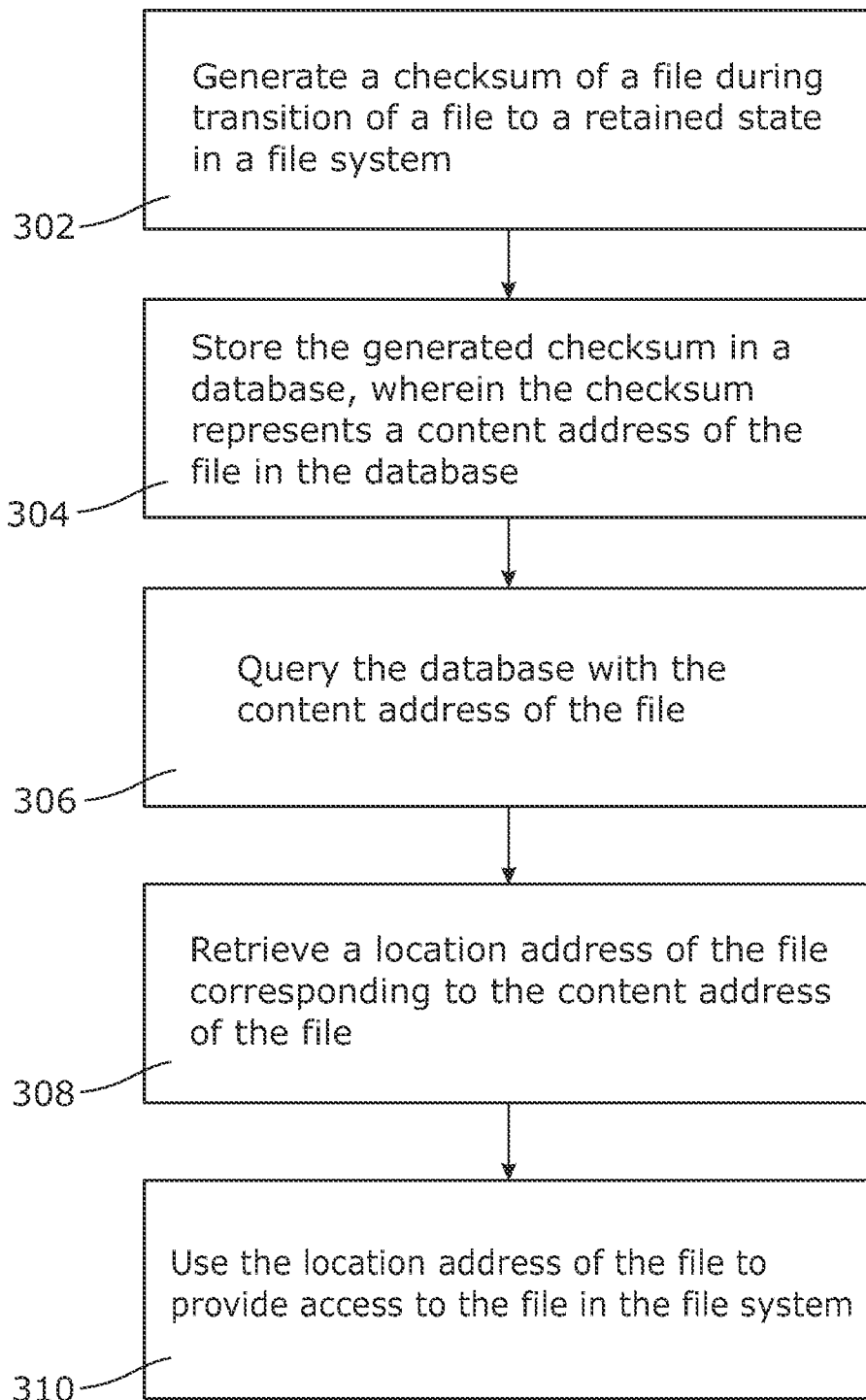
300

Fig. 3

4/5

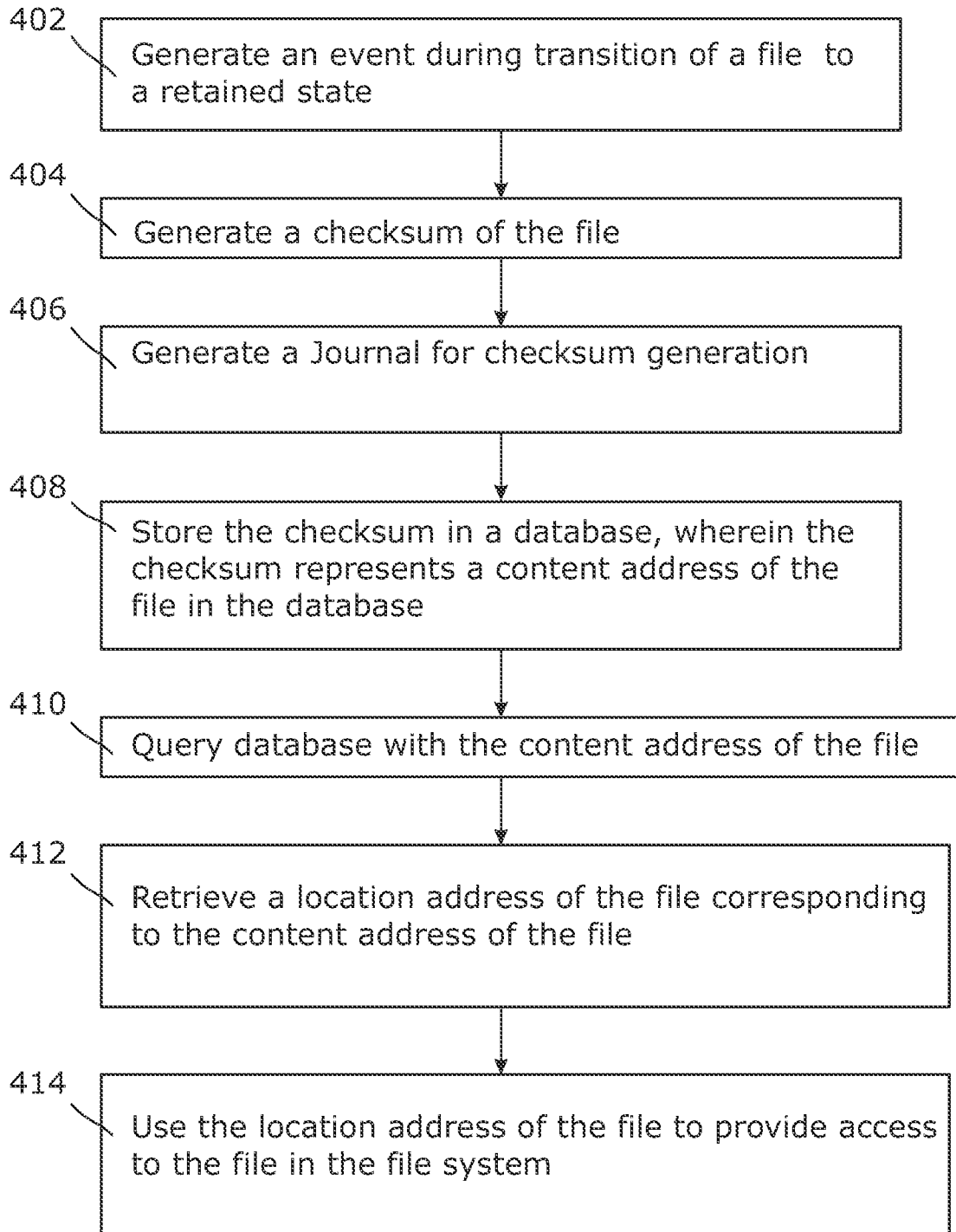
400

Fig. 4

5/5

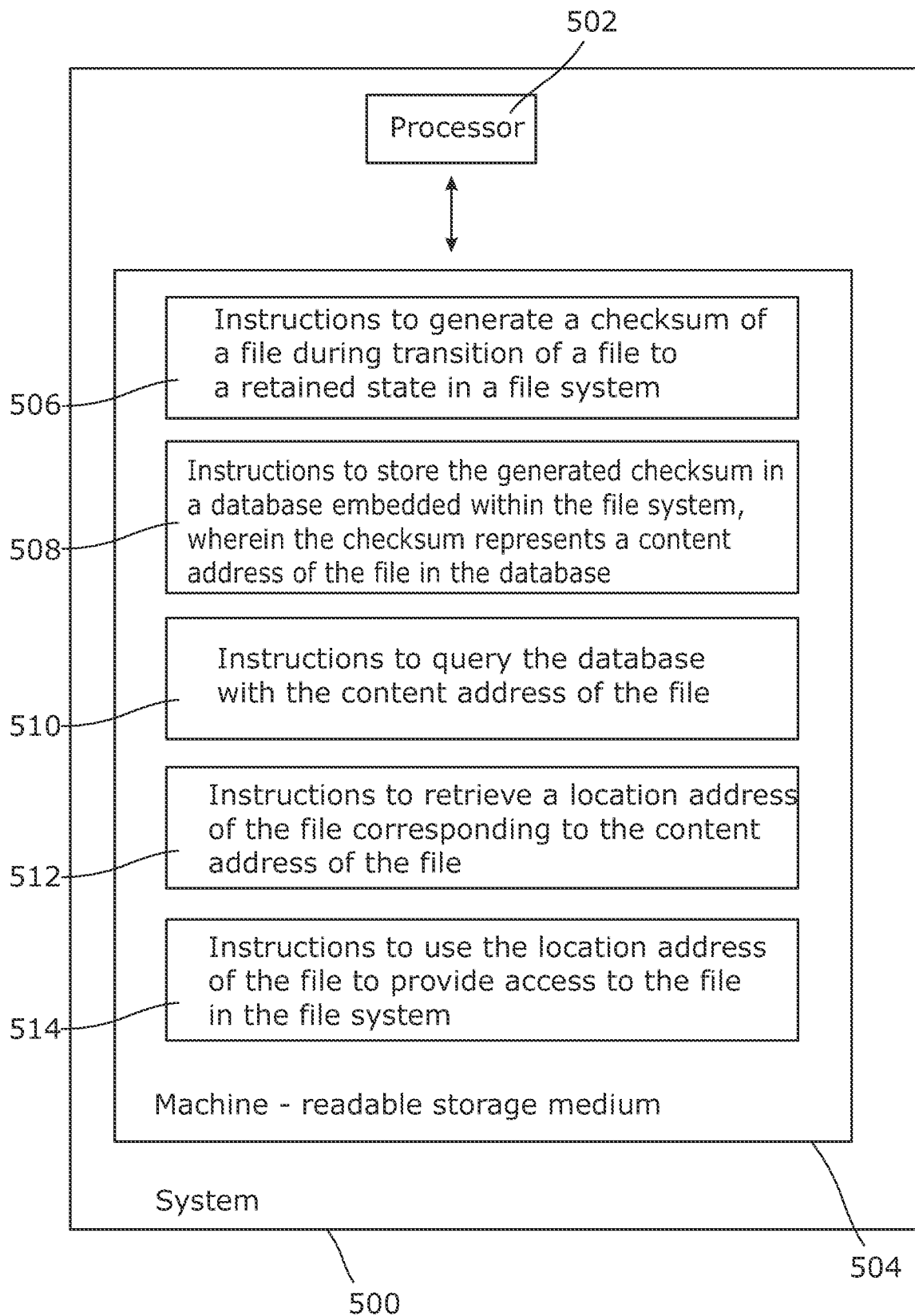


Fig. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/047596**A. CLASSIFICATION OF SUBJECT MATTER****G06F 17/30(2006.01)i**

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)

G06F 17/30; G06F 12/02; G06F 12/16; G06F 12/10; G06F 12/00; G06F 12/08

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

Korean utility models and applications for utility models

Japanese utility models and applications for utility models

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

eKOMPASS(KIPO internal) & Keywords: content addressable storage, database, retained state, hash, conversion, and similar terms

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 7,366,836 B1 (STEPHEN TODD et al.) 29 April 2008 See column 1, lines 21-27; column 8, lines 16-18; column 10, lines 3-6; column 12, lines 1-8 and 43-44; column 13, lines 2-6, 25, and 35-44; column 15, lines 31-33; column 17, lines 53-55; column 19, lines 12-21; column 21, lines 17-26; column 29, lines 46-52; claims 1, 15, and 17; and figure 1.	1-14
Y		15
Y	US 2008-0228697 A1 (ATUL ADYA et al.) 18 September 2008 See paragraphs [0009], [0114], and [0406]-[0408]; and figures 6 and 12.	15
A	US 2010-0070698 A1 (CRISTIAN UNGUREANU et al.) 18 March 2010 See paragraphs [0009] and [0072]-[0083]; and figures 2-3.	1-15
A	US 2013-0198462 A1 (BERTRAND SERLET et al.) 01 August 2013 See paragraphs [0013] and [0024]-[0034]; and figure 2.	1-15
A	US 2012-0215980 A1 (SCOTT C. AUCHMOODY et al.) 23 August 2012 See paragraphs [0022] and [0038]-[0040]; and figure 2.	1-15
A	US 2009-0228511 A1 (BENJAMIN ATKIN et al.) 10 September 2009 See paragraphs [0005] and [0029]-[0041]; and figure 3.	1-15



Further documents are listed in the continuation of Box C.



See patent family 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 application or patent 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

27 January 2015 (27.01.2015)

Date of mailing of the international search report

28 January 2015 (28.01.2015)

Name and mailing address of the ISA/KR

International Application Division
Korean Intellectual Property Office
189 Cheongsu-ro, Seo-gu, Daejeon Metropolitan City, 302-701,
Republic of Korea

Facsimile No. ++82 42 472 3473

Authorized officer

NHO, Ji Myong

Telephone No. +82-42-481-8528



INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2014/047596

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 7366836 B1	29/04/2008	None	
US 2008-0228697 A1	18/09/2008	None	
US 2010-0070698 A1	18/03/2010	CN 101676855 A	24/03/2010
		CN 101676855 B	02/04/2014
		CN 101685468 A	31/03/2010
		CN 101685468 B	27/11/2013
		EP 2164005 A2	17/03/2010
		EP 2164005 A3	29/09/2010
		EP 2164006 A2	17/03/2010
		EP 2164006 A3	11/06/2014
		JP 2010-079886 A	08/04/2010
		JP 2010-157204 A	15/07/2010
		JP 5506290 B2	28/05/2014
		JP 5539683 B2	02/07/2014
		US 2010-0064166 A1	11/03/2010
		US 7992037 B2	02/08/2011
		US 8335889 B2	18/12/2012
US 2013-0198462 A1	01/08/2013	CN 103226561 A	31/07/2013
		CN 103294610 A	11/09/2013
		US 2013-0198475 A1	01/08/2013
		US 8631209 B2	14/01/2014
US 2012-0215980 A1	23/08/2012	US 2012-215746 A1	23/08/2012
		US 2012-221524 A1	30/08/2012
		US 7788220 B1	31/08/2010
		US 8185554 B1	22/05/2012
		US 8386521 B2	26/02/2013
		US 8402063 B2	19/03/2013
		US 8443000 B2	14/05/2013
US 2009-0228511 A1	10/09/2009	CA 2717068 A1	11/09/2009
		CN 102016852 A	13/04/2011
		CN 102016852 B	27/03/2013
		EP 2252950 A1	24/11/2010
		EP 2252950 A4	25/07/2012
		JP 2011-513853 A	28/04/2011
		US 7747663 B2	29/06/2010
		WO 2009-110912 A1	11/09/2009