Considerations about an Oracle Database Multi-Master Replication

I. Filip*, C.Vasar* and R. Robu*
* Politehnica University from Timisoara, Timisoara, Romania
ioan.filip@aut.upt.ro, cristian.vasar@aut.upt.ro, raul.robu@aut.upt.ro

***Abstract* — In some cases of software-distributed applications, a large data amount must be accessed by a large number of users (dispersed geographically). In these cases, the database replication process provides an efficient solution to increase the application performances. This paper presents two methods to implement a multi-master database replication process. A replication example between two Oracle databases is also described, detailing step by step the operations that must be performed, providing a useful support to solve such problems.**

I. Introduction

Database replication process allows creating and maintaining copies of a database objects (usually tables, indexes, or even code) into a distributed database environment. Replication can improve the performance of a database application, increasing the availability using alternate data access options. Also, replication can be considered as a data backup method (in case of a hardware failure, software corruption, or even a natural disaster).

Oracle provides two main techniques to implement a database replication process. A basic replication is implemented using materialized view or snapshot and can make only data copies (not code). For example, a snapshot can be applied to create a read-only copy of a source table. Therefore replication is not bi-directional and the snapshot copies are read-only. An advanced replication is bi-directional (changes of the copies updating also the sources), therefore even the data copy can be updated. This last replication, often named multi-master replication, provides some advantages comparative with the snapshot replication: possibility to replicate data (tables, indexes) and code (stored procedures, packages, triggers), structural replication of the tables object, replication with a large number of databases [1]. But also some disadvantages can be mentioned: multi-master replication process is complex, the database server performances are reduced into a large data amount, and also the administration of replication process can be difficult.

II. Database multi-master replication

A database multi-master replication involves a set of two or more linked master sites. For example, figure 1 describes a database replication system contains one master definition site and three other master sites. A master site contains a complete copy of all replicated objects. These objects are stored into a replication group, facilitating the administration of related database objects and also the replication process. Many replication groups (referred sometimes as master groups) can exist simultaneous on the same master site. All master sites

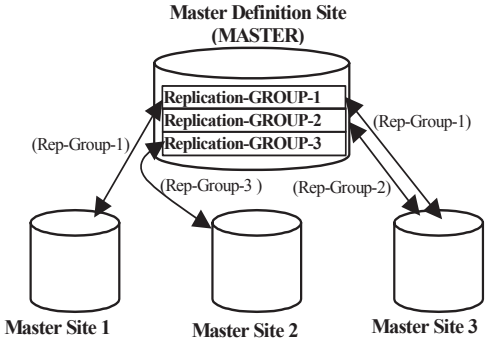

Figure 1. Database replication system

communicate with one another, propagating data/objects (or any schema changes). To facilitate the administration of the replication process, one of the sites is the definition master site, serving as control center to manage the master groups [2]. Many aspects must be taking into consideration when is decided to implement a database multi-master replication [3]:

- What is wanted/needed to replicate (is it necessary a multi-master replication or a simple snapshot replication is enough)?
- Verifying the configuration of data tables (existence of a primary key, foreign keys, and other constraints);
- Setting adequately the database parameters;
- Verifying if the necessary replication built-in packages are properly installed and loaded;
- Creating a replication administrator account (with the necessary privileges);
- Creating properly database-links (building the distributed environment);
- Creating the replication tasks (push-purge jobs queues, firing the replication, timing for tasks, etc).

A. Create Replication Administrator Accounts and Database Links

The default values of the database parameters allow implementing a multi-master replication. However, to the database *init.ora* file, the recommended values of some parameters can be verified and eventually changed (for example: GLOBAL_NAMES = true, SHARED_POOL_SIZE = 80Mb, REPLICATION_DEPENDENCY_TRACKING = true, etc).

The following query permit to verify if the minimal set of needed packages (DBMS_REPCAT, DBMS_DEFER_SYS and DBMS_REPCAT_ADMIN) that are activated:

978-1-4244-4478-6/09/$25.00 ©2009 IEEE

SELECT owner, object_name, status FROM dba_objects WHERE object_name = 'DBMS_DEFER_SYS' or object_name= 'DBMS_REPCAT' or object_name = 'DBMS_REPCAT_ADMIN' AND owner IN ('SYS', 'SYSTEM');

There are also other packages that can be used to implement a replication process, but this paper will refer only to those mentioned above, allowing the following tasks: DBMS_REPCAT_ADMIN - create replication administrator account, DBMS_REPCAT - administer the replication environment, DBMS_DEFER_SYS – manage the replication node lists, performing administrative tasks (scheduling, executing and deleting queued transactions) [4].

A replication of a tables group between two Oracle (10g) database servers (configured as master sites) is the goal of the following exemplification (see the figure 2). One of the first steps supposes setting of the GLOBAL_NAME for each database. Connected as SYSTEM user, the default GLOBAL_NAME can be retrieved by query:

*SELECT * FROM global_name;*

Also, this name can be modified, assigning a new GLOBAL_NAME to the current database (for example):

ALTER DATABASE RENAME GLOBAL_NAME TO orcl1.oracle1;

For the two considered databases (stored on two distinct Oracle servers), assume that the GLOBAL_NAME is *orcl1.oracle1*, respectively *lab.oracle1*. Also, for each database must be created a replication administrator account. (usually named *repadmin* and with a password that can be different for each site):

CREATE user repadmin identified by repadmin;

with the properly privileges. First, user *repdamin* gets usually privileges (GRANT comment any table, grant lock any table, comment any table, select any table, create sequence, create trigger, create view, create synonym, alter session, create materialized view, select any dictionary to *repadmin*). Second, the replication privileges are granted to the user *repadmin*, using procedures from the built-in packages: DBMS_REPCAT_ADMIN, DBMS_DEFER_SYS:

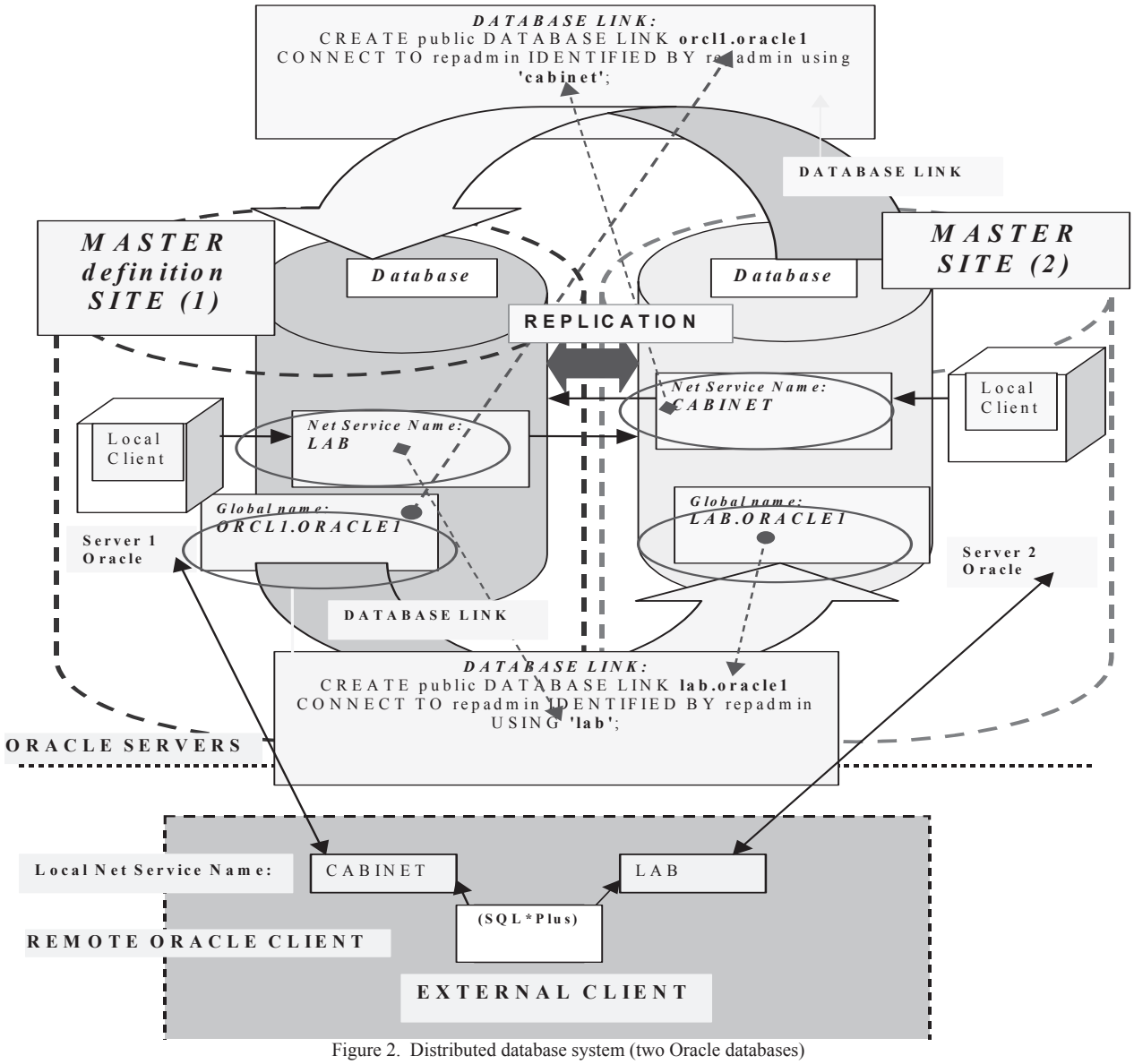

Figure 2. Distributed database system (two Oracle databases)

execute dbms_repcat_admin.grant_admin_any_schema ('repadmin');

exec dbms_defer_sys.register_propagator('repadmin');

execute dbms_repcat_admin.register_user_repgroup (username=> 'repadmin', privilege_type=> 'receiver', list_of_gnames=> null);

The execution of first procedure grants privileges to the replication administrator for administering all the replication groups from the local database (master site). The second procedure registers the user *repadmin* as propagator for the current database (grant also privileges as create session, create procedure, create database link, execute any procedure). As propagator, user *repadmin* can propagate changes to remote database. The last procedure grants the necessary privileges to operate with remote site as a receiver [5].

An important operation that must precede the replication process is the creation of links between all databases involves into replication process. Therefore, a distributed database environment /system must be created. Consider the following prerequisites (see figure 2): LAB is the name of the Net Service Name used as an alias descriptor to make a remote connection from master site 1 to master site 2, respectively CABINET to make a remote connection from master site 2 to master site 1. Thus a bi-directional remote access is allowed. Also, considering the GLOBAL_NAME for each of two servers: ORCL1.ORACLE, respectively LAB.ORACLE1, the creation of link between these master sites require the following commands:

- on master site 1 (logged as system administrator):

CREATE public DATABASE LINK lab.oracle1 CONNECT TO repadmin IDENTIFIED BY repadmin using LAB;

- on master site 2:

CREATE public DATABASE LINK orcl1.oracle1 CONNECT TO repadmin IDENTIFIED BY repadmin using 'CABINET';

In this moment, the build operation of replication process can begin. Considering a schema named *CLIENT* (on each of two master sites), the next exemplification will replicate a table of this schema. This operation can be performed in two ways: using the graphical tool *Oracle Enterprise Manager* (OEM) or using scripting commands involving procedures calls of already mentioned build-in packages. Working with OEM supposes adding databases involved into replication process to the options tree (using the system administrator account to connect). Also, the creation of distributed environment (databases links) could be done using OEM (see "Database Links" branch from options tree of OEM in figure 3). The scripting commands can be run using the command line tool *SQL*Plus*.

It is very important to check the functionality of databases links before start the replication process. This can be done using the OEM option "Advanced Replication–Administration", showing graphical the active or inactive links between databases from distributed environment (see figure 4): the continuous lines means an active link, and the red dotted lines means a broken link (the cause can be either the remote database server is shutdown, either the link is not correct configured).

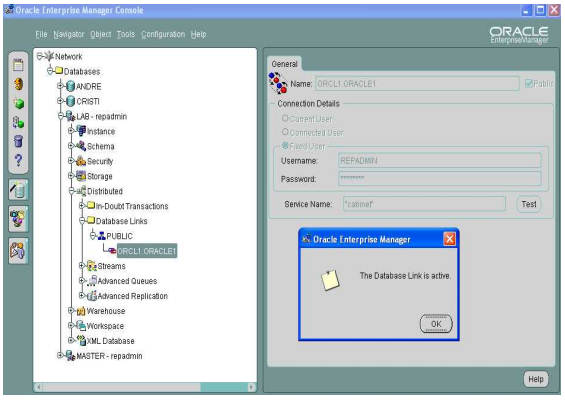

Figure 3. Create database link

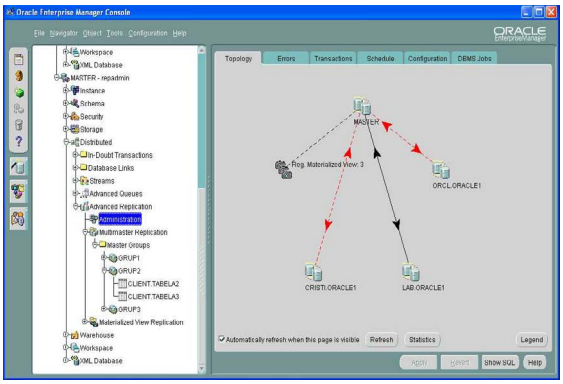

Figure 4. Check active or inactive database links

Both ways to implement a replication process are presented bellow: working with OEM interface (due to its simplicity), respectively the method involving procedures calls (requiring to work with properly built-in packages). All operating process with OEM begin with add databases to the menu tree of graphical interface (see the left side of OEM interface depicted in figure 3 and 4).

After creation of replication administrator accounts (on each master site) and builds the link between databases, the first step is to access the "Advanced Replication – Multimaster Replication" option from OEM options tree. This option allows two operations: Setup Master Sites and Create Master Group.

B. Defining the Master Sites

By accessing the Setup Master Sites wizard, the database on witch the replication is applied can be set as master site (using the *SYSTEM* administrator account), creating thus the first master site. Next, the already created *REPADMIN* account will be reconfirmed as default replication administrator (and also as default account for both propagate and receive role). The propagator role allows the propagation of the deferred transaction queue to other master site, respectively the receiver allows to receive the propagated deferred transactions (sent by the propagator from other master sites). Follow the next step of the wizard, there must be specified the master site schema that contains the objects to be replicated (for this exemplification, the CLIENT schema). The push jobs of a database link can be scheduled by setting how often the deferred transaction queue is propagated to each of the other master sites (see figure 5 – graphical interface to set the replication frequency).

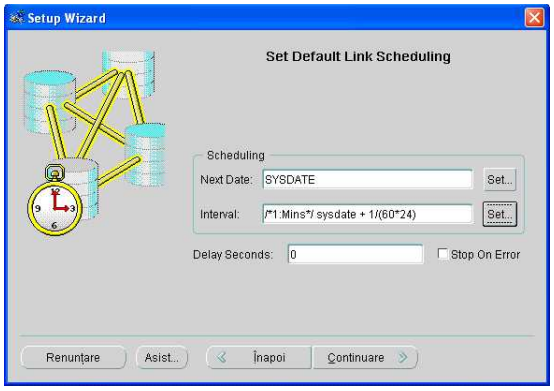

Figure 5. Set the push job scheduling (starting moment and time interval)

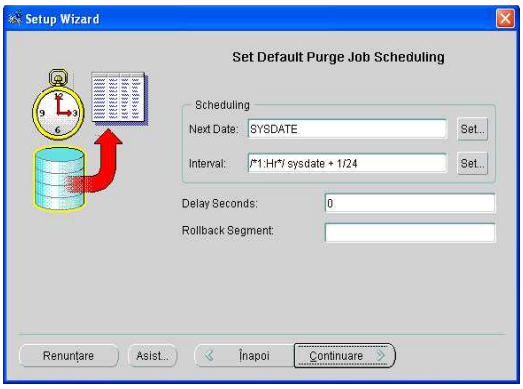

Figure 7. Set the purge job scheduling

This task may be done also using a direct call of procedure DBMS_DEFER_SYS.SCHEDULE_PUSH (using the command line interface SQL*Plus), as in the following example where replication is done every minute:

BEGIN

DBMS_DEFER_SYS.SCHEDULE_PUSH(destination => 'LAB.ORACLE1',

next_date => to_date('18-09-2008 08:04:50','DD-MM-YYYY HH24:MI:SS'),

*interval => 'sysdate + 1/(60*24)', stop_on_error => FALSE, delay_seconds => 0, parallelism => 0);*

END;

In figure 6 also is shown how the OEM allows to manage the replication frequency, change respectively disable the replication schedule (or even stop them on error).

After setting of a transaction push process, immediately a purge job scheduling must be set to periodically empty the deferred transaction queue (deleting the transactions which are already processed). In this mode, the size of transactions queue is maintained under control (figure 7 – set every hour a purge transaction job). A purge process can be also started by a procedure call (DBMS_DEFER_SYS.SCHEDULE _PURGE), as is described bellow (deleting successfully replicated transaction every hour to empty the jobs queue):

BEGIN

DBMS_DEFER_SYS.SCHEDULE_PURGE (next_date => sysdate, interval => 'sysdate + 1/24',

delay_seconds => 0, rollback_segment => '');

END;

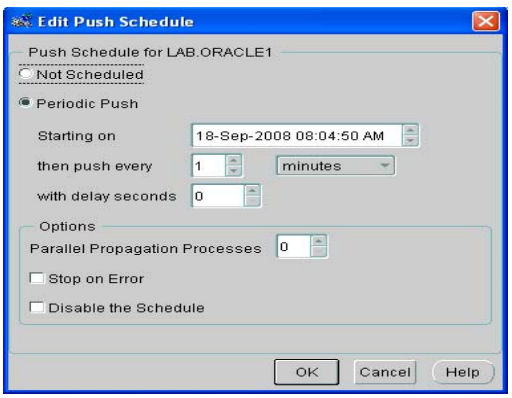

Figure 7. Change or stop the periodical jobs push

C. *Defining Replication Groups*

The next phase assumes the creation of a master group, which is a collection of replicated objects (logically related), allowing a centralized management of all the objects that will be replicated. This process can be done using either the OEM interface (selecting the option "Multimaster Replication - Master Group" from OEM options tree) or a procedure calls (working with built-in packages) and supposes the following: establishing a group name, adding the objects which will be replicated, adding the second master site to the group. This last operation set the destination site where the objects group will be also updated (figure 8). All these operations are done to the master definition site and generate the replication support for all considered objects. All databases linked with the master definition site can be selected as possible replication destination. Do not forget that this replication is bi-directional (so all master sites are also simultaneously destinations too).

As it can be seen in figure 8, the data changes propagation (to the remote site) can be asynchronous or synchronous. If the selected option is synchronous propagation, the data changes on all master sites operate as a distributed transaction, ensuring the sites synchronization. Therefore, a data update operation succeeds on both master sites or is canceled (for example, if the link with the destination site is broken, site failures) on both master sites (even on source site). This option maintains the data consistency. If the selected option is asynchronous propagation, the data changes at source are committed before replication to remote destination (even if the replication process to the destination site is impossible due to different causes) [2][6].

Choosing one or other of these types of propagations depends on the particularities of the application (the main reason take into consideration being the data consistency) [2][4].

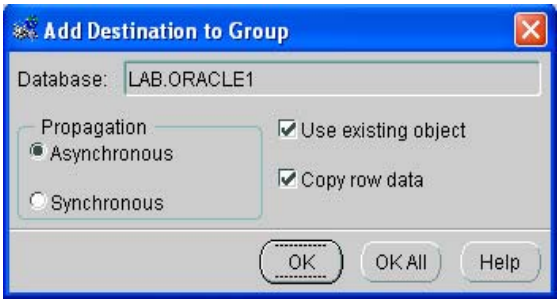

Figure 8. Set destination site and replication type

Previously was also stated that the group creation can be done using procedures calls (using the procedures built-in Oracle packages). Therefore the following procedures must be called (with properly input parameters):

- *DBMS_REPCAT.CREATE_MASTER_REPGROUP* (creates the group);
- *DBMS_REPCAT.CREATE_MASTER_REPOBJECT* (adds objects to the group; this call is done for every new object added to group);
- *DBMS_REPCAT.SET_COLUMNS* (optionally call, unless the replicated tables do not have a primary-key);
- *DBMS_REPCAT.GENERATE_REPLICATION_SUPPORT* (based on all previously procedures calls, generates replication support for every group's object, so such call must be done for each replicated object);

Obviously the replication activity is stopped during implementation of replication support, thus starting the replication activity must be done explicitly, submitting a request call of the procedure *DBMS_REPCAT.RESUME_MASTER_ACTIVITY* and using the group name as input parameter.

The Oracle replication technique based on materialized view or snapshot is easier, but its functional potential is more limited. Therefore, for complex problems, the multi-master replication remains the only solution. Other database management systems, such as *MS SQL Server*, *IBM DB2*, *Interbase*, *MySQL* also provides mechanisms (more or less sophisticated) for data replication. There are some commercial products for database replication [10] (for example, *DBBalance Cross-Database Studio* [11], *IBReplicator*, *Replication Master*, *Daffodil Replicator* [12], etc), but quite expensive and their use in a particular application is difficult sometimes.

The OEM interface offers a relatively easy way to implements a replication process. But a lot of steps must be rigorously followed. The paper proposes an automation solution consisting into implementation of the replication process using a commands file (batch file), which contains all the commands and the procedures calls, with all the input parameters, provided as input variables. This solution (using the appropriate build-in Oracle packages), once tested, can be used whenever is required. Such a SQL batch commands file has been implemented and tested, proving to be a viable and faster alternative (running from *SQL*Plus* command interface). Thus, the replication process can be automatically repeated between other Oracle servers, the only requirement is to specify new input parameters for the batch file.

The replication process exposed in this paper take into consideration a homogenous distributed database system (each database is an Oracle database). The replication into a heterogeneous distributed database system (at least one of the databases is not an Oracle database) is more difficult due to the requirements to integrate different techniques from several research fields including distributed systems, databases, network protocols and operating systems [7][8].

III. Conclusion

Despite the apparent simplicity of the replication process, especially by using OEM graphics interface (which simplifies the process implementation and administration), a multi-master replication is a very serious matter and not simple in terms of programming needs. If the replication process is done using scripting commands, strongly programming knowledge is required about how to use the necessary Oracle build-in packages. Depending on application, a database replication requires an adequate analysis and planning, respectively a complete understanding of replication mechanism [9]. Using a scripting batch file, which contains all the necessary commands to implement the replication, the process can be automated. The presented details of a database replication phases provide a useful guide to implement such complex process requiring careful set-up and monitoring. A multi-master database replication can be in some cases the suitable solution to design and implement distributed computing applications, improving the load balancing.

References

[1] Y. Hao, D. Xing-Chun, and J. Guo-Quan, "Research on Data Synchronization in Oracle Distributed System", *International Seminar on Future Information Technology and Management Engineering,* 20-20 Nov. 2008, pp. 540 – 542.

[2] D. K. Burleson, J. Garmany, and S. Karam, "*Oracle Replication. Expert Methods for Robust Data Sharing*", Rampant TechPress, 2003.

[3] http://www.oracle.com/technology/books/pdfs/book_rep_chap6_ce2.pdf

[4] S.Feurstein, C. Dye, and J. Beresniewicz, *Oracle Built-in Package*. O'Reilly, USA, 1998.

[5] http://www.dba-oracle.com/art_dbazine_mm_repl.htm

[6] http://www.oracle.com/technology/software/index.html

[7] A. Yair, C. Tutu, "From Total Order to Database Replication", *Proceedings of the 22 nd International Conference on Distributed Computing Systems (ICDCS'02)*, 2002, pp. 494-503.

[8] M. Wiesmann, A. Schiper, "Comparison of Database Replication Techniques Based on Total Order Broadcast"*, IEEE Transactions on Knowledge and Data Engineering*, Volume: 17, Issue: 4, pp. 551- 566, April 2005.

[9] M. Wiesmann, F. Pedone, A. Schiper, B. Kemme, G. Alonso, "Database Replication Techniques: a Three Parameter Classification*", Proceedings of the 19th IEEE Symposium on Reliable Distributed Systems*, Germany, 2000, pp. 206-215.

[10] T Kim Nguyen, "Comparing Replication Technologies", *The Data Administration Newsletter*, September 1, 1999, PeerDirect Inc. (http://www.tdan.com/view-articles/5268).

[11] http://www.dbbalance.com

[12] http://www.ibphoenix.com/main.nfs?a=ibphoenix&page=ibp_repl_tools