Asynchronous Data Replication: A National Integration Strategy for Databases on Telemedicine Network

Douglas D. J. de Macedo[1], Hilton W. G. Perantunes[2], Rafael Andrade[1],
Aldo von Wangenheim[2], M.A.R. Dantas[2]
[1] *Programa de Pós-Graduação em Engenharia e Gestão do Conhecimento – PPGEGC*
[2]*Departamento de Informática e Estatística – INE*
Universidade Federal de Santa Catarina (UFSC) – Florianópolis, SC – Brasil
{macedo,andrade,william,awangenh,mario}@inf.ufsc.br

Abstract

Telemedicine systems currently have increased the volume of information stored in their databases. A centralized telemedicine system project covers data sets that ranges from personal patient's information, physicians and institutions to all images of examinations performed. As they store large amounts of examinations, the medical databases can reach several terabytes of volume.

This paper presents a contribution characterized by an asynchronous replication model for medical distributed databases. The model, called Postgres Replication, is an extension to the relational database PostgreSQL. In our experiment, an engine has been created to manage all integration operations and information replication for the medical databases. The early results indicate that the model and its implementation have successfully reached a good performance level and interoperability.

1. Introduction

Telemedicine is based on the use of electronic information and telecommunication technologies to provide distance support for patients, health professionals, clinics and hospitals [1]. And is this data availability even when separated by long distances that has aroused great interest in telemedicine to health institutions and governments. This is also the reason that increases the use of Picture Archiving and Communication System (PACS). The PACS main component is assigned to images distribution and was originally developed for radiology services with the purpose of capturing medical images electronically, instead of using film [2]. Currently, PACS is not limited to the radiology services and has been extended to other clinical services of images as cardiology, pathology, and others [2]. This extension provides various types of medical data and with the help of telemedicine also makes possible remote manipulation and evaluation of this information.

Applications for telemedicine are very specialized, because they depend directly from the examination modalities and the application area. Taking into consideration the specialties, communication mechanisms, security, storage and interaction are designed, directly contributing to the development of a tool for diagnosis support.

Regarding storage mechanisms, the design of a telemedicine system expects that the databases will store large amounts of data. This data set ranges from personal patients information, physicians and institutions to all images of examinations performed, such as, hemodynamic, computerized tomography, magnetic resonance, electrocardiograms, ultrasonography, among others. Because they store such large amounts of examinations, the medical databases often have significant size, with volumes that can reach up several terabytes.

Aiming these problems the Federal University of Santa Catarina (UFSC) [3] together with the Santa Catarina's Secretary of State for Health (SES/SC) developed a research project called *Rede Catarinense de Telemedicina* (RCTM), which provides a number of services for patient health support. The RCTM has telediagnostic services, second medical opinion, and collaborative findings, among other services that are in the development stages.

In the model adopted by the RCTM, the sectors of the health area from geographically dispersed municipalities perform examinations in patients (electrocardiograms, hemodynamic, nuclear medicine, computerized tomography and magnetic resonances). Such examinations are sent in real time to a centralized

1063-7125/08 $25.00 © 2008 IEEE
DOI 10.1109/CBMS.2008.76

database, located in the University Hospital of UFSC. From this moment on, doctors can access the information of the examinations through a Telemedicine portal and, for example, view examinations, provide reports, or interact in a second opinion with other area's professional.

This portal has a database, where the patient's examinations have been stored since the beginning of the network operation, in 2005. Currently the bank presents a data volume of about 800GB.

For illustrating the purpose and importance of the size of this examinations database, Table 1 shows the relationship of examinations and their sizes. Every day several tests from several different patients are conducted and the volume of databases tends only to increase [4].

Currently the RCTM has a group of more than 63 connected municipalities dealing with 12 different examinations modalities and a total available volume of more than 70,000 examinations stored on the database. There are approximately 6,000 new exams each month. Daily, more than 600 people use the *Portal*.

Table 1. Exams and its typical sizes

Exam Type	Average Type	Number of Items	Total Average
Electrocardiogram (ECG)	100Kb	1	100Kb
Nuclear Medicine (NM)	1Mb	5	3Mb
Computed Tomography (TC)	500Kb	100	50Mb
Magnetic Resonances (MR)	250Kb	200	50Mb
X-Ray Angiography (XA)	70Mb	8	560Mb

However, to provide expansion to other states, this centralized database architecture does not support the integration at a countrywide level, because the data consistency would be compromised. This research proposes to study and identify ways to provide scalability, performance and data replication regarding the information integration, contained in distributed medical databases. The issues related to security and access control to patient's data will not be addressed in this stage of the work. The objective is to integrate various distributed databases in different states of a country, composing a single national telemedicine network.

2. Related Work

This section will present some examples of data replication on free software projects. They are: PGPool [5], PGCluster [6], Postgres-R [7] and Slony-l [8]. Though not discussed here, there are also proprietary softwares for this purpose, as IBM DB2 Replication [9], Oracle (RAC) Real Application Clusters [10] and SyBase Replication Server [11].

The Slony-l is a replication system with single-master for multiple slaves. This software is a system designed to work in datacenter and backup sites, where all the other nodes of the system are available and for these reasons implements pessimistic algorithms with an asynchronous replication [8]. If the operating environment is susceptible to be unavailable the Slony-l is not the ideal solution for the project.

The middleware PGPool runs between PostgreSQL and data clients. PGPool offers the following features: connectivity limiter, connectivity pooling, replication, load and parallel query balancing. PGPool replication permits to create a backup in real time for multiple nodes [5]. The major characteristic of the PGPool is to be a pool central server. One disadvantage is that PGPool does not provide adequate functionality for data replication.

The Postgres-R is an extension of the PostgreSQL system, and provides an efficient, fast and consistent replication for databases clusters [7]. This package is used for load balancing and high data availability. However, its replication mode is synchronous and determines a high consumption of data communication.

PGCluster is a system of synchronous replication composition of multi-master for PostgreSQL. In a server organized by multi-master, two or more storage systems can be accessed simultaneously by the user. Due to this synchronous feature, there is no considerably latency with the duplicate data between nodes of the cluster and considering its composition, so the system will be available to competition for access among multiple users [6]. The PGCluster is a system with an interesting architecture, but the software requires high bandwidth as a mandatory infra-structure.

However, because the architecture of the Telemedicine Portal is based on the system manager PostgreSQL database, the solutions here studied and discussed are based on this system. For this reason, a prototype was developed as a concept proof using PostgreSQL, called *Postgres Replication* (PGR).

3. Methodology

3.1 Data Replication.

Data replication has been extensively studied in the databases distributed context [12], [13], [14]. When it involves database studies, the main goal is how to achieve performance with a large number of nodes, and provide autonomy for various replication types [15].

Data replication is to maintain multiple copies of data, called reply, in different devices on one or more networks and is an important technology for

distributed services. Replication improves the data availability, allowing access even if some replicas are unavailable [13].

The traditional techniques of replication seek to maintain the data consistency and make the data be seen as a single highly available data copy to the user [16]. The techniques that implement this solution are called "pessimistic" and can be seen, for example, on systems that elect a primary replica that is responsible for all requests to an object in particular [16]. The pessimistic algorithms coordinate the synchronous access to the replica during the access and prevent other users to update it at the same time. This strategy is suitable for local area networks (LAN) where latencies are small and failures uncommon. However, in long-distance networks, or distributed (WAN) such as the Internet, pessimistic, replication techniques have low performance [13].

Moreover, the replication optimistic is the name given to a set of techniques for efficiently share the data in long distance networks or environments with mobile devices.

Optimistic algorithms allow data to be accessed without the need for prior synchronization, assuming that any problems will occur very rarely [17]. Updates are propagated in the background and the occasional conflicts are further processed. Examples of the use of optimist techniques can be seen in the system for managing Internet Domain Names (DNS) and systems asynchronous collaboration between users, as the CVS software [13].

The optimistic replication algorithms offer some advantages in relation to the pessimistic, as the possibility of the sites can continue operating normally with their replica, even if the communication between them is interrupted or become unstable, in addition to providing greater scalability because of communication between sites is less frequent [13]. The costs of these benefits arise in the data consistency area, since the optimistic replication, in a given time replicas may have divergent data [17]. Thus, the great challenge of the optimistic algorithms is to maintain the consistency of the data and to keep the replicas sufficiently similar among themselves.

One relevant aspect related to the data replication is the number of *replicas* that have permission to record information on other *replicas*. Systems with a single master (single-master or master slave) have all their updates originated from a single *replica* and passed to the other, called *slaves*. The multi-master systems allow the updates to be submitted by multiple *replicas* independently because they provide greater data availability, but also make your management more complex.

3.2 The PGR Proposal

We evaluated the needs presented by a telemedicine system regarding distributed databases and the tools to manage them, which do not provide the flexibility level required for the project. Also, we verified the necessity of applying the concepts of partial replication and integration of the data using the optimistic replication, multi-master and asynchronous databases.

In this context, the expansion of the Telemedicine Portal at a national level requires the need for a tool that assists the replication process between databases. For this, a prototype was developed, using partial multi-master replication, in asynchronous mode and with hybrid fragmentation.

The objective of the PGR is to allow specific subsets of the databases tables to be monitored. From these tables, it is possible to fragment data horizontally and vertically [18], or filter the records through conditions in requires and get only the attributes that interest to the replication application.

This fragmentation is necessary because there will be cases where certain sites will not replicate the medical images. This fact can be explained by the high cost of bandwidth and storage of an entire database to be published to other nodes. For this reason also, in this study we use partial replication.

The prototype is composed by a structure of relational databases in each *replica* of the environment that keeps control information of the system. There is also an application that will run in each server that hosts the databases, performs the connections in other *replicas* and coordinates the replications.

The application requires that the structure of relational tables monitored by the replication system is the same in all databases. This feature is necessary because the strategy used for the data propagation is a multi-master, and each *replica* expects to find on the tables from the other *replicas* fields with the same names and the same data types. In the environment, each *replica* stores information in its database on the other *replicas* at specific tables.

When an update occurs in the database and if this local database is a master on the synchronous environment, a connection starts immediately in the secondary databases and the information is transmitted. Instead of using this approach, the PGR only records the information about changes in your system in an events table. Such information makes reference about the operation performed and the actions that happened.

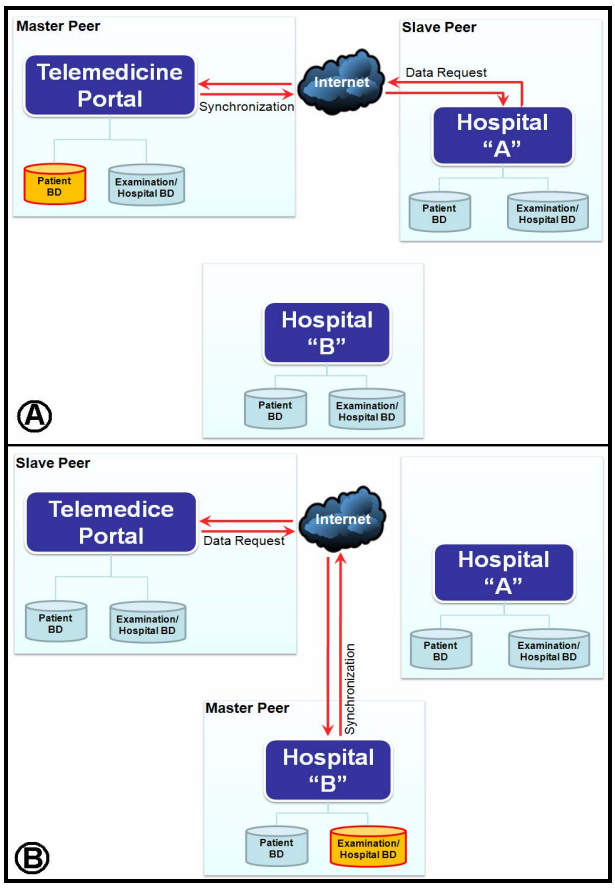


Figure 1. Replicas communication in the environment

In a specific moment, a second replica will connect to the local database and retrieve the information, getting the identifiers from records that have been updated and copying data from their tables to their own database. The moment of the connection is chosen by each *replica*, characterizing the data asynchronous propagation. The transaction timing is stored on the database of the PGR, this way in another connection only the updates are identified and captured.

Figure 1 shows the replication data process in the PGR environment. The propagation from changes realized in each *replica* is performed through transactions sequences that, individually, are similar to the method Master-Slave: a *replica* establishes connections with others, and verifies where there is a recent data updates in each replica (Figure 1A). The information about the queries performed is registered to guide the future synchronization of the *replica* that originated the request and to the other *replicas* that didn't do it yet. In a second moment, another replica will assume the master position (Figure 1B), and will realize the same procedures to propagate changes.

The data obtained on the *replicas* updates are stored in tables called *shadow*, with the same structure of the tables monitored by the application, with an extra column that lists each record to the origin of database. After this step, the queries to the tables in the database are made through associations between tables in their original *shadow* tables, including the data filtering according to their origin.

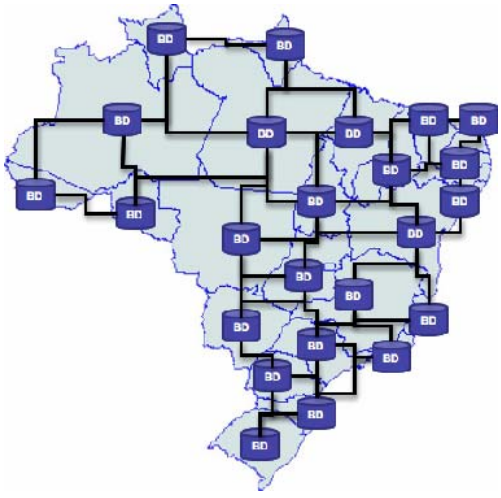
The philosophy is that each health institution should have two different databases: one with information from patients and other with the information of examinations and hospitals where the tests were performed. The information from the patient records are maintained on the institution's database that performed the exam and replicated to the master database.

Thus, the institution's database can replicate the information from the patient records to the master database and inform to the master when an update of the data examinations occurred. If some node needs to replicate the information about examinations, only at this moment the system will replicate the information.

This model can be divided into various databases, creating a branched structure. For example, you can have a centralized State database, one database in each municipal district, a database in each institution and so on, where each one can communicate directly with the master server in a transparent way to the user.

4. Results

To test the methods performance applied in the PGR tool, we prepared some simulation tests of data replication between distributed databases.



Picture 2. Database distribution and connection scenario for tests

Figure 2 shows the scenario that reflects an environment where real operating examinations, video images and other information are transferred.

For the application performance tests, we have used *chunks* with different sizes. Twenty tests were considered for the data replication between remote entities. In the first test we used chunks of *binary large objects (blob)* and in the second test plain text. Table 2, shows the set of data used in the tests.

Table 2. Chunks sizes

	10	100	1000	10000
Text	8 Kb	16 Kb	104 Kb	1128 Kb
Binary Objects	0,34 Mb	26,20 Mb	275,07 Mb	1712 Mb

We performed twenty replications of medical images and simple entries, divided in chunks size of 10, 100, 1,000 and 10,000 records. In the experiment, the time has been measured for transfers between two different sites of computational environment. Thus, it is possible to extract the time when the engine starts the process controller for data transfer and at the moment when the transaction is successful.

For the grouping of the records in collections, we used temporary tables at the site database that contains the original data for each chunk being transmitted. The primary controller site needs to get the structure of these tables to be able to reproduce it on its own database and then copy all data from each table through the procedures of PostgreSQL tables copy. The average time of the transactions, for the records in plain text and binary can be seen in Table 3.

Table 3. Replication Average Time in seconds

	10	100	1000	10000
Text	0.0048	0.0091	0.0170	0.1836
Binary Objects	0.0665	9.6952	104.4264	627.1346

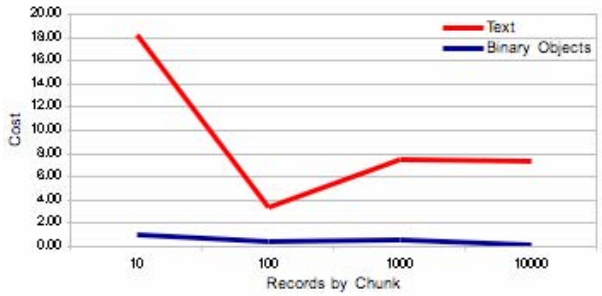
To help visualizing the data obtained from the replication, an algorithm was used to normalize the obtained data magnitude, according to the formula below:

$$y = \frac{|\bar{x} - x_i|}{x_{max}}$$

This normalization was obtained by the absolute value of the difference between each value and the average divided by the maximum of comments by each category. The formula results in values between 0 and 1 allowing a data comparison from different variables with dispersions. These values were multiplied by 100 to create a unit on the cost of shipment, and are listed in Table 4, for each data category.

Table 4. Replications cost per records

	10	100	1000	10000
Text	18.25	3.40	7.46	7.39
Binary Objects	0.97	0.47	0.59	0.08



Picture 3. Data transfer cost

Assessing the experimental results in Figure 3, we can see, when used the same scale, there is less variation in relation to average time of transfer in certain sizes sets, as chunks of text from 100 records compared to those of 10. This suggests that we can find values for a records number, enabling a lower cost in transfer time according with each data type, through the size choice of the sets used by the replication engine.

5. Conclusions and future work

In this paper we presented an interoperability model based on asynchronous replication between distributed medical databases. As concept proof we developed an engine that manages all the integration operations and information replication between medical databases.

Experimental tests were performed with the tool, where it was stable and functional, reaching the goals expected. As a theoretical result we validated the use tool replication techniques and partial optimistic, with fragmentation in a hybrid asynchronous replication environment with distributed databases.

As a practical result of this work, this tool provides the possibility of telemedicine applications and systems deployed in Brazil, to have all the patients data integrated. This model will facilitate the doctors decision-making, because the patient history can be consolidated, thus avoid unnecessary examination duplication, which directly result in cutting costs and indirectly broader patient care.

In experimental tests, the metrics extracted based on replication time between sites were satisfactory. It is important to note that the chunks replication of 100 records of plain text had better performance. For chunks of binary data, the compound per 10,000 records obtained better performance, which suggests

that the greater records of binaries number to be replicated, the better the replication performance.

This distributed storage model allows the user of the system to search for information on the Telemedicine Portal and to find the list of examinations of a certain patient. The examinations performed in another institution will be available through external access via the Telemedicine Portal that informs the user where the information is contained. If the user needs to view the exam, only now the system makes the data replication.

There are many challenges to be resolved for further work. As an example, we can cite the aspects involving the data consistency, the detection and conflicts resolution. Another example is the development of mechanisms to ensure the data integrity and data security and enabling a particular institution to know at any given time if your copy of the data is consistent with the data from other institutions.

6. References

[1] Mcneill K. M.; Weinstein R. S.; Holcomb M. J., Arizona Telemedicine Program: Implementing a Statewide Health Care Network, Journal of the American Medical Informatics Association, Volume 5 ,Number 5 , Sep / Oct 1998.

[2] H.K. Huang, Enterprise PACS and image distribution, Comput. Med. Imaging Graph. 27 (2003), pp. 241–253.

[3] Cyclops Group. http://cyclops.telemedicina.ufsc.br.

[4] Maia, R.S.; Wangenheim, A. von; Nobre, L.F., "A Statewide Telemedicine Network for Public Health in Brazil," *19th IEEE International Symposium on Computer-Based Medical Systems, 2006. CBMS 2006.*, pp.495-500, 2006.

[5] PGPool. http://pgpool.projects.postgresql.org.

[6] PGCluster. http://pgcluster.projects.postgresql.org.

[7] Postgres-R. Eager multi-master replication for Postgres. http://www.postgres-r.org.

[8] Slony-l. Enterprise-level replication system. http://www.slony.info.

[9] Gu, L.; Budd, L.; Cayci, A.; Hendricks, C.; Purnell, M.; Rigdon, C.; A Pratical Guide to DB2 UDB Data Replication V8. http://www.ibm.com/redbooks.

[10] Oracle. Oracle Real Application Clusters (RAC) 11g: *An Oracle Technical White Paper*. April, 2007. http://www.oracle.com.

[11] SyBase. Replication Server: Move and synchronize data across the enterprise with Replication Server. http://www.sybase.com.

[12] Ozsu, T.; Valduriez, P.: *Principles of Distributed Database Systems*. Ed. 2, Pretince Hall, 1999.

[13] Saito, Y. and Shapiro, M. 2005. Optimistic replication. ACM Computer Surveys (CSUR). Vol. 37, n. 1, p. 42-81, 2005.

[14] Mario A. R. Dantas, D. P. Cunha: An Experimental Case Study of Replication on Reconciliation in a Wireless Environment. HPCS 2004: 179-182, 2004.

[15] Coulon, C.; Pacitti, E.; Valduriez, P., "Consistency management for partial replication in a high performance database cluster," Proceedings of 11th International Conference on Parallel and Distributed Systems. Vol. 1, pp. 809-815, 2005.

[16] Bernstein, P. A.; Hadzilacos, V.; Goodman, N. Concurrency Control and Recovery in Database Systems. Massachusetts: Addison Wesley, 1987.

[17] Goel, A.; Pu, C.; Popek, G.J., "View consistency for optimistic replication," Proceedings of Seventeenth IEEE Symposium on Reliable Distributed Systems. pp.36-42, 1998.

[18] Coulon, C.; Pacitti, E.; Valduriez, P., "Scaling up the Preventive Replication of Autonomous Databases in Cluster Systems," VECPAR, LNCS 3402, pp. 170-183, 2004.