Iqbal *et al.LGU (IJECI) 2020*

LGU (IJECI)
ISSN: 2522-3429 (Print)
ISSN: 2616-6003 (Online)

LGU International Journal for
Electronic Crime Investigation

Research Article

Vol. 5 issue 3 Year 2021

Communication Mechanism in a Distributed System

Zain Ali, Sehar Afzal
University of Lahore
Zainali114@gmail.com
seharAfzal43@gmail.com

Abstract:

In this research, problems are discussed dynamically distributed systems that relate to the sharing of data and communication from one system to another over the network. A distributed system communicates with its related systems by sending and receiving messages over the internet and in this way, it fulfills its work. When we discuss dynamic distributed systems, it means that it includes many different changeable types of networks, different operating systems like android, mac, windows, different software processors portability, breaking down of WAN, and inter-process communication errors. Another problem that accrues in distributed systems is latency. So, it is very difficult to develop software for these types of environments. Proposed work is related to make message communication in distributed systems easy, reliable, and efficient. For the sharing of data, coherence is responsible. Every problem can be solved but that proper

appropriate methods and algorithms are required. We create a new method which is a dynamic atomic shared memory for message communication. A properly stated method is proposed for message communication and then implemented. According to this method, owners can be changed dynamically and their access to read and write also changes.

1. Introduction

In this research, problems are discussed dynamically distributed systems that relate to the sharing of data and communication from one system to another over the network. A distributed system communicates with its related systems by sending and receiving messages over the internet and in this way, it fulfills its work. When we discuss dynamic distributed systems then it means that it includes many different changeable types of networks, different operating systems like android, mac, windows, different software processors portability, breaking down of WAN, and inter-process communication errors. The communication mechanism in distributed systems is related to how the system sends and receives messages from one system to another over the internet. In distributed systems, multiple systems relate to each other's, so a specific and well-defined method is required

for efficient and accurate communication. To fulfill this purpose, we developed a method which insured the accuracy and reliability of communication in processes. The focus of this work is to gain favorable results and assuring of communication and it helps to develop application easier. The reliability provides a guarantee about communication and coherence provides a guarantee about the sharing of data. Algorithms results are checked at lower bound to explain the limitations and cost of problems that can be faced. We create a new method which is a dynamic atomic shared memory for message communication. A properly stated method is proposed for message communication and then implemented. According to this method, owners can be changed dynamically and their access to read and write also changed. In this proposed work, we provide the best solution for implementation for communications in dynamic distributed systems that avoid crashes. Its performance in evaluates timing and failure of communication.

2. Literature Review

In the past, work is done for the inquiry of a lot of complications, errors and duplication of data on distributed algorithms [6,7], and group membership and communication mechanisms are checked [5,8]. We also focused on the previously implemented studies [3, 13]. A lot of mechanisms are used in different academic and commercial systems for communication in processes. They act as a middleware for the process to process communication in distributed systems. Mostly they used in group communication distributed systems [4]. An inspirational work which helps to develop this work is atomic data from dynamic voting systems [15] protocols. This system can manage a lot of processors in distributed systems [14]. To handle the problems just like inconsistency in the breakdown of communications in processes, and some are used to demand the systems for configuration. It is noticed that it is very useful for problem detection and generates solutions for these problems. From recent studies and it is noticed that algorithms are very complex which are developed to detect the problems and errors in distributed systems. Such algorithms help a lot to decrease the complexity by using global services with good interface and practices. An example of this conscious algorithm which provides a baseline for other related works [9]. To check the performance and efficiency of this algorithm, decomposition is used.

3. Problem Statement:

Dynamic Distributed Systems

For data sharing and communication problems in dynamic distributed systems, we are focusing new direction. The environment will be less interacting for example an unlimited number of processors, Request from the user to join and leave the system. Our purpose is for coherent theory and we see service for the lower bound and upper bound algorithm results. With the passage of time, the number of processes and their connections is changed over a network. Processors can be added, recover if they fail in a network. Processors can be connected via mobile and wirelessly. An application may move from one place to another, on these conditions we will consider distributed running application which has identification and information of users include file, multimedia, real-world information, and

games.

4. Approach

As the high-level global services are solved, communication and data sharing problems are solved in the same way. According to the environmental performance expectation, these

services maintain the problems and fault tolerance.

Research on distributed services focuses on correctness, On the other hand, Algorithm focuses on performance. Our work will combine these two techniques which will produce algorithms that work efficiently will break down in dynamic distributed systems then performance and fault tolerance squeeze out by global services. We will include a balanced study of performance and service assurance. Atomicity is expensive so resolve consistency which may reduce the cost and will provide benefits.

The setting of these techniques is difficult, so it is also difficult to develop these algorithms which means that we have to break down bigger pieces into smaller pieces. These pieces will be viewed as lower-level global services. These services will provide data sharing and lower-level communication for example resource allocation, routing, failure, and progress detection. These services also include fault tolerance and performance which can be repeated again and again. The work we focus to achieve goals are given below:

-State new services focus on communication and data sharing in a Distributed environment

- Developing and analyzing an algorithm in a dynamic system to implement these services

This work is achieved by the mathematical framework based on state machines which include the feature to convey timing issues, behavior, and probabilistic behavior. Give assistance to Meta theory include models, measure performance and analysis proof methods will also be developed. Our theoretical work contributes features to the implementation and testing of distributed systems services. This work is conducted by examples that are selected from file management, application prototype, collected information, games, and computer cooperative work. During developing the specification for the system, we also focus on the developer's opinion and information for service assurance.

5. Atomic Memory Service

Reconfigurable memory service on the algorithm for a distributed system that can be used to reading and writing memory in a dynamic network [12]. Users can join and leave during the action of mathematical calculation. Examples are mobile and peer computing networks. The benefit of this service is that data can survive for a long time in a dynamic setting.

We identify and introduce atomic memory serves as a global service called RAMBO means Reconfigurable atomic memory for basic objects. Dynamic distributed algorithms implement this global service. To obtain presence objects are reproduce and also to

obtain repetition in the availability of small changes algorithm use configuration which includes read and write sets of belonging struc-

ture. To provide large and small changes algorithms use reconfiguration in which members are updated. These types of updates do not use any infraction and objects configuration can install any time.

The algorithm includes major actions like reading, writing, configuration, and out of date configuration. Algorithm merge in the main algorithm which handles garbage collection and global reconfiguration services. Reconnoiter provide the main algorithm to repeat configuration. Reconfiguration does not fir tightly in the main algorithm. The major configuration may be used one time but read and write use them all the time.

The main algorithm performs read and write operations. Information collected from reading operations and spread information from writing operations. Both operation use for active configuration. This communication put into the background which allows the algorithm to maintain information. Every stage is finished by a condition that includes objects from the configuration. Read and write actions may run simultaneously. Garbage collection is used when there is no type of configuration used for repetition.

Reconfiguration service is executed by a distributed algorithm that involves a general agreement to configuration. An object from a new configuration may introduce a new configuration, many invitations are consistent by running general agreement among objects. General agreement executed by the Paxos algorithm [11]. That type of general agreement is slow but in some condition may not be finished but they do not read and write action slowly.

Garbage collection uses two stages in which the first stage communicates with old configuration and the second stage communicates with the new configuration. Garbage collection action makes surety that objects of reading and writing old configuration learn new configuration.

We evaluate performance based on time and failure action means garbage collection occurs from time to time, reconfiguration is requested for garbage collection to keep up, objects of active configuration do not fail, then we show that read and write action perform in maximum latency time.

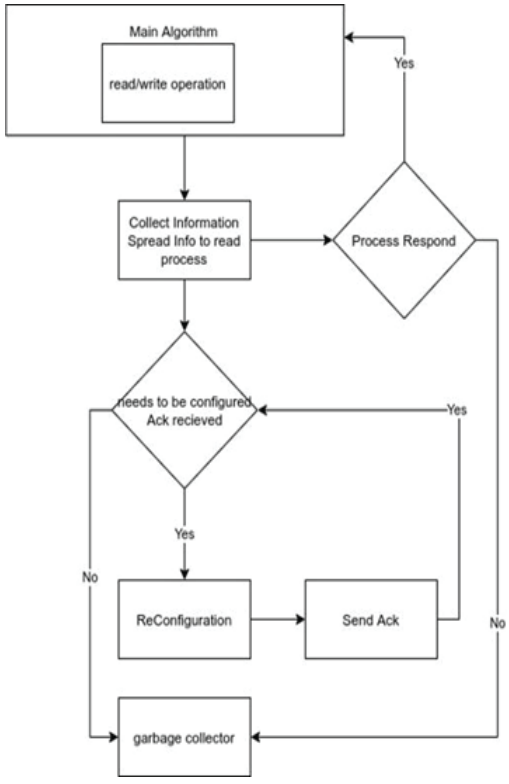

Figure 1

Our proposed method is to overcome problems found in the previous literature that is PAXOS algorithm used for the process to process

communication and when the failure of one process occurs it disturbs Communication. Global service called RAMBO Reconfigurable atomic memory for basic objects. Dynamic distributed algorithms implement this global service. Our method includes major action like reading writing, configuration, and reconfigure

out of date configuration as we can see in figure 1.The first step includes the Algorithm merge in the main algorithm. The main Algorithm handles garbage collection and the global reconfiguration services in the main algorithm perform read and write operation. Information collected from reading operations and spread information to writing operations. Both operation use for active configuration. This communication put into the background which allows the algorithm to maintain information. Every stage is finished by a condition that includes objects from the configuration. Read and write actions may run simultaneously. Garbage collection is used when there is no type of configuration used for repetition.

6. Conclusion

Our proposed work is different from other works because it acts as a middleware in communication in distributed systems. There are a lot of different frameworks such as CORBA, DCE, and Java/JIN which are used to develop different distributed systems. But they have small scope according to their components, their specification, and architecture according to their formal definition and informally from their behavior. By using these methods and services, the performance of dynamic distributed systems cannot be supported by these frameworks. On the other hand, our proposed work is very helpful to provide initial help to handle the faults and errors during communications. This proposed work is best because its interface and behavior are accurately elaborated. The performance and fault tolerance for communication in distributed systems are mentioned in the behavior of an algorithm. The reliability, accuracy, error detection and handling, and performance can be handled by our proposed method because it handles it during its computation.

We are assuring that our proposed work will be very helpful in the theory of algorithms construction. It will provide the analysis for complications accrue during communication in dynamic distributed systems. It will provide as much strong communication and accuracy as available in static distributed systems. This project is based on the theoretical. To develop the dynamic distributed systems, our algorithm and framework are very strong to ensure the accuracy and performance of systems. Some additional work is required just like ours into systems for some other purpose. It is as same as components that are created by the help of object-oriented and component technologies in software engineering [2].

7. References:

[1] H. Attiya, A. Bar-Noy and D. Dolev, “Sharing Memory Robustly in Message-Passing Systems", J. of the ACM, vol. 42, no. 1, pp. 124-142, 1996.

[2] Kenneth P. Birman. A review of experiences with reliable multicast. Software, Practice, and Experience, 29(9):741–774, September 1999.

[3] O. Chenier and A. Shvartsman, “Implementing an eventually-serializable data service as a distributed system building block,” in Networks in Distributed Computing, vol. 45, pp. 43–72, AMS.

[4] Communications of the ACM, special section on group communications, vol. 39, no. 4, 1996.

[5] R. De Prisco, A. Fekete, N. Lynch, and A. Shvartsman, “A dynamic primary configuration group communication service,” in Distributed Computing Proceedings of DISC’99

- 13th International Symposium on Distributed Computing, 1999, LNCS, vol. 1693, pp. 64–78.

[6] B. Englert and A. Shvartsman. Graceful quorum reconfiguration in a robust emulation of shared memory. In Proc. of the 20th IEEE Intl Conference on Distributed Computing Systems (ICDCS’2000), pp. 454-463, 2000.

[7] A. Fekete, D. Gupta, V. Luchangco, N. Lynch, and A. Shvartsman, “Eventually-serializable data service,” Theoretical Computer Science, vol. 220, no. 1, pp. 113–156, June 1999.

[8] A. Fekete, N. Lynch, and A. Shvartsman. “Specifying and using a partitionable group communication service." ACM Trans. on Computer Systems. vol. 19, no. 2, pp. 171-216, 2001.

[9] R. Guerraoui and A. Schiper, “The Generic Consensus Service," IEEE Trans. on Software Engineering, Vol. 27, No. 1, pp. 29-41, January 2001.

[10] S. Haldar and P. Vita´nyi, “Bounded Concurrent Timestamp Systems Using Vector Clocks", J. of the ACM, Vol. 249, No. 1, pp. 101-126, January 2002

[11] Leslie Lamport, "The Part-Time Parliament", ACM Transactions on Computer Systems, 16(2) 133-169, 1998.

[12] N. Lynch and A. Shvartsman, “RAMBO: A Reconfigurable Atomic Memory Service", in Proc. of 16th Int-l Symposium on Distributed Computing, DISC’2002, pp. 173-190, 2002.

[13] K. W. Ingols, “Availability study of dynamic voting algorithms,” M.S. thesis, Dept. of Electrical Engineering and Computer Science, MIT, May 2000.

[14] M. Merritt and G. Taubenfeld, “Computing with infinitely many processes (under assumptions on concurrency and participation),” In Proc.14th International Symposium on DIS- tributed Computing (DISC), October 2000.

[15] E. Yeger Lotem, I. Keidar, and D. Dolev. “Dynamic voting for consistent primary components." In Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing, pages 63–71, August 1997.