International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 04-10 www.ijaem.net ISSN: 2395-5252

Efficient Algorithms for Group Communication in a Distributed Computing Environment

Kpanuku chika-obi[1], Bennett, E. O[2], Igiri, C. G. [3]
Department of Computer Science, Rivers State University, Port Harcourt, Nigeria.

Date of Submission: 01-02-2023 Date of Acceptance: 10-02-2023

ABSTRACT
Group communication has been undoubtedly very useful paradigm for many distributed systems. Provision of a fast and scalable group communication mechanism can considerably improve the efficiency of these systems. The paper presents an Efficient Algorithm for Group Communication in a Distributed Computing Environment that guarantees scalability and speed. The algorithm has been designed to prevent double message delivery and to determine the Message Dispatcher Object (MDO) which they should send message to in each step of message propagation. A new group communication mechanism is proposed for large scale distributed objects over a huge heterogeneous inter-network. The mechanism uses Message Dispatcher Objects, objects co-located with each group member to distribute and parallelize the group communication load among all group members. The results show that our scheme scales well in a large number of group members in small interval of time given an average throughput of 38.9bps (bytes per seconds) with no significant loss of data

Keywords: Message Dispatch Object, group communication, networking

I. INTRODUCTION

Group communication is a communication, which takes place in groups. Within organizations, the individuals are required to work in groups, particularly when working on assignments, projects or preparing reports. The primary use of group communication is to share information. This information can vary from person to person, depending upon his role. Group communication can also be used to manage conflicts along with making decisions to overcome difficult circumstances. The exchange of ideas decides the future of the group and the goals which a group can achieve. Group cooperation is one of the most important actions in our human society. Without group cooperation, it is difficult to achieve any objective. It has been proven that cooperation among individual computers (peers) as a group are also really important in computer systems like database transactions, robot technologies and sensor-actuator networks (Corman, et. al, 2017).

Communication between two processes in a distributed system is required to exchange various data, such as code or a file, between the processes. When one source process tries to communicate with multiple processes at once, it is called Group Communication. A group is a collection of interconnected processes with abstraction. This abstraction is to hide the message passing so that the communication looks like a normal procedure call. Group communication also helps the processes from different hosts to work together and perform operations in a synchronized manner, therefore increases the overall performance of the system.

Among other environment of communication is Parallel computing. Parallel computing splits a task into sub-tasks that are executed at the same time, whereas distributed computing splits a task into sub-tasks that are executed at different locations, using different resources. Parallel computing typically refers to multiple processing elements existing within one machine with each processor being dedicated to the overall system at the same time (Leopold 2001). Distributed computing refers to a group of separate machines that each contributes processing cycles to the overall system, over a network, over time.

II. LITERATURE REVIEW

The task of a membership service is to maintain a list of the currently active and connected processes in a group. The output of the membership service is called a view. The reliable multicast services deliver messages to the current view

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 04-10 www.ijaem.net ISSN: 2395-5252

members. The first and best-known Group Communication Systems(GCS) was developed as part of the Isis toolkit (Birman, 2009); it was followed by over a dozen others. GCSs are powerful building blocks that facilitate the development of fault-tolerant distributed systems. Classical Group Communication Systems(GCS) applications include replication using a variant of the state machine/active replication approach (Montresor et al.,2013); primary-backup replication, for example, (Guerraoui and Schiper 2010); support for distributed and clustered operating systems distributed transactions and database replication system management (Amir et al.,2008) and monitoring (Al-Shaer et al.,2015); and highly available servers for example, (Mishra and Pang 2008; Fekete and Keidar 2001), and the video-on-demand servers. More recently, GCSs have been exploited for collaborative computing (Birman et al.,2009), for example, distance learning (Al-Shaer et al.,2015), drawing on a shared white board (Shamir 2016), video and audio conferences (Valenci 2008), application sharing and even distributed musical "jam sessions" over a network (Gang et al.,2019). Currently, real-time GCSs such as RTCAST (Abdelzaher et al.,2012) are being developed and are being exploited for real-time applications, for example, radar tracking, see (Johnson et al.,2010). Another emerging research direction focuses on the provision of object group services within the Common Object Request Broker Architecture (CORBA) framework, Furthermore, GCS has been recently identified as a key tool for supporting fault tolerance in CORBA: the new Fault-Tolerant CORBA specification (OMG 2000) recommends that view-oriented GCSs be used to support active object replication in CORBA.

Unifying the GCS properties

The guarantees of different GCSs are stated using different terminologies and modeling techniques, and the specifications vary greatly in their rigor. Moreover, many suggested specifications are complicated and difficult to understand, and some were shown to be ambiguous. This makes it difficult to analyze and compare the different systems. Furthermore, it is often unclear whether a given specification is necessary or sufficient for a certain application. We formulate a comprehensive set of specification "building blocks" which may be combined to represent the guarantees of most existing GCSs. In light of our properties, we survey and analyze over thirty published specifications which cover over a dozen leading GCSs the configurable service of (Amir and Stanton 2008), the study correlates the terminology used in different papers to our terminology. This yields a semantic comparison of the guarantees of existing systems. Another important benefit of our approach is that it allows reasoning about the properties of applications that exploit group communication.

Distributing computing environment

Distributed computing paradigms have been analysed by many researchers who described the various forms of distributed computing and provide definitions, characteristics and architecture of different computing environments which include cloud computing, jungle computing and fog computing.

Peer-to-Peer Computing: Peer- to Peer Computing is a decentralized network computing, where peers are connected through ad hoc networks and provide resources to the network. Peer-to-Peer computing mainly works on scalability issues of distributed computing. No master-slave relationship exists among the peers and all peers work as client and server as per requirement. Peer-to-Peer computing is used in file sharing, bioinformatics, artificial intelligence, and so on.

Cluster Computing: Cluster computing is a distributed computing system with a collection of interconnected stand-alone computers working together as single integrated computing resources. The key components of a cluster computing are personal computers, workstations, operating systems, fast communication protocol, services, and parallel programming environment. Selection of cluster depends on compatibility with cluster hardware and operating system, price, and performance which depends on bandwidth and latency.

Utility Computing: Utility computing is simple and based on the principle of centralization and sharing of physical resources. Utility computing provides the facility to users as per their demand and user pay for these facilities. Scalability, high availability, manageability, disaster recovery are some characteristics of utility computing. Utility computing provides the benefits of reduced hardware capital expenditure and operational cost. Utility computing solutions consist of virtual server, virtual storage, virtual software, backup and most IT solutions. Utility computing takes the benefits of grid computing, virtualization and provisioning technologies. On demand computing,

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 04-10 www.ijaem.net ISSN: 2395-5252

pay-per-use computing, pay-per-services computing are different types of utility computing.

Grid Computing: The main objective of grid computing is to share resources that are available from individuals and institutions in secure and flexible manner and to solve the problem of multi-institutional and virtual organizations. Geographical distribution and heterogeneity of resources, the large scale of infrastructure, resource coordination and sharing, multiple administration, autonomy, scalability, dynamicity are characteristics of grid computing.

Grid computing system is classified into various categories: Computational Grid, Distributed supercomputing grid, High-throughput grid, Scavenging grid, Data grid, Service grid, On-demand grid, Collaborative grid and Multimedia grid. The grid computing has a wide set of applications in engineering, life sciences, physical sciences, commercial areas, research and in other areas. Adaptive applications, real-time and on-demand applications, coordinated applications, poly-application, linked applications and minimal communication applications are the grid applications that use grid resources and grid infrastructure. Main challenges must be taken into consideration for effective implementation of application areas of grid computing: Resource management, Reliability, Security management, Data management, Service management.

III. SYSTEM DESIGN

A Tiering architectural design was used for the paper. The tiering architecture is a technique to organize functionality of a giving layer and place this functionality into appropriate servers and, as a secondary consideration, onto physical nodes. The two-tiered architecture refers to client/server architectures in which the user interface (presentation layer) runs on the client and the database (data layer) is stored on the server. The actual application logic can run on either the client or the server. Figure 1 show the system architecture on the system.

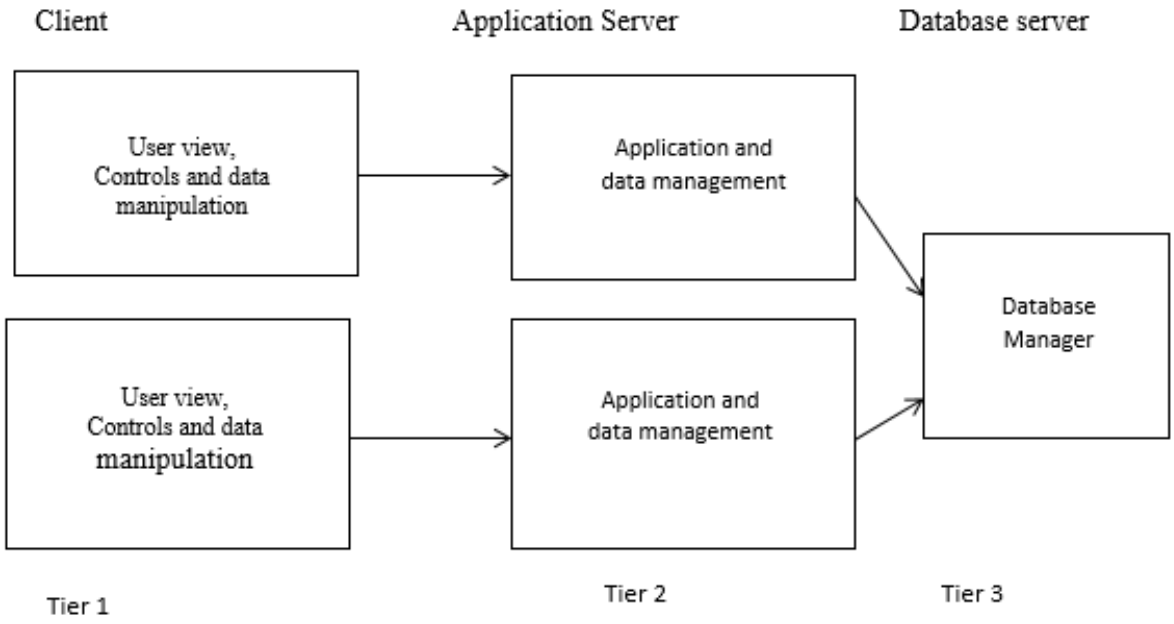

Figure 1: System Architecture

a) Tier 1: The user tier is the topmost level of the system. It is concerned with handling user interaction and updating the view of the application as presented to the user;

b) Tier 2: The logical tier is pulled out from the presentation tier and, as its own layer, it controls an application's functionality by performing detailed processing.

c) Tier 3: The data tier includes the data persistence mechanisms (database servers, file shares, etc.) and the data access layer that encapsulates the persistence mechanisms and exposes the data.

This paper adopted client-server architecture; the client-server architecture is the most common distributed system architecture which decomposes

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 04-10 www.ijaem.net ISSN: 2395-5252

the system into two major subsystems or logical processes.

Client: This is the first process that issues a request to the second process i.e., the server.

Server: This is the second process that receives the request, carries it out, and sends a reply to the client. In this architecture, the application is modelled as a set of services those are provided by servers and a set of clients that use these services. The servers need not to know about clients, but the clients must know the identity of servers.

Message propagation Among MDOs

Message dissemination among MDOs is done in a parallel manner and the propagation load is distributed among host with set boundaries in the system. Figure 2 shows the message propagation process among MDOs.

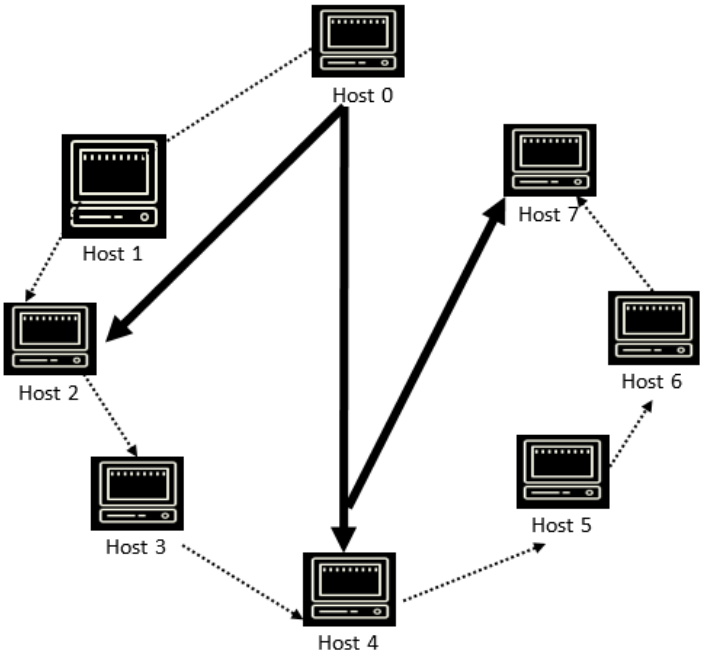

Figure 2: Message propagation among MDOs

The mechanism supposes there is one MDO on each host in the system. Each MDO knows all MDOs and their addresses. This information is stored in a list called MDO list. The index of each MDO is the same in all MDO lists in the system. The figure depicts an MDO list for a group with 8 members on 8 hosts. In such a system, there is one MDO object on each host and all of them have the depicted list. The first MDO in the list is called the proxy MDO. Each MDO also has a message storage queue that incoming messages are stored in. Using this queue, MDOs store a message for a limited period of time to ensure that all group members have received the message. If there are more than one group member on a host, its MDO knows all of them too. When a group member wants to send a message to the group it does not send the message to the members directly. Instead, it sends the message to one of the MDOs which is called the proxy MDO. Then, MDOs propagate the message among themselves. In the next step each MDO delivers the message to its local group member(s).

Algorithm 1: Message Propagation Algorithm

Group Communication Mechanism Process

This mechanism assumes that group members are on different hosts in a huge heterogeneous inter-network such as the Internet and there is a logical path between every two hosts. The communication channels between every two hosts are reliable and communications are done using application-level techniques such as remote method invocations.

Algorithm 2: Group Communication Mechanism Process

Algorithm 2 is a reliable multicast protocol that allows a group of processes to agree on a set of messages received by the group. Each message should be received by all members of the group or by none. The order of these messages may be important for some applications. All reliable

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 04-10 www.ijaem.net ISSN: 2395-5252

multicasts have the following three properties. **Validity**: If a correct process multicast a message m, then all correct processes eventually receive the message. **Agreement**: If a correct process receives a message m, then all correct processes eventually receive m. **Integrity**: For any message **m**, every correct process receives **m** at most once and only if **m** was multicast by the sender of m. In an asynchronous system.

Table 1: Concept and notations used

S/N	Concept	Symbols	Meaning
1.	Multicast	N/A	Multicast is group communication where data transmission
2.	Unicasting	N/A	is addressed to a group of destination computers
3.	Host	N/A	simultaneously.
4.	Peer-to-peer	P2P	unicast is a one-to-one transmission from one point in the
5.	applications	MDO	network to another point; that is, one sender and one
6.	Message	RD	receiver.
7.	Dispatcher Objects	TOD	This refers to any computer (server) that is interlinked with
9.	Reliable delivery	NSAPs	other machine(s) through an Internet connection.
			architecture that partitions tasks or workloads between peers.
			These are objects co-located with each group member, to
			route and deliver multicast messages to the group members

IV. RESULTS AND DISCUSSION

4.1 System Setup

For the sake of simplicity, we supposed that each host sends one message at each time. By some changes in the mechanism the method can exploit multithreading capabilities of hosts to send more than one message simultaneously. For instance, suppose each host can send 3 messages at the same time. Thus, each MDO instead of dividing its customized list into two halves by finding median; divides it into four sections and finds three MDOs to send the message. Consequently, message propagation among MDOs can be done in fewer steps. We have tested our mechanism with 8 hosts and with one member on each host. The results have been compared to the mechanism that sequentially sends a message to all group members. On a 100Mbps Land Area Network (LAN) and with a maximum burst time of 100ms and a message size ranging from 4000 to 8000 bytes. The proposed mechanism delivered the message to all group members in about ranging from 302s to 1020s for lower boundary and higher for the higher bound.

The results in the implementation show how messages are transmitted to groups from source IP addresses to destination(nodes).Table 2 shows the Packet Delivery time with a boundary of (500-10000).while Table 3 entails specific information of the multicast streams such as the packet in size, time in seconds and the message size.

Table 2: Packet Delivery Time (500-10000)

Table 2 shows the number of packets delivered with boundaries of 500 – 10000

Time (s)	Packets (Bytes)	MsgSize (Kb)
101	5000	4.8
123	5700	5.5
131	6450	6.2
145	7030	6.88
149	7480	7.37

Table 3: Packet Delivery Time (0-5000)

Table 3 shows the number of packets delivered with boundaries of 0 – 5000

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 04-10 www.ijaem.net ISSN: 2395-5252

Time (s)	Packets (Bytes)	MsgSize (Kb)
32	1118	1.1
40	1562	1.4
72	2256	2.1
80	3105	2.11
98	4503	7.1

Calculating Throughput of the system

Network throughput refers to the rate of message delivery over a communication channel, such as Ethernet or packet radio, in a communication network

packettime ∈ seconds …… (1)

Given TCP windows value = 64kb for computers running the Windows operating system

Converting to bits (64 KB x 8 = 524,288 bits)

Average Through put = $\frac{1118+1562+2256+3105+4503}{32+40+72+80+98} = \frac{12544}{322} = 38.94$bps…… (2)

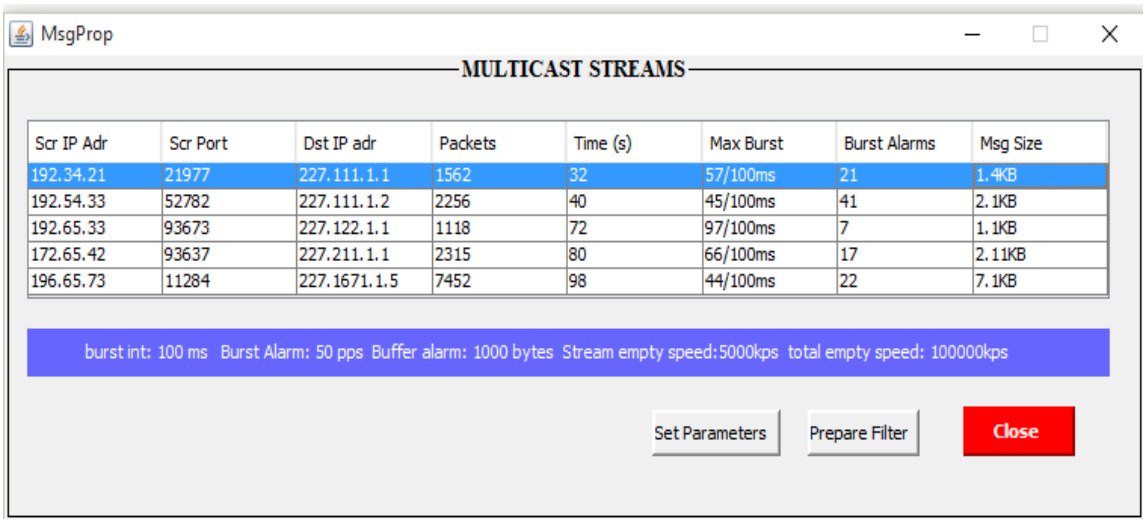

Figure 3: Multicast Streams Data

V. CONCLUSION

The approach used here achieves scalability and high message delivery speed by distributing and parallelizing the multicasting load among hosts in the system. It uses Message Dispatcher Objects (MDOs), which are objects co-located with each group member, to route and deliver multicast messages to the group members

REFERENCES

[1]. Abdelzaher, M., Birman, K. P. and Joseph, T. A. (2012). "Reliable Communication in the Presence of Failures," ACM Trans. Comp. Syst., 5(1), 47-76,

[2]. Al-Shaer, R., Marzullo, K. and Schmuck, F. (2015). "Supplying High Availability with a Standard Network File System," Proc. Eighth International Conference on Distributed Computing Systems, 447-453, San Jose, CA.

[3]. Amir, L. and Stanton, T. (2008). "Reliable Group Communication in Distributed Systems," Proc. Eighth International Conference on Distributed Computing Systems, 439-446, San Jose, CA.

[4]. Birman, K. P. (2009). "The Process Group Approach to Reliable Distributed Computing," Comm. ACM, 36(12), 36-53.

[5]. Birman, K. P., Cooper, R., Jospeh, T. A., Kane, K. P., Schmuck, F.. and Wood, M. (2009). "Isis - A Distributed Programming Environment," User's Guide and Reference Manual, Cornell University, Ithaca, NY, June.

[6]. Corman, I., Kaashoek, M. F., Michiels, R., Bal, H. E. and Tanenbaum, A. S. (2017). "Transparent Fault-Tolerance in Parallel

Orca Programs," Proc. Symposium on Experiences with Distributed and Multiprocessor Systems III, 297-312, Newport Beach, CA.

[7]. Fekete, E. and Keidar, R. (2001). "Replicated RPC Using Amoeba Closed Group Communication," Proc. of the 13th Conference on Distributed Computing Systems, 499-507

[8]. Gang, A., Bershad, B. N. and Zekauskas, M. J. (2019). "Midway: Shared Memory Parallel Programming with Entry Consistency for Distributed Memory Multiprocessors," CMU-CS91-170, Carnegie Mellon University, Pittsburgh, PA, Sep. 1991. Cited on page 96

[9]. Guerraoui, U. and Schiper, R. (2010). From Group Communication to Transactions in Distributed Systems. In Communications of the ACM 39(4), 84–87.

[10]. Johnson, L., Wooten, D. B. and Reed, A., II. (2010). A conceptual overview of the self-presentational concerns and response tendencies of focus group participants. Journal of Consumer Psychology, 9(3), 141-153.

[11]. Leopold, D. (2001). See Group Communication through Computers, Vols. 1 and 2, Institute for the Future Volume 2 describes the current techniques being used in the analysis of FORUM conferences.

[12]. Mishra, A. and Pang, L. (2008). "Transportation-Communication Substitutability: A Research Proposal," Bell Canada

[13]. Schiper, K. and Rynal, A. (2010). Trends in global virtual teams report. Retrieved from https://www.rw-3.com/virtual-teams-survey-0.

[14]. Shamir, M. (2016). Communication in Cross-functional Teams: An Introduction to the special issue. Professional Communication, (March), 19–21.

[15]. Valenci, C. (2008). The rules of virtual groups: Trust, liking, and performance in computer-mediated communication. Journal of Communication, 55(4), 828–846.