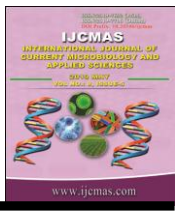
Int.J.Curr.Microbiol.App.Sci (2019) *8*(5): 446-451

International Journal of Current Microbiology and Applied Sciences
***ISSN: 2319-7706* Volume 8 Number 05 (2019)**
Journal homepage: http://www.ijcmas.com



Review Article https://doi.org/10.20546/ijcmas.2019.805.052

Replication in Distributed Systems and its Improvements

Tushar Kumar Pandey[1*], Ishita Singh[2] and Manoj Kumar[3]

[1]ARIS Cell, RPCAU, Pusa, India
[2]VC Cell, RPCAU, Pusa, India
[3]DoEE, RPCAU, Pusa, India

**Corresponding author*

Keywords

Update Propagation, Active replication, Passive replication

Article Info

Accepted:
07 April 2019
Available Online:
10 May 2019

A B S T R A C T

In distributed system data replication is needed in order to improve data availability and performance among replica and those replicas need to be periodically refreshed using some update propagation strategies because the data store may be physically distributed across multiple machines at multiple sites. Data availability means to provide data to requesting node. The key issue is to decide where, when and by whom replicas should be placed and mechanisms to use for keeping the replicas consistent. Replication is a way to enhance the performance of application that accesses the data. It is an area in technology which is of great interest to both distributed systems and databases. A system where data is replicated ensures consistency through sharing of information among redundant resources such as software and hardware. The solution developed from replication helps to improve accessibility, fault tolerance and accessibility. This paper introduces distributed systems giving its history and goes ahead to explain its benefits. It also describes the replication techniques and gives the comparison among them. The paper concludes with various models of replication which can be further improved.

Introduction

Distributed system is a collection of sites which are connected together via some kind (If communication network in which every sites has its own database system and users can access data anywhere in the network from any site, so there is a need to make data available at every site. Data availability is a significant factor in distributed system and it can be achieved by replication of data. Replication is a process of keeping copies of data at various locations for increasing data availability and performance. Once a system is established in distributed environment it is necessary to update it regularly so that freshness of data can be achieved (1).

The pace at which computer systems change continues to be overwhelming by the day. Since the inception of modern computers in 1945, until about 1985, computers were very huge and costly. Additionally, there was no way of connecting them, therefore the computers operated independently.

In the mid-1980s, there was a major advancement in technology which began to

change the narrative (2). The result of this technology advancement brought about a system where the computers have been able to be networked together. These machines are in reality physically detached; this is why they are said to form a distributed system. Innumerable meanings of distributed systems have been provided by researchers in literature, none of them has been adequate enough, and none of them in agreement with any of the others.

Some researchers say that a distributed system is a collection of separate computing elements that appears to its users as a single clear system (2).

This definition brings up two distinct features of distributed systems. The first one is that a distributed system is a collection of computing elements each being able to operate free of each other. The other feature is the abstraction in users believing they are dealing with a single system.

Studies have explained two benefits of the distributed systems (3). These are scalability and redundancy. Scalability, according to various researchers is the ability to add components into a system, redundancy, on the other hand, is the ability to reproduce the same results over time.

In order to build a fault-tolerant service in the distributed systems, data had to be into various servers. This process has been defined by researchers as replication.

Replication is an aspect of the distributed systems where data is copied at multiple locations (3). This copying of data ensures that this data is always available; secondly, it provides fault tolerance. This means that in case one of the machines fails, the others are still able to operate efficiently. When building a fault-tolerant service with replicated data and functionality the service should produce the same results as a non-replicated service and should also respond despite node crash.

Replication in databases and distributed systems depend on different norms and offer different assurances to the clients. There are two types of replication strategies; these are active and passive replication.

Active replication also referred to as the state machine method is a non-centralized replication procedure (4). The main notion of active replication is that all replicas receive and process the same order of client requests. This, therefore, ensures that consistency is certain.

Clients contact the servers not as one particular server, but as a group. In order for the servers to receive the same input in the same order, client requests are broadcasted to servers using an Atomic Broadcast. In cases where the semantic info about the operation is known, feebler communication primitives can as well be used. The main advantage of this type of replication is its simplicity that is the same code everywhere and transparency. The other advantage is that failures are fully concealed from the clients. This is so because if a replica fails, the requests are still processed by the other available replicas. Also in this type of replication, there is no need to change the existing servers. However, one of the arguable setbacks for this method is the consumption of many resources. This is because all the processing is done on all the replicas. This method also requires deterministic execution.

The following steps are involved in the processing of an update request in the Active Replication, the client sends the request to the servers using an Atomic Broadcast, replicas execute the request in the order they are delivered, no coordination is necessary since

all replica process the same request in the same order. Replica is deterministic; hence they all produce the same results. Finally, all replicas send back their result to the client, and the client only waits for the first answer ignoring all the others.

In passive replication, only one server exists. This is referred to as the primary (5). This server processes client requests. After processing a request, it updates the state on the other servers which act as a backup and sends back the response to the client. In case the primary server crashes, one of the backup servers automatically replaces it to ensure continuity. This type of replication can also be used for non- deterministic processes.

The pros of passive replication are that all operations pass through a primary that linearizes operations. The other advantage is that it works even if the execution is non-deterministic. One of the setbacks for passive replication in comparison to active is that in an instance of a failure there will be late response. The other setback is that delivering state change can be costly.

Both active and passive replication schemes require that servers are available. In case the server crashes, it will take some time to spot and take out the broken node. This is because they both depend on network

Typically, the steps in passive replication are as follows; the client sends the request to the primary. There is no initial coordination, the primary executes the request and then coordinates with the other replicas by sending the update information to the backups. Finally, the primary sends the answer to the client.

DFS is implemented on a cooperating set of server computer connected by a communication network, which together create the illusion of a single, logical system for the purpose of creation, deletion, and random accessing of data. In DFS, generally more than one server stores the data and is accessed over the network. Major Challenges that needs to be addressed by distributed file system are concurrency and partial failures. Problem of concurrency is due to multiple nodes that execute in parallel makes it difficult in keeping the data consistent. Partial failures comprises of node failures and network congestion. Passive replication can lead to data inconsistency if handled properly. Other problems are transparency, reliability, flexibility, scalability, security, performance, fault tolerance.

Transparency

Transparency hides the interaction between the client machine and server machines (6) and also does not provide the user regarding the details of what processing is being carried out in backend.

Reliability

Reliability is availability of data at any time when and where it is required without any changes to the original data. Distributed file system allows multi-processes that are pre-ferred by users because this can protect against single processor system crashes. If in case failure happens, the replicated copies can be used. if we use replicated copy, the copy present must be consistent with its content. Thus even on a failure, a backup of contents are available (7).

Flexibility

Flexibility can be achieved by using a monolithic kernel or microkernel on each machine. Kernels are used in order to manage activities like process management, resource management, memory management.

Monolithic kernel provides functionality of kernel by not considering whether all machine use it or not, the approach used is (kernel does it all). Micro-kernels are used to provide proper accessibility to other services as and when needed using minimalist modular approach (8-10).

Scalability

Scalability in which resources get added or removed in a network at any time. More clients are connected with the system day to day. An efficient system is needed in handling those users or client. There can be many CPU's get added to the distributed system. While designing, DFS can be made in two ways: one is centralized and the other is decentralized architecture. Centralized architecture requires more administration while scaling up in the distributed file system. Decentralized architecture scaling can be managed by an administrator itself (11, 12).

Security

The main three key aspects in order to achieve are: confidentiality, integrity and availability. Confidentiality uses authentication techniques to protect our system from unauthorized users. Integrity protects our system against data corruption by using message digest to identify data corruption. Availability ensures system to be accessible always and protects system against failure (13).

Performance

Performance metrics can be measured using response time, throughput, system utilization and amount of network capacity used. Response time is time of system or a functional unit to respond to the given input or request. Throughput is successful message delivery with respect to time measured out in (bps) bits per second. System utilization reports server OS configuration and its associated utilization information. Network capacity is maximum capacity of network path to convey data from one network location to another (14). Applications that run should be presented as if it running on a single processor. In LAN message transmission takes over milliseconds. Performance can be increased by reducing the number of message transmitted (15).

Fault tolerance

Fault tolerance has to be provided when a failure occurs (can be a hardware or software) in a distributed file system. It must be provided with fault-tolerant capability such that the system should be able to recover and tolerate faults that occur. Consider if a system has multiple servers, if any one fails, then the load has to be distributed among the servers in transparent way (16).

Proposed work

There are various models of data replication studied by various researchers. Each of these has its own properties and performance:

Transactional replication

This is the model for replicating transactional data. An example of this would be a database. A database is storage for data of in various forms. Transactional means that there are a lot of queries sent to the database. The data is frequently accessed by passing various queries. In this case, every transaction that is passed has to be replicated (3).

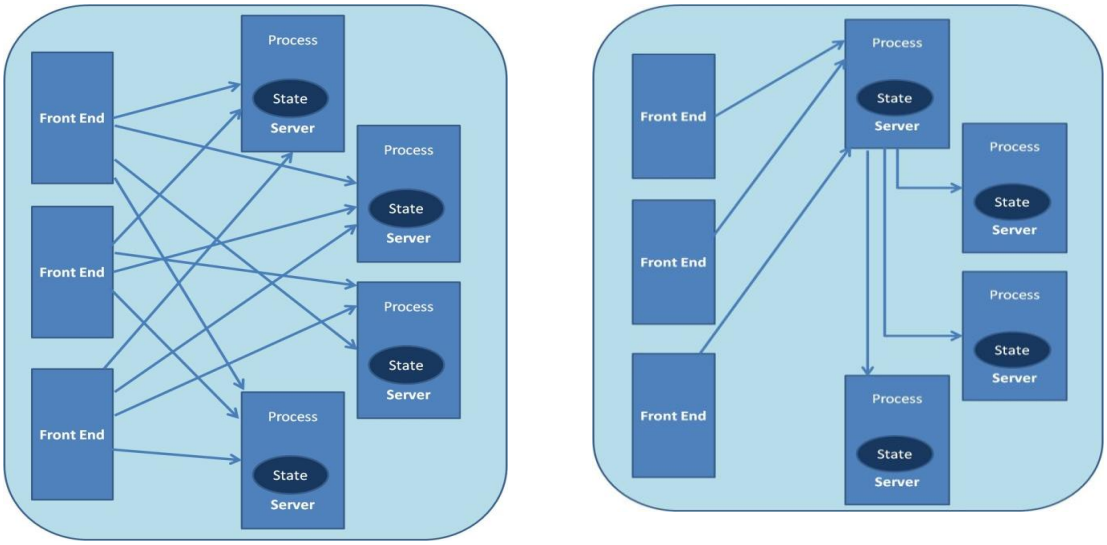
The one-copy serializability model is employed in this case, which defines legal outcomes of a transaction on replicated data in accordance with the overall properties that transactional systems seek to guarantee.

The problem with transactional replication is that sometimes the distribution agents fail with an error message "The row was not found at the Subscriber" when applying the replicated command or due to the Violation of PRIMARY KEY constraint i.e. "Cannot insert duplicate".

State machine replication

In this model, it is assumed that a replicated process is a deterministic finite automaton and that atomic broadcast of every event is possible. State machine replication is based on distributed consensus which is a distributed computing problem and has a great deal in common with the transactional replication model. This is sometimes erroneously used as a synonym for active replication. State machine replication is sometimes enforced by a replicated log consisting of multiple later rounds of the Paxos algorithmic program (Fig. 1).

Fig.1



Virtual synchrony

This computational model is used when a group of processes collaborates to replicate in-memory data or to coordinate actions. The model describes a distributed object called a process group. A process can join a group and is provided with a checkpoint comprising the current state of the data replicated by group members. Processes can then send multicasts to the group and will see inbound multicasts in the identical order. Membership changes are held as a special multicast that brings a new membership view to the processes in the group. The Limitation of virtual synchrony is the split-brain condition where the processes that belong to the same group get partitioned into sub-groups within each partition. In this condition it becomes impossible to guarantee the system consistency.

In conclusion, replication in distributed systems is still an area of active research. This means that many of the above discussed issues can be resolved in the near future. In summary, this paper presented forms of replication in a distributed system and their properties. None of them can be conclusively said to be better than the other. The findings of this research and the results that exist today hold the promise of a more improved replication system in the future.

References

1. Coulouris, George F., Jean Dollimore, and Tim Kindberg. *Distributed systems: concepts and design pearson education, 2015.*
2. Babaoglu, Ozalp, and Keith Marzullo. *"Consistent global states of distributed systems: Fundamental concepts and mechanisms." Distributed Systems 53 (2013).*
3. Guerraoui, Rachid, and André Schiper. *"Fault-tolerance by replication in distributed systems." In International Conference on Reliable Software Technologies, pp. 38-57.Springer, Berlin, Heidelberg, 2016.*
4. Tanenbaum, Andrew S., and Maarten Van Steen. *Distributed systems: principles and paradigms.Prentice-Hall, 2017.*
5. Coulouris, George F., Jean Dollimore, and Tim Kindberg. *Distributed systems: concepts and design. Pearson education, 2015.*
6. Zhang, T., Sun XZ, Xue W, Qiao N, Huang H, Shu J, Zheng W. ParSA*: High-throughput scientific data analysis framework with distributed file system. Future Generation Computer Systems. 2015; 51: 111–19.*
7. Hac, A., *A validated performance model for distributed file systems. Journal of Systems and Software. 1989; 10(3): 169–85.*
8. Lanjewar, U., Naik M, Tewari R. Glamor: *An architecture for file system federation. IBM Journal of Research and Development. 2008; 4(5): 329–39.*
9. Carretero, J., Perez F, de Miguel F, Alonso GL. *Performance increase mechanisms for parallel and distributed file systems. Parallel Computing. 1997; 23(4,5): 525–42.*
10. Hac, A., *A benchmark for performance evaluation of a distributed file system. Journal of Systems and Software. 1989; 4(9): 273–85.*
11. Cho, JY., Jin HW, Lee M, Schwan K. *Dynamic core affinity for high-performance file upload on Hadoop Distributed File System. Parallel Computing. 2014; 40(10): 722–37.*
12. Liu, T-J., Chung C-Y, Lee C-L. *A high performance and low cost distributed file system. Beijing: 2011 Proceedings of IEEE 2nd International Conference Software Engineering and Service Science (ICSESS). 2011; p. 47–50.*
13. Hung-Chang, H., Hsueh-Yi C, Haiying S, Yu-Chang C. *Load Rebalancing for Distributed File Systems in Clouds. IEEE Transactions on Parallel and Distributed Systems. 2013; 24(5): 951–62.*
14. Perez, F., Carretero J, Garcia F, de Miguel F, Alonso L. EvalutingParFiSys: *A high-performance parallel and dis-tributed file system. Journal of Systems Architecture. 1997; 43(8): 533–42.*
15. Zhou, X., He, L. A Virtualized Hybrid Distributed File System. Beijing: *Proceedings of International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberCC). 2013; p. 202–05.*
16. Lee, W., Su, D, Srivastava J. *QoS-based evaluation of file systems and distributed system services for continuous media provisioning. Information and Software Technology. 2000; 42(15): 1021–35.*

How to cite this article:

Tushar Kumar Pandey, Ishita Singh and Manoj Kumar. 2019. Replication in Distributed Systems and its Improvements. *Int.J.Curr.Microbiol.App.Sci.* 8(05): 446-451.
doi: https://doi.org/10.20546/ijcmas.2019.805.052