# Resource Management in Distributed Systems

Nalin Chaudhary[1], Dr.Pushpneel Verma[2]
[1]*Research Scholar, Dept. Of CSE, Bhagwant University, Ajmer, Rajasthan, india*
[2]*Associate Professor, Dept. Of CSE, Bhagwant Institute of Technology, Muzaffarnagar, U.P., India*

***Abstract*- A distributed system is a collection of autonomous computers with the aim of enhancing resource sharing. Resource management in a distributed system involves resource discovery and resource scheduling. Resource discovery can be implemented in a centralized or decentralized manner. Our application is based on handling resource discovery efficiently in a decentralised way in a distributed system. We have implemented a main information server which maintains information about all the servers in the network because in order to be the part of network all the servers have to get themselves registered with the information server. Each server keeps a record of all the resources present among its clients. Whenever a client issues a request for a resource it is firstly checked onto itself and if the requested resource is not available there then it makes a request to its immediate server. If it also fails at its immediate server then with the help of information server the request is broadcasted in the network. If this request reaches a server who is possessing the resource then an acknowledgement will be passed to the server in the network which was responsible for generating the request.**

***Index Terms*- Record, Distributed, Server, Autonomous etc.**

## I. INTRODUCTION

Distributed system is a collection of loosely coupled processors interconnected by a communication network where each processor is addressed as a machine or a node or a host or a 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 stand-alone computer systems. The basic architecture of a distributed system is shown as follow :-

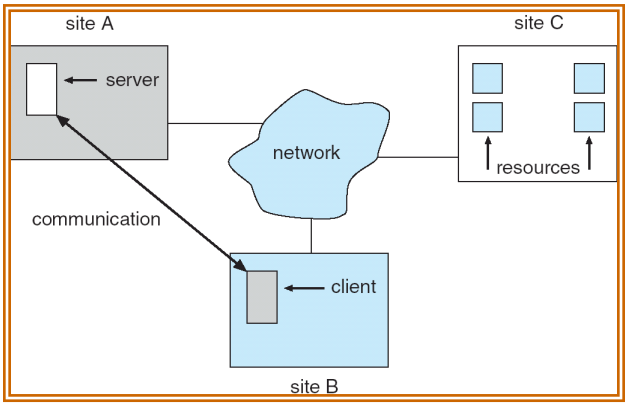


**Fig -1:** Distributed System Architecture

Various hardware and software architectures 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, 3-tier architecture, N-tier architecture, Distributed objects, loose coupling, or tight coupling.

A. *Client-server*

Smart client code contacts the server for data, then formats and displays it to the user. Input at the client is committed back to the server when it represents a permanent change[1].

B. *tier architecture*

Three tier systems move the client intelligence to a middle tier so that stateless clients can be used. This simplifies application deployment. Most web applications are 3-Tier[1].

C. *N-tier architecture*

N-Tier refers typically to web applications which further forward their requests to other enterprise services. This type of application is the one most responsible for the success of application servers[1].

*D. Tightly coupled*

(clustered) — refers typically to a cluster of 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 [1].

E. *Peer-to-peer*

an architecture where there is no special machine or machines that provide a service or manage the network resources. Instead all responsibilities are uniformly divided among all machines, known as peers. Peers can serve both as clients and servers [1].

F. *Space based*

Refers to an infrastructure that creates the illusion (virtualization) of one single address-space. Data are transparently replicated according to application needs. Decoupling in time, space and reference is achieved [1].

G. *Grid approach*

A grid uses the resources of many separate computers, loosely connected by a network (usually the Internet), to solve large-scale computation problems. Public grids may use idle time on many thousands of computers throughout the world. Such arrangements permit handling of data that would otherwise require the power of expensive supercomputers or would have been impossible to analyze[1].

## II. RESOURCE MANAGEMENT

Every node consists of finite number of resources that are to be distributed among a number of competing processes occurring at that very node. Memory space, CPU cycles, files and I/O devices are the example of resources being used by any process going within a node[6]. As the number of resources available to a system are limited so we can make use of distributed architecture which allows remote sharing of the various resources. Thus a system present at one location can access the resources of a system present at another location provided both the systems are a part of distributed architecture.

## III. RESOURCE DISCOVERY

Inorder that any process going within a system can access any resource of the system the very first phase for using of the resource is resource request. After making a request for the resource the process must wait till the time the resource can be allocated to the process. As here we are considering the distributed architecture so any process within the system can even make a request to the resource which is not present with that very system. Inorder to make request for a remote resource present within the distributed system the very first phase is resource discovery . This paper undertaken by us is going to give an idea of how a resource can be discovered in a distributed environment in an efficient manner such that it leads to management of resources in a decentralised way[6].

## IV. RESOURCE MANAGEMENT IN DISTRIBUTED SYSTEMS

"RESOURCE MANAGEMENT IN DISTRIBUTED SYSTEMS" is an application specifically designed to handle resource discovery efficiently in a decentralised way in a distributed system. We have implemented a main information server which maintains information about all the servers in the network because in order to be the part of network all the servers have to get themselves registered with the information server. Each server keeps a record of all the resources present among its clients. Whenever a client issues a request for a resource it is firstly checked onto itself and if the requested resource is not available there then it makes a request to its immediate server. If it also fails at its immediate server then with the help of information server the request is broadcasted in the network. If this request reaches a server who is possessing the resource then an acknowledgement will be passed to the server in the network which was responsible for generating the request[6].

A Client is a node or a simple part of the local network that can be part of the distributed system by connecting it to a server. This node is equipped with the privileges of adding, deleting, modifying a resource present with it. In order to access the resources of the network connected using distributed architecture the client makes request to its server which provides it with requested resource[6].

A Server is a node or a simple part of the local network that can be part of the distributed system by connecting it to a main information server. Thus it is a node that is particularly used to connect several clients (or nodes) together and is particularly use to deal with resource requests of its clients[6].

A Main Information Server is a node that is used to keep track of all the servers within the distributed architecture. Thus all the nodes which want to act as a server within the distributed architecture must be registered with this main information server. The main information server is used to route a resource request to all the servers in the distributed architecture if a particular server fails to handle the resource request of its client[6].

## V. DISTRIBUTED SYSTEM

In words of Tanenbaum : "A distributed system is a collection of independent computers that appear to the user of the system as a single computer". It is

indeed a bunch of "computers" connected by "wires". Nodes are (at least) semi-autonomous...but run software to coordinate and share resources. 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 stand-alone computer systems. In this focus is on systems to store/access/share data and operations on data. Data is moved around the network and is delivered to the right places at the right times, safely and securely. The prime focus is laid on Internet information services and their building blocks[6].

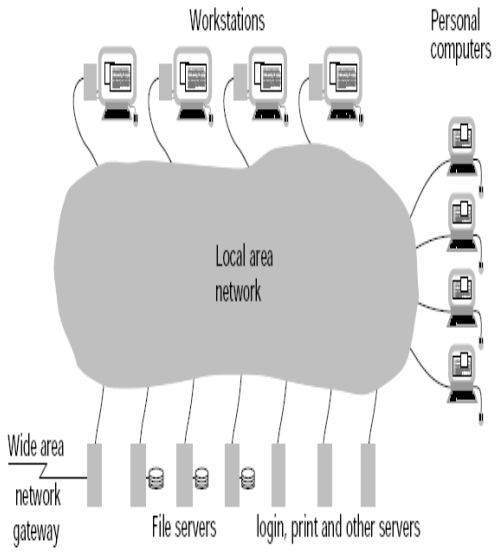


**Fig - 2 :** A Simple Distributed System

## VI. ADVANTAGES

There are many different types of distributed computing systems and many challenges to overcome in successfully designing one. Ideally this arrangement is drastically more fault tolerant and more powerful than many combinations of stand-alone computer systems. Following are the advantages of distributed systems :-

*A. Openness*

Openness is the property of distributed systems such that each subsystem is continually open to interaction with other systems. Web services protocols are standards which enable distributed systems to be extended and scaled. In general, an open system that scales has an advantage over a perfectly closed and self-contained system. Openness cannot be achieved unless the specification and documentation of the key software interface of the component of a system are made available to the software developer. Consequently, open distributed systems are required to meet the following challenges:-

B. *Monotonicity*

Once something is published in an open system, it cannot be taken back.

C. *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.

*D. Unbounded Nondeterminism*

Asynchronously, different subsystems can come up and go down and communication links can come in and go out between subsystems of an open distributed system. Therefore the time that it will take to complete an operation cannot be bounded in advance[6].

## VI. CONCLUSIONS

Ideally this arrangement is drastically more fault tolerant and more powerful than many combinations of stand-alone computer systems. In this focus is on systems to store/access/share data and operations on data. There are many different types of distributed computing systems and many challenges to overcome in successfully designing one. Ideally this arrangement is drastically more fault tolerant and more powerful than many combinations of stand-alone computer systems.

A Server is a node or a simple part of the local network that can be part of the distributed system by connecting it to a main information server. Thus it is a node that is particularly used to connect several clients (or nodes) together and is particularly use to deal with resource requests of its clients. This paper undertaken by us is going to give an idea of how a resource can be discovered in a distributed environment in an efficient manner such that it leads to management of resources in a decentralised way.

## REFERENCES

[1] Computer Networks – Andrew S Tanenbaum.
[2] Core Java 2 Volume 2-Advanced Features – By Cay S. Horstmann, Gary Cornell Sun Microsystems.
[3] The Complete Reference Java 2 – By Herbert Schildt, Tata McGraw-Hill.
[4] www.sun.java.com.
[5] http://java.sun.com/docs/books/tutorial.
[6] An introduction to Database Systems – By C.J. Date.

[7] Schneider, B., Hanges, P.J., Smith, D.B. and Salvaggio, A.N. (2003). Which comes first: Employee attitudes or organizational financial and market performance? Journal of Applied Psychology, Vol. 88(5), 836- 851.
[8] Taylor, J. C. & Stern, Gary M. (2009). The Trouble With HR: An Insider's Guide to Finding and Keeping the Best Talent. ISBN 9780814413449.
[9] American Management Association.New York, p65 Truss C (2001) Complexities and Controversies in Linking HRM with Organizational Outcomes, Journal of Management Studies 38(8): 1121-1149 Uyargil, C, 2010.
[10] İnsan Kaynakları Yönetimi, 5.Baskı, Beta Basım, İstanbul, p.3. Zellars, K.L. & Fiorito, J. (1999). Evaluations of organizational effectiveness among HR managers: Cues and implications. Journal of Managerial Issues, Vol. 11(1), 37- 55.