Computer Engineering and Intelligent Systems
ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online)
Vol 3, No.8, 2012
www.iiste.org

An Asynchronous Replication Model to Improve Data Available into a Heterogeneous System

Mohammad Naderuzzman (Corresponding Author)
Department of Computer Science & Engineering
Dhaka University of Engineering & Technology,
Gazipur, Dhaka
E-mail: nader_u@yahoo.com

Dr. Md. Nasim Akhtar
Department of Computer Science & Engineering
Dhaka University of Engineering & Technology,
Gazipur, Dhaka
E-mail: nasim_duet@yahoo.com

Abstract
Data replication is an important technique for high level data availability, especially applied for distributed systems. To enhance the data access and data reliability, data replication is one of the most important aspects. This paper is proposed a asynchronous replication model. This model also supports heterogeneous system which is currently a very promising system. In our proposed model the main server and replication servers are loosely coupled. Since the structure is loosely coupled and asynchronous model thus there is less dependency on each server. Moreover the proposed model is supporting heterogeneous system, so it will be highly cost minimizing solution for efficient data replication.
Keywords: Data replication, Data Persistency, Asynchronous, Heterogeneous Replication.

I. INTRODUCTION

The heterogeneous computing system is very promising platform. Since to exploit the parallelism, the single parallel architecture based systems may not be sufficient for a running application. In some cases, heterogeneous distributed computing (HDC) systems can achieve better performance than the super computer systems; moreover it involves much lower cost than the super computer. However the HDC system is more exception oriented which may have negative impact on the running application [1]. Basically the homogeneous computing system is easier to control, because the processing times are not dependent and identically with an arbitrary identically distribution [2].

A distributed heterogeneous computing (DHC) system is a collection of autonomous and dissimilar computing machines those are linked by a network; these computing machines are synchronized by software to function like a single powerful computing facility. Computer's tasks are broken into different parts, and because of this it is possible to distribute the task for parallel execution. A DHC system has some advantages over homogeneous computing, some parts of an application perform in one system while some parts may perform better in another system [3].

Heterogeneous computing system may 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 computing process that copy and maintain database objects, such as tables, simultaneously in multiple databases. A change in a main database is reflected, forwarded and applied in each of the replicated servers which might be located in a remote location [5]. Replicated database and a distributed database may sounds same. But the difference between them is that, in a distributed database, data are available at many locations, but a particular table resides only at one location [6]. For example, the employees table resides at only the pah.employee database in a distributed database system which also includes the kl.employee and kn.world databases. In replication a 100% same data in different locations [7]. Replication balances the data transactions and provides fast, local access to shared data over multiple sites through network [8]. Replication works as to balancing load.

Data replication can be driven by using application programs which transports data to and from other locations and then load at the receiving location. During the replication, data might be needed to be filtered and transformed. This replication should not interfere with existing applications and must have minimal impact on

production system. Replication processes should be under managed and monitored [9]. So, data replication improves the transaction time, data access time and improves fault tolerance by maintaining and managing multiple copies of data (e.g. files, objects, databases or part of databases etc.) at different locations [10].
A replication process could be either asynchronous or synchronous replication for copying data. In asynchronous replication, changes are made after a certain time from the master site to different other site. And in synchronous replication, changes are made immediately after some data transaction occurs to the master site. Using asynchronous replication, updates of transactions results efficiently update at all other sites [5]. So asynchronous replication should be appropriate solution for organizations: seeking the fastest data recovery, minimize data loss, and protection against database integrity problems. It ensures that a copy of data must be identical to the primary copy which was created at the time the primary copy updated. The primary or main benefit of synchronous replication is that, data can be recovered quickly. Operation on remote or mirrored site begins immediately at any time if primary site stops, all the operations should be disrupted. Only few operations in process at that instant of disruption may be lost. Because the primary site and the remote site neither of them will have record of those transactions, at this situation the database rolls back to the last confirmed state [11].
The architectural layer which is persistence layer, whose job is to provide an abstract interface to storage mechanisms. Such an interface in the API should be abstract and independent of storage technologies. It should typically have the features like: Store/ retrieve of whole database object 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. Also some basic query support, like: how many instances of a given type exists etc. such layer usually are built from at least two internal layers of software: first being the abstract interface and second being a set of bindings, one for each targeted database. In practice, there are three layers; there may be internal division between the logic of object and relational (or other) storage mechanisms [12].
Our proposed layer will use a multi-threaded application and will act like an interface between database and the main system. In the persistence layer there will have a single thread which will be responsible for making communication with the main server and will have a different thread running to manage the replicated databases. It will help the entire system to reduce dependency of the replicated server on the main server. If the main server crashes it will intelligently start using replicated server. In this way such a system will make no down time for the entire system and the persistence layer will be like an interface between the database and the entire remote system. In this way, adding more replicated server will be just like plug and play.

II. RELATED WORK

In grid community, lots of work has focused on providing efficient and safe replication management services for distributed and clustering system through designing of algorithm and systems. Enterprise business or industrial business uses replication for many reasons. Replication technology creates the replication of data on the node that is right for it and from where the data transmission becomes faster. For example in a network some remote nodes may be far from the main server and in this case data transmission rate will be too high. This case a replication server can be created on to a remote location which in terms helps the remote system reduce data transmission impediments and thus improve visit delay, bandwidth consumption and system reliability [13].
Hitoshi Sato et. al [14] proposed an approach for file clustering based replication for working in a grid environment. His goal was to create a technique to automatically determining the optimal file replication strategies. The approach outperformed stored group files in a grid file system according to the relationship of simultaneous file access and it determines locations and movement of replicas of file clusters from the observed performance data of file access and the implementation specification was in Linux OS. The authors did not consider the heterogeneous systems and the replication in grid environment needs lot of inter connection speed e.g. in gigabytes.
In [15] an intelligent replication framework in data grid was proposed by the author. The goal of their approach was to create a replica management service that interrogate replica's placement optimization mechanisms and dynamic replication techniques, which coupled with the algorithms of computations and job scheduling for better performance in data grid. They used dynamic ordinary replication strategies and replica placement schemes and their result shows that their approach improves the job execution time by 10% to 23%. However their work replica management is only coupled with computational job scheduling which actually perform better in Symmetric Multi-Processors Server (SMP).
In a study a reflected persistence data layer framework based on O/R mapping were designed and implemented by Yuan-Sheng Lou et. al [16]. In the information system design, persistence data layer is the most important part. This is the foundation for the performance of the system and its migration ability. In the paper the author presented five modules: data loadable module, data write module, database services module, primary key cache module and paging cache module for persistence layer. In this case the reflection is not native to the OS. A lot of

execution handling mechanism needed to be included in to the system. Besides the replication by using reflection mechanism is a very slow process and takes a lot of memory involvement and might cause buffer overflow.
For a peer-to-peer network a load sharing technique was proposed by using dynamic replication [17]. For providing and improving access performance they have proposed two load sharing techniques, which use data replication. In the first technique they have used a periodic push-based replication (PPR) to reduce the hop count (the number of legs traversed by a packet) and in the second technique it uses on demand replication (ODR), which performs and improves access frequency. They have proposed two algorithms to improve access performance on a P2P network. A P2P network is typically not very much secure and they didn't have any strategy for large distributed application.

III. FRAMEWORK OF HETEROGENEOUS ASYNCHRONOUS REPLICATION

A. Multithreading in persistence layer

In the proposed framework, the persistence layer will queue all the transactions of database for the replicated server that will run in a single thread. On other hand, the persistence layer will have another thread that got higher priority will make a sound transaction with the main server. All the database server connection will be configurable by using XML configuration. This way 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. In case of database server crashes, the persistence layer will be intelligent enough to start using sound transaction from one of the replicated server and it will send system an alert message to the administrator. This will make no down time for the system. This paper presents a new framework for using in the persistence layer for asynchronous data replication (Figure 1), and they implement the corresponding function. The various modules are describing as follows:

(1) **Exception:** The Exception module/layer will handle all kind of exception of the system. The database exception handler will incorporate with this proposed layer. Typically in a system an exception handler is defined by the occurrence of exceptions, e.g. special conditions that change the normal flow of program execution. Thus, null pointer, memory overflow, dividing by zero are those special conditions which in terms handled by the Exception layer. In this proposed system, if a database server become overloaded or crashes (which will be known as exception) it will look out of the box and create an exceptions thus send it to exception handling layer. For example, if the main server crash, the exception layer will get acknowledged immediately and it will then pass the exception type to the main engine which then take the proper action.

(2) **Global Configuration:** Global Configuration is a system resource where all the data, classes etc. will be defined by its values. For an example in a web enterprise application the data flow/routing class, the url(s) will be defined in the global configuration sections.

(3) **Connection String:** Connection string is a character string expression which uniquely identifies the data-store for using by a particular query or set of queries, and methods for connecting. In this proposed system, connection string will be used to hold all the connection expression between servers and persistence layer.

The entire system will help any enterprise system more secure and reliable by replicating the database. Data will be replicated asynchronously to different database server. The entire system will be much flexible to handle any kind of database server. For example, the system might be using Oracle server as the main server and other replicated servers can be SQL, MySQL, DB2, Informix or even MS Access server. The proposed system will develop such system, because a company might use SQL Server in Windows for the security reason and it will cost a lot of money, so if the company wants to make the replication through all the SQL Server then it will cost lots of money for each. But if it is possible to replicate with the open source database like MySQL, then it will drastically reduce the cost. Besides, in this proposed paper, it is possible to have something like plug-and-play ability into the system. For example in this system any new replicated server can be added whenever anyone wants and for this its not necessary to stop for long to all the servers, just reconfigure the replication and then restart the server. If a replication server is added to the system, it needs to add some connection string and the API details in the global configuration and will also have a lookup service on that configuration file. The lookup service will update the system the system as soon anything added or removed from the configuration file and also it will have another service that will be known as replication asynchronous service. The job of replication asynchronous service is to update the newly added replication server with the bulk data from the main server. This service collects all the data from the main server and update the newly added to the replication server.

A proposed algorithm will be use to run the asynchronous replication. In this replication, by using multi-threading we will make the replication asynchronous. A higher priority thread will keep consistent connection with the main server. After that the persistence layer will create other many threads for each of the replication servers. Among all threads a single thread will have the higher priority and all other will have the lower priority on the system. So, the record will be kept in queue of those data of saved, deleted or modified, then it will be make its own copy and do the transaction with that thread.

B. Heterogeneous Replication Process

In the heterogeneous replication system the network architecture is built with different OS. In this proposed system the software's main engine(SME) resides in the main server which contains the independent DLLs (dynamic link library in C#) or JAR files (in JAVA) to solve complex business logics. The data connectivity, complex query execution and ORM (Object Relational Mapping) executes on the persistence layer. This layer also responsible for more different service like Data Queue (DQ) and replication asynchronous service (RAS) that supports the heterogeneous system (Figure 2).

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

The high level model has been shown in figure 1.

IV. Result & Discussion

The conception of faster replication of data and use of lesser bandwidth is the principle of the persistence layer framework. The job of the persistence layer is to create an interface with main server, which communicates with different database servers and relates with them. The procedure ensures efficient way for data replication within a very short time with better and faster data transmission rate. A Host to Host and Disk to Disk is considered in this replication. The Persistence Layer Replication's data transfer rate can be compared with other existing replication methods, and it can be determined that it works faster than other replication techniques. All different replication algorithms applied and found different performance found on different data sets, although they took different time interval for transmission of various data length. In our paper we have proposed that all the data will be of fixed length. Different many replication mechanisms applied on them fixed length data and the resultant comparison shows that the asynchronous Persistence Layer Replication takes significantly less time than other replication methods. The comparison of various replication algorithms were based on the bit rate and data length. The comparison is shown in figure 2.

From the Figure 2, we can compare very easily that our proposed model is much faster than other replication models available.

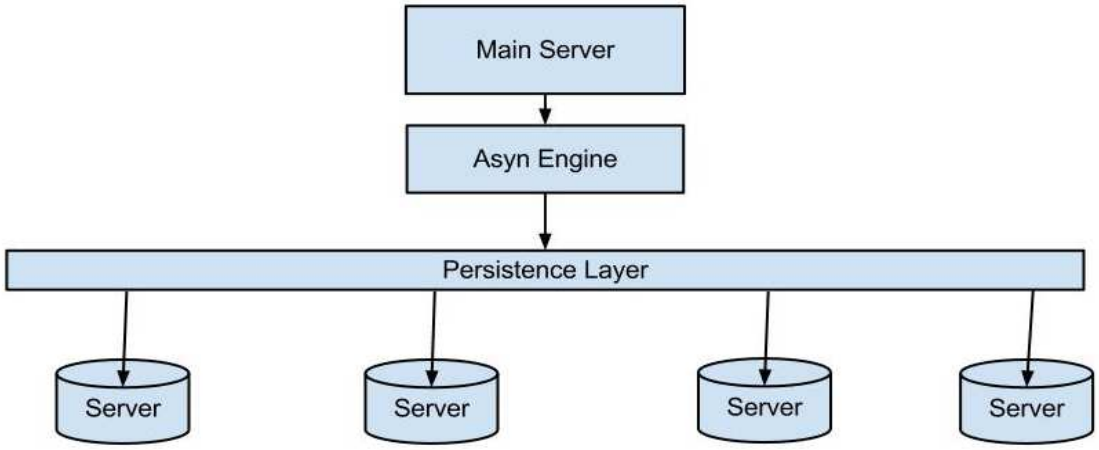

Figure 1: Asynchronous replication model

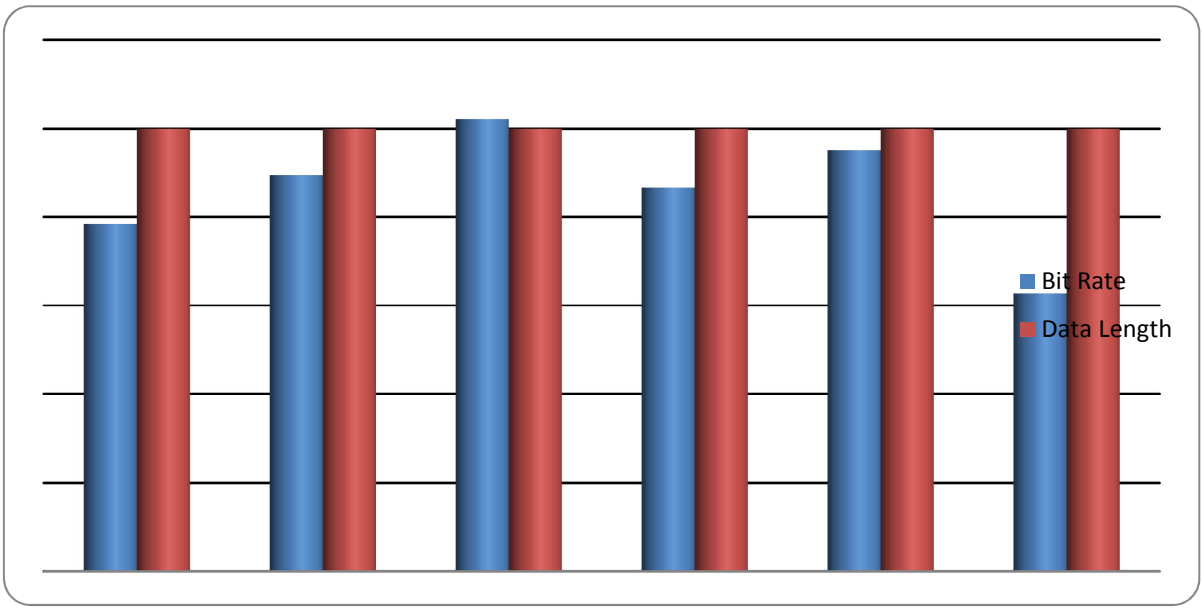

Figure 2: Graph of various replication algorithms

REFERENCES

[1] Xiaoyong Tanga, Kenli Li a,_, Renfa Li a, Bharadwaj Veeravalli,“ Reliability-aware scheduling strategy for heterogeneous distributedcomputing systems” J. Parallel Distrib. Comput. 70 (2010) 941_952
[2] Xiaonian Tong, Wanneng Shu, “An Efficient Dynamic Load Balancing Scheme for Heterogenous Processing System”IEEE Conference on Computational Intelligence and Natural Computing, 2009, pp. 319 – 322
[3] Wayne F. Boyera, Gurdeep S. Hura, “Non-evolutionary algorithm for scheduling dependent tasks in distributed heterogeneous computing environments” J. Parallel Distrib. Comput. 65 (2005) 1035 – 1046
[4] Guest editorial, “Heterogeneous computing” Parallel Computing 31 (2005) 649–652
[5] S Rajasekhar, B Rong, K Y Lai, I Khalil and Z Tari, “Load Sharing in Peer-to-Peer Networks using Dynamic Replication” Advanced Information Networking and Applications, AINA 2006. 20th International Conference, April 2006,pp. 1011 – 1016
[6] Randy Urbano, “Oracle Database Advanced Replication”, Part No.B10732-01,Oracle Corporation, 2003,ch. 1, Retrieved February 8, 2010, from http://www.databasebooks.us/oracle_0003.php
[7] Yijie Wang , Sijun Li, “ Research and performance evaluation of data replication technology in distributed storage systems” August 2006, Computers & Mathematics with Applications 51 (2006) 1625-1632
[8] Bost, Bernadette Charron, Fernando Pedone, Andr´e Schiper,“ Replication Theory and Practice” Berlin Heidelberg NewYork,
2009, Springer, ISBN-10 3-642-11293-5 Springer, ch. 2, Retrieved 14 February 2010, from http://www.ebook3000.com/Replication --Theory-and-Practice_36872.html
[9] Yi lin, ,“Practical and consistent database replic ation”, Mc Gill University Montreal, Quebec, 2007,ch. 1-2, Retrieved February 15,2010, from http://proquest.umi.com.ezproxy.ump.edu.my
[10] Lijun (June) Gu, LIoyd Budd, Aysegul Caycl, Colin Hendricks, Micks Purnell, Carol Rigdon,” Practical Guide to DB2 UDB Data Replication V8 ”.Durham, NC, USA, 2002, IBM ,ch.1,Retrieved February12, 2010, from http://site.ebrary.com/lib/kuktemlib/Doc?id=10112697&ppg=23
[11] Uros C ibej , Bostjan Slivnik, Borut Robic, “The complexity of static data replication in data grids”, 2005, Parallel Computing 31 (2005) 900–912
[12] Hitachi data system, Retrieved February 18, 2010, from http://www.hds.co.uk/assets/pdf/sb-synchronous-data-replication.pdf
[13] Open Ehr, Retrieved February 18, 2010, from http://www.openehr.org/208- E.html? branch=1&language = 1
[14] Tian Gao, Fang’ai Liu, “A Dynamic Data Replication Technology in Educational Resource Grid”, Information Technologies and Applications in Education, 2007. ISITAE '07. First IEEE International Symposium , Nov. 2007, pp. 287 – 291
[15] Hitoshi Sato, Satoshi Matsuoka, and Toshio Endo , “File Clustering Based Replication Algorithm in a Grid Environment”, 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, June 2009, pp. 204 – 211
[16] Ali Elghirani, Albert Y. Zomaya, Riky Subrata “An Intelligent Replication Framework for Data Grids”, Computer Systems and Applications, AICCSA '07. IEEE/ACS International Conference, June 2007, pp. 351 – 358
[17] Yuan-sheng Lou, Zhi-jian Wang, Long-da Huang and Lu-lu Yue, “The Study of A Reflected Persistence Data Layer Framework”, Software Engineering, 2009. WCSE '09. WRI World Congress , November 2009, pp. 291 – 294