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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
24 April 2014 (24.04.2014)

WIPO | PCT

(10) International Publication Number
WO 2014/060011 A1

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

(21) International Application Number:
PCT/EP2012/070399

(22) International Filing Date:
15 October 2012 (15.10.2012)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant *(for all designated States except US)*: **HUAWEI TECHNOLOGIES CO., LTD.** [CN/CN]; Huawei Administration Building, Bantian Longgang, Shenzhen, Guangdong 518129 (CN).

(72) **Inventor; and**
(71) **Applicant** *(for US only)*: **RECHES, Shlomo** [IL/DE]; Huawei Technologies Duesseldorf GmbH, Riesstr. 25, 80992 Munich (DE).

(74) **Agent**: **KREUZ, Georg, M.**; Huawei Technologies Duesseldorf GmbH, Dessauerstr. 3, 80992 Munich (DE).

(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**: METHOD AND APPARATUS FOR MANAGING THE COHERENCY FOR DATA STORED IN DISTRIBUTED DATABASES

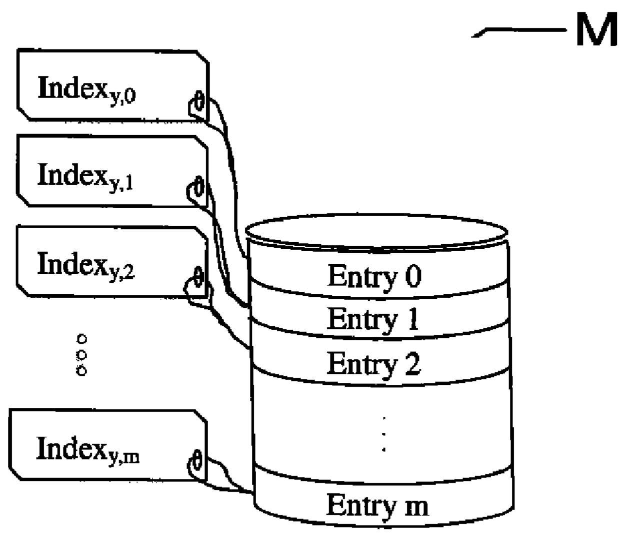

Fig.1

(57) **Abstract**: The invention relates to a method for managing the coherency for data stored in a predetermined number of distributed databases (M) which store distributed database copies of a data entry, wherein in a data entry write operation an data entry is written as a database copy of the data entry to a distributed database (M) along with a calculated data entry index which indicates that the database copy of the data entry comprises up-to-date data of the data entry, wherein in a data entry read operation the database number of the distributed database (M) which stores the database copy comprising the up-to-date data of the data entry is identified by evaluating the data entry indices of all database copies of the data entry and wherein the database copy is read from the distributed database (M) having the identified database number.

TITLE

Method and apparatus for managing the coherency for data stored in distributed databases

TECHNICAL BACKGROUND

The invention relates to a method and apparatus for managing the coherency for data stored in a predetermined number of distributed databases which store distributed database copies of a data entry.

In many applications data is stored in a distributed database. A distributed database is a database in which storage devices or memories are not all attached to a common central processing unit CPU. The data may be stored in multiple computers located in the same physical location or may be dispersed over a network of interconnected computers. Distributed databases can reside on network servers on the internet or corporate intranets or other company networks. The replication and distribution of databases can increase database performance at end-user worksites.

In a conventional distributed database system there are processes involved to ensure that the distributive databases are up-to-date and coherent. Replication involves the search for changes in the distributive database. If a change has been identified, the replication process tries to make the databases coherent to each other. Such a replication process can be very complex and time-consuming depending on the size and number of distributed databases. The conventional replication process can also require a lot of time and computer resources. Duplication on the other hand is not so complicated. Duplication basically identifies one database as a master and then duplicates that database. The duplication process is normally done at a predetermined set time. Both processes try to keep the data current in all distributed database locations.

In distributed databases partial copies of an overall information reside in physically different locations. These locations can be either remote or close to one another. By distributing the database the load can be distributed and a higher database bandwidth can be achieved as well. In a conventional data management system the distributed database management layer is typically a smart load balancer. A critical factor for a distributed data management system is data coherency. Data coherency is, for example, needed for a cache memory or for a multi-core CPU architecture. Further, data coherency is necessary, for example, for database servers cloud computing. For a fast database access the information or data should be replicated in a data storage that is physically close to a client. A coherency problem can arise when the information or data is altered or written by the client. In this scenario only a local data copy is up-to-date and all other data copies can partially be invalid, i.e. a part of the data entries is still valid while other entries are obsolete. After some time and many local data alterations all data copies might be partial and the whole information is no longer held by a single data copy stored in the distributed database.

To overcome the coherency problem it has been suggested to use a centralized supervisor that tracks all of the data access in all distributed data copies. The disadvantages of this conventional approach are the very high bandwidth requirements by the centralized unit. Another conventional approach to overcome the coherency problem has been by mandating that each information alteration must be carried out to all database copies. Further, also in this conventional approach the bandwidth requirements are high.

SUMMARY

Accordingly, it is an object of the present invention to provide an apparatus and a method providing data coherency between distributed database copies of a data entry with reduced bandwidth requirements.

According to a first aspect a method for managing the coherency for data stored in a predetermined number of distributed databases which store distributed database copies of a data entry is provided.

According to a first possible implementation form of the first aspect a method for managing the coherency for data stored in a predetermined number of distributed databases which store distributed database copies of a data entry is provided,
wherein in a data entry write operation an data entry is written as a database copy of the data entry to a distributed database along with a calculated data entry index which indicates that the database copy of the data entry comprises up-to-date data of the data entry,
wherein in a data entry read operation the database number of the distributed database which stores the database copy comprising the up-to-date data of the data entry is identified by evaluating the data entry indices of all database copies of the data entry and wherein the database copy is read from the distributed database having the identified database number.

A data entry comprising up-to-date data along with a data entry index written into only one distributed database by performing single one data entry write operation, and is not written into all other distributed databases, i.e. all other distributed databases are left untouched, during a read operation, the data entry index is identified and the data entry is read from the distributed database identified by the data entry index. By doing this, the data entry comprising up-to-date data can always be read from the distributed database identified by the data entry index, data coherency is thereby ensured.

In a possible second implementation form of the method for managing the coherency for data according to the first aspect the data entry index is identified according to the data entry indices of all database copies of the data entry comprises the data entry index is calculated in a modulo operation depending on the data entry indices of the other database copies of the same data entry stored in the remaining distributed databases.

In a possible third implementation form of the method for managing the coherency for data according to any of the preceding implementation forms of the first aspect, the data entry index for the updated data entry written as a database copy into the distributed database x is calculated as follows:

$$\text{index}_{x,i} = \text{modulo}\left(-\text{index}_{x,i} + \sum_{j=0}^{n-1} \text{index}_{j,i},\ n\right)$$

wherein x is the database number of distributed database in which the database copy of the data entry is written,
i is the data entry number of the data entry,
j is a database number of a distributed database, and
n is the predetermined number of distributed databases,
wherein modulo finds the remainder of division of one number by another.

In a possible fourth implementation form of the method for managing the coherency for data according to any of the preceding implementation forms of the first aspect, the database number of the distributed database which stores the database copy comprising the up-to-date data of the data entry is identified by calculating the database number in a modulo operation depending on the data entry indices of the other database copies of the same data entry stored in the remaining distributed databases.

In a possible fifth implementation form of the method for managing the coherency for data according to any of the preceding implementation forms of the first aspect, the database number is calculated as follows:

$$\text{database number } x = \text{modulo}\left(\sum_{j=0}^{n-1} \text{index}_{j,\,i},\ n\right)$$

wherein i is the data entry number of the data entry,
j is a database number of the distributed database, and
n is the predetermined number of distributed databases.

In a possible sixth implementation form of the method for managing the coherency for data according to any of the preceding implementation forms of the first aspect, an index size of in number of bits of the data entry index is given by:

$$\text{index size} = \text{Roundup}\ [\ \log_2 (n)\ ]$$

wherein n is the number of distributed databases, Roundup, is also called ceiling function which is the mathematical operation of rounding a number up to the next higher integer.

According to a second aspect a data management system for providing data coherency of data stored in a predetermined number of distributed databases which store distributed database copies of a data entry is provided.

In a possible first implementation form of the data management system according to the first aspect the data management system providing data coherency of data stored in a predetermined number of distributed databases which store distributed database copies of a data entry comprises
at least one processing element adapted to write in a data entry write operation an data entry as a database copy of the data entry to a distributed database of said data management system along with a calculated data entry index which indicates that the database copy of the data entry written to the distributed database comprises up-to-date data of the data entry, wherein said data management system further comprises at least one processing element adapted to identify in a data entry read operation the database

number of the distributed database which stores the database copy comprising the up-to-date data of the data entry and wherein the database copy is read from the distributed database having the identified database number.

A data entry comprising up-to-date data along with a data entry index written into only one distributed database by performing single one data entry write operation, and is not written into all other distributed databases, i.e. all other distributed databases are left untouched, during a read operation, the data entry index is identified and the data entry is read from the distributed database identified by the data entry index. By doing this, the data entry comprising up-to-date data can always be read from the distributed database identified by the data entry index, data coherency is thereby ensured.

In a possible second implementation form of the data management system according to the first implementation form of the second aspect the processing element is adapted to calculate the data entry index for an data entry written as a database copy into the distributed database in a modulo operation depending on the data entry indices of the other database copies of the same data entry stored in the remaining distributed databases of said data management system.

In a possible third implementation form of the data management system according to any of the preceding implementation forms of the second aspect, the processing element calculates the data entry index for the data entry written as a database copy to the distributed database as follows:

$$\text{index}_{x,i} = \text{modulo}\left(-\text{index}_{x,i} + \sum_{j=0}^{n-1} \text{index}_{j,i}, n\right)$$

wherein i is the data entry number (entry address) of the data entry,
j is a database number of a distributed database, and
n is the predetermined number of distributed databases,
wherein $\text{index}_{x,i}$ is the index of a database x and address i.

In a possible fourth implementation form of the data management system according to any of the preceding implementation forms of the second aspect, the database number of the distributed database which stores the database copy comprising the up-to-date data of the data entry is identified by calculating a database number in a modulo operation depending on the current data entry indices of the other database copies of the same data entry stored in the remaining distributed databases.

In a possible fifth implementation form of the data management system according to any of the preceding implementation forms of the second aspect, the database number is calculated as follows:

$$\text{database number } x = \text{modulo}\left(\sum_{j=0}^{n-1} \text{index}_{j,\,i},\ n\right)$$

wherein i is the data entry number of the data entry,
j is a database number of the distributed database, and
n is the predetermined of distributed databases.

In a possible sixth implementation form of the first to fifth implementation forms of the data management system according to the second aspect, said data management system comprises a plurality of processing elements, said plurality of processing elements are arranged in a processing element cluster and said distributed databases for said processing elements are arranged in a memory cluster,
wherein the processing element cluster is connected through a crossbar or interconnection to the memory cluster.

In a possible seventh implementation form of the first to fifth implementation forms of the data management system according to the second aspect the data management system comprises a plurality of processing elements each having its local distributed database.

In a possible eighth implementation form of the seventh implementation form of the data management system according to the second aspect a plurality of processing elements each having its local distributed database are connected to each other through a crossbar.

A data entry comprising up-to-date data along with a data entry index written into only one distributed database by performing single one data entry write operation, and is not written into all other distributed databases, i.e. all other distributed databases are left untouched, during a read operation, the data entry index is identified and the data entry is read from the distributed database identified by the data entry index. By doing this, the data entry comprising up-to-date data can always be read from the distributed database identified by the data entry index, data coherency is thereby ensured. Furthermore, the method according to the first aspect and the system according to the second aspect are reliable and robust. Moreover, the data entry index is small in size and occupies only a tiny fraction of the memory space of a distributed database. With the concept according to the first or second aspect it is possible to provide data coherency between data entries of a huge number of distributed databases M without significant data overhead. Accordingly, the concept according to the first and second aspects is resource efficient.

BRIEF DESCRIPTION OF FIGURES

In the following possible implementation forms of a method for managing data according to the first aspect and of the data management system according to the second aspect are described with reference to the enclosed figures in more detail.

Fig. 1	shows a diagram for illustrating a possible implementation form of a method for managing data according to the first aspect;
Fig. 2	shows a diagram for illustrating a possible implementation form of the data management system according to the second aspect;
Fig. 3	shows a diagram for illustrating a further possible implementation form of the data management system according to the second aspect.

DETAILED DESCRIPTION OF EMBODIMENTS

As can be seen in the schematic diagram of Fig. 1 a database M can be adapted to store database copies of data entries. Database M as illustrated in Fig. 1 can form part of a group of distributed databases which store distributed database copies of a data entry. In a possible embodiment there can be a number n of distributed databases. For every database of the n distributed databases each data entry can be attached with a data entry index. As can be seen in Fig. 1 the database M can store m data entries. Each data entry i refers to the same piece of information as a data entry j of another database copy stored in another database within the system. With the method and system according to the present invention not all data entries i hold the up-to-date data of the respective data entry but at least one data entry carries this up-to-date data. With the method according to the first aspect it is possible to identify which each database copy holds this valid data entry having the up-to-date data. In a possible implementation form of the method for managing data according to the first aspect the method returns the database number of the distributed database which stores a database copy comprising the up-to-date data of the respective data entry. As can be seen in Fig. 1 the database y ($y \in 0,\dots,n$) can comprise data entry indices attached to the respective data entries.

With the method for managing the coherency for data according to the first aspect it is possible to perform for each data entry a data entry write operation and a data entry read operation. In a data entry write operation an updated data entry is written as a database copy of the respective data entry to distributed database M as shown in Fig. 1 along with the calculated data entry index which indicates that the respective database copy of the data entry written to the distributed database comprises the up-to-date data of the respective data entry. This data entry index for an updated data entry written as a database copy into the distributed database is calculated in a possible implementation form of the method according to the first aspect in a modulo operation depending on the current data entry indices of the other database copies of the same data entry stored in the remaining distributed databases.

In a possible implementation form of the method according to the first aspect the data entry index of the updated data entry written as a database copy into a distributed database x is calculated as follows:

$$\text{index}_{x,i} = \text{modulo}\left(-\text{index}_{x,i} + \sum_{j=0}^{n-1} \text{index}_{j,i},\ n\right) \qquad (1)$$

wherein i is the data entry number (entry address) of the respective data entry, which is an integer equal to or greater than 0,
j is a database number of a distributed database, $0 \leq j < n$, and
n is the predetermined number of distributed databases within the system, which is an integer equal to or greater than 0,
wherein $\text{index}_{x,i}$ is the index of a database x and address i, $0 \leq x < n$,

wherein modulo finds the remainder of division of one number by another. Given two positive numbers, a (the dividend) and n (the divisor), a modulo n (abbreviated as a mod n) can be thought of as the remainder, on division of a by n. For instance, the expression "5 mod 4" would evaluate to 1 because 5 divided by 4 leaves a remainder of 1, while "9 mod 3" would evaluate to 0 because the division of 9 by 3 leaves a remainder of 0; there is nothing to subtract from 9 after multiplying 3 times 3.

In the method for managing the coherency for data according to the first aspect in a data entry read operation the data entry indices of all database copies of the respective data entry are evaluated to identify a database number of the distributed database which stores a database copy comprising the up-to-date data of the respective data entry. The database copy is read from the respective distributed database with the identified database number.

In a possible implementation form of the method according to the first aspect the database number of the distributed database which stores the database copy comprising the up-to-date data of the data entry is identified by calculating a database number in a modulo operation depending on the current data entry indices of the other database copies of the same data entry stored in the remaining distributed databases. In a possible implementation form the database number is calculated as follows:

$$\text{database number} = \text{modulo}\left(\sum_{j=0}^{n-1} \text{index}_{j,\,i},\ n\right) \qquad (2)$$

wherein i is the data entry number of the respective data entry, which is an integer equal to or greater than 0,
j is a database number of the distributed database, $0 \leq j < n$, and
n is the predetermined number of distributed databases which is an integer equal to or greater than 0,
wherein $\text{index}_{j,\,i}$ is the index of a database j and address i.

In the method for managing data according to the first aspect the index size of a data entry index is given by:

$$\text{index size} = \text{Roundup}\left[\log_2(n)\right] \qquad (3)$$

wherein n is the number of distributed databases equal to or greater than 0, Roundup, is also called ceiling function which is the mathematical operation of rounding a number up to the next higher integer.

The size of the data entry index is small, i.e. the memory space occupied by storing the data entry index along with the data entry is very low. Accordingly, the method for managing data using data entry indices which indicate where up-to-date data of a respective data entry can be found in the system is very efficient with respect to the low occupied memory.

In the following a specific exemplary implementation form for a method for managing the coherency for data stored in a predetermined number of distributed databases which store distributed database copies of a data entry is explained briefly for a simple scenario. If the system, for example, comprises only three databases M-0, M-1, M-2, there is a predetermined number of n=3 distributed databases M which can store distributed database copies of the same data entry. A processor element PE having valid up-to-date data of a data entry to be stored in the distributed databases M-0, M-1, M-2 does perform with the method according to the present invention only one single data entry write operation into one distributed database and labels the respective database as the database which holds the up-to-date data for the respective data entry. If in the given example the processor element PE stores the data entry into the second database M-1 and in a situation where the other two databases M-0 and M-2 have in a possible exemplary first scenario a stored data entry index of “1” for the respective data entry the processor can perform during the write operation a calculation of the data entry index for the updated data entry as follows:

$$(1+x+1) \bmod_3 = 1$$

Accordingly, the calculated data entry index for the second database M-1 of the three databases M-0, M-1, M-2 is calculated in this simple example as x=2.

This calculated data entry index can be written by the processor element PE along with the data entry into the respective distributed in this second exemplary scenario database M-1.

If for example in the same data management system the same processor element PE stores as writes the valid up-to-date data of the data entry into the second database M-1 but the first database M-0 has a data entry index of 1, whereas the third database M-2 has a data entry index of 0, the data entry index for the second database M-1 will be calculated as follows:

$$(1+x+0)\ \text{modulo}_3 = 1$$

so that the calculated data entry index for the second database M-1 is calculated to x=0. Accordingly, the processor element PE will write the data entry to the second database M-1 along with the calculated data entry index of 0.

Later, when the data entry has to be read from the databases M-0, M-1, M-2 the data entry indices of all database copies of the respective data entry are evaluated to identify the correct database number of the distributed database which stores a database copy comprising the up-to-date data of the respective data entry. In a possible implementation form the database number of the distributed database which stores the database copy comprising the up-to-date data of the data entry is identified by calculating the database number in a modulo operation depending on the current data entry indices of the other database copies of the same data entry stored in the remaining distributed databases. In a possible implementation form the database number is then calculated as the above equation (2). In the given simple example the number n of the predetermined number of distributed databases is n=3.

Accordingly, in this simple example with three databases M-0, M-1, M-2 three data entry indices are evaluated wherein the data entry index for the first database M-0 is 1, the data entry index for the second database M-1 is 0 and the data entry index for the third database M-2 is 0 if it assumed that for example the last write operation was performed as described above according to the second exemplary scenario.

Accordingly, the processor element PE calculates the database number according to the above equation (4) in this example:

$$\text{database number} = (1+0+0)\ \text{modulo}_3 = 1$$

Consequently, the processor element PE calculates the database number of the database which stores the database comprising the up-to-date data as database number = 1. This is in fact the database number of the second database M-1 which holds the up-to-date data after the last write operation as explained above. Accordingly, in this simple example the processor element PE will read the database copy storing on the second distributed database M-1 having the calculated database number database number = 1.

The index size of the data entry index is in the given simple example:

index size = Roundup [$\log_2(3)$]

wherein 3 is the number of distributed databases M-0, M-1, M-2.

Accordingly, in this simple example the index size comprises 2 bits making it possible to indicate up to four different databases.

According to a second aspect a data management system for providing data coherency of data stored in a predetermined number of distributed databases M is provided. This data management system can comprise a plurality of processing elements PE and databases or memory elements M as shown in Figs. 2, 3.

Fig. 2 shows a possible first implementation form of a data management system comprising a plurality of processing elements PE and memory or databases M. The distributed databases M are adapted to store distributed database copies of a data entry. In the implementation form of Fig. 2 the data management system comprises a processing element cluster PE-C having a plurality of processing elements PE as shown in Fig. 2. The processing element cluster PE-C is connected through a crossbar XB or interconnection to a memory cluster M-C comprising databases or memories M for the processing elements PE of the processing element cluster PE-C.

In an alternative possible implementation form of the data management system according to the second aspect the data management system comprises a plurality of processing elements PE each having a local database M as shown in Fig. 3.

The processing elements PE each having a local database M are connected to each other through a crossbar XB as shown in Fig. 3. In the data management system as shown in the implementation forms of Figs. 2, 3 each processing element PE can perform a data entry write operation or a data entry read operation. In a data entry write operation an updated data entry is written as a database copy of the respective data entry to the distributed database M along with the calculated data entry index which indicates whether the respective database copy of the data entry written to the distributed database comprises the up-to-date data of the respective data entry.

Further, the processing elements PE are also adapted to perform data entry read operations. In a data entry read operation of a processing element PE the processing element PE evaluates the data entry indices of all database copies of the respective data entry to identify a database number of the specific distributed database or memory M which stores the database copy comprising the up-to-date data of the respective data entry. After having evaluated the data entry indices the database copy is read from the respective distributed database or memory M with the calculated database number by the processing element PE.

In the field, the crossbar also known as crossbar switch, cross-point switch, crosspoint switch, or matrix switch is used for connecting multiple inputs to multiple outputs in a matrix manner. A crossbar switch is an assembly of individual switches between multiple inputs and multiple outputs. The switches are arranged in a matrix. If the crossbar switch has X inputs and Y outputs, then a crossbar has a matrix with X x Y cross-points or places where the "bars" cross. At each crosspoint is a switch; when closed, it connects one of X inputs to one of Y outputs.

In a possible implementation form of the method and system according to the present invention each distributed database or memory M is initialized. Even if an application does not require for a content to be defined the data entry indices which form control information are defined. In this implementation form before performing operations in the distributed databases M the data management system initializes all data entry indices.

In the system according to the present invention the number of distributed databases can be a power of 2 or not be a power of 2.

If the memory system comprises a number of databases which is not a power of 2, the initialization is mandatory because values without initialization could be out of the index range. For example, if there are three databases in the system with a two-bit index, an uninitialized index could be, for example, 3 which is inadmissible or illegal. Further to that, more than one index for a certain data entry could be illegal at the same time and hence cannot be updated in a single write command during operation. Only in the case where the number of databases is a power of 2 like 2, 4, 8,16… databases, initialization can be avoided.

An up-to-date data of a data entry to be stored in the distributed databases is written into only one distributed database by performing single one data entry write operation according to the first or second aspect, data coherency is thereby ensured. Furthermore, the method and system according to the first and second aspects are reliable and robust. Moreover, the used data entry index can be calculated in a modulo operation without occupying too much calculation resources of the processing elements PE. The calculated data entry index is small in size and occupies only a tiny fraction of the memory space of a distributed database M. With concept according to the first or second aspect it is possible to provide data coherency between data entries of a huge number of distributed databases M without significant data overhead. Accordingly, the method and system according to the first and second aspects are resource efficient, in particular with respect to calculation resources of a processing element PE and memory resources of the distributed databases M.

CLAIMS

1. A method for managing the coherency for data stored in a predetermined number of distributed databases (M) which store distributed database copies of a data entry,

   wherein in a data entry write operation an data entry is written as a database copy of the data entry to a distributed database (M) along with a calculated data entry index which indicates that the database copy of the data entry comprises up-to-date data of the data entry,

   wherein in a data entry read operation the database number of the distributed database (M) which stores the database copy comprising the up-to-date data of the data entry is identified by evaluating the data entry indices of all database copies of the data entry and wherein the database copy is read from the distributed database (M) having the identified database number.

2. The method according to claim 1,

   wherein the data entry index is identified according to the data entry indices of all database copies of the data entry comprises the data entry index is calculated in a modulo operation depending on the data entry indices of the other database copies of the same data entry stored in the remaining distributed databases.

3. The method according to claim 1 or 2 wherein the data entry index is calculated as follows:

$$\text{index}_{x,i} = \text{modulo}\left(-\text{index}_{x,i} + \sum_{j=0}^{n-1} \text{index}_{j,i},\ n\right)$$

   wherein x is the database number of distributed database (M) in which the database copy of the data entry is written,

   i is the data entry number of the data entry, j is a database number of a distributed database (M), and

n is the predetermined number of distributed databases (M),

wherein $\text{index}_{x,\,i}$ is the index of a database x and address i.

4. The method according to one of the preceding claims 1 to 3,

wherein the database number of the distributed database (M) which stores the database copy comprising the up-to-date data of the data entry is identified by calculating the database number in a modulo operation depending on the data entry indices of the other database copies of the same data entry stored in the remaining distributed databases (M).

5. The method according to one of the preceding claims 1 to 4,

wherein the database number is calculated as follows:

$$\text{database number } x = \text{modulo}\left(\sum_{j=0}^{n-1} \text{index}_{j,\,i},\ n \right)$$

wherein i is the data entry number of the data entry,

j is a database number of the distributed database, and

n is the predetermined number of distributed databases,

wherein $\text{index}_{j,\,i}$ is the index of a database j and address i.

6. The method according to one of the preceding claims 1 to 5 wherein an index size of the data entry index is given by:

index size = Roundup [$\log_2(n)$]

wherein n is the number of distributed databases (M).

7. A data management system for providing data coherency of data stored in a predetermined number of distributed databases (M) which store distributed database copies of a data entry,

wherein said data management system comprises at least one processing element (PE) adapted to write in a data entry write operation an data entry as a database copy of the data entry to a distributed database (M) of said data management system along with a calculated data entry index which indicates that the database copy of the data entry written to the distributed database (M) comprises up-to-date data of the data entry,

wherein said data management system comprises at least one processing element (PE) adapted to identify in a data entry read operation the database number of the distributed database (M) which stores the database copy comprising the up-to-date data of the data entry by evaluating the data entry indices of all database copies of the data entry and wherein the database copy is read from the distributed database (M) having the identified database number.

8. The data management system according to claim 7,

wherein said processing element (PE) is adapted to calculate the data entry index for an data entry written as a database copy into the distributed database (M) in a modulo operation depending on the data entry indices of the other database copies of the same data entry stored in the remaining distributed databases of said data management system.

9. The data management system according to claim 7 or 8,

wherein the processing element (PE) calculates the data entry index for the data entry written as a database copy to the distributed database x as follows:

$$\text{index}_{x,i} = \text{modulo}\left(-\text{index}_{x,i} + \sum_{j=0}^{n-1} \text{index}_{j,i},\ n\right)$$

wherein i is the data entry number of the data entry,

j is a database number of a distributed database (M), and

n is the predetermined number of distributed databases (M),

wherein $\text{index}_{x,i}$ is the index of a database x and address i.

10. The data management system according to one of the preceding claims 7 to 9,

wherein the database number of the distributed database (M) which stores the database copy comprising the up-to-date data of the data entry is identified by calculating the database number in a modulo operation depending on the data entry indices of the other database copies of the same data entry stored in the remaining distributed databases (M).

11. The data management system according to one of preceding claims 7 to 10,

wherein the database number is calculated as follows:

$$\text{database number } x = \text{modulo}\left(\sum_{j=0}^{n-1} \text{index}_{j,i},\ n\right)$$

wherein i is the data entry number of the data entry,

j is a database number of the distributed database (M), and

n is the predetermined number of distributed databases (M),

wherein $\text{index}_{j,i}$ is the index of a database j and address i.

12. The data management system according to one of the preceding claims 7 to 11,

wherein said data management system comprises a plurality of processing elements (PE), said plurality of processing elements (PE) are arranged in a processing element cluster (PE-C) and said distributed databases (M) for said processing elements (PE) are arranged in a memory cluster (M-C),
wherein the processing element cluster (PE-C)is connected through a crossbar (XB) or interconnection to the memory cluster (M-C).

13. The data management system according to one of the preceding claims 7 to 11,

wherein said data management system comprises a plurality of processing elements (PE) each having its local distributed database (M).

14. The data management system according to claim 13,

wherein said plurality of processing elements (PE) each having its local distributed database (M) are connected to each other through a crossbar (XB).

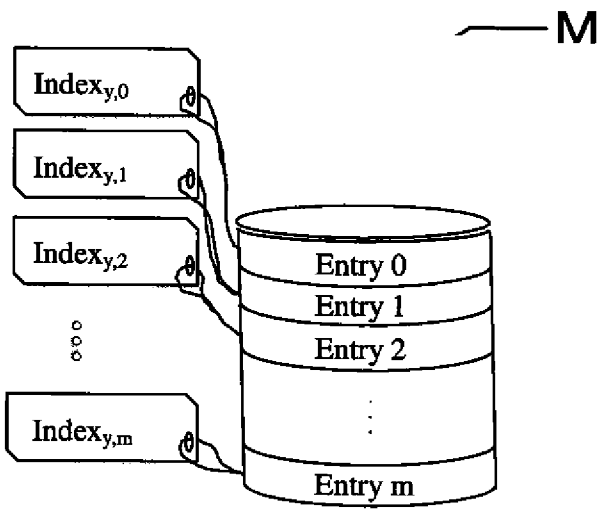
M
$\text{Index}_{y,0}$
$\text{Index}_{y,1}$
$\text{Index}_{y,2}$
$\text{Index}_{y,m}$
Entry 0
Entry 1
Entry 2
Entry m

Fig.1

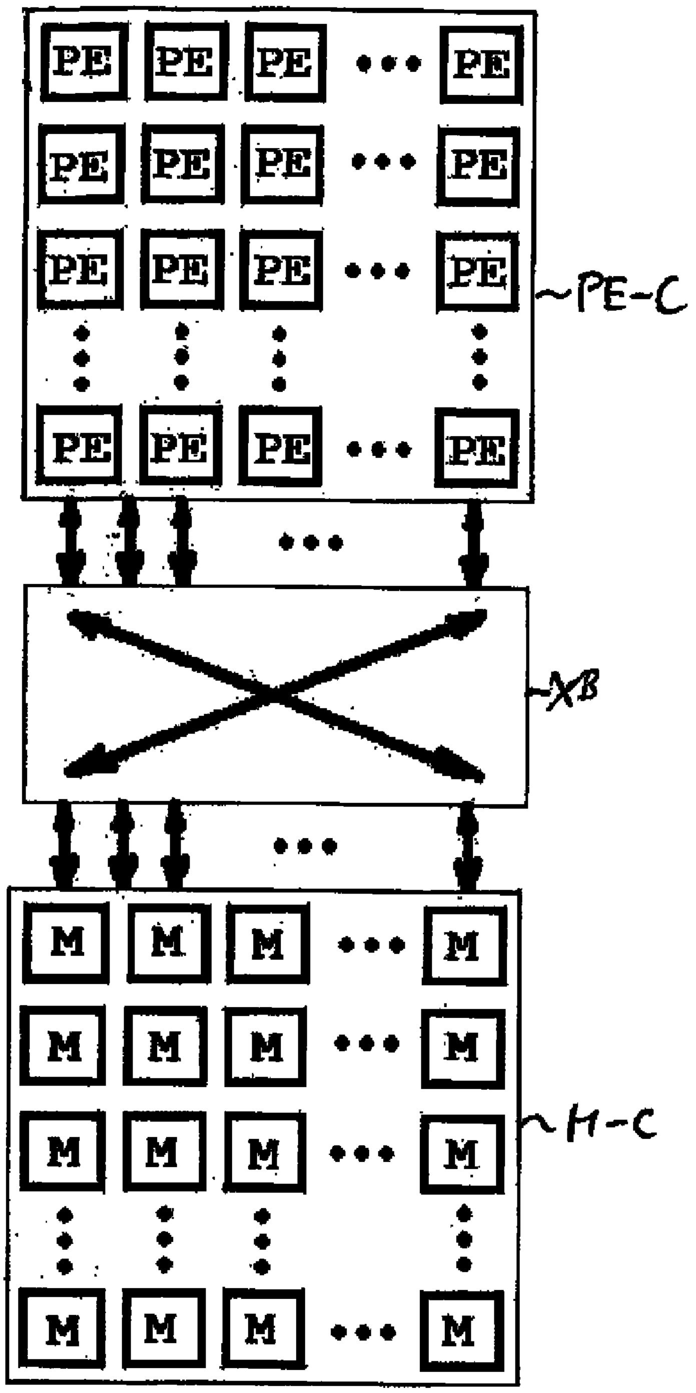

Fig 2

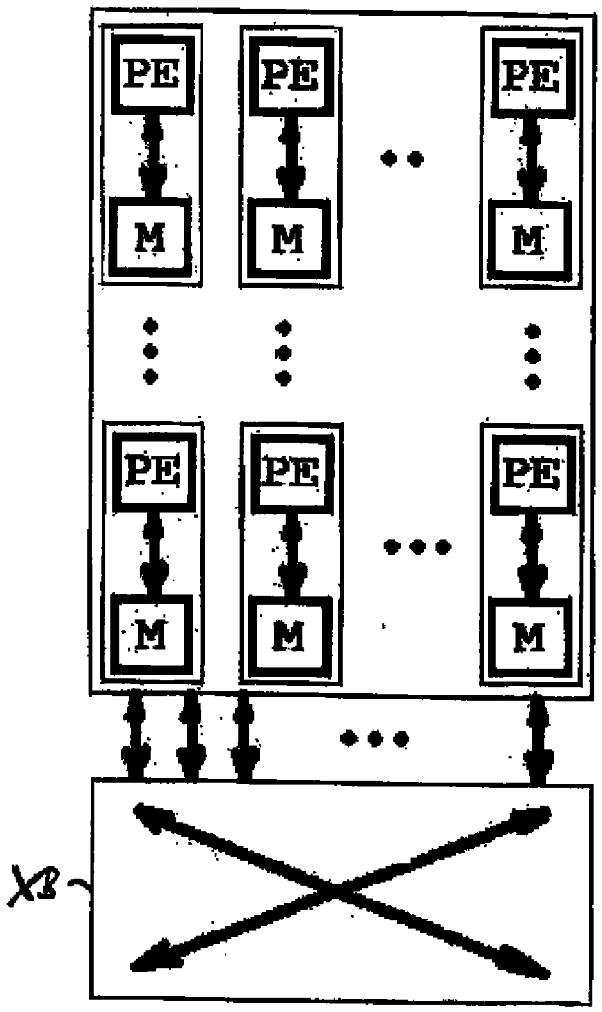

Fig 3

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2012/070399

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F17/30
ADD.

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

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)

EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 1 239 380 A2 (SHARP KK [JP]) 11 September 2002 (2002-09-11) paragraphs [0001], [0049], [0051], [0059], [0072] -----	1-14
X	US 6 944 636 B1 (STARBUCK BRYAN T [US]) 13 September 2005 (2005-09-13) paragraphs [0003], [0010], [0014], [0038], [0041], [0061], [0062], [0064] -----	1-14
A	US 5 710 922 A (ALLEY PETER E [US] ET AL) 20 January 1998 (1998-01-20) column 1, lines 9-12 column 2, lines 4-39; claims 1, 6 -----	1-14
A	US 6 330 568 B1 (BOOTHBY DAVID J [US] ET AL) 11 December 2001 (2001-12-11) column 2, lines 6-14, 33-51 -----	1-14

☐ 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 11 June 2013	Date of mailing of the international search report 18/06/2013
Name and mailing address of the ISA/ European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Authorized officer Haffner, Ronald

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/EP2012/070399

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 1239380 A2	11-09-2002	CN 1374587 A	16-10-2002
		EP 1239380 A2	11-09-2002
		JP 4225729 B2	18-02-2009
		JP 2002334006 A	22-11-2002
		US 2002129007 A1	12-09-2002
US 6944636 B1	13-09-2005	CN 101061475 A	24-10-2007
		JP 4722124 B2	13-07-2011
		JP 2007535757 A	06-12-2007
		KR 20120006082 A	17-01-2012
		US 6944636 B1	13-09-2005
		US 6950835 B1	27-09-2005
		US 2005246399 A1	03-11-2005
		US 2005246400 A1	03-11-2005
		US 2005246462 A1	03-11-2005
		US 2005256995 A1	17-11-2005
US 5710922 A	20-01-1998	NONE	
US 6330568 B1	11-12-2001	NONE	