THE IMPACT OF DISTRIBUTED DATABASES IN BUSINESS DEVELOPMENT

NICOLETA MAGDALENA IACOB (CIOBANU)
UNIVERSITY OF PITESTI, DEPARTMENT OF COMPUTER SCIENCE
TARGU DIN VALE STREET, ARGES, PITESTI, ROMANIA. 110040
nicoleta.iacob_2007@yahoo.com

Abstract:
The economy and the society are in a continuos change, the organizations are expanding and open new offices and branches in various places of the world. In this dynamic environment, the information infrastructure must adapt in order to ensure the smooth running of the business and also maintain the continuous development. In this way, the data management is done with low cost and also high performance. The paper presents the advantages of the distributed databases in business development as an alternative to centralized databases. The distributed databases are the solution to many of the new requirements: fast access and large amount of data. In addition, there are analyzed the available forms and strategies for data replication depending on when and how are needed.

Key words: *distributed database, fragmentation, replication, replication forms, strategies.*

JEL classification: *C61, C88, L21*

1. Introduction

In the *centralized databases* all data are located on a single node, only the users are distributed in the network, and are managed by a single database management system (DBMS). In this case, the disadvantages are, particularly, the high costs of communication and very low reliability and availability, since any error that occurs and blocks access to database breaks the entire network activity. Despite the complexity (compared with centralized databases), *distributed databases* eliminate disadvantages of the centralized database, being the solution to many new requirements, such as: quick access, large volume of data, improved performances.

A **distributed database (DDB)** is a collection of logically interrelated data which are physically distributed on computers (nodes) over a network. Every computer in the network has the autonomy to process local applications. Also, each computer participates in the execution stage of at least one global application that requires accessing data from multiple computers (Iacob, 2010).

A **distributed database management system (DDBMS)** is the software that manages the DDB and provides an access mechanism that makes this distribution transparent to all users.

To ensure compliance with **the DDB's specific objectives** such as: increase of system reliability and data availability, decentralization and better use of system resources and increase of system adaptability to changes in organizational structure, the DDB design aims to follow some *principles*:

- *Maximizing local processing of data* can be done by placing data closer to the applications that require them. In a properly designed DDB approximately 90% of the total data should be locally accessed and only 10% from remote locations.
- *Providing a high level of data security and availability* can be done by making

data replication to numerous sites. In this way, the system can use an alternative copy when the one which should have been accessed under regular circumstances is not available.

- *Parallel data processing* can maximize the use of CPU processing power for each station node.

2. Distributed data storage

A distributed database system is a database system which is fragmented or replicated on the various configurations of hardware and software, located usually at different geographical sites within an organization (Beynon-Davies, 2004).

Consider a relation *r* that will be stored in the database. There are two approaches to store this relation in the distributed database:

- **Replication**. The system maintains several identical replicas (copies) of the relation, and stores each replica at a different site. The alternative to replication is to store only one copy of relation *r*.
- **Fragmentation**. The system partitions the relation into several fragments, and stores each fragment at a different site.

The **fragmentation** is the partitioning of a global relation *R* in fragments R_1, R_2,...,R_n, containing enough information to reconstruct the original relation *R* (Tambulea, 2003).

Fragmentation and replication can be combined: A relation can be partitioned into several fragments and there may be several replicas of each fragment. In the following subsections, we elaborate on each of these techniques (Silberschatz, Korth & Sudarshan, 2010).

3. Data replication

Replication is a process that consists in making and distributing copies of data between different places, at remote or mobile users using Internet connection. In addition, allows the changes to be propagated consistently to the appropriate copies. Distribution of these replicas is done in order to process data locally. Furthermore, replication provides more reliability, minimizes the chance of total data loss, and greatly improves disaster recovery (Rahimi & Haug, 2010).

In a computer network is very important to not allow access to unauthorized persons to the information sent between computers (Defta, 2010). The replication process increases system security and improves speed of data processing operations. In addition, the application can run even if a local server would fail, but other servers with replicated databases remain accessible. In case of data replication, is more difficult to ensure data consistency because the update of a database fragment must be propagated to all copies of the fragment from the replicated databases. Replication decision in case of a data fragment is influenced by the report between read and write processes for that fragment.

3.1. Replication types

In terms of replication, a database can be:

a) **not replicated** - when the system has a single database, which is located on one node;

b) **partially replicated** – when there are data that are replicated and data that are not replicated;

c) **fully replicated** - when the entire database is fully replicated on one or more

network nodes.

In Table 1, we present a comparison of these types of data replication in a distributed database.

Table 1: Types of replication

	Not replicated	**Fully replicated**	**Partially replicated**
Data consistency	Large	Small	Small
Storage costs	Small	Large	Medium
Reliability and availability	Small	Large	Medium
Update speed	Medium	Small	Small
Communication costs	Large	Large	Small

An important aspect is that replication does not usually involve the entire database (in which case we are dealing with a full replication). The most common case is to replicate a table, but often replication of data subsets from one or more tables is also used, which make the mechanism of replication even more complex.

3.2. Forms of replication

There are two basic parameters to set in the design process of a replication strategy, *"where"* and *"when"* the updates are propagated. So, depending on ***when*** parameter, the replication can be: **synchronous** and **asynchronous.**

a) In **synchronous replication**, replicas are kept in sync at all times (Rahimi & Haug, 2010). Synchronous replication means that an update should be propagated and applied immediately in all replicas, so this way database consistency is assured all the time. In this approach, a transaction may access any copy because the accessed data are the same in all replicas from different locations.

b) Since synchronous replication is often related to distributed databases and involves a very high level of complexity, which often translates into very high costs (both for communication, design and operation), **asynchronous replication** is used as an alternative, because is more cheaper. In asynchronous replication, unlike synchronous replication, the replicas are not always kept synchronized and this means that synchronization process of databases occurs only periodically. Two or more replicas of the same data can sometimes have different values and in a transaction these different values may be seen. This is acceptable for some applications that don't require the updates to be made in real time.

Depending on ***where*** the updates can take place we have: **primary copy (master)** and **update everywhere (group).**

c) With an **update everywhere** approach, changes can be initiated at any replicas. This means that any of the *sites* that have a copy of the information can update its value. Each location may make the data updates, and changes are replicated to all other nodes. It is obvious that this configuration is the most exposed to integrity loss, due to conflicting updates, also called "conflicts".

d) With a **primary copy** approach, there is only one copy that can be updated (master copy), all others (secondary copies) being updated in order to reflect changes in the master copy.

3.3. Replication database strategies

Previous ideas can be combined in four different data replication strategies in a

distributed database, shown by comparison in Table 2 below:

- *Synchronous - Primary copy*
- *Synchronous - Update everywhere*
- *Asynchronous - Primary copy*
- *Asynchronous - Update everywhere*

Table 2: Replication database strategies

	Synchronous - Primary copy	**Synchronous - Update everywhere**	**Asynchronous - Primary copy**	**Asynchronous – Update everywhere**
Inconsistencies?	No	No	Yes	Yes
Response time	Large	Large	Small	Small
Updates coordinated?	No	Yes	It is not need for coordination	No

4. Conclusions

The current trends in the economy at regional, national and global level dictate a clear orientation towards the development of complex organizational structures, supported by continuous development of information technologies, that offer unlimited possibilities to support a sales market that exceeds and remove any kind of barriers (cultural, language, politics).

Regarding the organization of information systems in distributed environments, it is necessary to provide numerous features transparently to the users. Distributed systems are attractive for users with diverse and interactive activities which require large processing capabilities, resource sharing and high availability systems. The information systems architecture based on multiple nodes allows data to replicate between different datacenters, ensuring maximum availability of resources, but also business continuity in case of disasters. Replication is used to improve local database performance and protect the availability of applications because there always are alternative data access paths.

REFERENCES

1. Beynon-Davies, P. (2004). *Database systems.* (3rd ed.). New York: Palgrave-Macmillan.
2. Defta, L. (2010). "Network security attacks. ARP Poisoning case study.", *Journal "Constantin Brâncuşi" University of Târgu Jiu Annals* - Economics Series, No. 4/2010, pp. 174-181.
3. Iacob, N. (2010). "Distributed query optimization", *Journal "Constantin Brâncuşi" University of Târgu Jiu Annals* - Economics Series, No. 4/2010, pp. 182-192.
4. Rahimi, S.K.; Haug, F.S. (2010). *Distributed database management systems*. A Practical Approach, IEEE, Computer society, New Jersey: John Wiley & Sons, INC.
5. Silberschatz, A.; Korth, H.F.; Sudarshan, S. (2010). *Database System Concepts*. (6th ed.). McGraw-Hill.
6. Tambulea, L. (2003). *Databases*. (6th ed.). Cluj-Napoca: Univ. Babes-Bolyai.