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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
4 April 2013 (04.04.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/048969 A1

(51) International Patent Classification:
G06F 11/00 (2006.01)

(21) International Application Number:
PCT/US2012/056941

(22) International Filing Date:
24 September 2012 (24.09.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/541,052 29 September 2011 (29.09.2011) US
13/414,607 7 March 2012 (07.03.2012) US

(71) Applicant: **ORACLE INTERNATIONAL CORPORATION** [US/US]; 500 Oracle Parkway, M/S 5op7, Redwood Shores, California 94065 (US).

(72) Inventors: **LITTLE, Todd**; 1155 W. Illinois Avenue, Palatine, Illinois 60067 (US). **LI, Xiangdong**; 9-5-1, Changchunjiexili, Xuanwu District, Beijing (CN). **LV, Xianzheng**; 3-2-11, No. 39, Xisanqijiancaichengxilu, Haidian District, Beijing (CN).

(74) Agents: **MEYER, Sheldon R.** et al.; Fliesler Meyer LLP, 650 California Street, Fourteenth Floor, San Francisco, California 94108 (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, 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, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, 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, ML, MR, NE, SN, TD, TG).

Published:

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

(54) Title: SYSTEM AND METHOD FOR PERSISTING TRANSACTION RECORDS IN A TRANSACTIONAL MIDDLEWARE MACHINE ENVIRONMENT

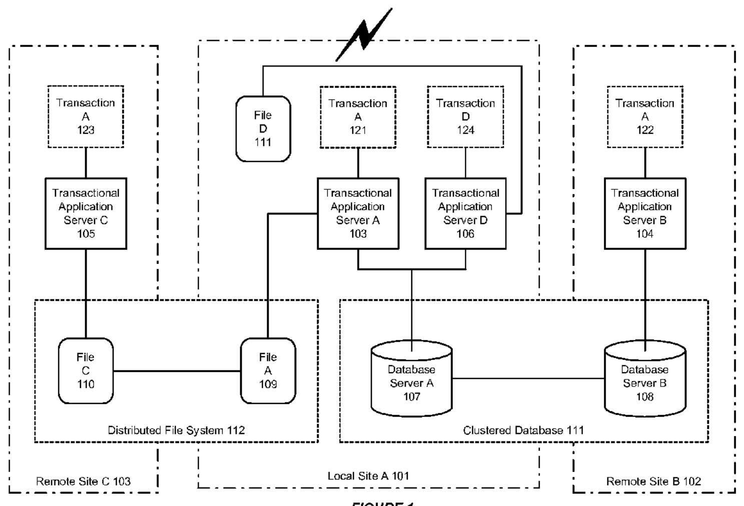

FIGURE 1

(57) **Abstract**: A transactional system can utilize the distributed storage and high availability (HA) capability provided by a clustered database to support easy and feasible disaster recovery. The transactional middleware machine environment comprises one or more transactional application servers associated with a transaction. The one or more transactional application servers operate to persist transactional log information associated with the transaction in a database that connects with said one or more transactional application servers at a local site. The database at the local site operates to replicate the persisted transactional log information to a remote database at a remote site. The remote database allows a different transactional application server at the remote site to recover the persisted transactional log information and complete the transaction, when a disaster disables the local site.

SYSTEM AND METHOD FOR PERSISTING TRANSACTION RECORDS IN A TRANSACTIONAL MIDDLEWARE MACHINE ENVIRONMENT

Copyright Notice:

Field of Invention:

[0002] The present invention is generally related to computer systems and software such as middleware, and is particularly related to supporting disaster recovery in a transactional middleware machine environment.

Background:

[0003] A transactional middleware system, or a transaction oriented middleware, includes enterprise application servers that can process various transactions within an organization. With the developments in new technologies such as high performance network and multiprocessor computers, there is a need to further improve the performance of the transactional middleware. These are the generally areas that embodiments of the invention are intended to address.

Summary:

[0004] Described herein is a transactional system that can utilize the distributed storage and high availability (HA) capability provided by a clustered database to support easy and feasible disaster recovery. The transactional middleware machine environment comprises one or more transactional application servers associated with a transaction. The transactional application servers can persist transactional log information associated with the transaction into a database that connects with one or more transactional application servers at a local site. The database at the local site can replicate the persisted transactional log information to a database at a remote site. When a disaster disables the local site, the remote database allows a different transactional application server at the remote site to recover the persisted transactional log information and complete the transaction.

Brief Description of the Figures:

[0005] **Figure 1** shows an illustration of a transactional middleware machine environment that supports disaster recovery, in accordance with an embodiment of the invention.

[0006] **Figure 2** illustrates an exemplary flow chart for supporting disaster recovery in a transactional middleware machine environment, in accordance with an embodiment of the invention.

[0007] **Figure 3** shows an illustration of a complex interface in a transactional middleware machine environment, in accordance with an embodiment of the invention.

[0008] **Figure 4** is a functional block diagram of an apparatus for supporting disaster recovery in a transactional middleware machine environment in accordance with some embodiments.

Detailed Description:

[0009] Described herein is a system and method for supporting a transactional middleware system, such as Tuxedo, that can take advantage of fast machines with multiple processors, and a high performance network connection. A transactional system can utilize the distributed storage and high availability (HA) capability provided by a clustered database to support easy and feasible disaster recovery. The transactional middleware machine environment comprises one or more transactional application servers associated with a transaction. Said one or more transactional application servers operate to persist transactional log information associated with the transaction to a database that connects with said one or more transactional application servers at a local site. The database at the local site operates to replicate the persisted transactional log information to a remote database at a remote site. The remote database allows a different transactional application server at the remote site to recover the persisted transactional log information and complete the transaction, when a disaster disables the local site.

[0010] In accordance with an embodiment of the invention, the system comprises a combination of high performance hardware, e.g. 64-bit processor technology, high performance large memory, and redundant InfiniBand and Ethernet networking, together with an application server or middleware environment, such as WebLogic Suite, to provide a complete Java EE application server complex which includes a massively parallel in-memory grid, that can be provisioned quickly, and can scale on demand. In accordance with an embodiment, the system can be deployed as a full, half, or quarter rack, or other configuration, that provides an application server grid, storage area network, and InfiniBand (IB) network. The middleware machine software can provide application server, middleware and other functionality such as, for example, WebLogic Server, JRockit or Hotspot JVM, Oracle Linux or Solaris, and Oracle VM. In accordance with an embodiment, the system can include a plurality of compute nodes, IB switch gateway, and storage nodes or units, communicating with one another via an IB network. When implemented as a rack configuration, unused portions of the rack can be left empty or occupied by fillers.

[0011] In accordance with an embodiment of the invention, referred to herein as "Sun Oracle Exalogic" or "Exalogic", the system is an easy-to-deploy solution for hosting middleware or application server software, such as the Oracle Middleware SW suite, or Weblogic. As described herein, in accordance with an embodiment the system is a "grid in a box" that comprises one or more servers, storage units, an IB fabric for storage networking, and all the other components required to host a middleware application. Significant performance can be delivered for all types of middleware applications by leveraging a massively parallel grid architecture using, e.g. Real Application Clusters and Exalogic Open storage. The system delivers improved performance with linear I/O scalability, is simple to use and manage, and delivers mission-critical availability and reliability.

[0012] In accordance with an embodiment of the invention, Tuxedo is a set of software modules that enables the construction, execution, and administration of high performance, distributed business applications and has been used as transactional middleware by a number of multi-tier application development tools. Tuxedo is a middleware platform that can be used to manage distributed transaction processing in distributed computing environments. It is a proven platform for unlocking enterprise legacy applications and extending them to a services oriented architecture, while delivering unlimited scalability and standards-based interoperability.

[0013] In accordance with an embodiment of the invention, a transactional middleware system, such as a Tuxedo system, can take advantage of fast machines with multiple processors, such as an Exalogic middleware machine, and a high performance network connection, such as an Infiniband (IB) network.

[0014] In accordance with an embodiment of the invention, the transactional system can utilize the distributed storage and high availability (HA) capability provided by a clustered database to support easy and feasible disaster recovery.

Persisting Transaction Records into a Clustered Database

[0015] In accordance with an embodiment of the invention, a transactional application server can persist transaction records, such as transactional log information, into a clustered database to enhance high availability (HA) capability for transactions in a transactional middleware machine environment. The clustered database can replicate such transactional information from a local site to a remote site to provide HA capability for the transactional middleware machine environment.

[0016] **Figure 1** shows an illustration of a transactional middleware machine environment that supports disaster recovery, in accordance with an embodiment of the invention. As shown in **Figure 1**, one or more transactional application servers (Server A 103 and Server D 106) can persist transactional information associated with a transaction (Transaction A 121) into a clustered database, e.g. a Database Server A 107 at a local site A 101. The clustered database 111 can replicate the transaction log information to a remote database server B 108 in a remote site B 102.

[0017] As shown in **Figure 1**, when a disaster strikes the local site A, a transactional application server B 104 at the remote site B can pick up the persisted transaction log information for transaction A 122 in database server B from a remote database and complete the transaction immediately, without delay. On the other hand, if the transaction log for a transaction D 124 is only written in a flat file D 111 at the local site A, then it may take longer time for the system to perform a disaster recovery and complete the transaction.

[0018] In accordance with an embodiment of the invention, when the transactional system is activated, the transactional system can open a database connection and write records to a database instead of writing to a file. The transactional system can be configured to write transaction log information into the database that the application is currently using in order to save the number of concurrent database connections.

[0019] Additionally, an abstracted transactional file operation interface can be used for storing the transaction log information in a file system, such as a distributed file system 112. The distributed file system can synchronize the transaction log information stored in a file A 109 at the local site A to a file C 110 at a remote site C 1103. Similarly, when a disaster strikes the local site A, a transactional application server C 105 at the remote site C can pick up the stored transaction log information for transaction A 123 in file C and complete the transaction.

[0020] Storing the transaction log information in a file system may not be convenient for disaster-recovery solution, since transaction log information in the flat file has to be synchronized on the recovery site with an unavoidable time window. Additionally, the support for a distributed file system can be limited.

[0021] In accordance with an embodiment of the invention, by persisting transaction log information in a clustered database, the system can exploit the replication and other HA aspects inherent from the underlying database system. The system can also enhance handling of disaster recovery scenarios, especially in a scenario such as cross-site recovery. Also, the need for a distributed file system solutions can be alleviated. The system also reduces complexity for a user, since configuration of a database is generally less complicated and/or already a necessity, e.g. for normal runtime/application work, compared to distributed file systems such as NFS, etc.

[0022] The servers 103-108 may be hardware compute nodes. The transactional application server A 103, the transactional application server D 106 and the database server A 107 may be provided in a rack of a middleware machine at the local site A 101. the transactional application server B 104 and the database server B 108 may be provided in a rack of a middleware machine at the remote site B 102. The transactional application server C 105 may be provided in a rack of a middleware machine at the remote site C 103.

[0023] **Figure 2** illustrates an exemplary flow chart for supporting disaster recovery in a transactional middleware machine environment, in accordance with an embodiment of the invention. As shown in **Figure 2**, at step 201, one or more transactional application servers can persist transactional log information associated with a transaction into a database that connects with said one or more transactional application servers at a local site. Then, at step 202, the database can replicate the persisted transactional log information from a local database server at the local site to a remote database server at a remote site. Finally, at step 203, the system allows a different transactional application server at the remote site to recover the persisted transactional log information from the remote database server and complete the transaction, when a disaster disables the local site.

Transaction Log (TLOG) in Tuxedo

[0024] In the example of Tuxedo, the Transaction Log (TLOG) contains a log record, in which information about transactions can be kept until the transaction is completed. Tuxedo can utilize the distributed storage and HA capability provided by a clustered database, for example an Oracle RAC database, to support easy and feasible disaster recovery. Additionally, the database can replicate the TLOG records to distributed sites to achieve even higher availability.

[0025] In accordance with an embodiment of the invention, a user can create a transaction log for different domains. The user can first create a Universal Device List (UDL) in order to create a TLOG in the Tuxedo environment. The UDL is a map of a transactional file system, such as the Tuxedo file system. The UDL can be loaded into a shared memory when an application is booted. To create an entry in the UDL for the TLOG device, the system can create a UDL on each machine using global transactions. The Bulletin Board Liaison (BBL) can initialize and open the TLOG during the boot process. Then, the user can define the related transaction-related parameters in a configuration file, for example the MACHINES sections in a Tuxedo configuration file, in order to configure the Tuxedo servers.

[0026] In accordance with an embodiment of the invention, an abstracted file operation interface can be provided in the Tuxedo environment. When the interface is activated, Tuxedo can open a database connection instead of opening a file. Thus, Tuxedo can write TLOG into the database instead of writing to file, and the database can replicate the TLOG records to distributed sites to get a higher availability.

[0027] In accordance with an embodiment of the invention, a clustered database can be used to support the persistence of transactional log information. The clustered database such as

Oracle RAC database allows multiple computers to run Oracle RDBMS software simultaneously while accessing a single database. Tuxedo can utilize the distributed storage and HA capability provided by the clustered database. In a clustered database environment, two or more computers (each with an instance) can concurrently access a single database. The system allows an application or a user to connect to either one of the computers and have access to a single coordinated set of data.

[0028] On the other hand, in a non-clustered database, a single instance accesses a single database. The database consists of a collection of data files, control files, and redo logs located on disk. The instance comprises the collection of Oracle-related memory and operating system processes that run on a computer system.

[0029] **Figure 3** shows an illustration of a complex interface in a transactional middleware machine environment, in accordance with an embodiment of the invention. Tuxedo supports two approaches to persist the TLOG records, one approach is to use information stored in a clustered database, and another approach is to use information stored in a file system. As shown in **Figure 3**, in order to support persisting a transaction log record, a Tuxedo application 301 can first access a high-level API, such as the TLOG APIs 302. The tlog APIs provide a set of uniform interfaces to different APIs in Tuxedo for persisting the TLOG record. This high level API, in turn, can choose to invoke one of the two sets of low level APIs that include db_tlog APIs 303, which deal with the TLOG record in ORACLE database, and file_tlog APIs 304, which deal with the TLOG record in file. The high level TLOG APIs can dynamically invoke one of the two low level APIs according Tuxedo's configuration, tuxconfig or dmconfig. This complex interface can be employed to replace an abstracted Tuxedo file operation interface, so that when it is activated, Tuxedo can open a database connection instead of opening a file, and write records to a database instead of a flat file.

[0030] In accordance with an embodiment of the invention, in order to save the number of concurrent database connections, a dedicated item in a configuration file can be used to configure an operation interface in the Tuxedo environment, so that the TLOG can be written into the same database as the Tuxedo application is currently using. Also, if configured, TLOG can be written into the same database as the application is using in the same connection to save the number of concurrent database connections. For example, Oracle XA provides an interface xaoSvcCtx(), which returns the database service handle for a given XA connection. The interface db_tlog does not open a new connection to the database server. Instead, the system makes use of the connection previously opened by the Transaction Processing Monitor *(*TPM*)*.

[0031] In accordance with some embodiments, **Figure 4** shows a functional block diagram of an apparatus 1000 for supporting disaster recovery in a transactional middleware machine environment configured in accordance with the principles of the invention as described above.

The functional blocks of the apparatus 1000 may be implemented by hardware, software, or a combination of hardware and software to carry out the principles of the invention. It is understood by persons of skill in the art that the functional blocks described in **Figure 4** may be combined or separated into sub-blocks to implement the principles of the invention as described above. Therefore, the description herein may support any possible combination or separation or further definition of the functional blocks described herein.

[0032] As shown in **Figure 4**, the apparatus 1000 for supporting disaster recovery in a transactional middleware machine environment comprises: a persisting unit 1100, a replicating unit 1200 and a recovering unit 1300. The persisting unit 1100 persists, via one or more transactional application servers, transactional log information associated with a transaction into a database that connects with said one or more transactional application servers at a local site. The replicating unit 1200 replicates, via the database, the persisted transactional log information from a local database server at the local site to a remote database server at a remote site. The recovering unit 1300 allows a different transactional application server at the remote site to recover the persisted transactional log information from the remote database server and complete the transaction, when a disaster disables the local site.

[0033] In some embodiments, the apparatus 1000 further comprises a reusing unit 1400 for allowing said one or more transactional application servers to reuse an existing database connection to the local database server.

[0034] In some embodiments, the apparatus 1000 further comprises a writing unit 1500 for writing the transactional log information into a transactional log file.

[0035] In some embodiments, the apparatus 1000 further comprises a supporting unit 1600 for supporting the transactional log file in a distributed file system.

[0036] In some embodiments, the apparatus 1000 further comprises a synchronizing unit 1700 for synchronizing the transactional log information in the transactional log file on a recovery site.

[0037] In some embodiments, the transaction is a two-phase transaction.

[0038] In some embodiments, the transactional log information contains persist state of a transaction manger after successfully committing a first phase of the two-phase transaction.

[0039] In some embodiments, the transactional log information is based on a map of a transactional file system and is maintained in a transactional log record.

[0040] In some embodiments, the apparatus 1000 further comprises a creating unit 1800 for creating the map of the transactional file system using a global transaction and opening the transactional log record when a transaction process is booted.

[0041] In some embodiments, the transactional log information contains information for multiple transactional domains.

[0042] Another embodiment comprises a system for supporting disaster recovery in a transactional middleware machine environment, comprising one or more transactional application servers at a local site associated with a transaction; and a database that connects with said one or more transactional application servers, wherein said one or more transactional application servers operate to persist transactional log information associated with the transaction to the database, and wherein the database operates to replicate the persisted transactional log information from a local database server at the local site to a remote database server at a remote site, and wherein the remote database allows a different transactional application server at the remote site to recover the persisted transactional log information and complete the transaction, when a disaster disables the local site.

[0043] Another embodiment comprises a system wherein said one or more transactional application servers reuse an existing database connection to the local database server.

[0044] Another embodiment comprises a system wherein the transactional log information can be written into a transactional log file.

[0045] Another embodiment comprises a system wherein the transactional log file is supported in a distributed file system.

[0046] Another embodiment comprises a system wherein the transactional log information in the transactional log file is synchronized on a recovery site.

[0047] Another embodiment comprises a system wherein the transaction is a two-phase transaction.

[0048] Another embodiment comprises a system wherein the transactional log information contains persist state of a transaction manger after successfully committing a first phase of the two-phase transaction.

[0049] Another embodiment comprises a system wherein the transactional log information is based on a map of a transactional file system and is maintained in a transactional log record.

[0050] Another embodiment comprises a system wherein the map of the transactional file system is created using a global transaction and the transactional log record is opened when a transaction process is booted.

[0051] Another embodiment comprises a system wherein the transactional log information contains information for multiple transactional domains.

[0052] Another embodiment comprises an apparatus for supporting disaster recovery in a transactional middleware machine environment, comprising means for persisting, via one or more transactional application servers, transactional log information associated with a transaction into a database that connects with said one or more transactional application servers at a local site; means for replicating, via the database, the persisted transactional log information from a local database server at the local site to a remote database server at a remote site; and means for allowing a different transactional application server at the remote site to recover the persisted transactional log information from the remote database server and complete the transaction, when a disaster disables the local site.

[0053] Another embodiment comprises an apparatus further comprising means for allowing said one or more transactional application servers to reuse an existing database connection to the local database server.

[0054] Another embodiment comprises an apparatus further comprising means for writing the transactional log information into a transactional log file.

[0055] Another embodiment comprises an apparatus further comprising means for supporting the transactional log file in a distributed file system.

[0056] Another embodiment comprises an apparatus further comprising means for synchronizing the transactional log information in the transactional log file on a recovery site.

[0057] Another embodiment comprises an apparatus wherein the transaction is a two-phase transaction.

[0058] Another embodiment comprises an apparatus of claim 36, wherein the transactional log information contains persist state of a transaction manger after successfully committing a first phase of the two-phase transaction.

[0059] Another embodiment comprises an apparatus wherein the transactional log information is based on a map of a transactional file system and is maintained in a transactional log record.

[0060] Another embodiment comprises an apparatus further comprising means for creating the map of the transactional file system using a global transaction and opening the transactional log record when a transaction process is booted.

[0061] Another embodiment comprises an apparatus wherein the transactional log information contains information for multiple transactional domains.

[0062] The present invention may be conveniently implemented using one or more conventional general purpose or specialized digital computer, computing device, machine, or microprocessor, including one or more processors, memory and/or computer readable storage media programmed according to the teachings of the present disclosure. Appropriate software coding can readily be prepared by skilled programmers based on the teachings of the present disclosure, as will be apparent to those skilled in the software art.

[0063] In some embodiments, the present invention includes a computer program product which is a storage medium or computer readable medium (media) having instructions stored thereon/in which can be used to program a computer to perform any of the processes of the present invention. The storage medium can include, but is not limited to, any type of disk including floppy disks, optical discs, DVD, CD-ROMs, microdrive, and magneto-optical disks, ROMs, RAMs, EPROMs, EEPROMs, DRAMs, VRAMs, flash memory devices, magnetic or optical cards, nanosystems (including molecular memory ICs), or any type of media or device suitable for storing instructions and/or data.

[0064] The foregoing description of the present invention has been provided for the purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise forms disclosed. Many modifications and variations will be apparent to the practitioner skilled in the art. The embodiments were chosen and described in order to best explain the principles of the invention and its practical application, thereby enabling others skilled in the art to understand the invention for various embodiments and with various modifications that are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the following claims and their equivalence.

Claims:

What is claimed is:

1. A system for supporting disaster recovery in a transactional middleware machine environment, comprising:

one or more transactional application servers associated with a transaction, wherein said one or more transactional application servers operate to persist transactional log information associated with the transaction to a database that connects with said one or more transactional application servers at a local site, and

wherein the database operates to replicate the persisted transactional log information from a local database server at the local site to a remote database server at a remote site, and wherein the remote database allows a different transactional application server at the remote site to recover the persisted transactional log information and complete the transaction, when a disaster disables the local site.

2. The system of claim 1, wherein said one or more transactional application servers reuse an existing database connection to the local database server.

3. The system of claim 1, wherein the transactional log information can be written into a transactional log file.

4. The system of claim 3, wherein the transactional log file is supported in a distributed file system.

5. The system of claim 3, wherein the transactional log information in the transactional log file is synchronized on a recovery site.

6. The system of claim 1, wherein the transaction is a two-phase transaction.

7. The system of claim 6, wherein the transactional log information contains persist state of a transaction manger after successfully committing a first phase of the two-phase transaction.

8. The system of claim 1, wherein the transactional log information is based on a map of a transactional file system and is maintained in a transactional log record.

9. The system of claim 8, wherein the map of the transactional file system is created using a global transaction and the transactional log record is opened when a transaction process is booted.

10. The system of claim 1, wherein the transactional log information contains information for multiple transactional domains.

11. A method for supporting disaster recovery in a transactional middleware machine environment, comprising:

persisting, via one or more transactional application servers, transactional log information associated with a transaction into a database that connects with said one or more transactional application servers at a local site;

replicating, via the database, the persisted transactional log information from a local database server at the local site to a remote database server at a remote site; and

allowing a different transactional application server at the remote site to recover the persisted transactional log information from the remote database server and complete the transaction, when a disaster disables the local site.

12. The method of claim 11, further comprising allowing said one or more transactional application servers to reuse an existing database connection to the local database server.

13. The method of claim 11, further comprising writing the transactional log information into a transactional log file.

14. The method of claim 13, further comprising supporting the transactional log file in a distributed file system.

15. The method of claim 13, further comprising synchronizing the transactional log information in the transactional log file on a recovery site.

16. The method of claim 11, further comprising allowing the transaction to be a two-phase transaction.

17. The method of claim 16, further comprising allowing the transactional log information to contain persist state of a transaction manger after successfully committing a first phase of the two-phase transaction.

18. The method of claim 11, further comprising allowing the transactional log information to be based on a map of a transactional file system and maintaining the transactional log information in a transactional log record.

19. The method of claim 18, further comprising creating the map of the transactional file system using a global transaction and opening the transactional log record when a transaction process is booted.

20. The method of claim 11, further comprising allowing the transactional log information to contain information for multiple transactional domains.

21. A program to perform the method of any of claims 11-20.

22. A non-volatile media that stores the program of claim 21.

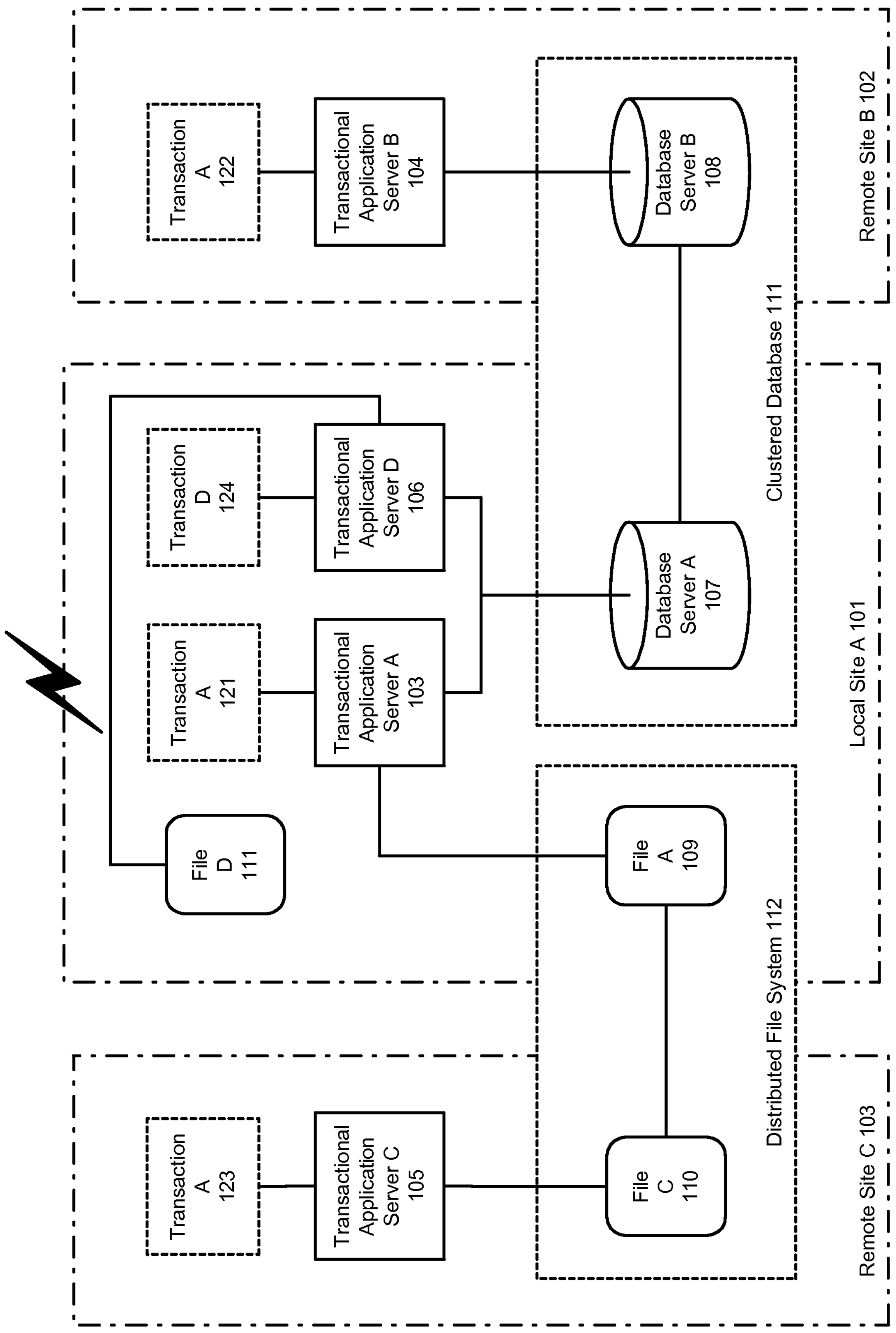
Remote Site B 102
Transaction A 122
Transactional Application Server B 104
Database Server B 108
Clustered Database 111
Local Site A 101
Transaction D 124
Transactional Application Server D 106
Transaction A 121
Transactional Application Server A 103
Database Server A 107
File D 111
File A 109
Distributed File System 112
Remote Site C 103
Transaction A 123
Transactional Application Server C 105
File C 110

FIGURE 1

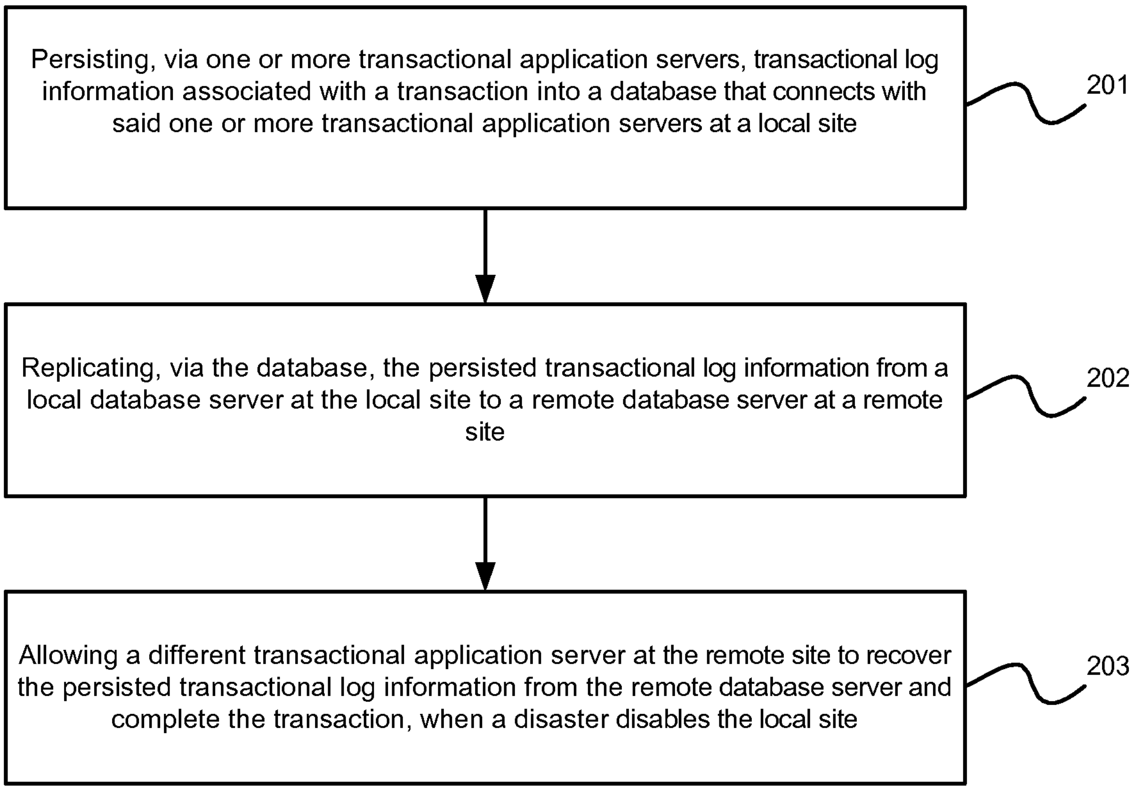
Persisting, via one or more transactional application servers, transactional log information associated with a transaction into a database that connects with said one or more transactional application servers at a local site
201
Replicating, via the database, the persisted transactional log information from a local database server at the local site to a remote database server at a remote site
202
Allowing a different transactional application server at the remote site to recover the persisted transactional log information from the remote database server and complete the transaction, when a disaster disables the local site
203

FIGURE 2

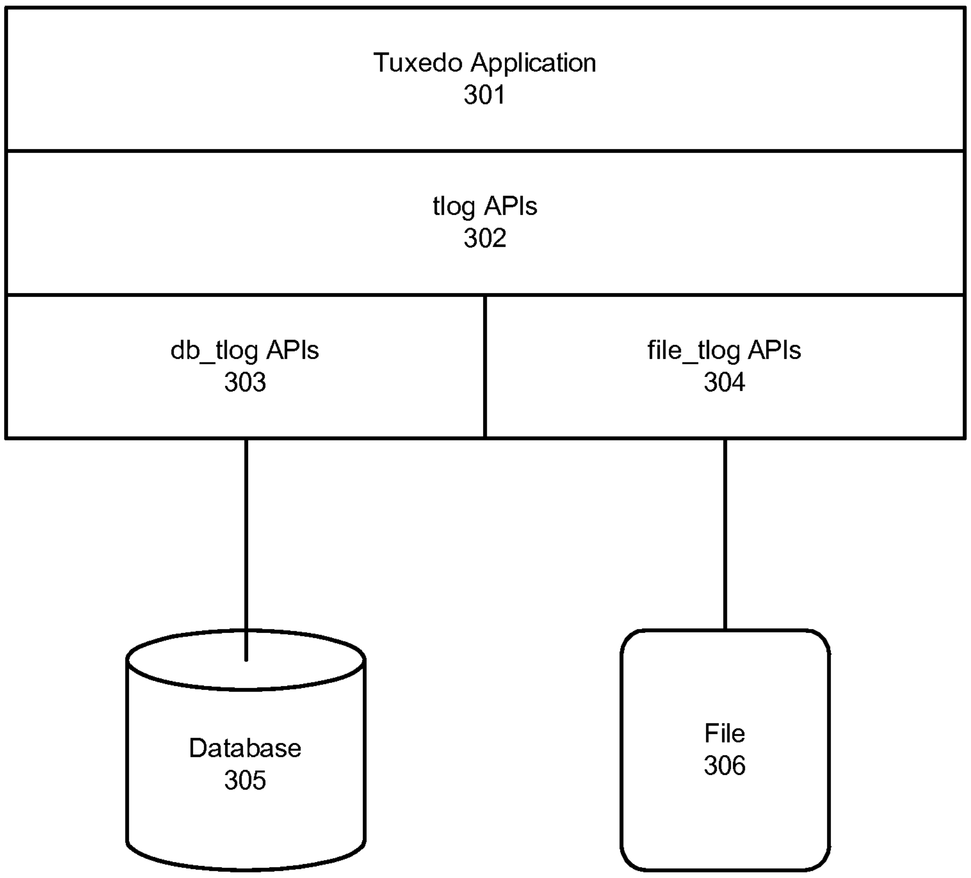
Tuxedo Application
301
tlog APIs
302
db_tlog APIs
303
file_tlog APIs
304
Database
305
File
306

FIGURE 3

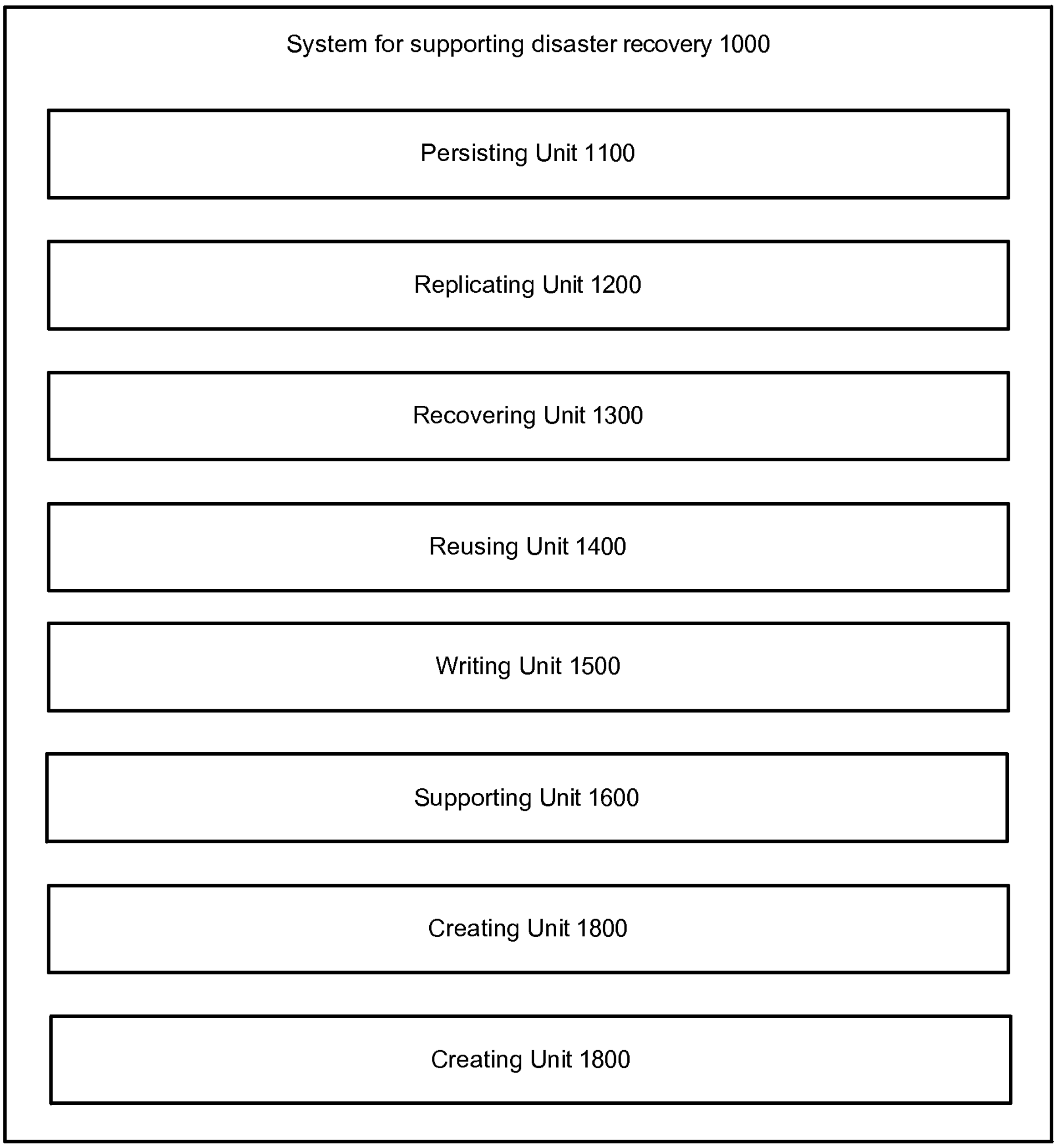
System for supporting disaster recovery 1000
Persisting Unit 1100
Replicating Unit 1200
Recovering Unit 1300
Reusing Unit 1400
Writing Unit 1500
Supporting Unit 1600
Creating Unit 1800
Creating Unit 1800

FIGURE 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US12/56941

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - G06F 11/00 (2012.01)
USPC - 714/4.11, 15, 19; 709/202, 219, 248
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC(8): G06F 11/00, 12/00, 11/14 (2012.01)
USPC - 714/4.11, 15, 16, 19; 707/999.007, 999.202; 709/201, 202, 203, 218, 219, 248

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

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
MicroPatent (US-G, US-A, EP-A, EP-B, WO, JP-bib, DE-C,B, DE-A, DE-T, DE-U, GB-A, FR-A); DialogPRO; IEEE; Google/Google Scholar; replicat*, transaction*, database, log*, remote*, disaster*, recover*, middleware, phase*, domain*

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X --- Y	US 2005/0114285 A1 (CINCOTTA, FA) May 26, 2005, abstract, figure 2-3, paragraph [0011], [0018], [0041], [0044]	1, 3, 5, 11, 13, 15 and 21-22 --- 2, 4, 6-10, 12, 14 and 16-20
Y	US 6,640,244 B1 (BOWMAN-AMUAH, MK) October 28, 2003, figure 102 and column 232, lines 23-35	2, 4, 8-9, 12, 14 and 18-19
Y	US 7,917,470 B2 (BARNES, TE et al.) March 29, 2011, abstract, figure 1A and column 50-59	6-7, 10, 16-17 and 20
Y	US 7,076,508 B2 (BOURBONNAIS, S et al.) July 11, 2006, figure 5, column 7, lines 1-11, column 11, line 61 to column 12, line 4 and column 19, lines 17-28	9 and 19

☐ Further documents are listed in the continuation of Box C. ☐

* Special categories of cited documents:	"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A" document defining the general state of the art which is not considered to be of particular relevance	"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E" earlier application or patent but published on or after the international filing date	"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&" document member of the same patent family
"O" document referring to an oral disclosure, use, exhibition or other means	
"P" document published prior to the international filing date but later than the priority date claimed	

Date of the actual completion of the international search 8 November 2012 (8.11.2012)	Date of mailing of the international search report 3 0 NOV 2012
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Authorized officer: Shane Thomas PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774