Framework of Persistence Layer Synchronous Replication to Improve Data Availability into a Heterogeneous System

A. H. Beg, A. Noraziah, A. N. Abdalla, and K. F. Rabbi

Abstract—**Data replication is an important technique in peer to peer network, data grid architecture, clustering and distributed system, where it increases data availability and enhances data access and reliability and minimizes the cost of data transmission. In this paper, we proposed a framework and structure of synchronous replication from the persistence layer that supports heterogeneous system. In this framework, we developed the multithreading based persistence layer. Our objective is to make the persistence layer more adaptive. In this adaptive persistency system, the replication server will not depend on the main server, so forth, adding a new replication server will be easier than ever, easy to cope with heterogeneous system, cost minimizing and finally there will be no down time.**

Index Terms—**Data replication, data persistency, synchronous, heterogeneous replication, multithreading technique.**

I. Introduction

The heterogeneous computing system is a very promising platform, as the single parallel architecture based systems may not be sufficient for a running application to exploit the parallelism. In some cases, heterogeneous distributed computing (HDC) systems can achieve higher performance than the single super computer systems; moreover, it puts the lower cost than the super computer. However, the HDC system is more exceptions oriented so it may put a negative impact on the running application [1]. The homogeneous computing system is easier to control: the reason is the processing times are no dependent and identically with an arbitrary identically distribution [2].

A distributed heterogeneous computing (DHC) is a collection of autonomous dissimilar computing machines that are linked by a network and synchronized by software that functioning as a single powerful computing facility. As Computer tasks can be broken into different parts so it is possible to distribute the task for parallel execution. A DHC system has some advantages over homogeneous computing because some parts of an application perform better in some system, and some parts may perform better in another system [3].

The heterogeneous Computing systems can achieve both capability based (aimed at minimizing the completion time of one big job), and capacity based (aimed at maximizing the number of completions of small jobs within a given time) jobs [4].

Data replication is a process that copies and maintains database objects, such as tables, in multiple databases. A change of the main database is reflected, forwarded and applied at each of the replicated server which might be in a remote location [5]. Replicated database and a distributed database sound the same. However, in a distributed database, data is available in many locations, but a particular table resides at only one location [6]. For example, the employee's table resides at only the pah.employee database in a distributed database system that also includes the kl.employee and kn.world databases. Even so, replication means that the 100% same data at another location [7]. Replication balances the data transaction, and it provides fast, local access to shared data over multiple sites [8]. So replication works as a load balancing.

Data replication can be drives by programs which transport data to some other location and then loaded at the receiving location. Data may be filtered and transformed during replication. Replication must not interfere with existing applications and should have the minimal impact on production systems. The replication processes to need to be managed and monitored [9]. So finally, data replication improves data access time, transaction time and provides fault tolerance by maintaining and managing multiple copies of data (e.g. files, objects, databases or parts of databases) at different locations [10].

A replication environment can use either asynchronous or synchronous replication to copy data. With asynchronous replication, changes are made one after a certain time with a lot of data from the master site to the different other site. With synchronous replication, changes made immediately once some data transaction occurs to the mater site. Using synchronous replication, an update of transaction results immediately replication of the update at all other sites [5]. Synchronous replication is the appropriate solution for organizations: seeking the fastest possible data recovery, minimal data loss, and protection against database integrity problems. It ensures that a remote copy of the data, which is identical to the primary copy, is created at the time the primary copy is updated. The main benefit of synchronous replication is that data can be recovered quickly. Operations at the remote, mirrored site can begin immediately at the point in time where the primary site stopped should operation at the primary site be disrupted. Only the few operations within the process at the instant of disruption may be lost.

Manuscript received November 11, 2012; revised January 23, 2013. This work was supported in part by the Ministry of Higher Education Malaysia for project financing under Fundamental Research Grant Scheme RDU100109; and University Malaysia Pahang under UMP Short Term Grant RDU080328

A. H. Beg, A. Noraziah, and K. F. Rabbi are with the Faculty of Computer Systems and Software Engineering, University Malaysia Pahang, Gambang-26300, Kuantan, Malaysia (e-mail: ahbeg_diu@yahoo.com, noraziah@ump.edu.my, zahid_r_i@yahoo.com).

A. N. Abdalla is with the Faculty of Electrical and Electronics Engineering University Malaysia Pahang, Gambang-26300, Kuantan, Malaysia (e-mail: ahmed@ump.edu.my).

DOI: 10.7763/IJCTE.2013.V5.759

Because neither the primary nor the remote site will have a record of those transactions, the database rolls back to the last confirmed state [11].

The persistence layer is an architectural layer whose job is to provide an abstract interface to information storage mechanism(s). An API to such an interface should be abstract, and independent of storage technologies. It would typically have features such as: Store / retrieve of the whole database objects by key, Logical database cursor abstraction for accessing some / all instances of a given type, Transaction support: open, commit, and abort, rollback and Session management. Some basic querying support, e.g. how many instances of a given type exist, etc. usually such layers are built from at least two internal layers of software: the first being the abstract interface, the second being a set of bindings, one for each target database. In practice, there may be three layers since there may be an internal division between the logic for object and relational (and other) storage mechanisms [12].

The proposed layer will use as a multi-threaded application and acts like an interface between database and the main system. The persistence layer will have a single thread which will be responsible for making communication with the main server and will have another thread running to manage the replicated databases. So it will help the entire system to reduce dependency of the replicated server on the main server. When the main server crashes it will intelligently start using replicated server. So it will make no down time for the entire system and the persistence layer will be like an interface between the database and the entire system. So adding more replicated server is as like as plug and play.

II. Related Work

In the grid community, distributed and clustering system lot of work have focused on providing efficient and safe replication management services through designing of algorithms and systems. Businesses or specially Enterprise business or industrial business use replication for many reasons. Replication technology creates data replication on the right node from where the data transmission becomes faster. Like a network is in some remote location separated from the main server, and the data transmission rate is too high. Thus, a replication server can be created on that remote location which in terms helps the remote system reduce data transmission impediments and improve visit delay, bandwidth consumption and system reliability [13].

Hitoshi Sato et al. [14] proposed an approach for file clustering based replication algorithm in a grid environment. The goal is to create a technique to automatically determine optimal file replication strategies. Their approach outperformed groups file stored within a grid file system according to the relationship of simultaneous file access and determines locations and movement of replicas of file clusters from the observed performance data of file access and implementation specification was in Linux 2.6.1.8. The authors do not consider the heterogeneous system and also the replication in grid environment needs a lot of inter connection speed (gigabyte).

In [15] an intelligent replication framework for data grid has been proposed. The main goal of their approach is to create a replica management service that interrogates replica placement optimization mechanisms and dynamic replication techniques, coupled with computation and job scheduling algorithms for better performance in data grids. They use dynamic ordinary replication strategies and replica placement schemes. Their result shows that their approach improves the job execution time by 10-23%. Their work, however, replica managements are only coupled with computational job scheduling, which actually better performs in Symmetric Multi-Processors Server (SMP).

Yuan sheng Lou et al. [16] studied a reflected persistence data layer framework based on O/R mappings were designed and implemented. Persistence data layer is the most important part to the information system design. It is the foundation of the performance from the system and its migration ability. In this paper, they presented five modules: data loadable module, data write module, database services module, primary key cache module and paging cache module for persistence layer. However, reflection is not native to the OS. A lot of execution handling mechanisms should be included within the system. Besides replication using reflection mechanism is a very slow process and takes a lot of memory and might cause the buffer overflow.

In the peer-to-peer network using dynamic replication proposed a load sharing technique [17] providing and improving access performance there has been proposed two load sharing techniques, which use data replication. At the first technique there has been used a periodic push-based replication (PPR) to reduce the hop count (the number of legs traversed by a packet) and at the second technique, it uses On-Demand Replication (ODR) that performs and improves access frequency. However, they proposed two algorithms: improve access performance on a P2P network. A P2P network is typically very insecure, and also they don't have any strategy to large distributed application.

III. Framework of Heterogeneous Replication

A. Multi-Threading in Persistence Layer

In the proposed framework, the persistence layer will queue all the database transaction for the replicated server which will run in a single thread. On the other hand, the persistence layer will have another higher-priority thread, which will make a sound transaction with the main server. All the database server connection will be configurable via XML configuration. This will help the system that if the system needs another replicated server, it just needs to modify the configurable XML and plug it to the system, and the data replication will start immediately. When the database server crashes then the persistence layer will be intelligent enough to start using sound transaction from one of the replication servers and will send system alert messages to the administrators. It will make no down time to the system. This paper presents a framework of persistence layer for synchronous data replication (Fig. 1), and they implement the corresponding function. The describing of the various modules as follows:

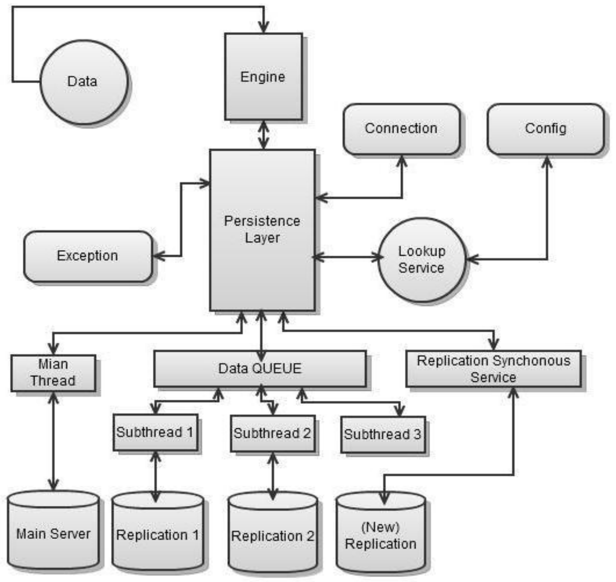

Fig. 1. Framework of persistence layer

1) Exception: The module/layer Exception will handle all kinds of exception to the system. A database exception handler layer will be incorporated within this layer. In a system typically an exception handler is defined by the occurrence of exceptions, special conditions that change the normal flow of program execution. Thus, memory overflow, null pointer, divided by zero is the special conditions which in terms handled by the Exception layer. In this proposed system, when a database server became to overload or crashes and/or main server or any of the replication servers crashes; (It will be known as an exception) it will look out of the box and create an exception thus send it to exception handling layer. As an example, when the main server crashes, the exception layer will get acknowledged it immediately, and it will then pass the exception type to the main engine which will then take the proper actions.
2) Global Configuration: Global Configuration is a system resource where all the static data, class, etc. will be introduced by its values. As an example in a web enterprise application of the data flow/routing class and the URL has been defined in the global configuration section.
3) Connection String: Connection string is the technical definition of software-development practices. Connection string is an expression string that identifies the data store to use for a particular query or set of queries and ways to connect. Connection string holds all the connection expression of all the replication servers and main server.

The entire system will help an enterprise system to become more secure, reliability through database replication. All the data will be synchronously replicated to the different database server. In the entire system will be too much flexible to handle any kind of database server. As an example, the system might be using SQL Server as the main server and other replicated server can be Oracle, MySQL, DB2, Informix even MS access. The proposed system will create such a system because a company might use SQL Server in Windows for the security shake, and it will cost a lot of money, so if the company wants to make the replication though all the SQL Server, then it will cost lots of money. However, if it is possible to replication with the open source like MySQL, then it will cut down the costs. Besides, it is possible to have as like as the plug and play ability of the system. As an example: in the system, any replicated server can be added whenever someone wants and for that not need to stop the server and configure the replication and then restart the server. When a replication server will be added to the system then needs to add some connection string and the API details in the global configuration. It will also have a lookup service on that configuration file. So when something is added or removed from the configuration file then the lookup service will update the system and also it will have another service will be known as replication synchronous service. The replication synchronous service will update the newly added replication server with the bulk data from the main server. This service will add all the data from the main server and update the newly added replication server.

The synchronous replication will be run by the proposed algorithm. The replication will be synchronous, and we will make it by use of multi-threading. A thread with higher priority will keep consistent connection with the main server. Then the persistence layer will create other many threads for all the replication servers. One thread will have the higher priority, and all other will have the lower priority in the system. So data that should be saved or deleted or modified will be kept in a queue, then it will make its own copy and do the transaction with that thread.

B. Heterogeneous Replication Process

In the heterogeneous replication system, the network architecture built through different OS. In the proposed system, the main server contains the software main engine (SME) which contains the independent dlls (dynamic link library) (in C#) or jar files (in Java) to solve complex business logics. On the persistence layer; the data connectivity, complex query execution and the ORM (Object Relational Mapping) executes. The persistence layer is also responsible for more different service like Data Queue (DQ) and replication synchronous service (RSS) that supports the heterogeneous system shown in Fig. 2.

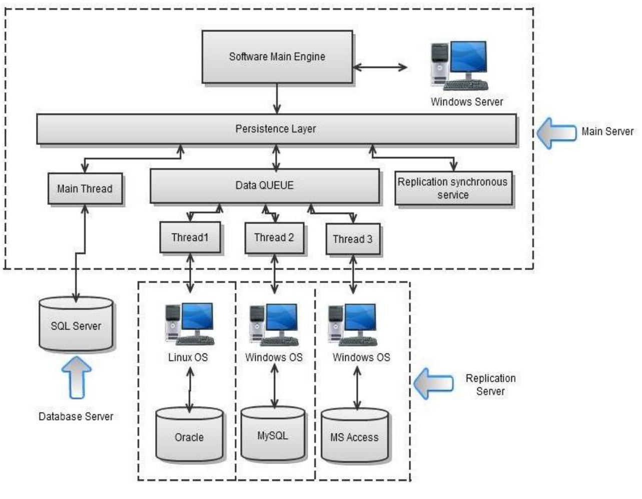

Fig. 2. Heterogeneous replication process

In the heterogeneous system, all the replication servers are connected from the persistence layer through the DQ. A multithreading is introduced from the DQ. So in this system; data will flow from persistence layer to the main server in a high-priority thread and to the different replication server through lower priority thread.

IV. Result and Discussion

The persistence layer framework principle lied upon the conception of faster replication of data as well as fewer bandwidths. Persistence layer creates an interface with the main server who communicates with the different database server relates with them. This ensures an efficient way to replicate data in a very short time with faster data transmission rate. Replication is considered in terms of Host to Host and Disk to Disk. Comparing the Persistence Layer Replication data transfer rates with other existing replication methods it can be seen that it works faster than the other replication methods. Different replication algorithm applied and performs on different data sets, although they took the different time interval for the transmission of various length data. In this paper has been considered a fixed length data. Various replication mechanisms applied on them, and comparison shows that Persistence Layer replication for synchronous data takes significantly less time than other replication methods. Various replication algorithms are compared with the basis of bit rate, data length are shown in the Fig. 3.

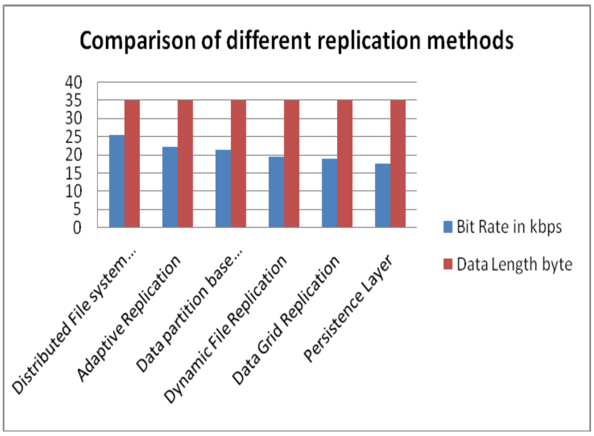

Fig. 3. Graph of various replication algorithms

The graph data represents that, in distributed file system data transfer occurred at 22.3 kbps, in data partition base replication it is 21.5, in dynamic file transfer replication it is 19.6 and as well as in data grid replication the transfer rate is 19 kbps. The above shows that, persistence layer replication transfer data on the speed of 17.5 kbps which is more considerable than the other replication algorithms.

V. Conclusion

In this paper we have proposed a structure and framework of synchronous replication that supports heterogeneous system, which will help enterprise application more secure and reliable data transmission. One of the main goals is to make the database replication more and easier to handle thus make it vastly configurable and also the whole architecture is service oriented means; it will use latest technology trends, and the replication will be from the persistence layer. Persistence layer is a part of software engine, and it will use the latest customizable fourth generation language like C# or Java. Therefore, a new era can begin related to networking and as well as database programming. Further work of this paper will include the creating of the algorithm which supports synchronous replication and as well as creating a distributed application that will test the entire replication system.

Acknowledgment

Appreciation conveyed to Ministry of Higher Education Malaysia for project financing under Fundamental Research Grant Scheme RDU100109; and University Malaysia Pahang under UMP Short Term Grant RDU080328.

References

[1] X. Tanga, L. Kenli, L. Renfa, and B. Veeravalli, “Reliability-aware scheduling strategy for heterogeneous distributed computing systems,” *Journal of Parallel Distributed Computing*, vol. 7, pp. 941-952, 2010.

[2] X. Tong and S. Wanneng, “An efficient dynamic load balancing scheme for heterogenous processing system,” *IEEE Conference on Computational Intelligence and Natural Computing*, pp. 319-322, 2009.

[3] W. F. Boyera and G. S. Hura, “Non-evolutionary algorithm for scheduling dependent tasks in distributed heterogeneous computing environments,” *J. Parallel Distrib. Comput,* vol. 65, pp. 1035-1046, 2005.

[4] Guest editorial, “Heterogeneous computing,” *Parallel Computing*, vol. 31, pp. 649- 652, 2005.

[5] S. Rajasekhar, B. Rong, K. Y. Lai, I. Khalil, and Z. Tari, “Load sharing in peer-to-peer networks using dynamic replication,” in *Proc. 2oth International Conference on Advanced Information Networking and Applications*, pp. 1011-1016, 2006.

[6] R. Urbano. (2003). Oracle Database Advanced Replication. Oracle Corporation. [Online]. Available: http://www.databasebooks.us/oracle_0003.php

[7] Y. Wang and S. Li, “Research and performance evaluation of data replication technology in distributed storage systems,” *Computers and Mathematics with Applications*, vol. 51, pp. 1625-1632, 2006.

[8] B. Charron, F. Pedone, and A. Schiper, “Replication theory and practice,” *Springer Berlin Heidelberg, NewYork*, Springer, 2009.

[9] Y. Lin, *Practical and Consistent Database Replic Ation*. Mc Gill University Montreal, Quebec, 2007, ch. 1-2.

[10] G. Lijun, L. Budd, A. Caycl, C. Hendricks, P. Micks, and C. Rigdon. *Practical Guide to DB2 UDB Data Replication V8. Durham*, NC, USA, 2002.

[11] C. Uros, B. Slivnik, and B. Robic, “The complexity of static data replication in data grids,” *Parallel Computing*, vol. 31, pp. 900- 912, 2005.

[12] Hitachi data system. (2007). Synchronous Data Replication. [Online]. Available: http://www.hds.co.uk/assets/pdf/sb-synchronous-data-replication.pdf

[13] Open EHR. Copyright (2007). [Online]. Available: http://www.openehr.org/208-OE.html?branch=1&language=1

[14] T. Gao and F. Liu, “A dynamic data replication technology in educational resource grid. information technologies and applications in education,” *First IEEE International Symposium*, pp. 287-291, 2007.

[15] H. Sato, S. Matsuoka, and T. Endo, “File clustering based replication algorithm in a grid environment,” in *Proc. 9th IEEE/ACM International Symposium on Cluster Computing and the Gri*d, pp. 204-211. June 2009.

[16] A. Elghirani, A. Y. Zomaya, and R. Subrata, “An intelligent replication framework for data grids. computer systems and applications,” *AICCSA '07. IEEE/ACS International Conference*, pp. 351-358. June 2007.

[17] Y. Lou, Z. Wang, L. Huang, and L. Yue, “The study of a reflected persistence data layer framework software engineering,” *WCSE '09. WRI World Congress*, pp. 291-294, 2009.

Abul Hashem Beg was born in 3rd November, 1981 in Tangail, Bangladesh. He received Master of Computer Science (Research) from University Malaysia Pahang, Malaysia in 2011. Currently he is doing Ph.D. in Signal Processing at the Faculty of Computer Systems & Software Engineering, University Malaysia Pahang. He is working as a Teaching Assistance at the same faculty. He performed as a Network Engineer at Global Network Inc., Internet Service Provider (ISP) in Bangladesh from 2005 to 2006. He also served his duty as an Assistant System Administrator, at CPM Blue Online Limited, Internet Service Provider (ISP) in Bangladesh from 2006 to 2009. He has published more than 20 research papers in the journals and international conference proceedings. His current research interests include distributed database, distributed systems, database replication, heterogeneous system, cloud computing and biofeedback.

Noraziah Ahmad was born in Penang, Malaysia on 1980. She received Ph.D in Distributed Database from University Malaysia Terengganu (UMT) in 2007. Currently, she is an Associate Professor and Head of Program (Master) in Faculty of Computer Systems and Software Engineering, University Malaysia Pahang. She has published more than 180 papers in the journals and conference proceedings. In addition to serving as international program committee member and reviewers in many conferences, she is currently an editorial board member of the International Journal of Engineering and Technology (IJET), International Journal of Web Application (IJWA) and Journal of Emerging Technologies in Web Intelligence (JETWI). She is also an advisory board of The Society of Digital Informatics and Wireless Communications (SDIWC). Her current research interests include distributed database systems, distributed systems, data grid, cloud computing and biofeedback.

Ahmed N. Abdalla was born in 19th April 1972 in Baghdad, Iraq. He received Ph.D in Electrical Engineering from HUST, Wuhan, China in 2002. He has published more than 150 papers in several journals and conference proceedings. Currently, he is the Associate Professor of the Faculty of Electrical and Electronics Engineering, University Malaysia Pahang. He is an honorable member of IEEE, USA, PDEA, FKEE, and UMP. He is delighted as the Member of research and PG, FKEE, UMP. He was also a member of Consulting Bureau at University of Technology, 2002, Iraqi Engineers Union in 1997. Previously, he served as a Lecturer, at the Department of Electronics & Electrical Engineering. UOT, Iraq in 2002. He was performed as a Dean of Electronics Center of Training and workshops, UOT, Iraq in 2007. His current research include system optimization, identification, system control, application of intelligent Technique's, wireless application, electrical machine and driver's and biofeedback