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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
3 March 2016 (03.03.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/032548 A1

(51) International Patent Classification:
G06F 15/16 (2006.01)

(21) International Application Number:
PCT/US2014/060705

(22) International Filing Date:
15 October 2014 (15.10.2014)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
4146/CHE/2014 25 August 2014 (25.08.2014) IN

(71) Applicant: **HEWLETT PACKARD ENTERPRISE DEVELOPMENT LP** [US/US]; 11445 Compaq Center Drive West, Houston, TX 77070 (US).

(72) Inventors: **SHASTRY, Krishnaprasad Lingadahalli**; Sy. No.192, Whitefield Road, Mahadevapura Post, Bangalore, Karnataka 560048 (IN). **MADHYASTHA, Sandesh V.**; Sy. No.192, Whitefield Road, Mahadevapura Post, Bangalore, Karnataka 560048 (IN).

(74) Agents: **SHOOKMAN, Jeb A.** et al.; Hewlett Packard Enterprise, 3404 E. Harmony Road, Mail Stop 79, Fort Collins, CO 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, 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, ST, 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).

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))*

(54) Title: PROVIDING TRANSACTIONAL SUPPORT TO A DATA STORAGE SYSTEM

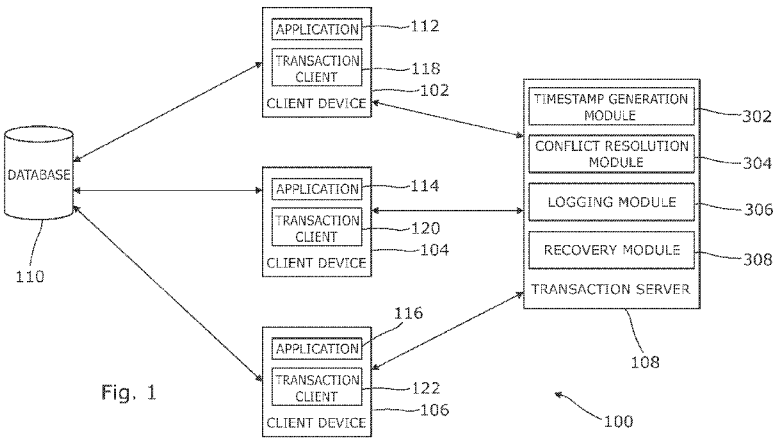

Fig. 1

(57) Abstract: Some examples described herein relate to providing transactional support to a data storage system. In an example, a transaction server may receive, from a client, a transaction request related to a record in a data storage system. The transaction sever may provide a unique transaction ID for the transaction request and last commit timestamp related to the record to the client, wherein the transaction ID is a timestamp value equal to or less than epoch time. The client may obtain a version of the record, corresponding to the last commit timestamp, from the data storage system. The client may update the version of the record, corresponding to the last commit timestamp, to generate an updated record in the data storage system, wherein the updating comprises including the unique transaction ID in the updated record. The transaction server may receive a request from the client to commit the updated record. The transaction server may determine whether a conflict related to the transaction request exists. In response to the determination that no conflict related to the transaction request exists, the updated record may be committed in the data storage system.

WO 2016/032548 A1

PROVIDING TRANSACTIONAL SUPPORT TO A DATA STORAGE SYSTEM

Background

[001] Organizations may need to deal with a vast amount of data these days, which could range from a few terabytes to multiple petabytes of data. Storage systems therefore have become central to an organization's IT strategy not withstanding whether it is a small start-up or a large company. Storage devices or systems (often used interchangeably) are no longer perceived as just a piece of hardware, but rather devices that help meet present and future information needs of an organization.

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 environment for providing transactional support to a data storage system;

[004] FIG. 2 illustrates a sequence diagram for an example transaction;

[005] FIG. 3 is a block diagram of an example computing device for providing transactional support to a data storage system;

[006] FIG. 4 is a flowchart of an example method for providing transactional support to a data storage system; and

[007] FIG. 5 is a block diagram of an example system for providing transactional support to a data storage system.

Detailed Description

[008] Data storage systems have evolved over the years from traditional Relational Database Management Systems (RDBMS) to recent NoSQL (Not only SQL) databases such as HBase, Cassandra, MongoDB, etc. that can quickly scale and meet increasing demand for storing and handling large volumes of data (or "Big Data"). Big data may include data sets with sizes beyond the ability of commonly used software applications and tools to capture, manage, and process the data within a reasonable time. For example, big data may include a few dozen terabytes to petabytes of data in a single data set.

[009] A NoSQl database, such as HBase, Cassandra, etc. may be a distributed database. A distributed database is a database in which sections of the database may be stored on multiple computer systems within a network. Data may be stored on multiple computers or storage devices, located at same physical location or different geographical locations, connected over a network. A distributed database, however, may lack "transactional support" functionality, typically provided by a Relational Database Management System.

[0010] A "transaction" may be defined as a sequence of read and/or write operations that may be conceptually related. Transactions are designed to maintain database integrity in a known, consistent state, by ensuring that interdependent operations on the system complete successfully or all the

operations are cancelled. However, there may be instances when concurrent transactions may occur. Transactions are said to be "concurrent" if they are being executed simultaneously. To avoid concurrency, database transactions are expected to satisfy a set of properties. These include atomicity, consistency, isolation, and durability. They are also commonly known by the acronym ACID.

[0011] One of the mechanisms for managing concurrency is optimistic concurrency model. In this model, a transaction is never blocked from executing its operations. However, before committing, each transaction verifies that no other transaction has modified the data it has read. If the check reveals conflicting modifications, the committing transaction is aborted and restarted again.

[0012] The "isolation" property defines that no transaction would interfere with another concurrent transaction. In other words, the intermediate steps of a transaction are invisible to other transactions. One of the mechanisms for managing isolation includes Snapshot Isolation (SI). In snapshot isolation, a transaction operates on a "private" snapshot of the database, taken at the start of the transaction. When the transaction concludes, it will successfully commit only if the values updated by the transaction have not been changed externally since the snapshot was taken. Snapshot isolation may involve use of timestamps.

[0013] "Timestamps" are unique sequence of characters or encoded information identifying when a certain event occurred (for example, by usually providing date and time of day). Timestamps can facilitate multiple versions of data in a database. They may be defined by a user when the data value is inserted, or implicitly assigned by the system. In SI, a transaction acquires a start timestamp, at the beginning of its execution, and acquires a commit timestamp, at the end of its execution. A transaction is said to be "committed" when the modifications done by the transaction are made visible to other

transactions. A transaction that commits successfully is called a committed transaction.

[0014] A transaction support based on optimistic concurrency is required to implement a mechanism to hide the intermediate modification to the data from other concurrent transactions. Some of the present mechanisms to handle this requirement either create additional metadata tables in the database to store information about intermediate modifications, or modify an existing database schema. Needless to say, such intrusion into a database is undesirable.

[0015] To prevent these issues, the present disclosure describes various examples for providing transactional support to a data storage system. In an example, a transaction server may receive, from a client, a transaction request related to a record in a data storage system. The transaction sever may provide a unique transaction ID for the transaction request and last commit timestamp related to the record to the client, wherein the transaction ID is a timestamp value equal to or less than epoch time. The client may obtain a version of the record, corresponding to the last commit timestamp, from the data storage system. The client may update the version of the record, corresponding to the last commit timestamp, to generate an updated record in the data storage system, wherein the updating comprises including the unique transaction ID in the updated record. The transaction server may receive a request from the client to commit the updated record. The transaction server may determine whether a conflict related to the transaction request exists. In response to the determination that no conflict related to the transaction request exists, the updated record may be committed in the data storage system.

[0016] FIG. 1 is a block diagram of an example computing environment 100 for providing transactional support to a data storage system. Computing

environment 100 may include a client computing devices 102, 104, and 106, a transaction server 108, and a database (or data storage system) 110.

[0017] Client computing devices (102, 104, and 106) may be communicatively coupled to transaction server 108 and database 110 via a computer network. Computer network may be a wireless or wired network. Computer network 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 may be a public network (for example, the Internet) or a private network (for example, an intranet). In an example, client computing devices (102, 104, and 106) may directly communicate with database 110.

[0018] Client computing devices 102, 104, and 106 generally represent any type of computing systems capable of reading machine-executable instructions. Examples of client computing devices 102, 104, and 106 may include, without limitation, a server, a desktop computer, a notebook computer, a tablet computer, a thin client, a mobile device, a personal digital assistant (PDA), a phablet, and the like. The number of client computing devices shown in FIG. 1 is for the purpose of illustration only and their number may vary in other implementations.

[0019] Client computing devices 102, 104, and 106 may each include one or more computer applications (machine-executable instructions) 112, 114, and 116 for carrying out a transaction(s) on a database 110. In this regard, client computing devices 102, 104, and 106 may each include a transaction client (for example, a library) 118, 120, and 122 that may provide transaction management APIs such as beginTransaction, endTransaction, etc. Transaction client (e.g., 102, 104, or 106) may also extend generic API's related to database, such as get, put, etc. to provide transaction support.

[0020] In an example, in case a computer application (example, 112) on a client computing device (example, 102) wish to perform a transaction related to a record in a database 110, transaction client (example, 118) may generate a transaction request for carrying out a transaction related to the record. Once a transaction request is generated, the client computing device (example, 102) may send the request to a transaction server (example, 108). In response, the transaction server (example, 108) may generate and assign a unique transaction ID to the transaction request, which is then shared with the client computing device (example, 102). In an instance, the unique transaction ID is a timestamp value equal to or less than epoch time (or Unix time). Epoch time (or Unix time) is a system for describing instants in time, defined as the number of seconds that have elapsed since January 1, 1970.

[0021] In an instance, the transaction server (example, 108) may also provide, to the client computing device (example, 102), the last commit timestamp value related to the record. A "last commit timestamp value" of a record is a value that was assigned to the record when it was last committed. Upon receipt, the client computing device (example, 102) may use the last commit timestamp of the record to obtain, from the database (example, 110), a version of the record, corresponding to the last commit timestamp. This version would be the latest version of the record in the database (example, 110). To provide a further explanation, let's consider an example where database may be a multi-version data storage system. In other words, database may store multiple versions of a record. In such case, in an instance, each version of the record may include a unique timestamp value. The latest version of the record may include the last commit timestamp value. Thus, in the present context, the client computing device (example, 102) may use the last commit timestamp value to obtain the latest version of a record from a database (example, 110).

[0022] Once the latest version of the record is received, the client computing device (example, 102) may modify or update the latest version of the record

to generate an updated or modified record in the database (example, 110). Updating of the record may include modifying an existing value in the record or inserting a new value in the record. In an instance, while updating, the client device (example, 102) may insert or include the unique transaction ID, which it had received from the transaction server (example, 108), in the updated record. A record updated in this manner by a transaction will not be visible to other transactions since any other transaction (for example, a read operation) would use timestamp value of January 1, 1970 (or a later value) as the starting value for a timestamp range. Fig. 2 illustrates a sequence diagram 200 for an example transaction.

[0023] Referring to FIG. 2, an application on a client device may initiate a transaction call (begin Trx) 202 related to a record (in a database, for example, Hbase) to a transaction client, which in turn may generate a transaction request based on the transaction call and send it to transaction server. On receipt of the transaction request, transaction server may generate and assign a unique transaction ID (Return Txid) 204 to the transaction request, which is then shared with the client computing device. In an instance, the unique transaction ID is a timestamp value equal to or less than epoch time (or Unix time). Transaction server may also provide, to the client computing device, the last commit timestamp (LCT) value 206 related to the record. Now, the client device may use the LCT value of the record (for example, via Get (ekey1, LCT)) to obtain 208, from the database, a version of the record (for example, Return (rKey1, val1)), corresponding to the last commit timestamp. This version would be the latest version of the record in the database. Once the latest version of the record is received 210, the client computing device may modify or update the latest version of the record (for example, using put (rKey1, val2, Txid) 212 to generate an updated or modified record in the database. While updating, the client device may insert or include the unique transaction ID (for example, Txid), which it had received from the transaction server, in the updated record. While reading the data, in “get” and “scan” operations, the client may first look for the rows

corresponding to its transaction ID thereby only reading its own changes. If a row is not modified by other transactions, the “get” and “scan” operations may read data which is greater than the timestamp value of January 1, 1970 and less than the last commit timestamp of the transaction. Thus, an intermediate “put” operation done by a transaction will not be visible to other transactions.

[0024] Once the latest version of a record is updated, the client computing device may send a request (for example, commitTransaction) 214 to the transaction server to commit the updated record. In response, the transaction server may determine whether a conflict related to the transaction request exists 216 (for example, ChkConflict (Txid, M_r)). In case no conflict related to the transaction request exists, the transaction server may provide a success response to the transaction client (for example, Return (Success, CT) 218, and the updated record may be committed in the database (for example, put (rKey1, val2, CT)). After the final “put” (for example, put (rKey1, val2, CT)), the status in the transaction server is marked as “committed” and the LCT is updated 222. At this point, the updated record is visible to other transactions.

[0025] Transaction server (example, 108) may include machine-readable instructions to generate a unique transaction ID for each transaction request that it may receive from a client computing device (example, 102). In an instance, the unique transaction ID is a timestamp value equal to or less than epoch time (or Unix time), as explained above.

[0026] Transaction server (example, 108) may maintain the start timestamp and the commit timestamp of a transaction. A transaction acquires a start timestamp, at the beginning of its execution, and acquires a commit timestamp, at the end of its execution. Transaction server (example, 108) may maintain the state of transactions and include instructions to resolve conflicts during the transactions.

[0027] In an instance, the transaction server (example, 108) may also provide, to the client computing device (example, 102), from which it had received a transaction request related to a record in a database (example, 110), the last commit timestamp value related to the record. A "last commit timestamp value" of a record is a value that was assigned to the record when it was last committed. Upon receipt, the client computing device (example, 102) may use the last commit timestamp of the record to obtain, from the database, a version of the record, corresponding to the last commit timestamp. The client computing device (example, 102) may modify or update said version of the record to generate an updated or modified record in the database. In an instance, while updating, the client device (example, 102) may insert or include the unique transaction ID, which it had received from the transaction server (example, 108), in the updated record. Once the latest version of a record is updated, the transaction server (example, 108) may receive a request from the client device (example, 102) to commit the updated record. In response, the transaction server (example, 108) may determine whether a conflict related to the transaction request exists. In case no conflict related to the transaction request exists, the updated record may be committed in the database. Various example components (e.g., a timestamp generation module 302, a conflict resolution module 304, a logging module 306, and a recovery module 308) of transaction server are described in detail below in reference to FIG. 2. In an example, computing environment 100 may include an additional backup transaction server to take care of the failure of transaction server. If the master transaction server fails the backup may take over.

[0028] Database 110 may be a repository that stores an organized collection of data. Database 110 may store one or more records. In an example, database 110 may be a distributed database. In another example, database 110 may use a key-value data structure for storing data. Such data structure provides scalability to the database and allows storage of large amounts of data, for instance, from a few dozen terabytes to petabytes of data in a single data set.

[0029] In an example, database 110 may be a multi-version data storage system. In other words, database may store multiple versions of a record. In such case, in an instance, each version of the record may include a unique timestamp value. The latest version of the record may include the last commit timestamp value.

[0030] FIG. 3 is a block diagram of an example computing device 300 for providing transactional support to a data storage system. Examples of computing device 300 may include, without limitation, a server, a desktop computer, a notebook computer, a tablet computer, a thin client, a mobile device, a personal digital assistant (PDA), a phablet, and the like. In an example, computing device 100 is transaction server 108 of FIG. 1.

[0031] In the example of FIG. 1, computing device 300 may include a timestamp generation module 302, a conflict resolution module 304, a logging module 306, and a recovery module 308. 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 and configured to interact with a processor of a computing device (e.g. 300).

[0032] Timestamp generation module 302 may receive, from a client, a transaction request related to a record in a distributed database. Timestamp generation module 302 may include instructions to provide a unique transaction ID for the transaction request and last commit timestamp related to the record to the client. In an example, the transaction ID is a timestamp value equal to or less than Unix time.

[0033] Conflict resolution module 304 may receive, from a client, a request to commit an updated record in the distributed database. In an instance, the updated record is generated by the client by obtaining a version of the record, corresponding to the last commit timestamp, from the distributed database. The client may modify said version of the record to generate an updated record. In an example, a modification of said version of the record comprises inserting the unique transaction ID in the updated record. Conflict resolution module 304 may further determine whether a conflict related to the transaction request exists. The conflict resolution module 304 may use the modified record and the transaction start timestamp to determine whether any other transactions modified and committed the same row. In response to the determination that no conflict related to the transaction request exists, conflict resolution module 304 may communicate the same to the client, which in turn may commit the updated record in the distributed database. In such case, the conflict resolution module 304 may also generate a new commit timestamp and send the new commit timestamp to the client.

[0034] In case it is determined that a conflict related to the transaction request exists, conflict resolution module 304 may mark the transaction for abort and send this information to client device.

[0035] Logging module 306 may log the functionalities performed by timestamp generation module and conflict resolution module into a persistence space for recovery. If a transaction is marked for abort due to conflicts, the client may return this information without cleaning the temporary updates to the database tables. If the client fails during the transaction execution, recovery module 308 may mark the transaction for abort after a specified duration. The transaction server may implement a background thread to clean all temporary "put" operations from the aborted transactions. If the client fails after the transaction is marked for "commit" then the transaction server background thread may update the corresponding records in the database table using the transaction commit timestamp.

[0036] FIG. 4 is a flowchart of an example method for providing transactional support to a data storage system. The method 400, which is described below, may at least partially be executed on a client computing device (example, 102, 104, or 106) , a transaction server (example, 108), and a database (example, 110) of FIG. 1 or transaction server 300 of FIG. 3. However, other computing devices may be used as well. At block 402, a transaction server may receive, from a client, a transaction request related to a record in the data storage system. At block 404, the transaction sever may provide a unique transaction ID for the transaction request and the last commit timestamp related to the record to the client. In an example, the transaction ID is a timestamp value equal to or less than epoch time. At block 406, the client may obtain a version of the record, corresponding to the last commit timestamp, from the data storage system. At block 408, the client may update the version of the record, corresponding to the last commit timestamp, to generate an updated record in the data storage system. In an instance, the updating comprises including the unique transaction ID in the updated record. At block 410, a transaction server may receive a request from the client to commit the updated record. At block 412, the transaction server may determine whether a conflict related to the transaction request exists. At block 414, in response to the determination that no conflict related to the transaction request exists, the updated record may be committed in the data storage system.

[0037] FIG. 5 is a block diagram of an example system 500 for providing transactional support to a data storage system. System 500 includes a processor 502 and a machine-readable storage medium 504 communicatively coupled through a system bus. 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, 514, 516, and 518. In an example, instructions 506 may be executed by processor 502 to receive, at a transaction server, a transaction request related to a record in a data storage system, from a client. Instructions 508 may be executed by processor 502 to provide, by the transaction sever, a unique transaction ID for the transaction request and last commit timestamp related to the record to the client, wherein the transaction ID is a timestamp value equal to or less than epoch time. Instructions 510 may be executed by processor 502 to obtain, by the client, a version of the record, corresponding to the last commit timestamp, from the data storage system. Instructions 512 may be executed by processor 502 to update, by the client, the version of the record, corresponding to the last commit timestamp, to generate an updated record in the data storage system, wherein the updating comprises including the unique transaction ID in the updated record. Instructions 514 may be executed by processor 502 to receive, at the transaction server, a request from the client to commit the updated record. Instructions 516 may be executed by processor 502 to determine, by the transaction server, whether a conflict related to the transaction request exists. Instructions 518 may be executed by processor to committing the updated record in the data storage system, in response to the determination that no conflict related to the transaction request exists.

[0038] Storage medium 504 may further include instructions to generate, in response to the determination that no conflict related to the transaction request exists. Storage medium 504 may also include instructions to send, by the transaction sever, the new commit timestamp to the client, and

instructions to update, by the client, the unique transaction ID in the updated record with the new commit timestamp.

[0039] For the purpose of simplicity of explanation, the example method of FIG. 4 is 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, 3, and 5, and method of FIG. 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.

[0040] 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 for providing transactional support to a data storage system, comprising:
receiving, at a transaction server, a transaction request related to a record in the data storage system, from a client;
providing, by the transaction sever, a unique transaction ID for the transaction request and last commit timestamp related to the record to the client, wherein the transaction ID is a timestamp value equal to or less than epoch time;
obtaining, by the client, a version of the record, corresponding to the last commit timestamp, from the data storage system;
updating, by the client, the version of the record, corresponding to the last commit timestamp, to generate an updated record in the data storage system, wherein the updating comprises including the unique transaction ID in the updated record;
receiving, at the transaction server, a request from the client to commit the updated record;
determining, by the transaction server, whether a conflict related to the transaction request exists; and
in response to the determination that no conflict related to the transaction request exists, committing the updated record in the data storage system.

2. The method of claim 1, further comprising, in response to the determination that no conflict related to the transaction request exists, generating, by the transaction server, a new commit timestamp.

3. The method of claim 2, further comprising, sending, by the transaction server, the new commit timestamp to the client.

4. The method of claim 3, further comprising, updating, by the client, the unique transaction ID in the updated record with the new commit timestamp.

5. The method of claim 1, wherein the data storage system is a distributed data storage system.

6. The method of claim 1, wherein the updating comprises modifying an existing value in the record or inserting a new value in the record.

7. A system for providing transactional support to a distributed database, comprising:
a timestamp generation module to:
receive, from a client, a transaction request related to a record in a distributed database; and
provide a unique transaction ID for the transaction request and last commit timestamp related to the record to the client, wherein the transaction ID is a timestamp value equal to or less than Unix time; and
a conflict resolution module to:
receive, from the client, a request to commit an updated record in the distributed database, wherein the updated record is generated by the client by obtaining a version of the record, corresponding to the last commit timestamp, from the distributed database, and modifying said version of the record, wherein the modifying comprises including the unique transaction ID in the updated record;
determine whether a conflict related to the transaction request exists; and
in response to the determination that no conflict related to the transaction request exists, provide an appropriate response to the client, for the client to commit the updated record in the distributed database.

8. The system of claim 7, wherein the distributed database stores multiple versions of the record.

9. The system of claim 7, wherein the conflict resolution module is to generate, in response to the determination that no conflict related to the transaction request exists, generating, a new commit timestamp.

10. The system of claim 9, wherein the conflict resolution module is to send the new commit timestamp to the client.

11. The system of claim 10, wherein the modifying comprises changing an existing value in the record or inserting a new value in the record.

12. A non-transitory machine-readable storage medium comprising instructions for providing transactional support to a data storage system, the instructions executable by a processor to:
receive, at a transaction server, a transaction request related to a record in a data storage system, from a client;
provide, by the transaction sever, a unique transaction ID for the transaction request and last commit timestamp related to the record to the client, wherein the transaction ID is a timestamp value equal to or less than epoch time;
obtain, by the client, a version of the record, corresponding to the last commit timestamp, from the data storage system;
update, by the client, the version of the record, corresponding to the last commit timestamp, to generate an updated record in the data storage system, wherein the updating comprises including the unique transaction ID in the updated record;
receive, at the transaction server, a request from the client to commit the updated record;
determine, by the transaction server, whether a conflict related to the transaction request exists; and
in response to the determination that no conflict related to the transaction request exists, committing the updated record in the data storage system.

13. The storage medium of claim 12, further comprising instructions to generate, in response to the determination that no conflict related to the transaction request exists, a new commit timestamp by the transaction server.

14. The storage medium of claim 13, further comprising instructions to send, by the transaction sever, the new commit timestamp to the client.

15. The storage medium of claim 14, further comprising instructions to update, by the client, the unique transaction ID in the updated record with the new commit timestamp.

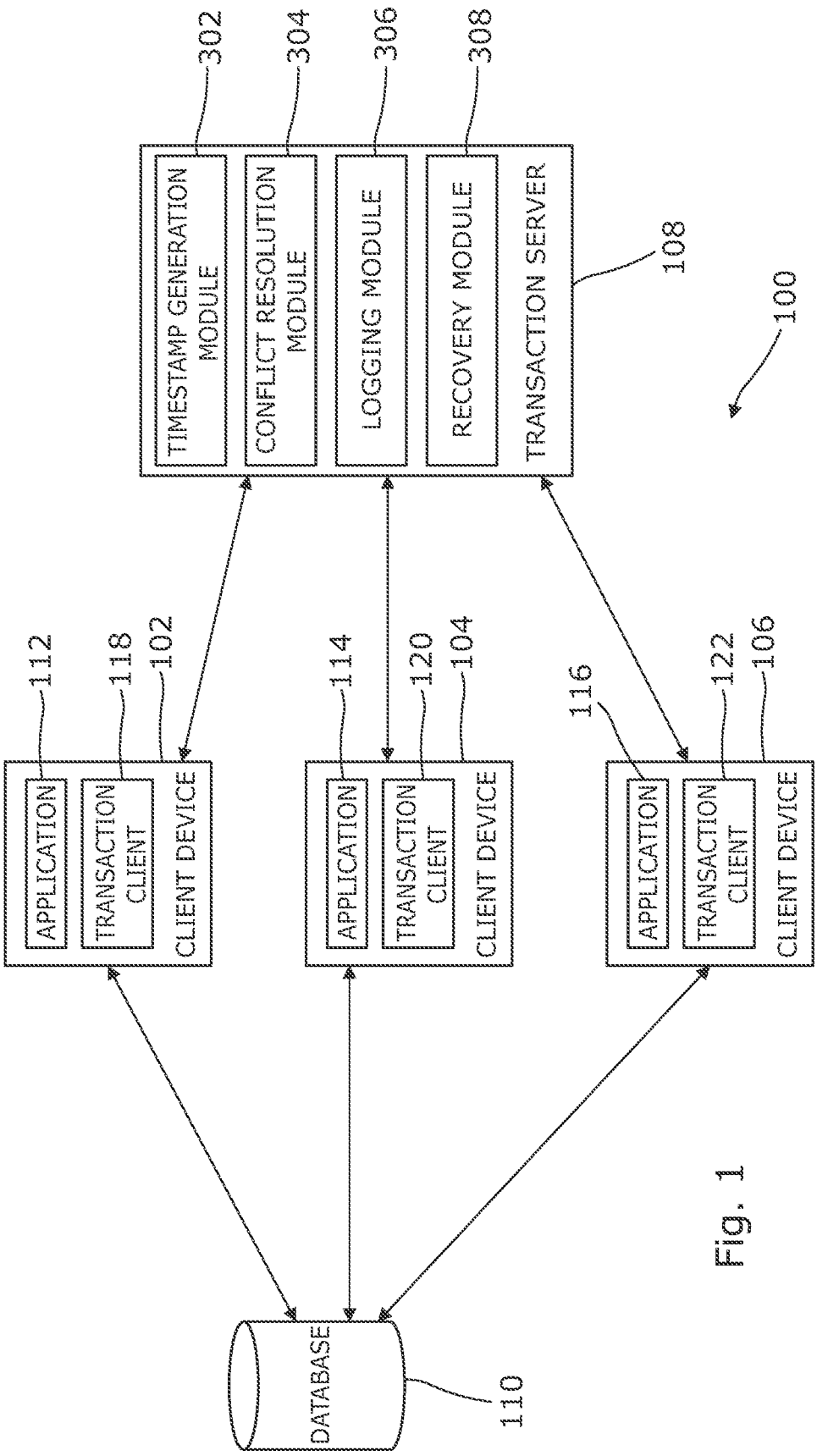
302
304
306
308
TIMESTAMP GENERATION MODULE
CONFLICT RESOLUTION MODULE
LOGGING MODULE
RECOVERY MODULE
TRANSACTION SERVER
108
100
112
118
102
APPLICATION
TRANSACTION CLIENT
CLIENT DEVICE
114
120
104
APPLICATION
TRANSACTION CLIENT
CLIENT DEVICE
116
122
106
APPLICATION
TRANSACTION CLIENT
CLIENT DEVICE
DATABASE
110

Fig. 1

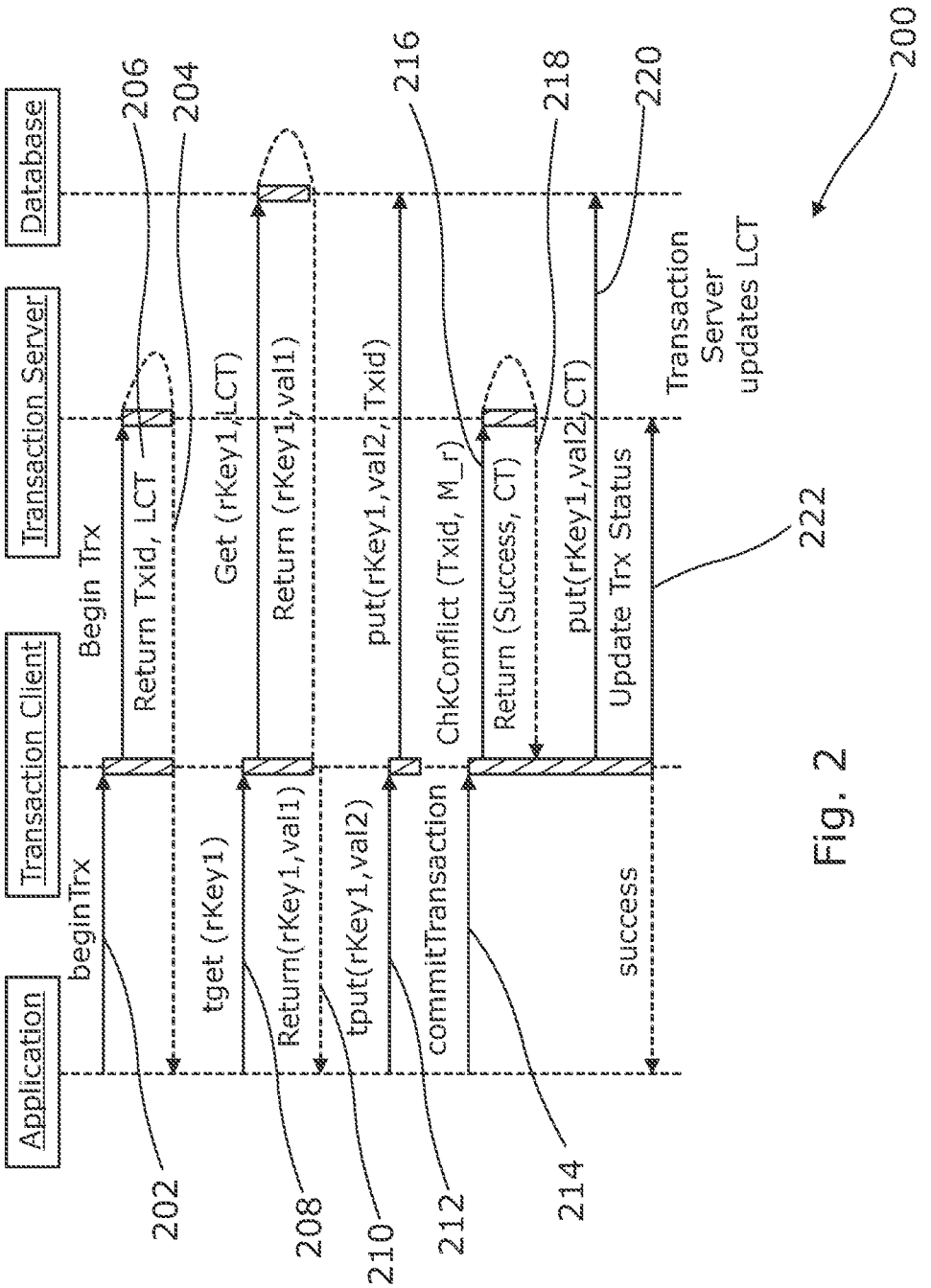
Application
Transaction Client
Transaction Server
Database
beginTrx
Begin Trx
Return Txid, LCT
Get (rKey1, LCT)
Return (rKey1, val1)
put(rKey1, val2, Txid)
ChkConflict (Txid, M_r)
Return (Success, CT)
put(rKey1, val2, CT)
Update Trx Status
Transaction Server updates LCT
tget (rKey1)
Return(rKey1, val1)
tput(rKey1, val2)
commitTransaction
success
202
204
206
208
210
212
214
216
218
220
222
200

Fig. 2

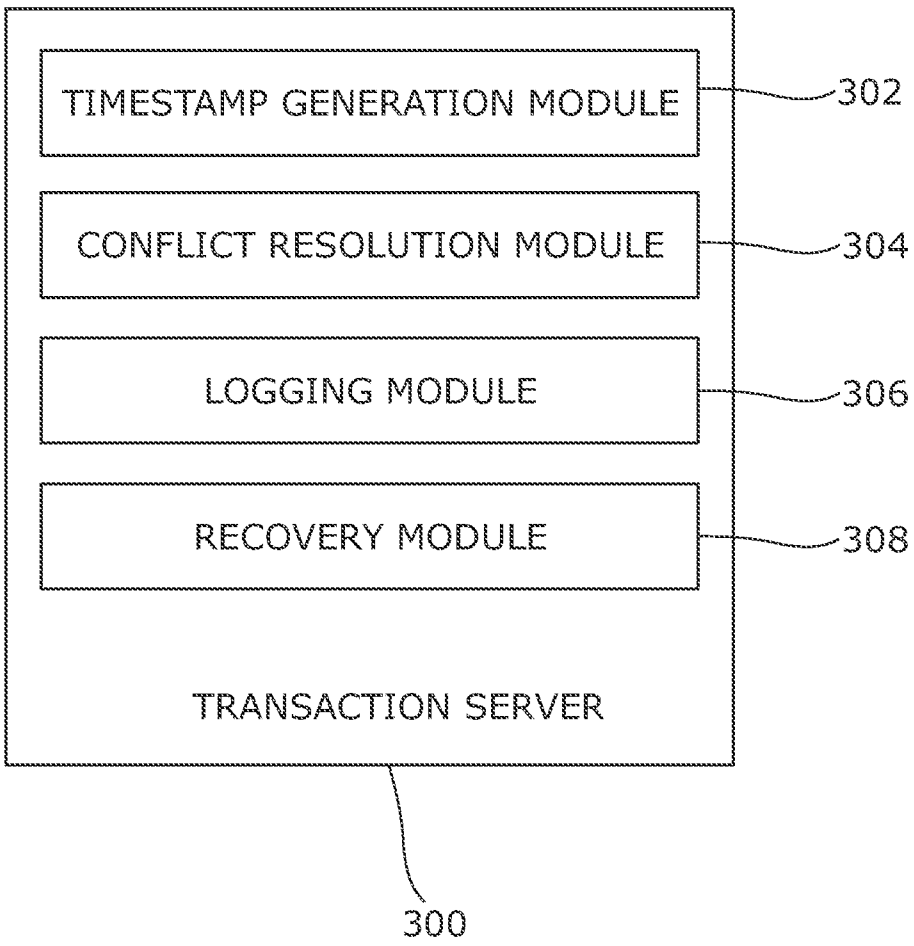
TIMESTAMP GENERATION MODULE
302
CONFLICT RESOLUTION MODULE
304
LOGGING MODULE
306
RECOVERY MODULE
308
TRANSACTION SERVER
300

Fig. 3

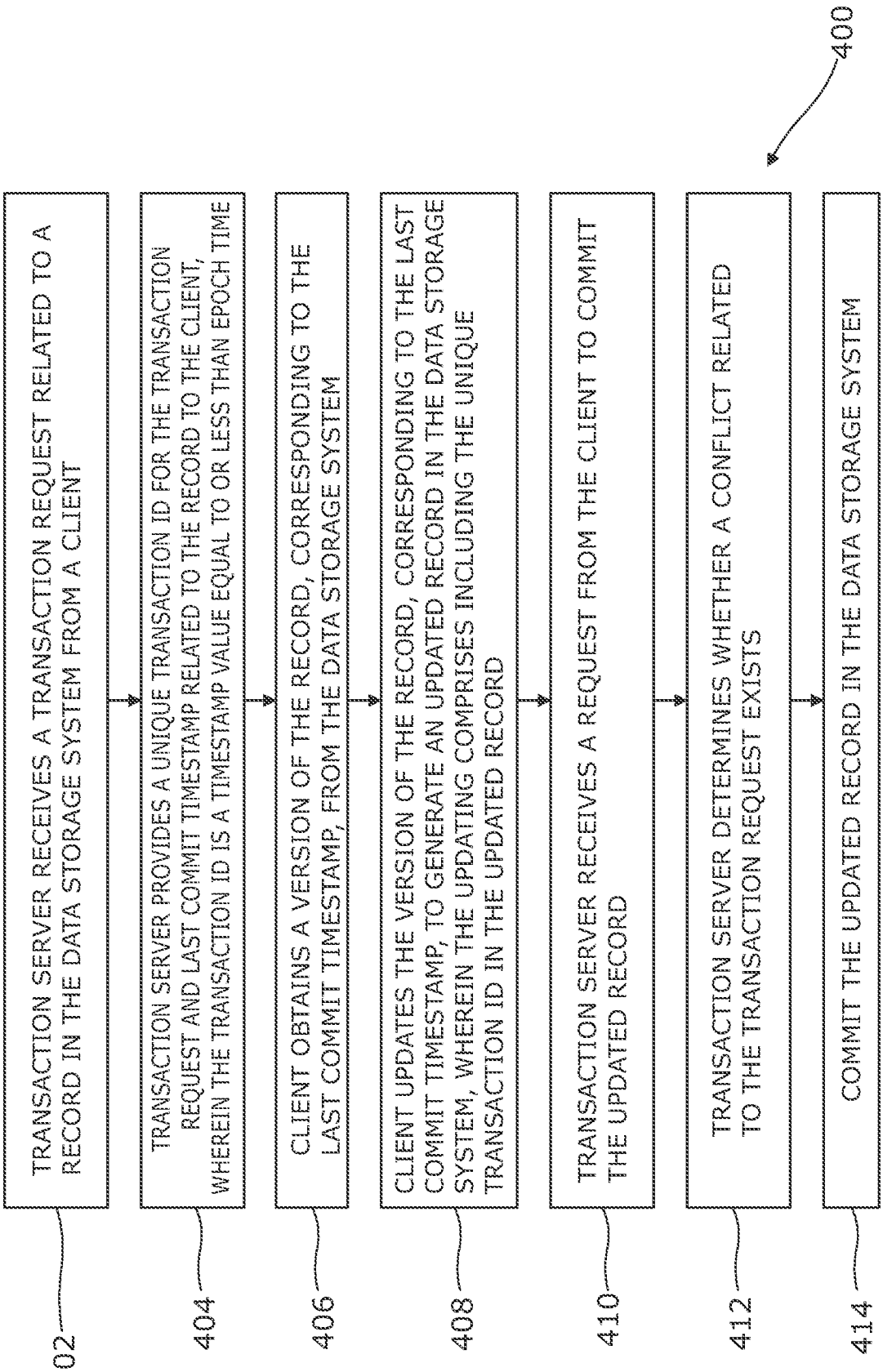
400
TRANSACTION SERVER RECEIVES A TRANSACTION REQUEST RELATED TO A RECORD IN THE DATA STORAGE SYSTEM FROM A CLIENT
402
TRANSACTION SERVER PROVIDES A UNIQUE TRANSACTION ID FOR THE TRANSACTION REQUEST AND LAST COMMIT TIMESTAMP RELATED TO THE RECORD TO THE CLIENT, WHEREIN THE TRANSACTION ID IS A TIMESTAMP VALUE EQUAL TO OR LESS THAN EPOCH TIME
404
CLIENT OBTAINS A VERSION OF THE RECORD, CORRESPONDING TO THE LAST COMMIT TIMESTAMP, FROM THE DATA STORAGE SYSTEM
406
CLIENT UPDATES THE VERSION OF THE RECORD, CORRESPONDING TO THE LAST COMMIT TIMESTAMP, TO GENERATE AN UPDATED RECORD IN THE DATA STORAGE SYSTEM, WHEREIN THE UPDATING COMPRISES INCLUDING THE UNIQUE TRANSACTION ID IN THE UPDATED RECORD
408
TRANSACTION SERVER RECEIVES A REQUEST FROM THE CLIENT TO COMMIT THE UPDATED RECORD
410
TRANSACTION SERVER DETERMINES WHETHER A CONFLICT RELATED TO THE TRANSACTION REQUEST EXISTS
412
COMMIT THE UPDATED RECORD IN THE DATA STORAGE SYSTEM
414

Fig. 4

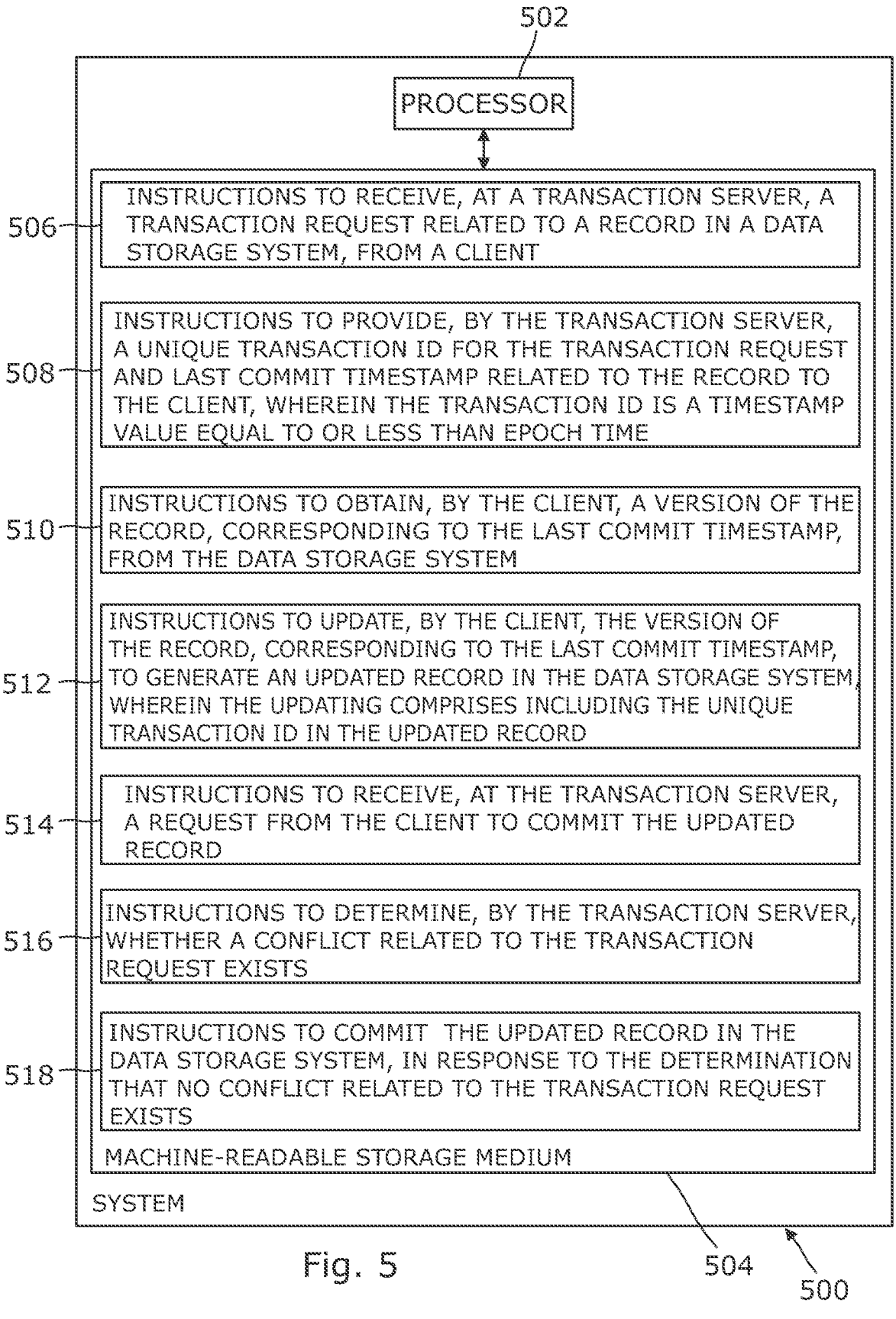
502
PROCESSOR
506
INSTRUCTIONS TO RECEIVE, AT A TRANSACTION SERVER, A TRANSACTION REQUEST RELATED TO A RECORD IN A DATA STORAGE SYSTEM, FROM A CLIENT
508
INSTRUCTIONS TO PROVIDE, BY THE TRANSACTION SERVER, A UNIQUE TRANSACTION ID FOR THE TRANSACTION REQUEST AND LAST COMMIT TIMESTAMP RELATED TO THE RECORD TO THE CLIENT, WHEREIN THE TRANSACTION ID IS A TIMESTAMP VALUE EQUAL TO OR LESS THAN EPOCH TIME
510
INSTRUCTIONS TO OBTAIN, BY THE CLIENT, A VERSION OF THE RECORD, CORRESPONDING TO THE LAST COMMIT TIMESTAMP, FROM THE DATA STORAGE SYSTEM
512
INSTRUCTIONS TO UPDATE, BY THE CLIENT, THE VERSION OF THE RECORD, CORRESPONDING TO THE LAST COMMIT TIMESTAMP, TO GENERATE AN UPDATED RECORD IN THE DATA STORAGE SYSTEM, WHEREIN THE UPDATING COMPRISES INCLUDING THE UNIQUE TRANSACTION ID IN THE UPDATED RECORD
514
INSTRUCTIONS TO RECEIVE, AT THE TRANSACTION SERVER, A REQUEST FROM THE CLIENT TO COMMIT THE UPDATED RECORD
516
INSTRUCTIONS TO DETERMINE, BY THE TRANSACTION SERVER, WHETHER A CONFLICT RELATED TO THE TRANSACTION REQUEST EXISTS
518
INSTRUCTIONS TO COMMIT THE UPDATED RECORD IN THE DATA STORAGE SYSTEM, IN RESPONSE TO THE DETERMINATION THAT NO CONFLICT RELATED TO THE TRANSACTION REQUEST EXISTS
MACHINE-READABLE STORAGE MEDIUM
SYSTEM
504
500

Fig. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/060705

A. CLASSIFICATION OF SUBJECT MATTER

G06F 15/16(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 15/16; G06F 17/30; G06F 11/10; G06F 17/00

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: transaction, unique, ID, timestamp, distribut*

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 6026413 A (CHALLENGER; JAMES ROBERT HAROLD et al.) 15 February 2000 See abstract;Col.8 line 22 - Col.11 line 67,Col.16 line 37-Col.18 line21, Col.32 line 47 - Col.33 line 50; figures 1A, 3-5, 13-17, 30E, 34.	1,5-8,12
A		2-4,9-11,13-15
Y	KR 10-2009-0102788 A (FUSION MULTISYSTEMS, INC. (DBA FUSION-IO)) 30 September 2009 See abstract; paragraphs [0066]-[0137]; figures 1A-1C, 2A.	1,5-8,12
A		2-4,9-11,13-15
A	US 2002-0065848 A1 (RICHARD WALKER et al.) 30 May 2002 See abstract; paragraphs [0063]-[0186]; figures 4-6.	1-15
A	US 2008-0222296 A1 (LIPPINCOTT LISA ELLEN et al.) 11 September 2008 See abstract; paragraphs [0045]-[0055]; figure 3.	1-15
A	US 7702739 B1 (CHENG WESLEY et al.) 20 April 2010 See abstract; Col.13 line 33 - Col.15 line 22; figure 6.	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	Date of mailing of the international search report
19 May 2015 (19.05.2015)	**19 May 2015 (19.05.2015)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701,
Republic of Korea
Facsimile No. ++82 42 472 7140

Authorized officer
KYUNG, Youn Jeong
Telephone No. +82-42-481-3452

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2014/060705

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 06026413 A	15/02/2000	CN 1173270 C	27/10/2004
		CN 1213800 A	14/04/1999
		GB 2328535 A	24/02/1999
		JP 11-120066 A	30/04/1999
		JP 3606542 B2	05/01/2005
		KR 10-0310066 B1	17/12/2001
		TW 400494 A	01/08/2000
		TW 400494 B	01/08/2000
KR 10-2009-0102788 A	30/09/2009	CA 2672035 A1	12/06/2008
		CA 2672100 A1	12/06/2008
		CN 101622594 A	06/01/2010
		CN 101622594 B	13/03/2013
		CN 101622595 A	06/01/2010
		CN 101622596 A	06/01/2010
		CN 101622606 A	06/01/2010
		CN 101622606 B	09/04/2014
		CN 101636712 A	27/01/2010
		CN 101646993 A	10/02/2010
		CN 101646994 A	10/02/2010
		CN 101657802 A	24/02/2010
		CN 101681282 A	24/03/2010
		CN 101689130 A	31/03/2010
		CN 101689131 A	31/03/2010
		CN 101689131 B	20/03/2013
		CN 101690068 A	31/03/2010
		CN 101715575 A	26/05/2010
		CN 102084330 A	01/06/2011
		CN 102597910 A	18/07/2012
		CN 102598019 A	18/07/2012
		CN 102696010 A	26/09/2012
		CN 103049058 A	17/04/2013
		CN 103098034 A	08/05/2013
		EP 2100214 A1	16/09/2009
		EP 2100214 B1	30/10/2013
		EP 2108143 A2	14/10/2009
		EP 2109812 A2	21/10/2009
		EP 2109822 A1	21/10/2009
		EP 2109822 B1	25/06/2014
		EP 2115563 A2	11/11/2009
		EP 2126679 A2	02/12/2009
		EP 2126680 A2	02/12/2009
		EP 2126698 A2	02/12/2009
		EP 2126709 A2	02/12/2009
		EP 2286326 A1	23/02/2011
		EP 2476039 A2	18/07/2012
		EP 2476055 A2	18/07/2012
		EP 2476079 A2	18/07/2012
		EP 2598996 A2	05/06/2013

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2014/060705

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
		JP 2010-512568 A	22/04/2010
		JP 2010-512584 A	22/04/2010
		JP 2010-512586 A	22/04/2010
		JP 2010-515116 A	06/05/2010
		JP 2011-518380 A	23/06/2011
		JP 2013-504142 A	04/02/2013
		JP 2013-504820 A	07/02/2013
		JP 5518197 B2	11/06/2014
		JP 5523835 B2	18/06/2014
		JP 5611597 B2	22/10/2014
		KR 10-1490327 B1	05/02/2015
		KR 10-2009-0087119 A	14/08/2009
		KR 10-2009-0087498 A	17/08/2009
		KR 10-2009-0095641 A	09/09/2009
		KR 10-2009-0102789 A	30/09/2009
		KR 10-2011-0039417 A	18/04/2011
		KR 10-2012-0090965 A	17/08/2012
		KR 10-2012-0093869 A	23/08/2012
		KR 10-2013-0026517 A	13/03/2013
		US 2008-0137284 A1	12/06/2008
		US 2008-0183953 A1	31/07/2008
		US 2008-0225474 A1	18/09/2008
		US 2008-0229079 A1	18/09/2008
		US 2008-0256292 A1	16/10/2008
		US 2009-0125671 A1	14/05/2009
		US 2009-0132760 A1	21/05/2009
		US 2009-0150641 A1	11/06/2009
		US 2011-0022801 A1	27/01/2011
		US 2011-0047356 A2	24/02/2011
		US 2011-0047437 A1	24/02/2011
		US 2011-0058440 A1	10/03/2011
		US 2011-0060887 A1	10/03/2011
		US 2011-0060927 A1	10/03/2011
		US 2011-0066808 A1	17/03/2011
		US 7713068 B2	11/05/2010
		US 7778020 B2	17/08/2010
		US 7934055 B2	26/04/2011
		US 8019938 B2	13/09/2011
		US 8074011 B2	06/12/2011
		US 8195912 B2	05/06/2012
		US 8289801 B2	16/10/2012
		US 8402201 B2	19/03/2013
		US 8429436 B2	23/04/2013
		US 8443134 B2	14/05/2013
		US 8578127 B2	05/11/2013
		US 8706968 B2	22/04/2014
		US 8719501 B2	06/05/2014
		WO 2008-070172 A2	12/06/2008
		WO 2008-070172 A3	24/07/2008
		WO 2008-070173 A1	12/06/2008

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2014/060705

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
		WO 2008-070174 A2	12/06/2008
		WO 2008-070174 A3	11/09/2009
		WO 2008-070175 A2	12/06/2008
		WO 2008-070175 A3	18/12/2008
		WO 2008-070191 A2	12/06/2008
		WO 2008-070191 A3	24/07/2008
		WO 2008-070796 A2	12/06/2008
		WO 2008-070796 A3	10/12/2009
		WO 2008-070798 A1	12/06/2008
		WO 2008-070799 A2	12/06/2008
		WO 2008-070799 A3	23/10/2008
		WO 2008-070800 A1	12/06/2008
		WO 2008-070802 A2	12/06/2008
		WO 2008-070802 A3	09/10/2008
		WO 2008-070803 A1	12/06/2008
		WO 2008-070811 A2	12/06/2008
		WO 2008-070811 A3	07/08/2008
		WO 2008-070812 A2	12/06/2008
		WO 2008-070812 A3	23/12/2009
		WO 2008-070813 A2	12/06/2008
		WO 2008-070813 A3	23/12/2009
		WO 2008-070814 A2	12/06/2008
		WO 2008-070814 A3	13/11/2008
		WO 2008-127458 A2	23/10/2008
		WO 2008-127458 A3	07/01/2010
		WO 2009-126542 A1	15/10/2009
		WO 2011-031796 A2	17/03/2011
		WO 2011-031796 A3	30/06/2011
		WO 2011-031899 A2	17/03/2011
		WO 2011-031899 A3	16/06/2011
		WO 2011-031900 A2	17/03/2011
		WO 2011-031900 A3	16/06/2011
		WO 2011-031903 A2	17/03/2011
		WO 2011-031903 A3	21/07/2011
		WO 2011-143628 A2	17/11/2011
		WO 2011-143628 A3	23/02/2012
		WO 2012-016089 A2	02/02/2012
		WO 2012-016089 A3	31/05/2012
		WO 2012-016209 A2	02/02/2012
		WO 2012-016209 A3	02/08/2012
		WO 2012-021847 A2	16/02/2012
		WO 2012-021847 A3	31/05/2012
		WO 2012-083308 A2	21/06/2012
		WO 2012-083308 A3	01/11/2012
US 2002-0065848 A1	30/05/2002	AU 8742101 A	04/03/2002
		CA 2424713 A1	28/02/2002
		CA 2424713 C	04/12/2007
		US 2007-0186157 A1	09/08/2007
		US 7249314 B2	24/07/2007

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2014/060705

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
		WO 02-17115 A2	28/02/2002
		WO 02-17115 A3	21/08/2003
US 2008-0222296 A1	11/09/2008	US 2008-228442 A1	18/09/2008
		US 7962610 B2	14/06/2011
US 7702739 B1	20/04/2010	US 2014-098758 A1	10/04/2014