IOSR Journal of Computer Engineering (IOSR-JCE)
ISSN : 2278-0661, ISBN : 2278-8727, PP : 53-56
www.iosrjournals.org

Real Time And Distributed Computing Systems

Ms.Kamble A.L[1],Ms.Shinde.T.K[2], Ms Kothiwale N[3], Khot.S.S[4]
[1]*(Computer Science , Shree Datta Polytechnic, India)*
[2]*(Computer Science, Dr.J.J.Magdum College of Engineering, India)*
[3]*(Computer Science, Dr J.J.Magdum Polytechnic, India)*
[4]*(Computer Science, Dr.J.J.Magdum College of Engineering, India)*

ABSTRACT : *General-Purpose Operating Systems (GPOSes) are being used more and more extensively to support interactive, real-time, and distributed applications, as found in the multimedia domain. In fact, the wide availability of supported multimedia devices and protocols, together with the wide availability of libraries and tools for handling multimedia contents, make them an almost ideal platform for the development of this kind of complex applications. However, contrarily to Real-Time Operating Systems, General-Purpose ones used to lack some important functionality needed for providing proper scheduling guarantees to application processes. Recently, the increasing use of GPOSes for multimedia applications is gradually pushing OS developers towards enriching the kernel of a GPOS so as to provide more and more real-time functionality, thus enhancing the performance and responsiveness of hosted time-sensitive applications. In this chapter, an overview is performed over the efforts done in the direction of enriching GPOSes with real-time capabilities, with a particular focus on the Linux OS. Due to its open-source nature and wide diffusion and availability, Linux is one of the most widely used for such experimentations*

Keywords *– real time operating systems, distributed computational systems*

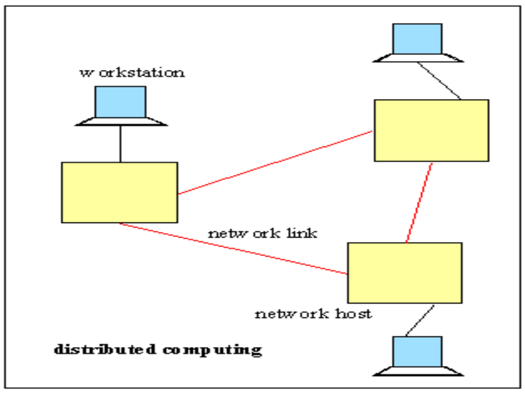
I.INTRODUCTION

The word distributed in terms such as “distributed systems”, ”distributed programming” and ”Distributed algorithm” originally referred to as computer networks where individual computers were physically distributed within some geographical area.

A distributed system consists of multiple autonomous computers or nodes that communicate which have its own local memory. Information is exchanged by passing messages between the processors through a computer network. The computer interacts with each other in order to achieve a common goal. In this, a problem is divided into many tasks, each of which is solved by one computer.

The main goal of a distributed computing system is to connect users and resources in a transparent, open and scalable way. Ideally this arrangement is drastically more fault tolerant and more powerful than many combinations of standalone computer systems. It is very difficult to define distributed system, but each of these systems, do have certain properties which are a part of the systems. Although there are a number of autonomous computational entities, each of these has their own local memory.

The different entities who are a part of the system communicate between each other by passing messages.These are, among others, types of applications that have been undergoing in the last few years a very steep demand on the side of end-to-end interactivity level, response time, and throughput. This is also due to the growing availability of affordable broadband Internet connections.



There can be different kinds of computers and network links which can be a part of the system. At the same time also there are chances of the system changing completely or partially during the execution of a distributed program..

II. Principle

This principle of distributed computing underlying the design of distributed systems:
1] Communication:_Communication does not come for free; often communication cost dominates the cost of local processing or storage.
2] Coordination: How can you coordinate a distributed system that it performs some task efficiently?
3] Fault-tolerance:_Major advantage of a distributed system is that even in the presence of failures the system as a whole may survive.
4] Locality:_Networks keep growing. Luckily global information is not always needed to solve a task; often it is sufficient if nodes talk to their neighbors.
5] Parallelism:_How fast can we solve a task if we throw more hardware at the problem? How much parallelism is possible for a given problem?
6] Symmetry Breaking: Sometimes some nodes need to be selected to orchestrate the others. This is done by a technique called symmetry breaking.
7] Synchronization:_How can you implement a synchronous algorithm in an asynchronous system?
8] Uncertainty:_If we need to agree on a single term describing the course, it probably is “uncertainty”. As the whole system is distributed no node knows what other nodes are doing at this exact moment.

III. Goals and Advantages

There are many different types of distributed computing systems and many challenges to overcome in successfully designing one. The main goal of a distributed computing system is to connect users and IT resources in a transparent, open, cost-effective, reliable and scalable way.

Ideally this arrangement is drastically more fault tolerant and more powerful than many combinations of standalone computer systems.

There are following advantages of distributed systems.

1] Openness:

Openness is the property of distributed systems such that each subsystem is continually open to interaction with other systems. There are some protocols and standards which enable distributed systems to be extended and scaled open distributed are inspired to meet the following challenges:
a] Monotonicity: Once something is published in an open system, it cannot be taken back.
b] Pluralism: Different subsystems of an open distributed system include heterogeneous, overlapping and possibly conflicting information. There is no central arbiter of truth in open distributed systems.
2] Scalability: A scalable system is one that can easily be altered to accommodate changes in the number of users, resources and computing entities affected to it. Scalability can be measured in three different dimensions:
a] Load Scalability: A distributed system should make it easy for us to expand and contract its resource pool to accommodate heavier or lighter loads.
b] Administrative scalability: No matter how many different organizations need to share a single distributed system, it should still be easy to use and manage.

IV. **Drawbacks / Disadvantages**

If not planned properly, a distributed system can decrease the overall reliability of computations, if the unavailability of a node can cause a disruption of the other nodes because the analysis may now require connecting to remote nodes or inspecting communications being sent between nodes.

There are following disadvantages of distributed systems.
1] Complexity-_Extra work must be done by the DBAs to ensure that the distributed nature of the system is transparent. Extra database design work must also be done to account for the disconnected nature of the database.
2] Economics-Increased complexity and a more extensive infrastructure mean extra labor costs.
3] Security-_Remote database fragments must be secured and they are not centralized, so the remote sites must be secured as well.

V. Difficult to maintain integrity-

In a distributed database, enforcing integrity over a network may require too much of the network’s resources to be feasible

VI. Inexperience- Distributed databases are difficult to work with and as a young field there is not much readily available experience on proper practice.

VII. Lack Of standards – There are no tools or methodologies yet to help users convert a centralized DBMS into a distributed DBMS.

VIII. Types of Distributed Systems

1] Clusters: A computer cluster is a group of linked computers, working together closely, forming a single computer. The components of a cluster are commonly connected to each other through fast local area networks. Clusters are usually deployed to improve performance and availability over that of a single computer.
2] Grids: Clusters can be combined to form a grid, a system of massive collective computing power which is designed to be easily used by “plugging in” to it.
3] Peer-to-Peer: A system where by individual users or nodes can communicate with each other by themselves.

IX. Applications

1) Telecommunication networks
Distributed system can be used in various Telephone networks, cellular networks, computer networks such as the Internet, wireless sensor networks and Routing algorithms.
2) Network applications
Distributed systems has a wide scope in World wide web and peer-to-peer networks, Massively multiplayer online games and Distributed information processing systems such as banking systems and airline reservation systems.
3) Real-time process control
Distributed systems are used in Aircraft control systems and Industrial control systems.
4) Parallel computation These distributed systems can also be used in Scientific computing, including cluster computing and grid computing and various volunteer computing projects and distributed rendering in computer graphics.

X. Architectural Models Related To Distributed Computing

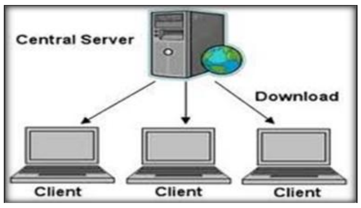
Various hardware and software architectural models are used for distributed computing. At a lower level, it is necessary to interconnect multiple CPUs with some sort of network regardless of whether that network is printed onto a circuit board or made up of loosely-coupled devices and cables.

At a higher level, it is necessary to interconnect processes running on those CPUs with some sort of communication system.

Distributed programming typically falls into one of several basic architectures or categories:

- Client-server,
- Tier architecture-tier architecture,
- Tightly coupled architecture,
- Peer-to-peer architecture,

1] Client-Server model
The client server model of computing is distributed application structure that partitions tasks of workloads between the providers of a resource or service called servers and service requesters called clients. Often clients and servers communicate over a computer network on a separate hardware but both client and server may reside in the same system.



A server machine is a host that is running one or more client does not share any of its resources, but requests a server’s contents or service function. Clients therefore initiate communication sessions with servers which a wait incoming requests.

2] 3-tier Architectural Model
Three – tier architecture is an extension to the client-server architecture and is defined by the following three component layers:

1) Presentation Tier

Also known as the client or front end, deals with the interaction with the user. Usually, there can be any number of clients which can all access the server at the same time. Clients process user input, send requests to the server, and show the results of these requests to the user. Client implements the presentation logic.

2) Application Tier

Also known as the server or the back-end or middleware, it processes the requests of all clients. It is the actual web application that performs all functionality specific to the web application. The business logic is implemented on an application server(s).Whenever it needs data; it contacts the database server(s).

3] N-tier architecture model

N-tier architecture is really 3 tier architectures in which the middle tier or application tier is split up into new tiers. A tier is one of two or more rows, levels or ranks arranged one above another. N-tier refers typically to web applications which further forward their requests to other enterprise services. This type of application is one of the most responsible for the success of application servers'. N-tier applications have the advantage that any one tier can run on an appropriate processor or operating system platform and can be updated independently of the other tiers.

4] Tightly coupled model :

Tightly coupled Distributed computing model actually means two entities are closely associated. It typically refers to a cluster of highly integrated machines that closely work together, running a shared process in parallel. The task is subdivided in parts that are made individually by each one and then put back together to make the final result.

The components of a cluster are commonly but not always connected to each other through fast networks. All clusters share common main memory and are usually deployed to improve performance and availability over that of a single computer while typically being much more cost effective than single computers of comparable speed or availability.

5] Peer-to-peer model:

Peer means a host computer. When many hosts are connected through to share files, computing capabilities, network bandwidth and storage, it calls peer to peer networking system. In peer-to-peer computing, each computer can serve the function of both client and server. Any computer in the network can initiate questions (like a client), receive questions and transmit data (like server).

In some cases, peer-to peer computing is implemented by actually giving each computer on the network both server and client capabilities. Peer-to-peer encompasses two distinct types of technology:1) The sharing of different computer processing and storage capabilities.

2) The sharing of digital files and data between two computers.

XI. Conclusion

Distributed computing encompasses many of the entities occurring in today's' world. No one entity could ever control the Internet.

It's just too distributed, in every sense.

The Internet will continue to spread it's already global reach for our foreseeable future, routinely touching every aspect of our daily lives.

As we continue our exploration of a truly unknowable future, we must try to keep in mind that in all the important ways. We must help to define the future of computing. While there are both physical and non-physical elements that make up this thing, we call the Internet.

The Internet and the World Wide Web ignore borders and instead allow people the freedom to view the World as a continuum and to see them as a part of the whole picture.

We are all part of a rapidly evolving complex system that is constantly changing and adapting, both as a whole and as individual points in a dynamic matrix of objects, processes and events. With each point in the matrix playing an integral part, we will continue to collectively define, create and explore the fantastic future of computing.

References

[1] A Note on Distributed Computing
[2] Algorithms and Distributed
[3] Computing Distributed Computing: Introduction
[4] Distributed Computing Utilities, Grids & Clouds
[5] www.google.com
[6] Distributed Object Systems
[7] The Clouds Project: Designing and
[8] Implementing a Fault Tolerant
[9] Distributed Operating System.
[10] Proceedings: Distributed Computing (Fall 1980).