Comparative Study on Active and Passive Replication

Karanjit Singh Tuwana
Computer Science Department
Christ University,
Bangalore

Geetha B K
Computer Science Department
Christ University,
Bangalore

Shreesha Bhaskar
Computer Science Department
Christ University,
Bangalore

Rakshita G S
Computer Science Department
Christ University,
Bangalore

P Beaulah Soundarabai
Associate Professor,
Computer Science Department,
Christ University, Bangalore

***Abstract*--Constant availability of data is very important. Storing all data in a single database is risky and very inefficient. So to make work faster and achieve tasks as quickly as possible we require multiple databases. It is imperative that we keep all of these databases up-to-date, with consistent data. The process of keeping all database up-to-date is known as replication. There are 2 types of replication active and passive, each approaches has its own advantages and disadvantages. With this paper we will compare commonly used algorithms, and specify in which scenario each approach should be used.**

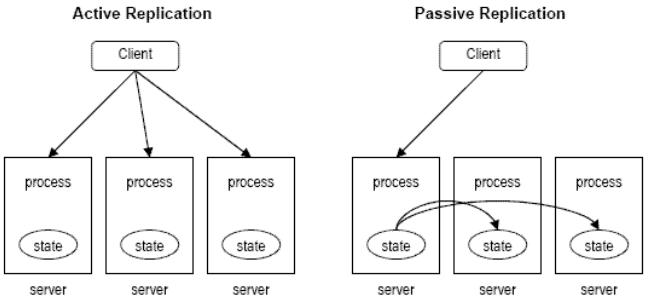
INTRODUCTION

With introduction of devices that can connect to the internet, the amount of data being produced is huge, and to manage this data effectively we need new techniques and technologies. In the distributed systems replication is mainly used to provide fault tolerance, and availability.

Availability: Availability refers to the fact that data is always present and available wherever required by the user. This is achieved by keeping up-to date copies of the data in multiple servers.

Fault Transparency or Tolerance:Errors might occur in our system. A system might fail this creating problems(atleast in a normal network). Replication helps to retain the work or data being done on a system so any error that occurs will not affect the performance of the system as a whole. In this paper we will talk about various algorithms (both passive and active). Compare them and present our findings.

Type of replication:



Active Replication:

Leslie Lamport introduced Active Replication under the name **state machine replication.** Here each client request is processed by all the servers.Active replication is **Deterministic**- The request hosted by the servers is given in same initial state and a request sequence, so that all processes will produce the same response sequence and end up in the same final state. An automatic broadcast protocol must be implemented in order to make all the servers receive the same response of operation which guarantees that either all the servers receive a message in the same order or none.

Passive Replication:

Here only one server i.e. primary server that processes request from client and it keeps the backup by updating the state on the other servers and sends back the response to the client. Whenever primary ever fails, one of the backup server takes its place. Passive replication may be used for

non-deterministic.

Passive replication is also called as primary back up. In which it plays a main role that receives the requests

from the clients and returns to it. This replication technique is quite useful than active replication since it requires less processing power and makes no assumption on the determinism of processing a request.

Passive replica is also called as primary backup. In which that plays a major role in which that receives the requests from the client and return backs to it. This replica is quite useful than the active replica, Since it requires less processing power and makes no assumption on processing of a request.

Passive replica needs to agree on the primary. If primary fails , one of the backups takes places If primary crashes before sending a response to client, the client will be time-out. Then the replica should know that the identity of new primary and can be reuse the request. This leads to increased response time in case of failure which makes unsuitable in the context of time-critical applications .However , the passive replica don't totally mask the failures to the client side.

Passive Replication, Membership And Cost A membership is a useful abstraction that as a basic component of communication systems e.g. Consul, Transies in which it is based on the asynchronous system model. It is used to maintain the list of members in the system. In which the service handles the explicit process requests to join a group and leave a group. The service also removes that are suspected to be crashed. .This point arises a question that what time-out value should be manage to exclude members?

- Managing an aggressive time-out value is expensive
- Managing a conservative time-out value is also a very bad idea

Working of active replication:

1. Request: The front end attaches a unique identifier to the request and multicasts it to the group of replica managers, using a totally ordered, reliable multicast primitive. The front end is assumed to fail by crashing at worst. 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. Since they are state machines and since requests are delivered int he same total order, correct replica managers all process the request identically. The response contains the client's unique request identifier.
4. Agreement: No agreement phase is needed, because of the multicast delivery semantics.
5. Response: Each replica manager sends it response to the front end. The number of replies that the front end collects depends upon the failure assumptions and on the multicast algorithm. If, for example, the goal is to tolerate only crash failures and the multicast satisfies uniform agreement and ordering properties, then the front end passes the first response to arrive back to the client and discards the rest.

Active replication

ADR ALGORITHM

The algorithm uses 2 variables req and id. The algorithm has 8 methods/functions.

1. ADR_r
2. ADR_w
3. ADR_t
4. Expansion
5. Switch
6. Contraction
7. ADR_x
8. ADR_j
9. ADR_s

ADR_r:

This procedure is called when a read request is issued locally, or message from a neighbor is received indication that the neighbor wants to read the object (in this case the Id will be neighbors Id)

If I have the replica of the object then

Retrieve the object from the local database, send the object to processor id

Else

Find the neighbor NB, submit the read request to NB and wait for the reply; after getting the object, send it to processor Id

ADR_w:

If I have a replica of the object

Update the local replica

Else

Find the neighbor NB, such that NB has the id 'ID', update the replica of NB

ADR_t:

If I am in the Singleton replication scheme

Namely, I have a local replica of the object for all neighbors

If expansion() == 0

Switch()

Else if I have local replication of atleast one neigbour

I am an R_fringe processor if and only if I have a replica of the object for exactly one neighbor

End if

Expansion()

This procedure returns 1 if the expansion test succeeds, namely at least one of the neighbor joins replication scheme in the procedure call otherwise it returns 0

Switch()

This procedure return 1 if the switch test succeeds, namely the singleton replication scheme switches to a neighbor otherwise it returns 0

Contraction()

This procedure is called by an R_fringe processor. Observe that a pair of processors which constitutes the whole replication scheme may call this procedure at the same time for the contraction. Thus, special care needs to be taken to prevent the contraction of both.

ADR_x

This procedure is called when an exit message is received from an R_fringe neighbor id. A process q will grant such request.

If q is not executing then contraction test at the same time

Send message to the neighbor Id saying 'Yes you may exit'

ADR_j(ID)

This procedure is called when a 'join' message is received from neighbor ID

Save the local object in the local database

ADR_s(ID)

This procedure is called when a join as singleton scheme message is received from neighbor ID saves the local object in the local database.

Three Tire active replication

A three-tier architecture is a context for active replication which has 3 components

1] Clients (the client-tier)

2] Middle tier(the mid-tier)

3] Server (end-tier)

Passive Replication

Semi passive Algorithm

- In passive replica technique primary manages the request, and after processing of each request by the client.
- After processing of each request, sends an update message to the backups.
- Every message decides on the update message, so this needs an initial value for the problem to be updated a value and tells the require of problem.
- Every problems tells or decides on the content of the content of updated messages
- However, only primary canhandle clients requests and response to it.
- There will be no problem till the primary crashes, When the primary crashes then one of the backups need to handle the request to obtain an initial value for every ones number.

Semi passive Algorithm(client-server based)

This algorithm for semi passive replication is based on lazy consensus problem. Every server will be executed by this algorithm and these server i.e., denoted by s will handle the integer k and the variables *recvQ*and *hand*

recvQ- It is the sequence of received requests. Initially it will be empty.

Hand- It is the set of handled request.

Whenever the server gets new request from the client it will be collected to *recvQ*. Whenever a server process calls the function *handlerequest*it will act as primary server in semi passive technique. In this function it will select the client request and handle that request. It will return the selected request and updated message which is being handled and also the response message. When the received request recvQ is not empty, the integer k will get incremented. At the end the request that have been processed will be removed from the recvQ and will be put in to hand. The server will wait until the value is decided. The decision values are (req,updreq,respreq). *req*is the value which is already handled. Updreq is the updated request from the handling request. Respreq is the request that should be sent back to the client. The status of the server will be updated according to the update message *updreq*.The request that has been handles will be appended to the hand.

CONCLUSION

Replication has many important implication such as fault tolerance, availability, performance, etc. It is important to understand the types of replication and how each of them work and the application of each of these types. During our research we have found that passive replication is used where performance is not an issue and redundancy needs to be reduced while active replication is where redundancy is not an issue and, you wish to utilize resources to the maximum. To conclude, there is no better or worse approach, each approach has its own advantages and dis-advantages, it all depends on the area of usage.

REFERENCES

[1] University of Illianos, Chicago and NASA/CESDIS Goddard space flight Center, Sushil Jajodia, George Mason. University of Illianos.
[2] ORDER: A Dynamic Replication Algorithm forPeriodic Transactions in Distributed Real-Time Databases Speci_cation of Replication Techniques,
[3] Semi-Passive Replication, and Lazy Consensus. School of Knowledge ScienceJapan Advanced Institute of Science and Technology
[4] Semi-Passive Replication_XavierD´efagoAndr´eSchiper Nicole Sergent D´epartementd'InformatiqueE´colePolytechniqueFe´de´rale de Lausanne, Switzerland Sushil Jajodia and David Mutchler. Dynamic voting. In Proc. ACM SIGMOD 1987.
[5] Michael L. Kazar. Synchronization and caching issues in the Andrew_le system. In Winter Conference Proceedings, 1988.
[6] Sushil Jajodia and David Mutchler. Dynamic voting. In Proc. ACM SIGMOD 1987 Annual Conference, pages 227{238. ACM SIGMOD, May 1987.
[7] Asynchronous active replication in three-tier distributed systems