Journal of Computer Science 6 (10): 1095-1098, 2010
ISSN 1549-3636

Data Replication Using Read-One-Write-All Monitoring Synchronization Transaction System in Distributed Environment

[1]Noraziah Ahmad, [2]Ahmed N. Abdalla and [1]Roslina Mohd Sidek
[1]Faculty of Computer Systems and Software Engineering,
University Malaysia Pahang, Kuantan, Malaysia
[2]Faculty of Electric and Electronic Engineering,
University Malaysia Pahang, Pekan, Malaysia

Abstract: Nowadays, the demand of transmitted information over networks increase rapidly and the demand for steady bandwidth seems to be out of control. Particularly organizations need to provide updated data to users that might be geographically remote and handling a vast amount of requested data distributed in multiple sites. **Problem statement:** Replication in distributed environment has become increasingly popular due to its high degree of availability, fault tolerance and enhance the performance of a system. These advantages of replication are important because it enables organizations to provide users with access to current data anytime or anywhere even if the users are geographically remote. However, this way of data organization also introduces low data consistency among replicas when changes are made during transactions. The need to have a system to monitor this data replication arises. **Approach:** Read-One-Write-All Monitoring Synchronization Transaction System (ROWA-MSTS) has been developed to solve this problem by using Rapid Application Development (RAD). **Results:** ROWA-MSTS helped to monitor the replicated data distribution over multiple sites while maintaining the data consistency. **Conclusion:** Results showed that ROWA-MSTS solved the distributed concurrency transactions and guarantees the data consistency in distributed systems.

Key words: Distributed system, replication, consistency, transaction

INTRODUCTION

In the world where internet does all the business nowadays, the demand of transmitted information over networks increase rapidly and the demand for steady bandwidth seems to be out of control. Particularly, organizations need to provide updated data to users that might be geographically remote while handling vast amount of requested data distributed in multiple sites. In modern distributed systems, replication receives particular attention for providing high data availability, fault tolerance and enhance the performance of the system (Gao *et al*., 2005; Mat Deris *et al*., 2004; Tang *et al*., 2006). It is an important mechanism because it enables organizations to provide users with access to current data where and when they need it. Ensuring efficient access to such a huge network and widely distributed data is a challenge to those who design, maintain and manage the grid network. The availability of a certain data at a huge network is one of the issues that still unsolved.

An ideal distributed file system provides applications strict consistency, i.e., a guarantee that all I/O operations yield identical results at all nodes at all times (Bernstein and Goodman, 1984; Zhang and Honeyman, 2004). In a replication system, the value of each logical item is stored in one or more physical data items, referred to as its copies (Zhang and Honeyman, 2004). Each read or write operation on a logical data item must be mapped to corresponding operations on physical copies. Of course this way of data organization introduces low data consistency and data coherency as more than one replicated copies need to be updated. Expensive synchronization mechanisms are needed to maintain the consistency and integrity of data among replicas when changes are made by the transactions. This suggests that proper strategies with minimum communication cost are needed in managing replication and transactions in distributed systems.

There are many examples of replication schemes in distributed systems. Among them are based on synchronous replication (Holliday *et al*., 2003;

Corresponding Author: Noraziah Ahmad, Faculty of Computer Systems and Software Engineering,
University Malaysia Pahang, Kuantan, Malaysia

Stockinger, 2001; Noraziah *et al*., 2006), which deploy quorum to execute the operations with high degree of consistency and ensure serializability. Synchronous replication can be categorized into several schemes, i.e., all -data-to-all-sites (full replication) and some-data-items-to-all-sites. However, full replication causes high update propagation, high storage capacity and difficult to maintain the data consistency (Gao *et al*., 2005; Budiarto and Tsukamoto, 2002; Mat *et al*., 2004). One of the simplest techniques for managing replicated data is Read-One-Write-All (ROWA) technique. Read operations on an object are allowed to read any copy and write operations are required to write all copies of the object (Budiarto and Tsukamoto, 2002).

In this study, we recall the replication concept and ROWA model. Next, we present the framework of ROWA-MST and show the implementation of ROWA-MSTS. Next, we present and analyze the result of implementation.

MATERIALS AND METHODS

System model: Replication is an act of reproducing. It also addresses the management of the complete copying process (Buretta, 1997). Any type of data processing object can be implemented. These include the data files; entire databases, specific tables, data within a specific tablespace; service types such as Telnet, File Transfer Protocol (FTP) and Simple Mail Transfer Protocol (SMTP).

Read-One-Write-All (ROWA): The simplest technique for managing replicated data is Read-One-Write-All (ROWA) technique. A read operation is allowed to read any copy of data. Meanwhile, a write operation is required to write all copies of data. All of the replicas have the same value when an update transaction commits. ROWA works correctly since a transaction processes from one correct state to another correct state. This technique has the lowest communication cost of read operation. This is because only one replica is accessed by a read operation. An available copies technique proposed by Bernstein and Goodman (1984) is an enhance version of ROWA technique, in terms of an availability of the write operations. Every read is translated into read of any replica of the data object. Meanwhile, every write is translated into write of all available copies of that data object. This technique handles each site either it is operational or down. All operational sites can communicate with each other. Therefore, each operational site can be independently determined which sites are down, simply by attempting to communicate with them. If a site does not respond to a message within the timeout period, then it is assume to be down. Nonetheless, writing is very expensive when all copies are available. The read-write transactions have been forced to write all replicas.

RESULTS AND DISCUSSION

Implementation of ROWA-MSTS: The framework of ROWA-MSTS involves 5 phases lock that includes initiate lock, propagate lock, obtain lock, update, commit and release lock. Figure 1 shows the locking phases of ROWA-MSTS.

Each of server used this phases when there are invoking transaction request to update data. Figure 2 shows the framework of ROWA-MSTS between 2 servers.

This experiment was done in shell programming with File Transfer Protocol as the communication agent. Bourne Again Shell (Bash) is used as command line editing since support command line editing and jobs control facilities. The job control includes great flexibility in handling the background process. Ubuntu 9.04 Jaunty distribution is used as the platform for the replicated server. Lock status is important in run level. It indicates the current status for the server in the distributed system. Table 1 shows the status lock set in ROWA-MSTS.

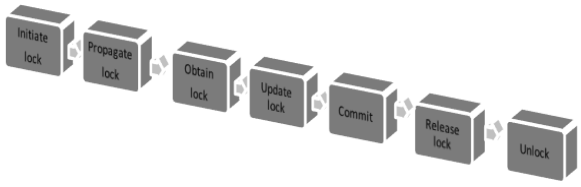

Fig. 1: Locking phase

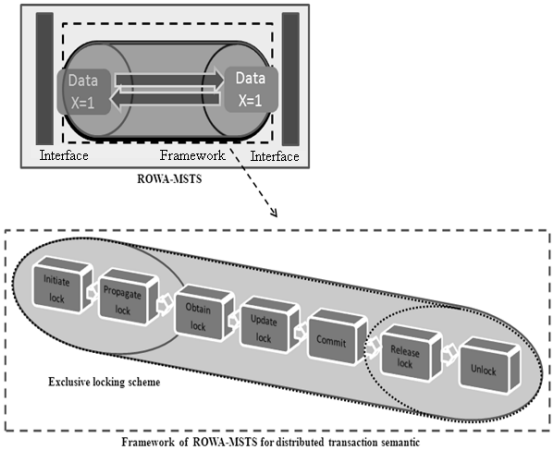

Fig. 2: ROWA-MSTS between 2 servers

Table 1: Status lock for ROWA-MSTS

Lock status	Description
0	If the lock status is set to 0, means that the server is ready to accept any transaction from other server.
1	If the lock status is set to 1, then the server is not available for any transaction or might probably busy with other transaction.
2	If the lock status is set to 2, the server is initiating the transaction or become the primary server.

Table 2: Master-neighbor coordination

Primary	Neighbor	
B:	A:	C:
172.21.140.192	172.21.140.44	172.21.140.33

Table 3: ROWA-MSTS handle the transaction

Replica time	A	B	C
t1	Unlock(x)	Unlock(x)	Unlock(x)
t2		Begin transaction Write_lock(x) Counter write(x) = 1 wait	
t3	Propagate lock: A		Propagate lock: C
t4	Lock (x) from B		Lock (x) from B
t5		Get lock A Count_write = 2 Get lock c Count_write = 3	
t6		Obtain quorum	
t7		Acknowledge client Update x	
t8	Commit Tx,t = 8	Commit Tx,t = 8	Commit Tx,t = 8
t9	Unlock(x)	Unlock(x)	Unlock(x)

In this experiment, no failures are considered during the transaction execution. The experiment aims to preserve the file consistency of during the execution. To execute the system, 3 replication servers are deployed. Each of the servers was connected to each other via fast Ethernet router. Table 2 shows the coordination between master and neighbor coordination.

Consider a case where transaction come to update data file x at server B. The ROWA-MSTS will implement the framework of ROWA-MSTS during the file replication.

PC A have the highest time taken for each lock to complete most probably because the location of the server is further compared to distance of PC C to master server, PC B. PC C has less time taken to complete each lock since it is located nearer to the PC B. However, the time for PC B is decrease to 0 because as master server, PC B only need to initiate lock and monitor the transaction and not involved in the locking phase.

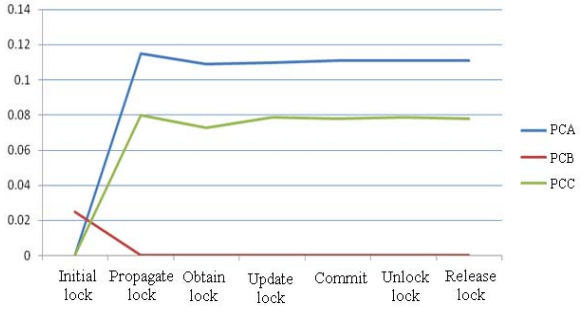

Fig. 3: Graph locking phase over time

Figure 3 shows the graph looking phase over time. From Fig. 3, it can be concluded that the time taken for each lock to be completed are consistent.

CONCLUSION

Based on experiment and result, it shows that ROWA-MSTS solves the distributed concurrency transactions and guarantees the data consistency in distributed systems. This is due to the transaction execution is equivalent to one-copy-serializability.

REFERENCES

Bernstein, P.A. and N. Goodman, 1984. An algorithm for concurrency control and recovery in replicated distributed databases. ACM Trans. Database Syst., 9: 596-615. DOI: 10.1145/1994.2207

Budiarto, S.N. and M. Tsukamoto, 2002. Data management issues in mobile and peer-to-peer environment. Data Knowl. Eng., 41: 183-204. DOI: 10.1016/S0169-023x(02)00040-X

Buretta, M., 1997. Data Replication: Tools and Techniques for Managing Distributed Information. 1st Edn., John Wiley and Sons, New York, ISBN: 10: 0471157546, pp: 384.

Gao, L., M. Dahlin, A. Nayate, J. Zheng and A. Iyengar, 2005. Improving availability and performance with application-specific data replication. IEEE Trans. Knowl. Data Eng., 17: 106-200. DOI: 10.1109/TKDE.2005.10

Holliday, J., R. Steinke, D. Agrawal and A. El Abbadi, 2003. Epidemic algorithms for replicated databases. IEEE Trans. Knowl. Data Eng., 15: 1218-1238. DOI: 10.1109/TKDE.2003.1232274

Mat Deris, M., A. Noraziah, M.Y.M. Saman, A. Noraida and Y. Yuan, 2004. High system availability using neighbor replication on grid. IEICE Trans. Inform. Syst. Soc., E87-D: 1813-1819.

Mat, D., J.H. Abawajy and H.M. Suzuri, 2004. An efficient replicated data access approach for large scale distributed systems. Proceeding of the International IEEE/ACM Conference on Cluster Computing and Grid, Apr. 19-22, IEEE Computer Society, Washington DC., USA., pp: 588-594. DOI: 10.1109/CCGrid.2004.1336663

Noraziah, A., M. Mat Deris, R. Norhayati, M.Y.M. Saman and M. Rabiei *et al*., 2006. Managing transactions on grid-neighbor replication in distributed system. Int. J. Comput. Networks, 86: 1624-1633. DOI: 10.1080/00207160801965198

Stockinger, H., 2001. Distributed database management systems and the data grid. Proceeding of the 18th IEEE Symposium on Mass Storage Systems and Technologies, Apr. 17-20, IEEE Computer Society, Washington DC., USA., pp: 1-12. DOI: 10.1109/MSS.2001.10003

Tang, M., B.S. Lee, X. Tang and C.K. Yeo, 2006. The impact on data replication on job scheduling performance in the data grid. Future Generat. Comput. Syst., 22: 254-268. DOI: 10.1016/j.future.2005.08.004

Zhang, J. and P. Honeyman, 2004. Replication control in distributed file systems. University of Michigan. http://www.citi.umich.edu/techreports/reports/citi-tr-04-1.pdf