DISTRIBUTED SYSTEMS DIT-5511

UNIT-1

Introduction :→

There are various definitions of the distributed systems.

1. A distributed system is a collection of independent computers that appears to its users as a single coherent system. — Tanenbaum

2. A distributed system is one in which components located at networked computers communicate and coordinate their actions only by passing messages. — Coulouris, Kindberg

3. Loosely coupled systems are called as distributed systems. Loosely coupled systems are those in which the processors do not share memory.
   - Each processor has its own local memory.

Characteristics of distributed systems →

According to the definitions of distributed system characteristics are as following -

(i) concurrency of components — concurrency of components

(ii) Lack of a global clock — Lack of a global clock

(iii) Independent failures of components — Independent failure of components

(iv) multiple autonomous computers — multiple autonomous computers

(v) speed — speed

(i) Concurrency :→

Since there are n computers in the distributed environment each sharing the resources, all resources are available to all users without the feeling of interaction.

(ii) No Global clock :→

- The notion of global time does not exist.
- Suppose a global clock is available for all the processes in the system.
- Then two different processes can observe a global clock value at different instants due to unpredictable message transmission delays.
- Therefore, two different processes may falsely perceive two different instants in physical time to be a single instant in physical time.
- In other alternative i.e. physical clock can drift from the physical time and the drift rate may vary from clock to clock due to technological limitations.

(iii) Independent failures :→

- If a computer in a distributed system fails, this failure is not immediately made known to other components in the system.
- Fault in the computer isolates it from the system

but the system does not stop running.

- Each component of the network can fail independently, still leaving others running successfully.

(iv) Multiple autonomous computers :-

- In a distributed system there are multiple computers and it is an individual autonomous node having its own software and resources.
- It can also share components and resources from the distributed environment and might not share some of the private resources.

(v) speed :→

- Proper utilization of shared resources makes work done very fast.
- As in distributed system capacity and processing powers are very high, work can be done in much less time.

Goals of distributed System

There are following goals of distributed system.

(i) connect users and making resources accessible
(ii) Distribution transparency
(iii) Support openness
(iv) Scalability

(i) Connect users and making resources accessible :→ (5)

- The main goal of a distributed system is to make it easy for the users to access remote resources and to share them in a controlled and efficient way.
- Resources can be printers, computers, storage facilities data files, web pages and networks.

(ii) Distributed transparency :→

- Another goal of a distributed system is to hide the fact that its processes and resources are physically distributed across multiple computers.
- A distributed system that is able to present itself to users and applications as if it were only a single computer system is said to be transparent.

Types of transparency

(a) Access :- Hide differences in data representation and how a resource is accessed

(b) Location :- Hide where a resource is located

(c) migration :- Hide that a resource may move to another location

(d) Relocation :- Hide that a resource may be moved to another location while in use

(e) Replication :- Hide that a resource is replicated

(g) concurrency :- Hide that a resource may be shared by several competitive users

(h) Failure :- Hide the failure and recovery of a resource

(iii) Support openness :→

- An open distributed system is a system that offers services according to standard rules that describe the syntax and semantics of those services.
- For example, in computer networks, standard rules govern the format, contents and meaning of messages sent and received.
- Such rules are formalized in protocols.
- In a distributed system there is an Interface Definition Languages (IDL) which specifies -
  - Name of the functions that are available together with types of the parameters
  - Return values of each function.
  - Any exception to be raised on a specific condition.

Advantages :-

(a) Flexibility :- A flexible open distributed system is one which allows easy configuration of the system with different components from

different developers.

(b) Interoperability :- Open system can work together. This feature allows two implementations of system or components from different manufacturers to co-exist and work together.

- Both of them rely on each others services as specified by a common standard.

(c) Portability :- Portability is an ability to transform an application from one software or hardware platform to another.

(d) Extensibility :- Distributed system allows to add new components to the system, replaces the existing ones.

- This is done without affecting those components those are in their original place.

(IV) Scalability :→

Scalability is measured along three different dimensions.

(a) Scalability with respect to size :-

This means that in a scalable distributed system users and resources can be easily added to the system.

(b) Scalability with respect to geographic area –

- In a scalable distributed system, users and resources may lie geographically at different locations.
- But still a communication between them is flexible and addition of new users and new resources is possible.

(c) scalability with respect to administration –

- A distributed system is to be properly scaled.
- A distributed system with various types of

- A distributed system with this feature allows easy and proper management of the system even if the system is widely split amongst various organizations.

Examples of Distributed Systems

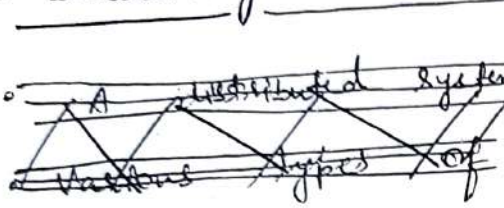
(i) Internet :- →

- Internet is a very large distributed system.
- It enables users, wherever they are, to make use of services such as the world wide web email and file transfer.

(b) Scalability with respect to geographic area –

- In a scalable distributed system, users and resources may lie geographically at different locations.
- But still a communication between them is flexible and addition of new users and new resources is possible.

(c) scalability with respect to administration –



- A distributed system with this feature allows easy and proper management of the system even if the system is widely split amongst various organizations.

Examples of Distributed Systems

(i) Internet :- →

- Internet is a very large distributed system.
- It enables users, wherever they are, to make use of services such as the world wide web email and file transfer.

- Internet service providers (ISPs) are companies that provide modem links and other types of connection to individual users and small organizations, enabling them to access services anywhere in the Internet.
- Also provides local services such as email and web hosting.

(ii) [redacted] Intranet :→

- An intranet is a portion of the internet that is separately administered and has a boundary that can be configured to enforce local security policies.
- It is composed of a several local area network (LANs) linked by backbone connection.
- The network configuration of a particular intranet is the responsibility of the organization.
- An intranet is connected to the internet via router, which allows the users inside the intranet to make use of services elsewhere such as the web or email.
- It allows the users in other intranets to access its services of "prowlers".
- Many organizations protects their services from unauthorized use by firewall.

Some organizations do not wish to connect their internal networks to the Internet at all such as police and other security and law enforcement agencies.

(iii) mobile and ubiquitous computing :→

- Technological advances in device miniaturization and wireless networking have led increasingly to the integration of small and portable computing devices into distributed systems.

These devices includes -

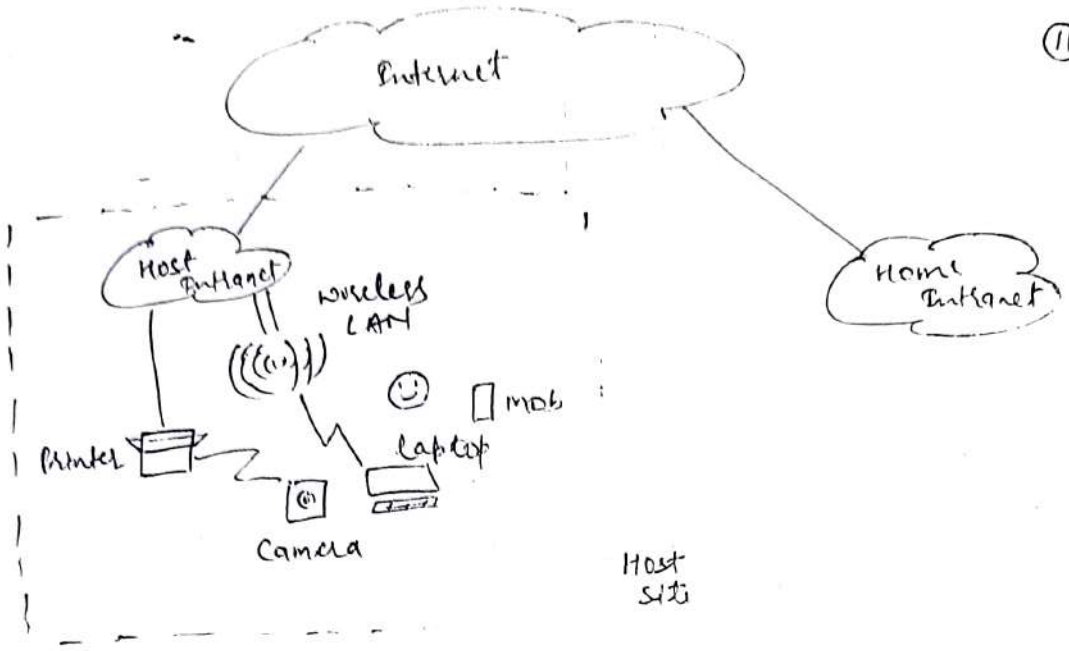
- Laptop
- Personal digital assistants (PDAs), mobile phones, pagers, video cameras and digital cameras
- Wearable device, such as smart watches with functionality similar to a PDA.
- Devices embedded in appliances such as washing machines, hi-fi systems, cars and refrigerators.

- The instability of many of these devices, together with their ability to connect conveniently in networks in different places, makes mobile computing possible.
- mobile computing is the performance of computing tasks while the user is on the move, or

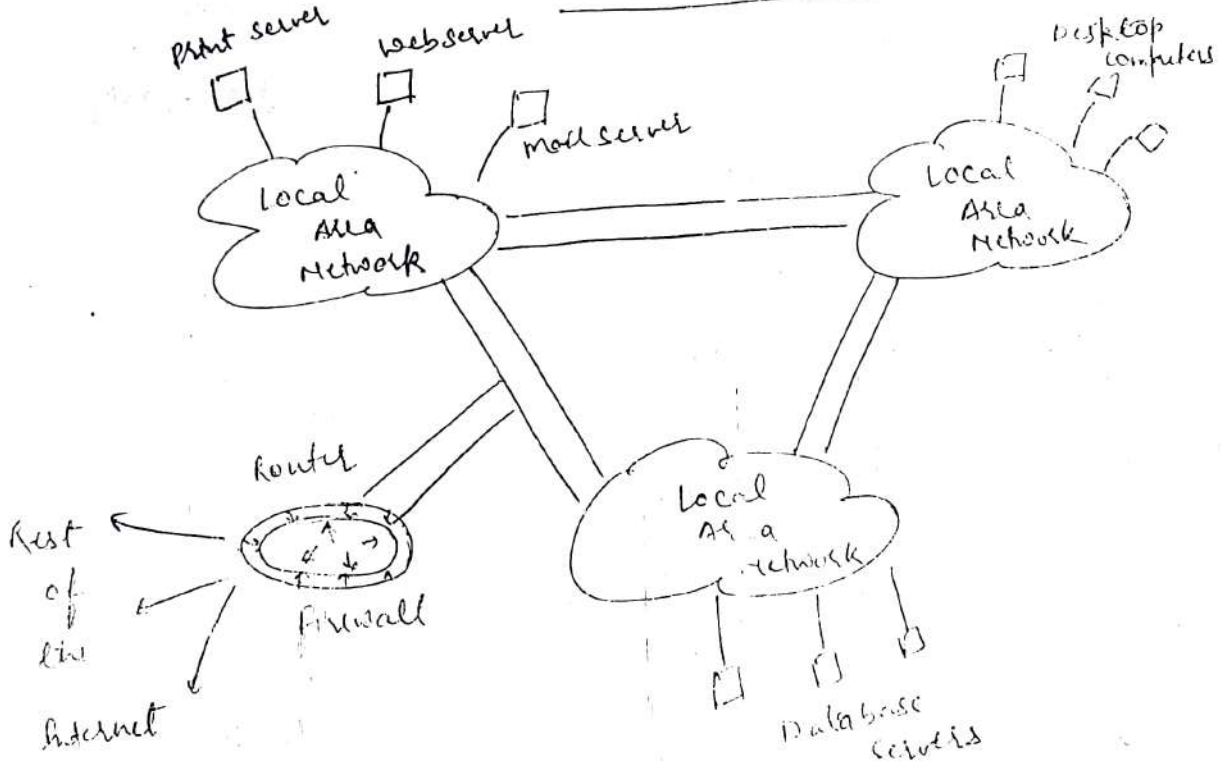
visiting places other than their usual environment (i.e.)

- In mobile computing, users who are away from their home intranet are still provided with access to resources via the devices they carry with them.
- They can continue to access the Internet; they can continue to access resources in their home intranet.

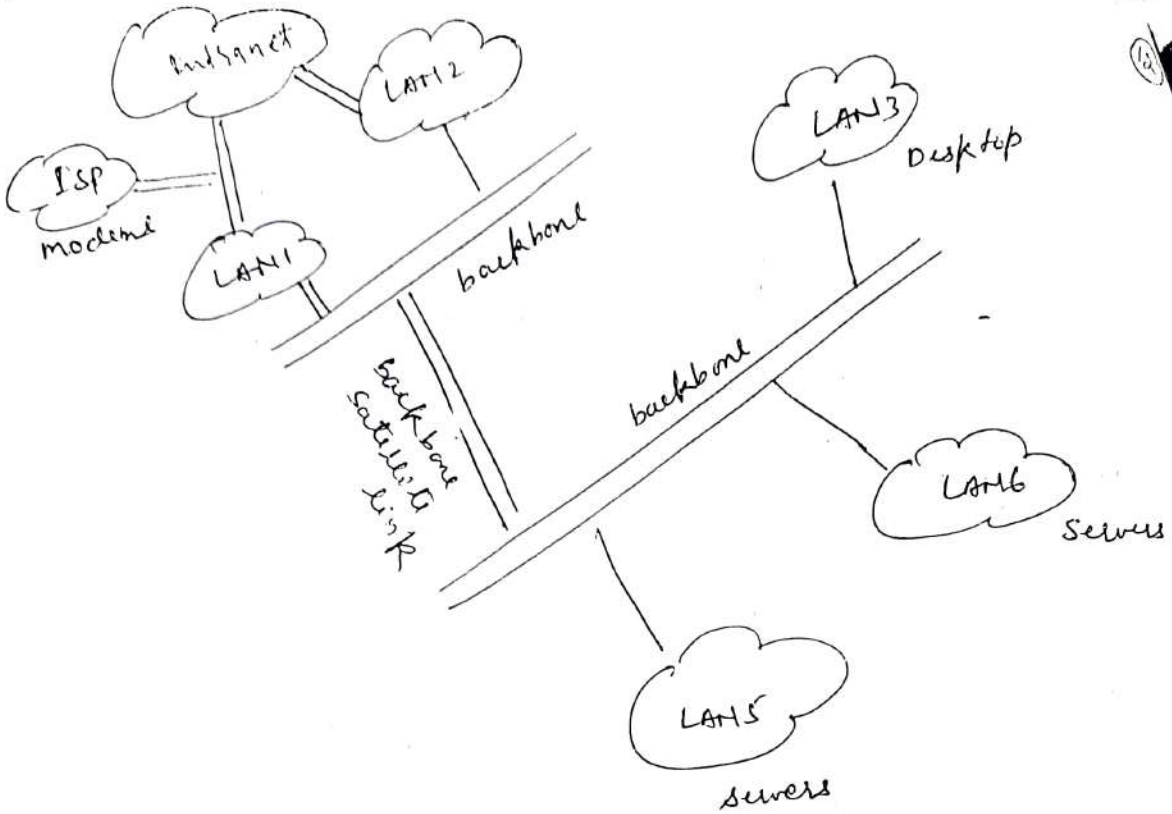
- Ubiquitous Computing – is the harnessing of many small, cheap computing devices that are present in user's physical environments including the home, office and even natural setting.
- The term 'ubiquitous' is intended to suggest that small computing devices will become so pervasive in everyday objects that they are scarcely noticed.
- The presence of computers everywhere only becomes useful when they can communicate with one another.
- For example, it would be convenient for users [redacted] to control their washing machine and hi-fi system from a 'universal remote control' device in the home.



Portable and handheld devices in a distributed system



A typical Intranet



A typical Portion of internet

Some more examples of distributed systems are –

- WWW
- Air traffic control
- Stock brokerage systems
- Banking
- Distributed file systems
- Network storage
- File sharing

Gaming

Challenges :→

The construction of distributed systems produces many challenges.

(i) Heterogeneity :→

- They must be constructed from a variety of different networks, operating systems, computer hardware and programming languages.
- The internet communication protocols mask the difference in networks, and middleware can deal with the other differences.
- The term middleware applies to a software layer that provides a programming abstraction as well as masking the heterogeneity of the underlying networks, hardware operating systems and programming languages.
- Some middleware such as Java Remote Method Invocation (RMI) supports only a single programming language.

(ii) Openness :→ Distributed systems should be extensible- the first step is to publish the interfaces of the components, but the integration of components written by different programmers is real challenge.

(iii) Security :- · Encryption can be used to provide adequate protection of shared resources and to keep sensitive information secret when it is transmitted in message over a network.

- Denial of service attacks are still a problem.

(iv) Scalability :→

- A distributed system is scalable if the cost of adding a user is a constant amount in terms of the resources that must be added.
- The algorithms used to access shared data should avoid performance bottlenecks and data should be structured hierarchically to get the best access times.

(v) Failure handling :→

- Any process, computer or network may fail independently of the others.
- Therefore each components needs to be aware of the possible ways in which the components it depends on may fail and be designed to deal with each of these failures appropriately.

Following techniques can be used to deal with failures

(a) Detecting failures — Some failures can be detected. For eg. checksums can be used to detect corrupted data in a message or a file.

(b) Masking failures - Some failures that have been detected can be hidden or made less severe. For eg.

- messages can be retransmitted when they fail to arrive.
- File data can be written to a pair of disks in that if one is corrupted, the other may still be correct.

(c) Tolerating failures -

- most of the services in the internet do exhibit failures
- Means it would not be practical for them to attempt to detect and hide all of the failures that might occur in such a large network with so many components.

(d) Recovery from failures →

- It involves the design of software so that the state of permanent data can be recovered or 'rolled back' after a server has crashed.

(e) Redundancy → services can be made to tolerate failures by the use of redundant components. For eg.

- There should always be at least two different routes between any two routers in the internet.
- In the Domain Name System, every name table is replicated in at least two different servers.
- A database may be replicated in several servers to ensure that the data remains accessible after the failure of any single server.

(f) Distributed systems provide a high degree of availability in the face of hardware faults.

- The availability of a system is a measure of the proportion of time that it is available for use

(vi) Concurrency :-

- The presence of multiple users in a distributed system is a source of concurrent requests to its resources.
- Each resource must be designed to be safe in a concurrent environment.

(vii) Transparency :→

- The aim is to make certain aspects of distribution invisible to the application programmer so that they need only be concerned with the design of their particular applications. Examples are :-

(a) Access transparency
(b) Location transparency
(c) Replication transparency
(d) migration transparency / mobility transparency
(e) Relocation transparency
(f) Failure transparency – enables the concealment of faults, allowing users and application programs to complete their tasks despite the failure of h/w or s/w components.
(g) Performance transparency – Allows the system to be reconfigured to improve performance as load vary.
(h) Scaling transparency :– Allows the system to expand in scale without change to the system structure or the application algorithms.

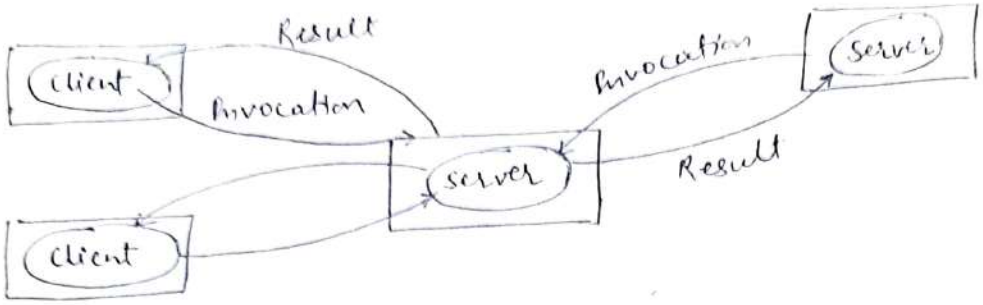
Architectural models

- An architectural model of a distributed system is concerned with the placement of its parts and the relationships between them.
- An architectural model first simplifies and abstracts the functions of the individual components of a distributed system and then it considers --
  - the placement of the ~~compnt~~ components across a network of computers - seeking to define useful patterns for the distribution of data and workload.
  - the relationships between the components - their functional roles and the patterns of communication between them.
- An initial simplification is achieved by classifying processes as server processes, client processes and peer processes.
- Examples of architectural model are client-server model and the peer to peer model.

(i) client - server model :→

- In this client processes interact with individual server processes in separate host computers in order to access the shared resources that they manage.
- Servers may in turn be clients of other servers.

- For example, a web server is often a client of a local file server that manages the files in which the web pages are stored.
- Web servers and most other Internet services are clients of the DNS service, which translates Internet Domain Names to network addresses.

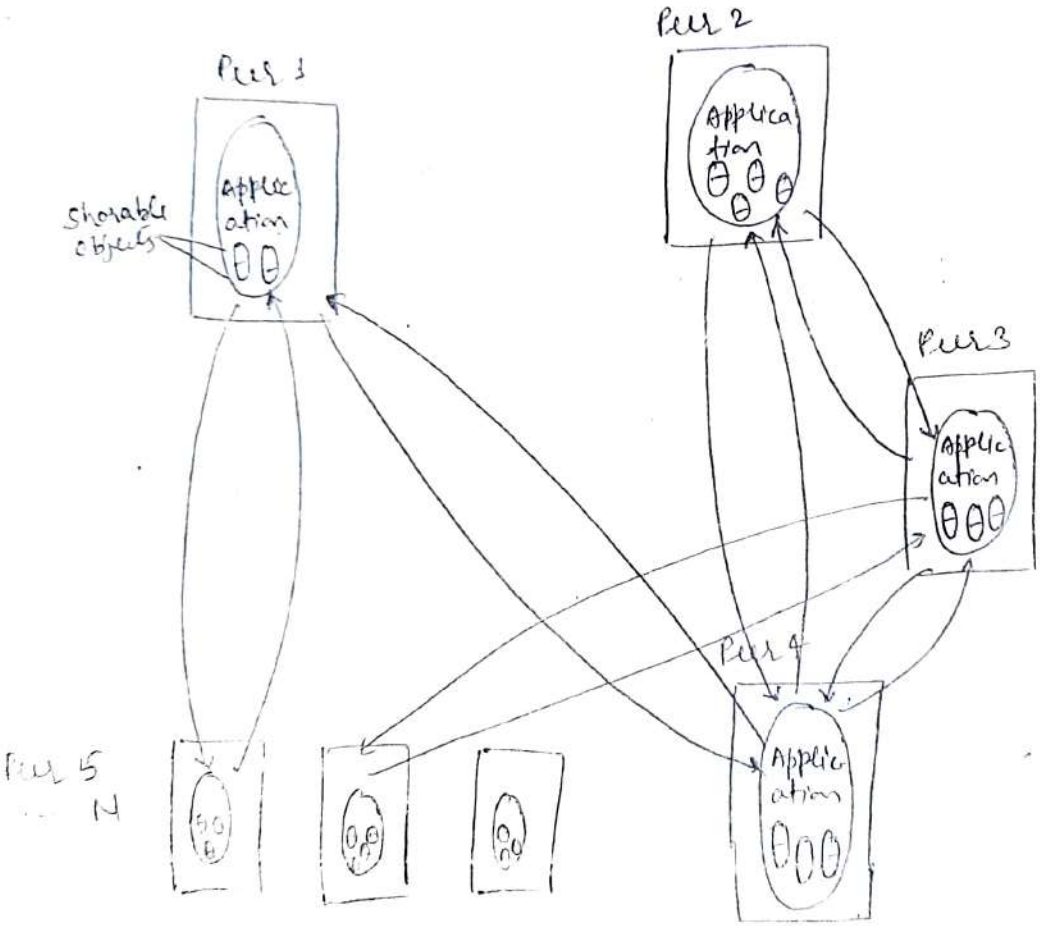


Client - server model

(ii) Peer to Peer model :→

- In this architecture all of the processes involved in a task or activity play similar roles, interacting cooperatively as peers without any distinction between client and server processes.
- Applications are composed of large numbers of peer processes running on separate computers and the pattern of communication between them depends entirely on application requirements.
- A large number of data objects are shared, an individual computer holds only a small part of the application database.

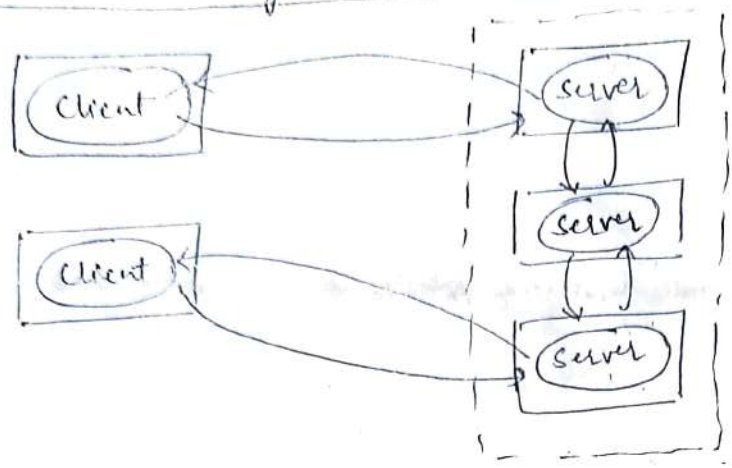
- And the storage, processing and communication loads of such access to objects are distributed across many computers and network links.
- Each object is replicated in several computers to further distribute the load and to provide resilience in the event of disconnection of individual computers.
- The need to place individual objects and retrieve them and to maintain replicas amongst many computers renders this architecture more complex than the client server architecture.



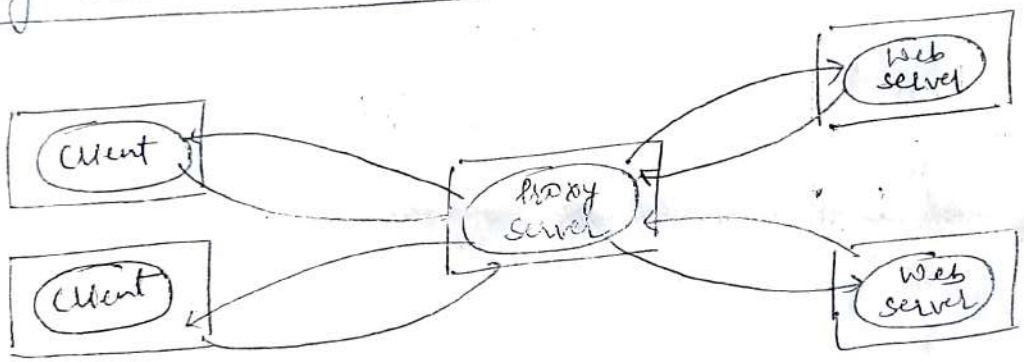
Peer to Peer Architecture

Variations → Variations of above models are as following -

(a) Services provided by multiple servers →

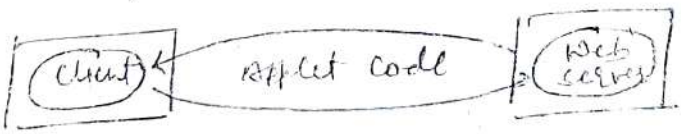


(b) Proxy servers and caches →



(c) Web Applets →

(i) Client request results in the downloading of applet code



(ii) Client interacts with the applet