Online Share Trading System Using Active Replication

Chan Mya Aye, Thet Su Mon
Computer University (Taung-Ngu)
7chanmyaaye@gmail.com

Abstract

In distributed system, the database can be replicated in multiple servers stored at different sites. Replication is a key to providing high availability and fault tolerance in distributed systems. A fault-tolerant service always guarantees strictly correct. There are two fault-tolerant replication systems: passive replication and active replication. In this paper, we propose online share trading system using active replication. To be reliable and fault tolerant, three active-based replicas will be used for this system.

Keywords: Fault-tolerant, Active Replication, Passive Replication, Group Communication

1. Introduction

Replication is the process of making a replica (a copy) of something. It is the process of copying data from a data store or file system to multiple computers that store the same data for the purpose of synchronizing the data. Replication is a key to provide high availability and fault tolerance in distributed systems [3].

A share is a document issued by a company, which entities its holder to be one of the owners of the company. A share is issued by a company or can be purchased from the stock market. By owning a share, one can earn a portion. By selling shares, he gets capital again. So his return is the dividend plus the capital gain. However, he also runs a risk of making a capital loss if you have sold the share at a price below his buying price. A company's stock price reflects what investors think about the stock. Market forces and general investor opinions can also affect share price. So, share prices are changing with the time.

In this system, there will be three active replicas of online share-trading servers to be reliable. They maintain the latest share price of registered companies and show users the updated share price. They can also handle the share trading among registered users.

This paper is organized as follows. Session2 presents the related work of replication. Background theory of replication and introduction of shares are discussed in session3.Proposed system design is presented in the session4. Session5 shows the implementation of the proposed system. Conclusion of the system is presented in session 6.

2. Related Works

The related works of replication are discussed in this session.

Andre Brito and Pascal Felber focused on active replication as an approach to provide fault-tolerance to Event Stream Processing (ESP) operators. More precisely, they addressed the performance costs of active replication for operators in distributed ESP applications. They used a speculation mechanism based on Software Transactional Memory (STM) to achieve the following goals: (i) enable replicas to make progress using optimistic delivery; (ii) enable early forwarding of speculative computation results; (iii) enable active replication of multi-threaded operators using transactional executions. Experimental evaluation showed that, using this combination of mechanisms, one can implement highly efficient fault-tolerant ESP operators. They have proposed new techniques to efficiently support active replication in fault-tolerant distributed event processing systems. By using an STM-based speculation mechanism, they allowed nodes to optimistically start processing events before their final delivery (before their respective order is know with certainty) [2].

Yair Amir and Ciprian Tutu presented a complete algorithm for database replication over partitionable networks sophistically utilizing group communication and proved its correctness. Their avoidance of the need for end-to-end acknowledgment per action contributed to superior performance. They showed how to incorporate online instantiation of new replicas and permanent removal of existing replicas. They also demonstrated how to efficiently support various types of applications that required different semantics [1].

M.Wisemann, F.Pedone and A.Schiper provided an abstract and "neutral" framework to compare replication techniques from both communities (distributed systems and databases). The framework has been designed to emphasize the role played by different mechanisms and to facilitate comparisons. Their paper described the replication techniques used in both communities, compared them, and pointed

out ways in which they can be integrated to arrive to better, more robust replication protocols [6].

3. Background Theory

The background theory of replication is discussed in this session.

3.1 Replication

Replication is a technique for enhancing services. The motivations for replication are to improve a service's performance, to increase its availability, or to make it fault tolerant. Replication consists of two or more replicas. Replicas are physical objects, each stored at a single computer. Replicas are held by distinct replica managers. Replica managers are components that contain the replicas on a given computer and perform operations up on them directly [3].

There are two approaches for fault-tolerant service

- passive (primary-backup) replication and
- active replication.

In this system, active replication will be used for fault tolerant. General model of replica management is shown in figure 1.

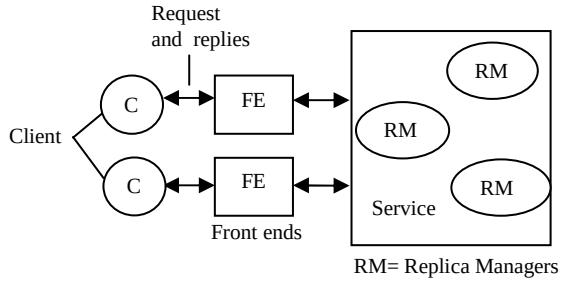

Figure 1. General Model of Replica Management

3.2 Passive (primary-backup) replication

The passive model for fault tolerant is shown in figure 2.

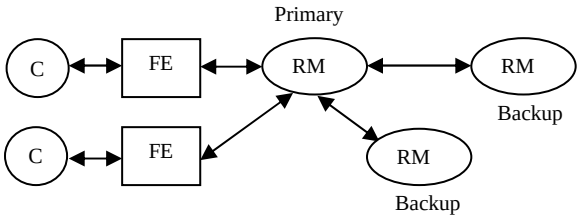

Figure 2. Passive Replication

In this model, there is at any one time, a single primary replica manager and one or more secondary replica managers – 'backups' or 'slaves'. In the pure form of the model, front ends communicate only with the primary replica manager to obtain the service. The primary replica manager executes the operations and sends copies of the updated data to the backups. If the primary fails, one of the backups is promoted to act as the primary [3].

3.3 Active Replication

The model for active replication is shown in figure 3. In this model, the replica managers play as a group. Front ends multicast their requests to the group of replica managers and all the replica managers process the request and reply. If any replica manager crashes, then the remaining replica managers continue to respond in the normal way. This configuration can reduce network load dramatically.

1. *Request*: The front end attaches a unique identifier to the request and multicasts it to the group of replica managers. It does not issue the next request until it has received a response.
2. *Coordination*: The group communication system delivers the request to every correct replica manager in the same (total) order.
3. *Execution*: Every replica manager executes the request identically. The response contains the client's unique request identifier.
4. *Agreement:* No agreement phase is needed.
5. *Response:* Each replica manager sends its response to the front end. The front end passes the first response to arrive back to the client and discards the rest [3].

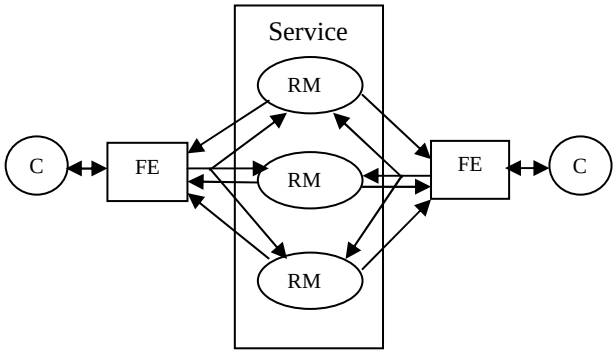

Figure 3. Active Replication

The active replication technique requires that the invocations of client processes be received by the non-faulty replicas in the same order. This requires an adequate communication primitive, ensuring the order and atomicity property. This primitive is called total order multicast or atomic multicast [4].

3.4 Group Communication

The group communication, as depicted in figure 4, is an adequate framework for providing the

multicast primitives required to implement both active and primary-backup replication. Several distributed systems provide this abstraction [5].

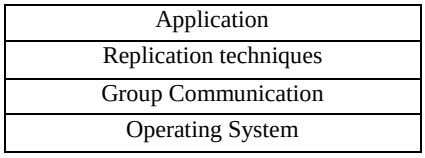

Figure 4. Group communication: the infrastructure for implementing replication

The group communication service has to manage changes in the group's membership while multicasts take place concurrently. A group membership service has four main tasks, as follows:

- Providing an interface of group membership changes
- Implementing a failure detector
- Notifying members of group membership changes
- Performing group address expansion [3]

3.5 Share

Share is a finite number of equal portions in the capital of the company. In Financial Markets, Share is a "unit of account" for various financial instruments. The person who owns shares is called shareholder.

3.6 Types of Share

There are four main types of shares.

1. **Ordinary shares** – standard shares with no special rights or restrictions. They have the potential to give the highest financial gains, but also have the highest risk. Ordinary shareholders are the last to be paid if the company is wound up.
2. **Preference shares** – typically carry a right that gives the holder preferential treatment when annual dividends are distributed shareholders. Shares in this category have a fixed value, which means that a shareholder would not benefit from an increase in the business's profits. However, Usually they have rights to their dividend ahead of ordinary shareholders if the business is in trouble. Also, where a business is wound up, they are likely to be repaid the par or nominal value of shares ahead of ordinary shareholders.
3. **Cumulative preference shares** - give holders the right that, if a dividend cannot be paid one year, it will be carried forward to successive years. Dividends on cumulative preferred shares must be paid, despite the earning levels of the business.
4. **Redeemable shares** - come with an agreement that the company can buy them back at a future date, this can be at a fixed date or at the choice of the business. A company cannot issue only redeemable shares.

4. Overview of the proposed system

The overview of the proposed system is shown in figure 5.

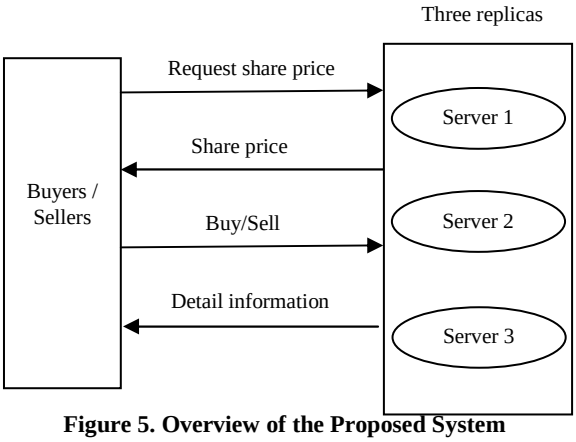

Figure 5. Overview of the Proposed System

. In this system, users can sell or buy shares. Share prices are changing all the time. Three replicas provide latest information about shares to users. Users can use this information to decide whether he should sell his own share or buy from others. All replicas do read and write operations with the same process and order. Therefore all of them keep the correct updated information.

5. System Implementation

The user needs to register to sell or buy shares via this system. Then, he can log in to see the share catalog of the system. The system shows the registered user the price and type of the company's share. Then the user can decide to buy that share or to sell his owns shares. For selling or buying, the user will receive the detail message of the bought share or sold share. The structure of group communication used in this system is shown in figure 6.

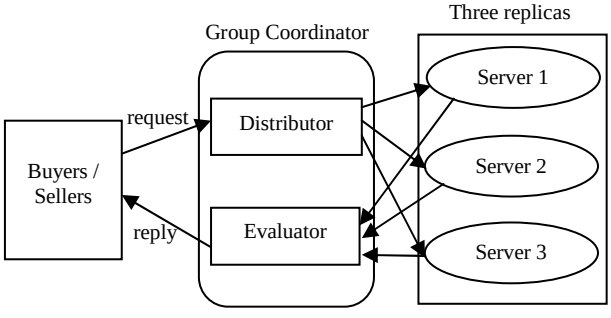

Figure 6. Group Communication of Proposed System

In this system, three replicas are grouped as a group, and the clients know a group address. The group communication knows all address of its members (replicas). When the clients make requests from the system, he needs to send the request by using the group address. The group (distributor) sends this request to all replicas. All replica do the same processing independently and reply the result to the group (evaluator). The evaluator chooses the first reply and send it back to the client. Replies from the rest two replicas are discarded by the evaluator.

As the failure can occur unexpectedly, any server can crash. If there is one server and it fails unexpectedly, this system will be no longer work. Therefore this system can guarantee for this condition because the three servers are used in this proposed system.

5.1 The Process of the System

In this system, the client as SQL statements can make query or update request to the servers through the group coordinator. For query request, the group coordinator will find to all servers and the servers will reply the information to the group coordinator. The group coordinator will reply this information to the client. And therefore, the group coordinator will get reply message from at least one of the servers even other servers are in crash.

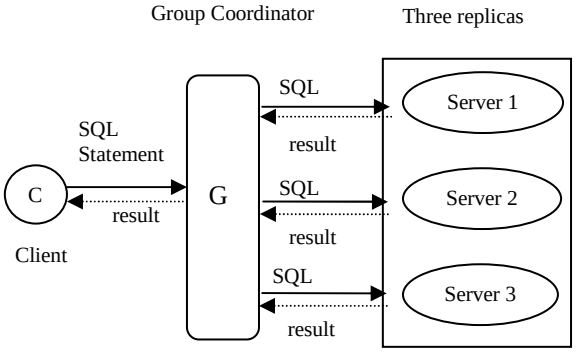

Figure 7. Request Flow Diagram

For update request, all of the servers need to be updated their databases according to client's update request. Group coordinator send this request to all servers with separate threads and wait the server's reply. If all server's reply are 'ok', this update is successfully completed in all servers. If one of the servers is in crash, this server are not reply 'ok' message. In this case this SQL update statements and the server name are store in group coordinator. When the failure server is restarted, this server sent request to the group coordinator to find wherether the update request for this server exists or not. If it is exist, group coordinator will send the SQL statement to this server to update servers' database for consistency.

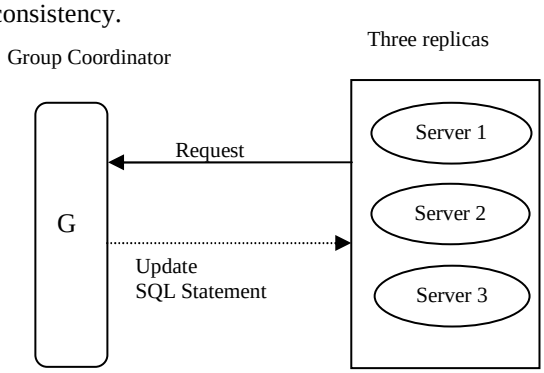

Figure 8. Updating Server Diagram

6. Experimental Result

Experiment was carried out on machines to measure the overheads of the system when a service group is deployed over a Local Area Network connection (LAN). In this experiment, the replicas are connected by LAN. To simulate that the replicas are connected by a LAN, each message exchanged amongst our system is held in a buffer before it is processed. Overhead is defined as (tn-t1)/t1, where tn is the service response time when the service group has n replicas and t1 is the service response time when no replication is used. The arrival rate and the processing rate are both simulated to follow an exponential distribution. The generated clients' requests are randomly distributed to the replicas. It is assumed that the arrival rate is less than the processing rate. That is, the system has sufficient processing capacity to handle the clients' requests.

- The crash of a replica is transparent to the client processes; the client never needs to reissue a request.
- Response time is low even in the case of a crash.
- No need of agreement between servers.
- Reliability increases.
- Both reads and updates are local.

Experiment is built on the same networking environment. Group communication is implemented as an active replication fault tolerant group. The server replicas are installed on three hosts on the same LAN, described in figure 6. The purpose of the experiment is to measure the response time delay of client invocation. Experiment shows that the response time rises when the number of replicas is increased. We assume that the number of replicas varies when replicas fail. Response time is measured in millisecond.

This system test with data from Stock & Share Market Educational option for investing Online & Online trading system. ICICI online trading system is the scenario: This scenario is in a form of a web optimized tutorial and it is rooted mostly on experience and budding knowledge. ICICI online trading system has been used as an example.

Below are some typical examples of how this stock market program helps user make money. In these actual trades, the software program led us every step of the way. It told us when to buy the stock and when to sell.

Table 1

Company	Code	Buy	Price	Sell	Price	Profit
Telkom Sa Limited	TKG	10/03/2009	9715	05/05/2009	12000	**23.5%**
Pick n Pay Stores Limited	PIK	09/03/2009	2815	04/06/2009	3600	**27.9%**
Rainbow Chicken Limited	RBW	13/02/2009	1300	20/05/2009	1690	**30.0%**
Mtn Group Limited	MTN	20/02/2009	8390	22/05/2009	12990	**154.8%**
SASOL LIMITED	SOL	23/01/2008	30300	21/05/2008	51400	**169.6%**
Merafe Resources Ltd	MRF	23/01/2008	160	30/06/2008	425	**265.6%**
Drdgold Ltd	DRD	20/11/2008	340	09/03/2009	943	**277.4%**
Anooraq	ARQ	06/03/2009	270	09/06/2009	1000	**370.4%**

6. Conclusion

In this system, online share trading system is implemented by using active replication strategy. Three active based replicas are used to build this system as a reliable share trading system. This System can be extended with many group coordinator. All replicas do the same process steps with same order for both read and write operations so that all have been update any time. This information can be browsed by the registered clients using Web browsers and search engines. Therefore, the user can get up-to-date share prices from this system. They can also make share selling and buying via this reliable system.

7. References

[1]. A.Brito and P.Felber, "Multithreading-Enabled Active Replication for Event Stream Processing Operators", Technische University Dresden, Germany.

[2]. G.Coulouris, J.Dollimore, T.Kindbrge, "Distributed Systems Concepts and Design", Third Edition, ISBN 0201-61918-0, 2001.

[3]. M.Wisemann, F.Pedone and A.Schiper, "Understanding Replication in Databases and Distributed Systems", EPFL-ETHZ DRAGON project.

[4]. R.Guerraoui and A.Schipe, "Fault-Tolerance by Replication in Distributed Systems", Proc. Reliable Software Technologies, 1996.

[5]. R.Gueraoui and A.Schiper,"Software-Based Replication for Fault Tolerance", Swiss Federal Institute of Technology, IEEE, 1997.

[6]. Y.Amir and C.Tutu, "From Total Order to Database Replication", Johns Hoplins University, Department of Computer Science.

[7]. L. Lingxia , X.Jingbo , M.Zhiqiang and L.Ruixin "Rapid-Response Replication: A Fault Tolerant Algorithm Based on Active Replication".

[8]. Y.Xinfeng, "Providing Reliable Web Services through Active Replication",Department of Computer Science, Auckland University, New Zealand.