Database Replication: A Survey of Open Source and Commercial Tools

Salman Abdul Moiz
Research Scientist
Centre for Development of Advanced Computing, Bangalore.

Sailaja P.
Senior Staff Scientist
Centre for Development of Advanced Computing, Bangalore.

Venkataswamy G.
Staff Scientist
Centre for Development of Advanced Computing, Bangalore.

Supriya N. Pal
Associate Director
Centre for Development of Advanced Computing, Bangalore.

ABSTRACT
The most critical and important aspect of disaster recovery is to protect the data from application fail over, natural disasters and infrastructure failures. Taking frequent backups of the huge volumes of data and storing it is also an integral part of the disaster recovery plan. Various scenarios of database replication strategies and techniques are provided in this survey paper addressing the need for replication of data. A wide range of open source and commercial tools have evolved over a period of time to deal with the issues and challenges involved during the process of database replication. This paper describes the general characteristics of replication & disaster recovery. Later the features of open source and commercial replication tools are compared.

Keywords
Database replication, Replication strategies, Open source replication tools, Commercial replication tools.

1. INTRODUCTION
The growing need towards decentralization in any enterprise has created a strong necessity for database replication. This is because businesses today are more geographically dispersed and it is expected to provide location transparency to the employees of an organization. Unlike centralized environments, replication may help in availability of data by processing transactions from remote site and later restoring the data in the event of failures. Database replication tools can provide efficient mechanism to replicate the data and maintain backups that can be used when a disaster occurs.

This survey paper presents different types & characteristics of replication. In addition the characteristics of open source and commercial replication tools are presented and the challenges ahead are described.

The remaining part of this paper is organized as follows: Section 2 describes the fundamentals of replication. Section 3 presents the features and abstract architecture of open source replication tools, and the comparison of each of these tools with respect to different parameters of replication. Section 4 portrays the commercial replication tools, their functionality and the comparison of each of these tools with respect to different parameters of replication. Section 5 describes the Issues and Challenges in database replication aligned with multiple replication tools and Section 6 concludes the paper.

2. REPLICATION CONCEPTS
Replication is the process of creation and maintenance of duplicate versions of database objects in a distributed database system [1]. Replication improves the performance and increases the availability of applications by providing alternate data access options. For example, users can access a local database rather than a remote server to minimize network traffic and provide location transparency. Furthermore, the application can continue to function if parts of the distributed database are down as replicas of the data might still be accessible.

Database replication is needed in the case of a system failure where in if a primary copy of the database is failed the secondary copy will be still there to retain the data. A replication service is required to maintain data consistency across these diverse environments. Distribution reduces the network costs for query access, and it improves application availability and consistency.

2.1 Types of Replication
The replication tools may be selected based on type of replication it supports. The capabilities and performance characteristics varies from one type of replication to another. A replication strategy may be selected based on two basic characteristics: *Where* and *When.*

When the data is updated at one site, the updates have to be propagated to the respective replicas. When the updates can be propagated can be achieved by Synchronous (eager) and Asynchronous (lazy) methods and where the updates can take place can be achieved by update everywhere and primary copy (master-slave) methods.

Synchronous replication (Master-Slave replication) works on the principle of Two-Phase commit protocol. In a two-phase commit protocol, when an update to the master database is requested, the master system connects to all other systems (slave databases), locks those databases at the record level and then updates them simultaneously. If one of the slaves is not available, the data may not be updated. The consistency of data is preserved; however it requires availability of all sites at the time of propagation of updates.

There exists two variations of *Asynchronous replication* (Store and Forward replication) i.e. Periodic and Aperiodic. In Periodic replication, the updates to data items are done at specific intervals and in aperiodic replication the updates are propagated only when necessary (usually based on firing of event in a

trigger). The time at which the copies are inconsistent is an adjustable parameter which is application dependent

In U*pdate anywhere* method, the update propagation can be initiated by any of the sites. All sites are allowed to update the copy of the datum whereas in a *Primary Copy* method there is only one copy (primary copy or master) which can be updated and all other (secondary or slave) copies are updated reflecting the changes to the master.

Various forms of replication strategies are as follows:

- *Snapshot Replication*: In snapshot replication, a snapshot or copy of data is taken from one server and moved to another server or to another database on the same server. After the initial synchronization, snapshot replication can refresh data in published tables periodically. Though snapshot replication is easiest form of replication, it requires copying all data items each time a table is refreshed.
- *Transactional Replication*: In transactional replication, the replication agent monitors the server for changes to the database and transmits those changes to the other backup servers [2]. This transmission can take place immediately or on periodic basis. Transactional Replication is used for server-server scenarios.
- *Merge Replication*: Merge replication allows the replicas to work independently [2]. Both entities can work offline. When they are connected, the merge replication agent checks for changes on both sets of data and modifies each database accordingly. If transaction conflict occurs, it uses a predefined conflict resolution algorithm to achieve consistency. Merge replication is used mostly in wireless environments.
- *Statement based replication*: The statement based replication intercepts every SQL query and sends it to different replicas [3]. Each replica (server) operates independently. To resolve conflicts, Read-Write queries are sent to all servers where as read only queries can be sent to only one server. This enables the read workload to be distributed. Statement based replication is applicable for optimistic approaches where each cache maintains the same replica.

2.2 Fault Tolerance

Fault tolerance describes a procedure/technique that keeps the system in a consistent state in event of a failure by using redundancy. If a site fails the data it contains may become unavailable. By keeping several copies of the data at different sites, single site failure ensures the overall availability of the system.

2.2.1 Failover

Failover is the process of moving database, metadata and user connections from a failed or shutdown primary site to a secondary site so that users can still access the data thus preserving the availability. Failover may be an implicit or an explicit process. Full functionality of alternate site or secondary site is to be ensured in case if disaster happens at the primary site.

2.2.2 Failback

When the primary site resumes operations, the user performs a failback to restore the primary site in consistent state. The restoration process is usually initiated when the load on the secondary server is less. Otherwise it degrades the performance of the system.

2.2.3 Active-Active

In Active-Active mode, two servers are configured with independent workloads [12]. They run on primary and secondary nodes respectively until one fails. When failover occurs, the secondary node takes over the primary and resumes activities until primary server fails back.

2.2.4 Active-Passive

In Active-Passive mode, the services are provided either by the primary node or by the secondary node. Initially the services are provided by a primary node before a fail over and by secondary node after fail over.

3. OPEN SOURCE REPLICATION TOOLS

Open source replication tools provided in the public domain may help in implementation of the database replication activities in our own network based environments in a collaborative manner.

3.1 Postgres-R

Postgres-R is an extension to PostgreSQL database which provides efficient database replication for clusters while preserving the consistency [4] [10]. It is designed to be as transparent as possible to the client, stable and secure. Postgres-R is primarily used for load balancing and high availability of database systems. The stretchy architecture of Postgres-R enables to scale the replication process. Further compared to a centralized system, Postgres-R is more reliable and flexible.

Postgres-R can replicate large objects and uses a store and forward asynchronous replica engine. The use of Group Communication System by Postgres-R helps in adding or removing nodes, when the system is in running state. The crashed nodes are detected and removed automatically without affecting the operations of database system. Postgres-R uses two-phase locking for concurrency control with locking at relation level.

Transaction execution in Postgres-R uses read-one-write-all approach and the transaction execution is divided into four phases. In Local Phase, the transaction is executed at local replica (local database). In Send Phase, the updates to data are propagated to replicas. In Synchronization Phase Serializability of the transactions is preserved. In write phase remote replicas executes the writes and respective replicas execute the transactions.

The basic architecture of Postgres-R is depicted in figure 1. The functionality and interfaces supported with PostgreSQL are available in Postgres-R. Additionally it supports replication. Ensemble is used for Group Communication System (GCS).

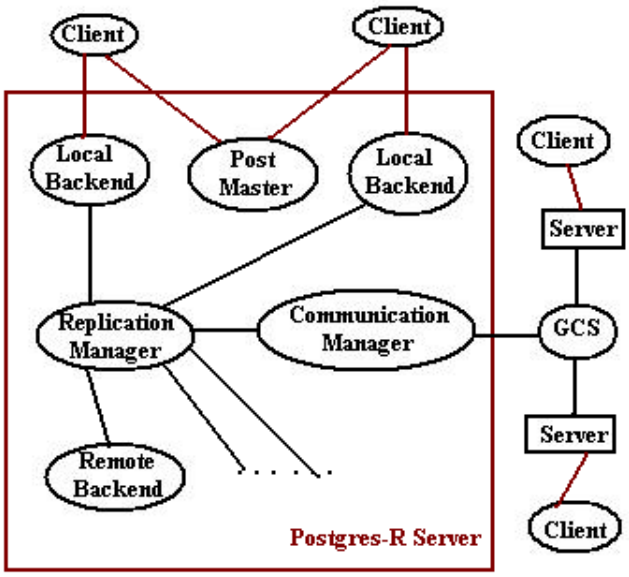

Figure 1. Architecture of Postgres-R [5]

The replication process comprises of several servers each of them running an instance of Postgres-R [5]. When clients want to access the databases it sends a request to the postmaster (listener). The postmaster creates the backend process for each client, then the communication is established between client and the backend. The client may connect to any of the servers. The transactions of a client are called local transactions at the server to which it is connected. When a request for remote transactions is initiated, it is handed over to one of the remote back end. In a local or a remote transaction, the client has to wait when a threshold of number of requests is reached. The replication manager is responsible for message handling process between local and remote sites [5].

3.2 Slony-I

Slony- I is a "master to multiple slaves" replication system supporting cascading and slave promotion [6]. Slony-I is a master-slave system capable of replicating large databases to a limited number of slave systems. If the number of servers grows beyond a certain specified limit (Threshold), the communication cost may increase. The purpose of Slony-I is to maintain data centres and backup sites where the sites are secured and normally are available all the time. However if the nodes in system are frequently prone to failures, Slony-I is not the better solution for replication.

Slony-I replication system can replicate tables and sequence numbers [7]. The objects (Tables, Sequence numbers etc.) are logically grouped into sets. Each set is independent of other which is originating from the same master. The tables, their relationships and the sequences used to generate serial numbers are grouped into their respective sets. Figure 2 describes the replication configuration of Slony-I.

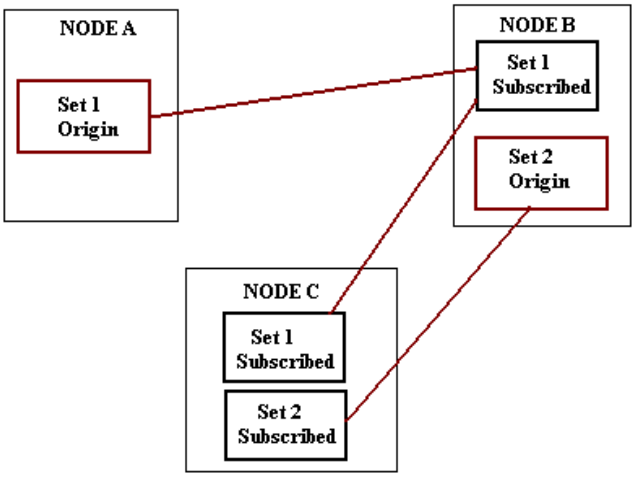

Figure 2. Replication Configuration of Slony-I [7]

The configuration specifies two sets with different origins. Inorder to replicate both sets on Node C, it is not necessary that the Node C has to communicate with origin of Set 1. If Node C fails the master of Set1 and Set 2 are still alive. If Node A fails, Node B gets promoted as master of both sets. However if Node B fails, Node C has to be promoted to be the master of Set 2 and continue replication of Set 1 from Node A. A Node stores replication log till all subscribes have restored the replication information. Slony-I uses "After Row" triggers, which analyses New and Old rows required to maintain the consistency. To identify a row, it is expected to have a unique constraint for a column in a table or it has to be supported by row-id.

3.3 ESCADA Replication Server

ESCADA is a portable, reliable and flexible database replication system that provides good performance and applicability tradeoffs [8]. Its features range from shared -nothing clusters to disaster prevention using remote servers.

ESCADA replication server uses a synchronous master/slave replication and a zero data-loss inter-cluster replication over WAN .The tool also provides Consistent multi-master/update everywhere replication and is Build on GORDA architecture and interfaces. It relies on a generic group communication service for Java, named jGCS. The GORDA Architecture and the Programming Interfaces GAPI is presented in [9].

ESCADA replication server is designed and implemented on the principle of component based technology that can be combined with several interoperable components to provide a range of replication solutions. GAPI allows effective implementation of replication protocols and can be reused with different database management systems.

In GORDA, the transaction processing is abstracted as a pipeline. The different processing stages of major meta level interfaces are depicted in figure 3.

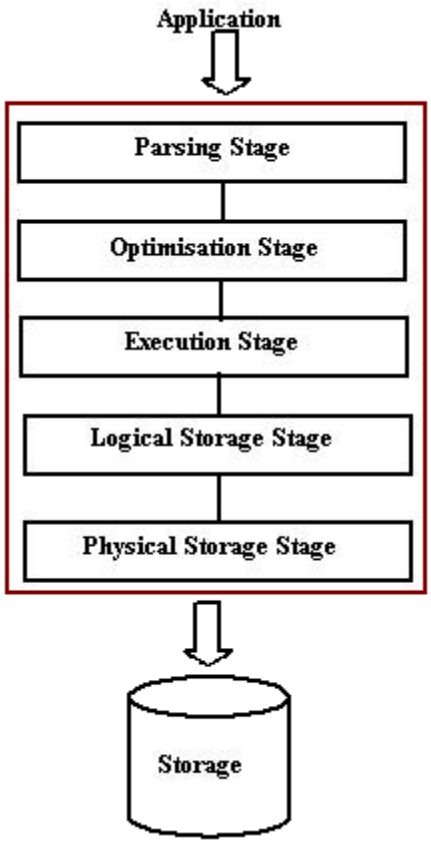

Figure 3. Major meta level interfaces: Processing stages [9]

The Parsing stage parses a given raw statement from the application and produces a parse tree. The parse tree is then transformed by optimization stage based on most appropriate optimization techniques into an execution plan. The execution stage executes the plan and produces object sets.

The logical storage stage maps logical objects into physical objects and the physical storage stage deals with I/O process and synchronization. The interface expose Meta object for each stage and for data that moves between them.

3.4 DB Replicator

DBReplicator is a powerful open-source & network-based multi-master heterogeneous database replication tool developed in Java. It provides an API designed to perform asynchronous update anywhere database replication.

Since DB Replicator is heterogeneous in nature, it synchronizes the same database between multiple RDBMS. It supports bi-directional data synchronization between any of the supported back ends where the synchronization process is initiated by the subscriber. If a relation doesn't exist in the subscriber database, it creates one to match those in publisher database.

DB Replicator provides automatic conflict detection and resolution strategy with minimal involvement of a DBA. Basic conflict resolution is implemented as publisher wins/subscriber wins. However the future versions of DB Replicator are expected to implement more complex conflict resolution strategies.
It works on the principle of merge replication where the transactions can be executed locally. Later at the time of reconciliation the instances of publisher and subscriber databases are compared to achieve consistency. The transactions which are executed locally may involve different databases because DB Replicator supports the heterogeneous databases.
DB Replicator provides built in scheduling facility. This helps in scheduling the operation for its execution. Irrespective of the failures, the operation is executed at a particular moment of time.

3.5 Pgpool-II

Pgpool-II is a middleware that acts as an interface between PostgreSQL database Client and PostgreSQL servers [11]. Since Pgpool-II is a middleware, the database application i.e. the client feels that Pgpool-II is the actual Postgres server and the server visualizes Pgpool-II as one of its client.

Pgpool-II manages multiple servers. The backup is taken on two or more disks. The replication features of Pgpool-II helps to availability of the system by providing efficient fault tolerance mechanism in case of failures.

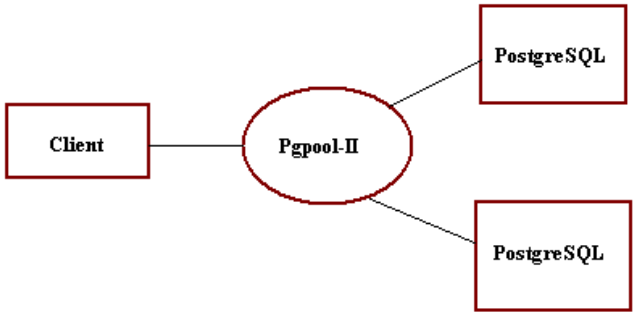

Figure 4. Replication Architecture of Pgpool-II

Figure 4 presents the generic replication architecture of Pgpool-II. Client sends queries which are duplicated and forwarded to PostgreSQL servers. The replication at different sites by Pgpool-II helps in achieving automated failover. In Automated failover if one of the two PostgreSQL servers goes down, Pgpool-II will enable the other server to take over and continue the services provided to its applications. Pgpool-II also supports online recovery which helps in restoration of the failed node without stopping the operations of the system.

When client requests huge number of read operations, Pgpool-II implements Load Balancing effectively by distributing the SELECT queries across multiple servers. This helps in execution of multiple queries at the same time. Further as the requested operation is a read operation, every server will return the same output.

However there is a limit (Threshold) for maximum number of client connections at the same time. If the number of clients binded to server increases the threshold the connection is rejected. Increasing the number of connections may not be the solution of these instances. As it requires more resources and the conflict resolution strategy also needs to be modified. Hence

there should be a balance between the performance and scalability. This factor could be adjusted based on the client's requirements.

Table 1 illustrates the comparison

of open source replication tools based on the various characteristics.

Table 1. Comparison of open source replication tools

	Postgres -R	**Slony-I**	**Escada**	**DB Replicator**	**Pgpool-II**
Type	Async	Async	Async	Async	Sync
Update any-where	Yes	No	Yes	Yes	Yes
Work-load partitioning	Yes	No	Yes	No	Yes
Master/ Slave	Yes	Yes	Yes	Yes	Yes
WAN support	Yes	Yes	Yes	Yes	Yes
Conflict managmnt	Yes	No	No	Yes	Yes
Auto detection of node failure	Yes	No	Yes	No	Yes

4. COMMERCIAL REPLICATION TOOLS

The Disaster recovery solutions provided by the open source replication tools are much cheaper as compared to commercial replication tools. However as fault tolerance is an important domain, few disaster recovery practitioners feel that fault tolerance shouldn't be compromised with cost of the replication tools.

4.1 IBM Informix Replication

Informix replication is a suite that synchronizes data across the enterprise. It is an asynchronous log based tool for replicating data between Informix Dynamic database servers. Transactions are captured on source server by reading logical log, storing it and reliably transmitting data to the target server [13]. At target server, Enterprise replication receives and applies each transactional data to the appropriate databases, tables and logs the current state of the transaction.

Enterprise replication uses asynchronous data replication to update the replicated database when the primary database has commits the changes. The delay to update data on a secondary site may vary. However it is later synchronized to a consistent value at all sites. The advantage of this approach is that in the event of a failure, the replication process continues and all transaction in the replication system will be committed. In Informix, asynchronous replication supports primary-target and update-anywhere replication models.

In the log based data capture process; the enterprise replication reads the logical log to obtain row images of tables and then evaluates them. The changes from logical log are captured and the same is updated in the base tables accordingly. In addition Informix supports trigger based data and transaction capture.

4.2 DB2 Data Propagator

DB2 Data Propagator provides replication solutions to distributed database environments and data warehousing. The replication server in DB2 Data Propagator supports consolidation (moving data from many databases to one) and distribution (moving data from one database to many) scenarios [14]. The horizontal or vertical fragment of the base table can be replicated i.e. the subset of the columns of a relation or subset of tuples of a given relation can be replicated.

DB2 Data Propagator can replicate between heterogeneous relational database management systems and platforms and distribute data to applications in the same or different environments. DB2 Data Propagator unites the distributed relational databases into a cohesive and integrated database solution [15]. It implicitly captures changes on data items and propagates the same to specific target databases, thus preserving consistency of the data.

The DB2 data propagator uses a log based change capture that minimizes the impact on transaction performance and as a result avoids conflict with source tables and online transaction processing.

4.3 Sybase Replication Server

Sybase uses three components for achieving solution to the data distribution problems i.e. SQL Anywhere databases, Replication agent for SQL Anywhere & SQL remote distribution solution [16].

Sybase replication server allows data to be shared and replicated between databases. The replication server manages the operations of data distribution which helps in reducing the burden of database administrators thereby paves way in meeting business goals.

Sybase replication server is a set of tools that moves and synchronizes data from one server to another. It ensures availability of data and a prompt recovery when disaster strikes. It integrates and synchronizes operations carried out at multiple locations.

The deployment architecture of Sybase replication is depicted in figure 5. In addition to the primary and secondary sites each for a

Sybase and a Non-Sybase database, it consists of three entities: Replication Server, Replication Agent and Direct Connect.

Replication Server is responsible for distribution and synchronization of data across the enterprise. If both primary and secondary databases are Sybase, replication can be handled only by using replication server.

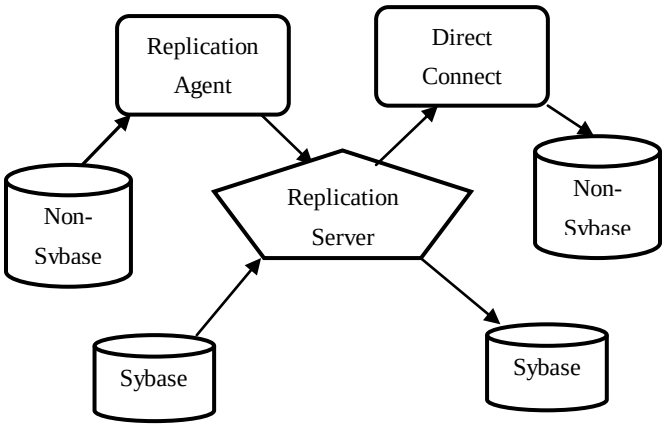

Figure 5. Deployment Architecture of Sybase Replication [16]

Replication Agent captures transactions from non-Sybase database and transfers them to the replication server. It is needed to replicate data from non-Sybase Server.

Direct Connect is a Sub component that enables replication to a variety of LAN based heterogeneous and other data targets [16].

There are three types of replications available with Sybase Replication Server:

Publish-and-subscribe model: The transactions occurring in a source database are detected by replication agent and transfer it to the replication server which distributes the information to other replicated servers. The primary data is the source of the data that replication server replicates in other databases. User "publish" data at primary sites to which Replication servers at other sites "subscribe."

Warm standby: Replication server provides warm standby capability with a pair of databases, one as the active database and one as the standby database, which is supported by Replication server functionality. As clients update the active database, replication server copies transactions to the standby database, maintaining consistency between the two. If the active database fails for any reason, you can switch to the standby database, making it the active database, and resume operations with little or no interruption.

Heterogeneous data server support: Sybase Replication Server also supports replicating data to and from non-Sybase data servers. Data can be replicated to non-Sybase data servers such as Oracle, Informix, IBM DB2, and Microsoft SQL Server using Direct Connect gateways. Transactions can be captured and forwarded from non-Sybase data servers to Sybase data servers using Sybase Replication Agents. Data can also be replicated from a non-Sybase source, through Replication server to a non-Sybase destination [16].

4.4 Q- Replication Tools

Q-replication uses message queues to transfer changes to the data between the source and target databases [17]. It has two major components: Q capture and Q Apply. The different components of Q-replication are depicted in figure 6.

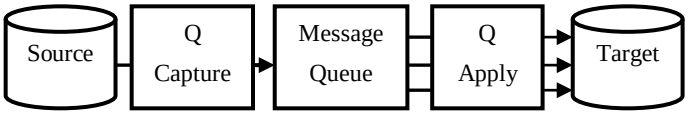

Figure 6. Q Replication

Q capture reads the transactional log, identifies the changes in data from the committed transactions and embeds into a message. These messages are then written to one or more message queues (Web Sphere). Q Apply then receives the transactional messages from the message queue and applies the changes to the target system as a single transactional unit. The message queue facilitates transfer of data across heterogeneous databases and platforms.

Q replication allows maintaining multiple copies of data and the transactional updates on data items; this may result in occurrence of a conflict. However it provides mechanisms to resolve such conflicts.

4.5 Oracle's Stream

Oracle Streams are one of the unified solutions used for information sharing. Stream captures and distributes database updates, events and messages [18]. It triggers the updates to the destination database or transfers events and messages. This capability of oracle streams helps in development of flexible solutions for replication, message queuing and event notification.

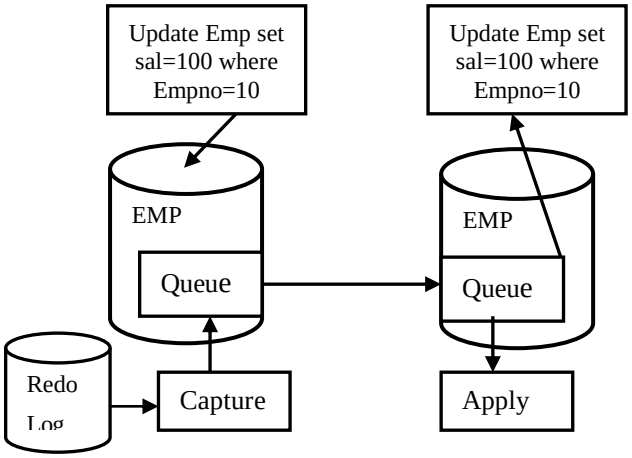

Figure 7. Architecture of Oracle's stream replication

The architectural elements of replication using oracle's stream is depicted in figure 7. Oracle's stream capture changes from

source database, puts it in a queue and propagates these changes to remote database.

The changes are then applied to each of the target databases. The execution of the update statement changes the state of a row in Emp relation whose Empno is 10. These changes are recorded in the redo log. The capture process collects these changes, retrieves from the redo log. This information is added to a queue which is delivered to another database that has a back up of Emp relation. Once the changes are successfully propagated the apply process applies the changes to the secondary databases at the remote site.

Table 2 presents the comparison of various commercial replication tools. It specifies the replication strategy used and the mechanism adopted to recover the data in case of failures.

Table 2. Comparison of commercial replication tools

Replication Tool	Replication Strategy	Recovery Process
IBM Informix Replication	Supports both "active-passive" and "active-active" replication	Continuous Log Restore [CLS] Remote Standalone Secondary servers [RSS]
DB2 Data Propagator	The Data Propagator replication engine uses a change-capture mechanism and a log to replicate data between a source system and a target system.	Distribute data to applications in the same or different environments Replicate all or a subset of your data to one or more locations.
Q-Replication Tools	Allows multidirectional replication	Emergency restart (ERE) failure recovery.
Sybase replication server	Publish and Subscribe model Warm Standby Heterogeneous data server support	Replication server and Replication Agents are responsible for synchronization & recovery of data between similar or heterogeneous databases
Oracle stream	Asynchronous Master-Master Replication Oracle Streams Replication captures changes from a source database stages and propagates those changes to one or more remote databases	Recovery of primary or secondary database is achieved with the help of redo logs.

5. ISSUES & CHALLENGES

The major challenge for the data processing centres is to ensure that all the applications and business users use the consistent copies of the redundantly stored data. Several replication tools are proposed to maintain the consistency by implementing several conflict resolution strategies. However the time lag between the failure and execution of the conflict resolution strategy is an issue. In some of the replication tools, though the data may be inconsistent for some time but later the conflict resolution strategy gets the data into a consistent state. This makes the data unavailable for certain period of time which is an issue to be resolved.

Since replicated databases works on distributed systems technology, there will be issues in maintaining reliable communication channels as data availability need to be preserved.

Locking and performance are harder issues in statement-based replication. In particular, locking granularity is usually at the table level, as table information can be obtained through simple query parsing; however, this limits performance. Finer granularity (i.e., row level) would require re-implementing a large fraction of the database logic inside the middleware. Moreover, the middleware-locking regime might not be compatible with the underlying database locking, leading to distributed deadlocks between the databases and the middleware.

Connections pools are usually major issue for failback. At failure time all connections to a bad replica will be reassigned to another replica or just removed from the pool.

Any RDBMS can manage multiple database instances. Research efforts have primary focused on replicating independent database instances leading to a concept of virtual databases [19]. Virtualization of an entire RDBMS has not been addressed. Further queries spanning multiple databases are not effectively handled by replication that works on peer database instances.

Stored procedures have been used for many legacy applications initially by Sybase. However the replication of stored procedure raises several issues. Stored procedure replication is a domain that is to be addressed as it may provide more secure environment rather than the entire data base replication strategy.

Finally as the commercial replication tools are not cost-effective there is a need to use the open source replication tools by providing the patches needed based on the replication strategy to handle the disaster situations.

6. CONCLUSION

Data base replication is the process of maintaining exact copy of a database for high availability, recoverability and maintenance. To achieve the fault tolerance in several applications that uses databases many replication tools both open source and commercial came into existence. Synchronous and Asynchronous replication strategies have their own advantages and disadvantages. There is a need to implement a better hybrid replication approach in the existing tools.

The replication tools need to ensure that the database consistency is preserved across the systems which are available at a

particular moment of time. This may help in guaranteeing high availability and at the same time may achieve a high degree of consistency. Further the tools have to support an efficient fail back mechanism that helps in knowing the current state of the database by the node which has recovered from the failure.

7. REFERENCES

[1] May Mar Oo, "Fault Tolerance by Replication of Distributed Database in P2P System using Agent Approach", International Journal of Computers, Issue 1. vol.4, 2010.

[2] Sujoy P.Paul, "Pro SQL Server 2008 Replication"

[3] "PostgreSQL 8.4 Server Administration", Volume II- The PostgreSQL Global Development Group, 1996.

[4] Shuqing Wu, Bettina Kemme, "Postgres-R(SI): Combining Replica Control with Concurrency Control Based on Snapshot Isolation", ICDE, 2005.

[5] Bettina Kemme, Gustavo Alonso, "Don't be lazy, be consistent: Postres-R, A new way to implement Database Replication", VLDB, 2000.

[6] Brad Ediger, " Advanced Rails", 2007

[7] Jan Weick, "Slony-I, A replication system for PostgreSQL"

[8] Rui Oliveira,"GORDA open replication of databases"- A report, 2008.

[9] Alfranio Correio Jr. et. Al, "GORDA: An Open Architecture for Database Replication", IEEE, NCA, 2007.

[10] Bettina Kemme, Gustavo Alonso, "Database Replication: A Tale of Research across Communities", VLDB, Vol.3, No. 1, 2010.

[11] Jean-Paul Argudo, " Pgpool-II", 2010.

[12] Will Schmied, Orin Thomas, "Implementing & Managing Exchange Server 2003", pp.219, 2004.

[13] IBM Informix Dynamic Server Enterprise Replication Guide, Version 9.4, 2003, Part No. CT1T2NA.

[14] Mrktheni, "Data Propagator Setup"

[15] Bart Steegmans et.Al, IBM DB2 for z/OS and OS/390 Data Management Tools Update, 2002.

[16] Charles B.Clifford, "Sybase Replication Server –Primer", 1995.

[17] Nagraj Alur et.Al, "Websphere Information Integrator Q-Replication: Fast Track Implementation Scenarios", 2005.

[18] Ashish Ray, "Implementing Replication with Oracle Streams"

[19] Emanuel Cecchet et.Al, "Middleware-based Database Replication: The Gaps between Theory & Practice", SIGMOD, 2008.