Distributed Computing (2020)

Introduction Distributed Systems

Definition of a Distributed System (1)

A distributed system is:

A collection of independent computers that appears to its users as a single coherent system.

Definitions of a Distributed System:

- One important characters of DS is that differences between the various computers and the way in which they communicate are hidden from users. (Transparency)
- Users and applications can interact with a distributed system in a consistent and uniform way, regardless of where and when interaction take place.(Coherency)
- D.S should also be easy to expand or scale. (Scalability)
- A DS will normally be continuously available, although perhaps certain parts may be temporarily out of order. (Reliability)

Transparency

A important goal of a distributed system is to hide the fact that its process and resource are physically distributed across multiple computers

A DS that is able to present itself to users and applications as if it were only a single computer system is said to be transparent

Transparency in a Distributed System

Transparency	Description
Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located
Migration	Hide that a resource may move to another location Without affecting how that resources can be accessed
Relocation	Hide that a resource may be moved to another location while in use
Replication	Hide that a resource may be shared by several competitive users
Concurrency	Hide that a resource may be shared by several competitive users
Failure	Hide the failure and recovery of a resource
Persistence	Hide whether a (software) resource is in memory or on disk

Different forms of transparency in a distributed system.

Scalability

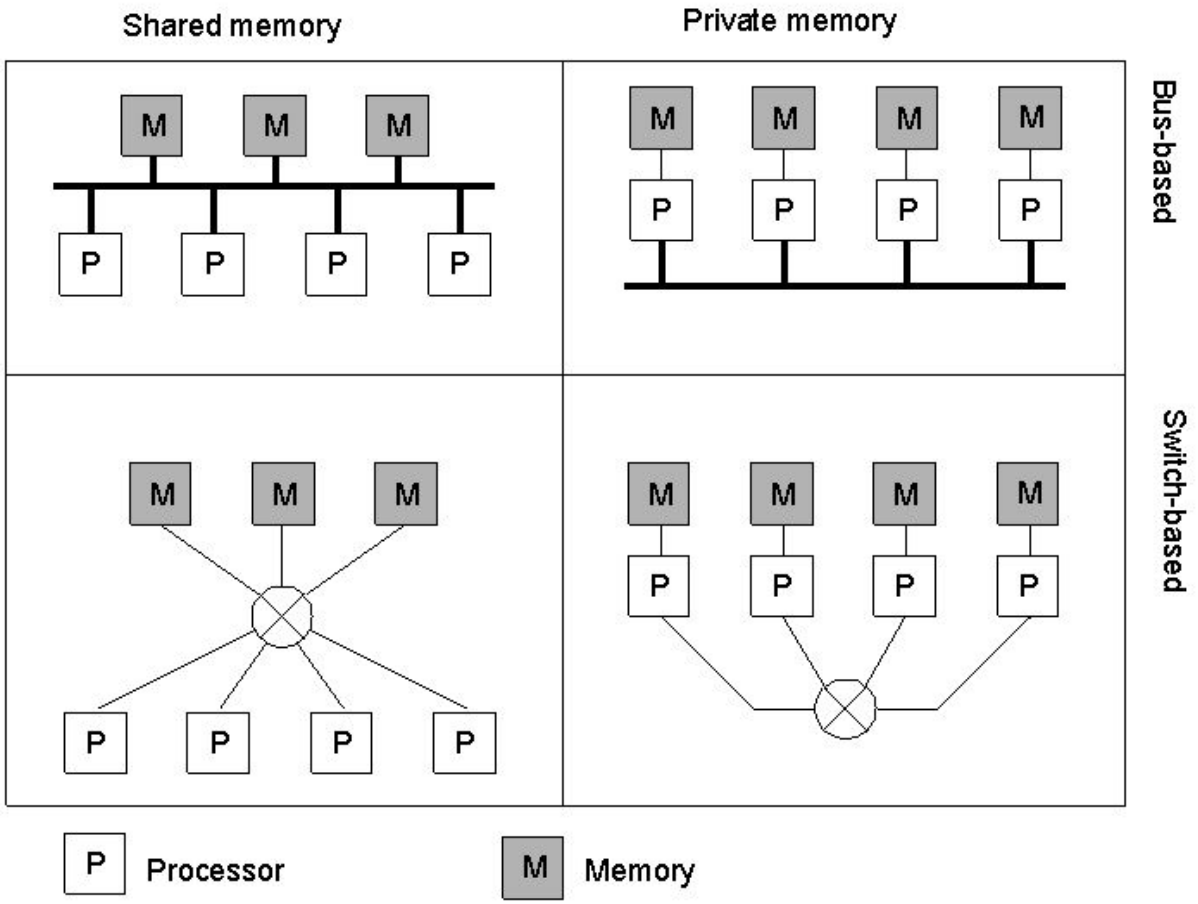
- Scalability of a system can be measured along at least three dimensions
- Size: add more users and resources to the system
- geographically scalable : system is one in which the users and resources may lie far apart
- Administratively scalable: easy to manage even if it spans many independent administrative organization

Openness

An open distributed system is a system that offers services according to standard rules that describes syntax and semantics of those services

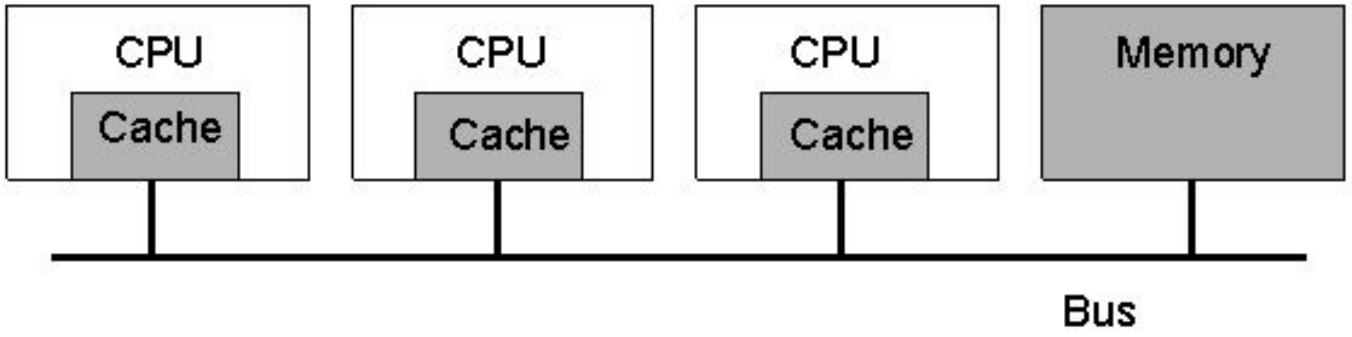
- Interfaces
- IDL

Hardware Concepts



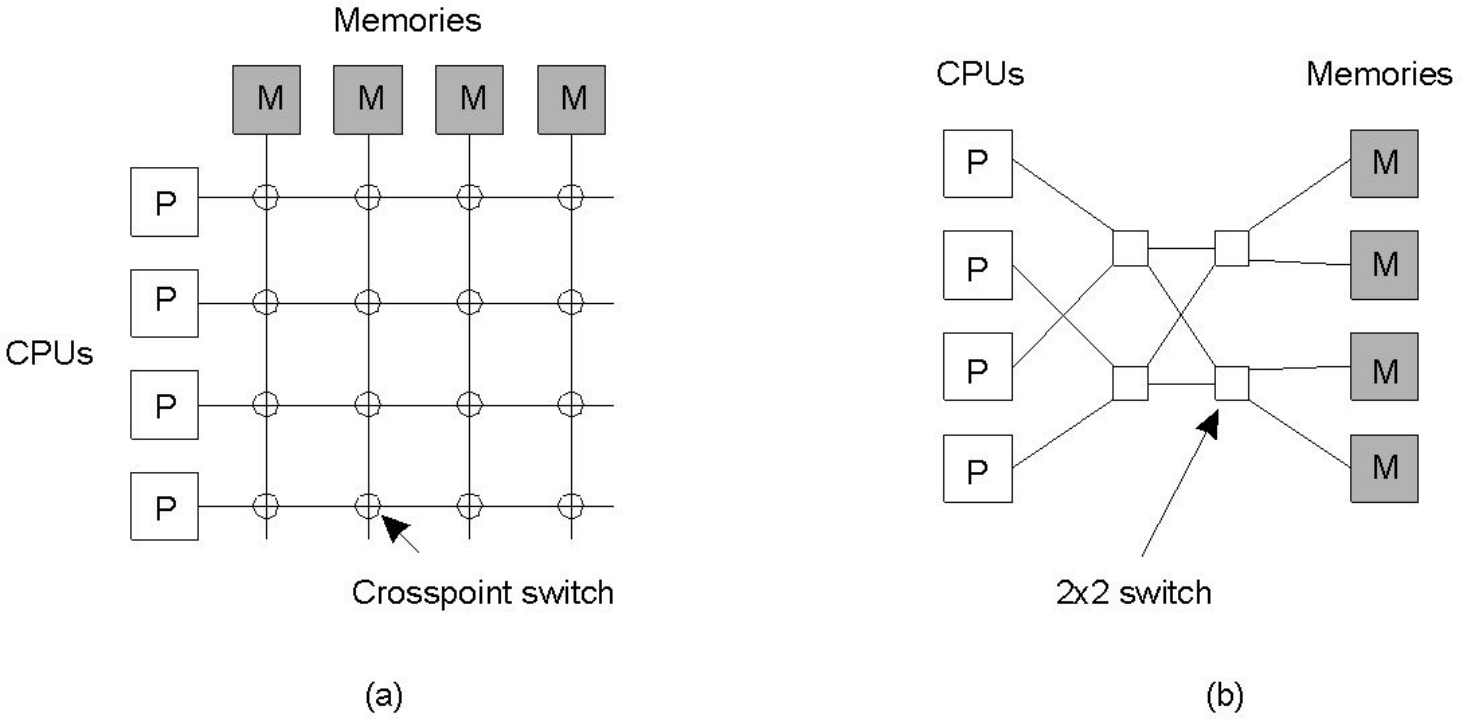
Different basic organizations and memories in distributed computer systems

Multiprocessors (1)



A bus-based multiprocessor.

Multiprocessors (2)



a) A crossbar switch
b) An omega switching network

Distributed System Types (Functionally)

1. Distributed computing system.

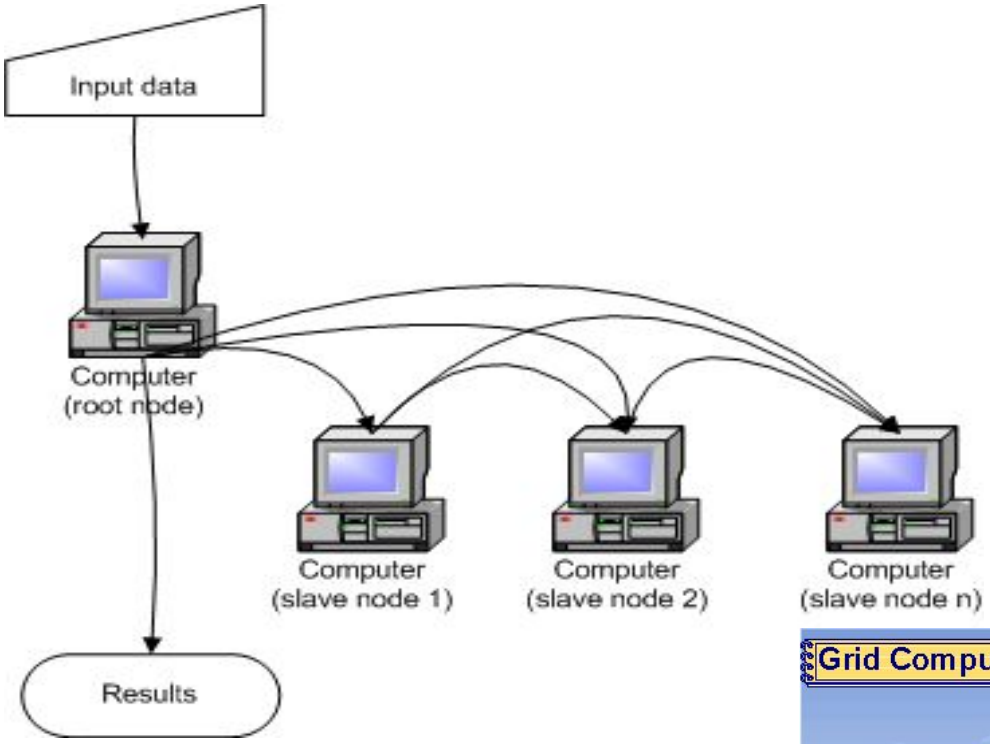
-Cluster Computing

- Grid Computing

2. Distributed information system

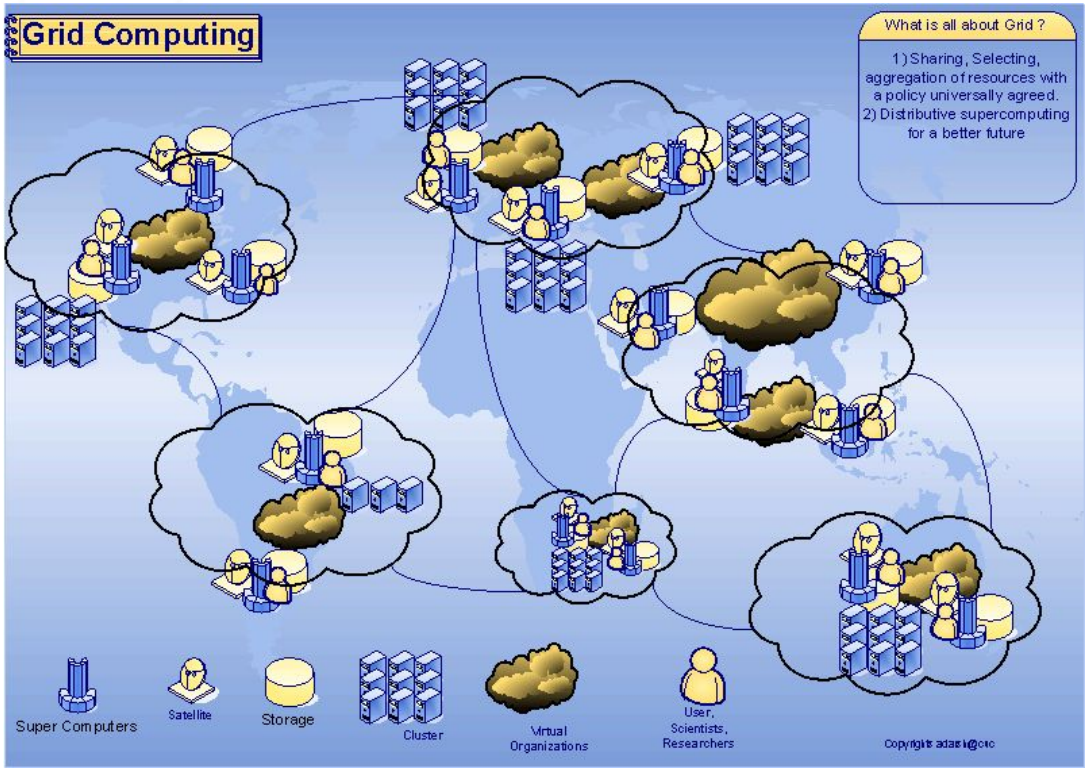
- Transaction processing system (TPS) and
- Enterprise Application Integration (EAI)

3. Distributed pervasive system.



Cluster Computing

Grid Computing



Characteristics	Computing Environment / Systems		
	Clusters	**Grids**	**Clouds**
Population	Commodity computers	High-End computers (Servers, Clusters)	Both and also web-base storage
Scalability	100s	1000s	100s to 1000s
Service Negotiation	Limited	Yes, SLA based	Yes, SLA based
User Management	Centralized	Decentralized and also virtual organization based	Centralized or can be delegated to third party
Resource Management	Centralized	Distributed	Centralized/Distributed
Capacity	Stable and Guaranteed	Varies, but high	Demand-base Provisioned
Pricing of Services	Limited, not open market	Dominated by public good or privately assigned	Utility pricing, discounted for larger customers
Node Operating System (OS)	Standard OS (Windows, Linux)	Standard OS (Dominated by Unix)	A hypervisor (VM) which can run multiple OSs
Failure Management (Self-Healing)	Limited (Often failed task/application are restarted)	Limited (Often failed task/application are restarted)	Strong support for failover and content replication. VMs can be easily migrated from one node to other.
Ownership	Single	Multiple	Single
Interconnection Network/Speed	Dedicated, high-end with low latency and high bandwidth	Mostly Internet with high latency and low bandwidth	Dedicated, high-end with low latency and high bandwidth
Discovery	Membership service	Centralized indexing and decentralized info services	Membership services

Distributed information system

- Transaction
- ACID
- A- Atomic
- C- Consistent
- I- Isolated
- D-Durable

Distributed information system

- Enterprise Application Integration
- RPC
- RMI
- Message Oriented Middleware

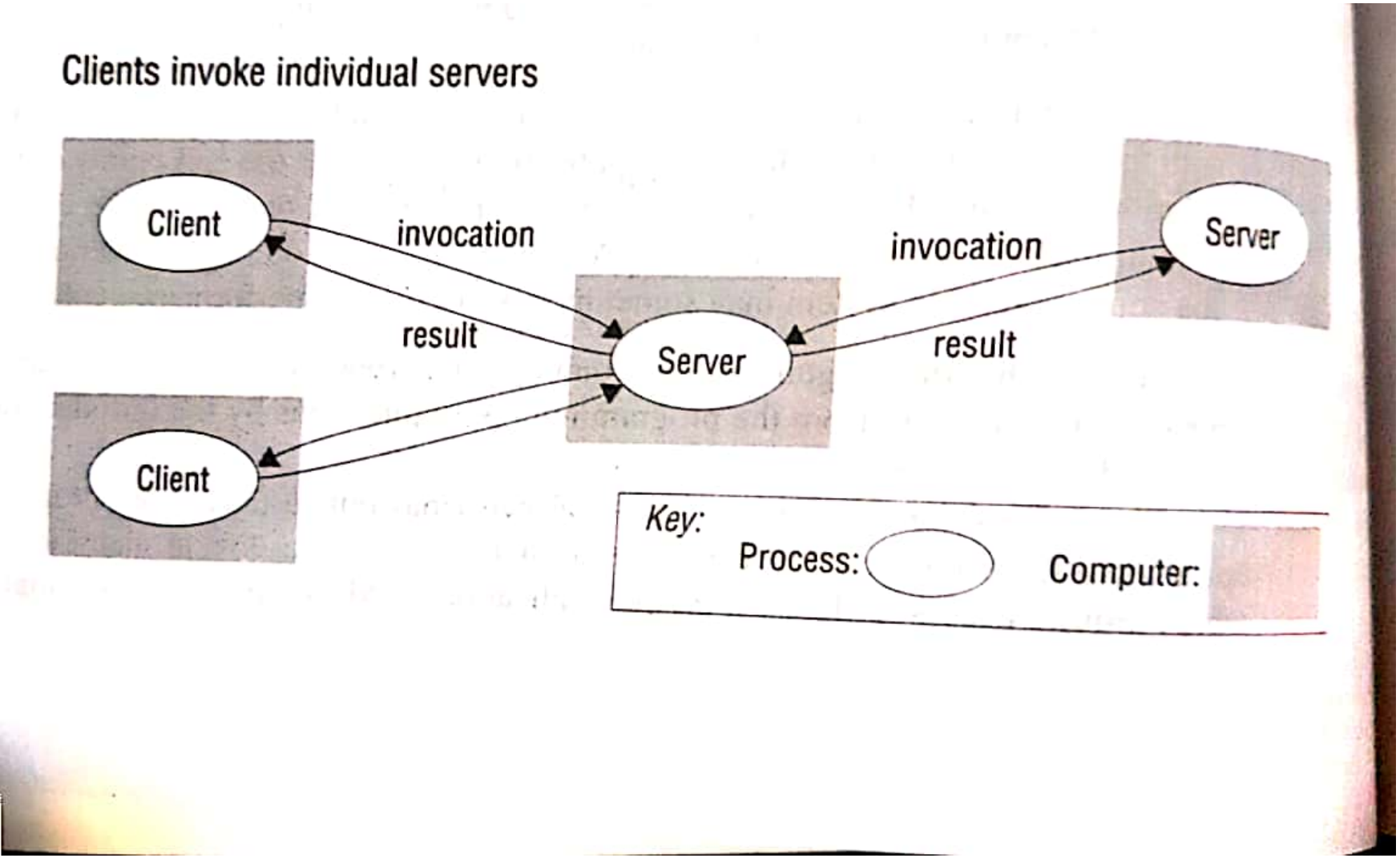
Distributed pervasive system

- Home Systems- Universal Plug and Play (UPnP)
- Electronic Health Care System
- Sensor Networks

Distributed Computing Models

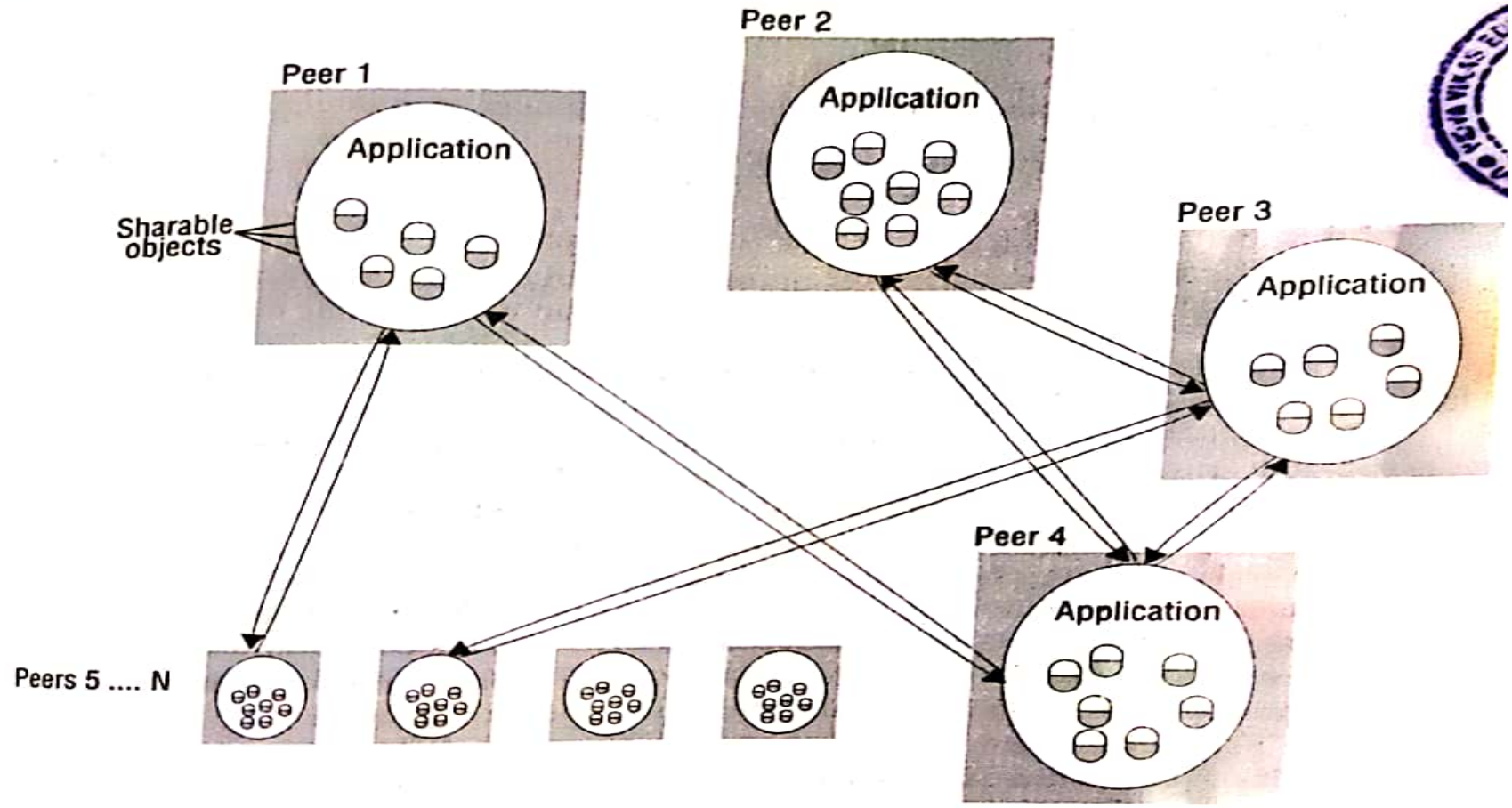
1. Architectural model : Client-server model and the peer-to-peer (P2P)
2. Interaction model : synchronous distributed systems and asynchronous distributed systems
3. Fault model: Omission fault, Arbitrary fault and timing fault
4. Security model: use combination of authentication and encryption methods

Client-server model



peer-to-peer (P2P)

re 2.3 A distributed application based on the peer-to-peer architecture



Fault model

Omission and arbitrary failures

Class of failure	*Affects*	*Description*
Fail-stop	Process	Process halts and remains halted. Other processes may detect this state.
Crash	Process	Process halts and remains halted. Other processes may not be able to detect this state.
Omission	Channel	A message inserted in an outgoing message buffer never arrives at the other end's incoming message buffer.
Send-omission	Process	A process completes a *send*, but the message is not put in its outgoing message buffer.
Receive-	Process	A message is put in a process's incoming message buffer, but omission that process does not receive it.
Arbitrary (Byzantine)	Process or channel	Process/channel exhibits arbitrary behaviour: it may send/transmit arbitrary messages at arbitrary times, commit omissions; a process may stop or take an incorrect step.

Fault model

Timing failures

Class of Failure	*Affects*	*Description*
Clock	Process	Process's local clock exceeds the bounds on its rate of drift from real time.
Performance	Process	Process exceeds the bounds on the interval between two steps.
Performance	Channel	A message's transmission takes longer than the stated bound.

Distributed Computing System Models Interconnections

- Mini Computer Model
- Workstation Model
- Workstation-Server Model
- Processor-Pool Model
- Hybrid Model

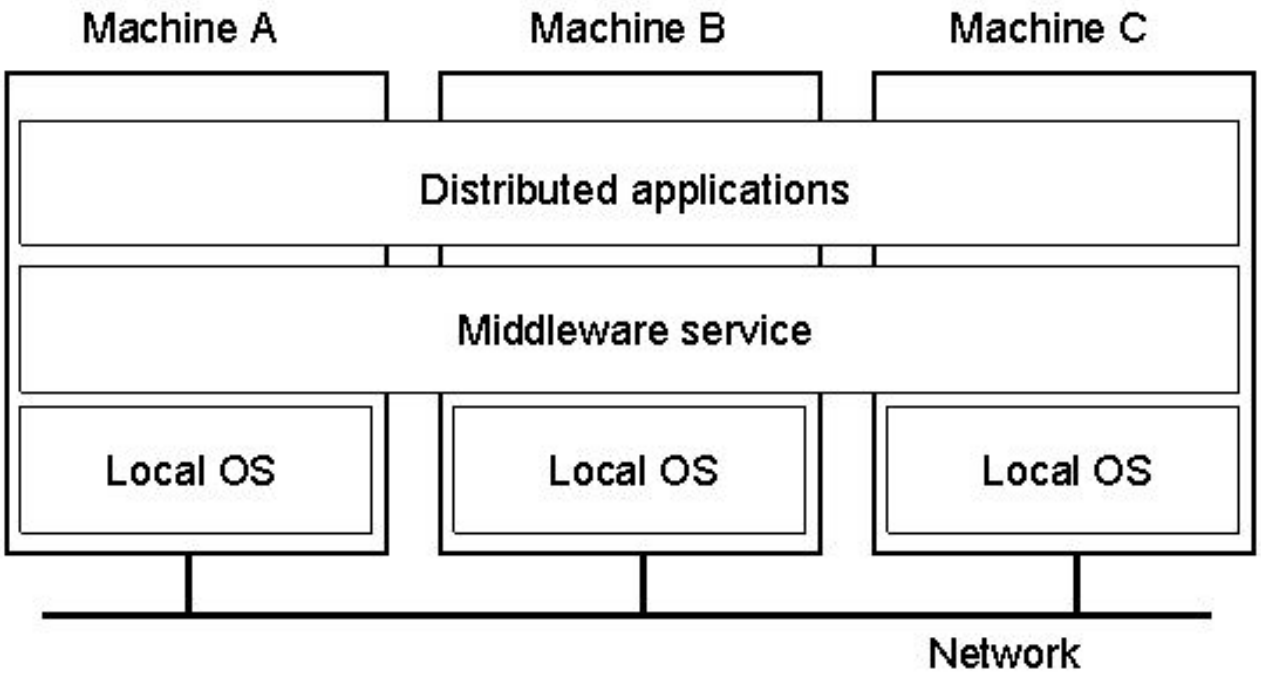
Software Concepts

System	Description	Main Goal
DOS	Tightly-coupled operating system for multi-processors and homogeneous multicomputers	Hide and manage hardware resources
NOS	Loosely-coupled operating system for heterogeneous multicomputers (LAN and WAN)	Offer local services to remote clients
Middleware	Additional layer atop of NOS implementing general-purpose services	Provide distribution transparency

An overview between

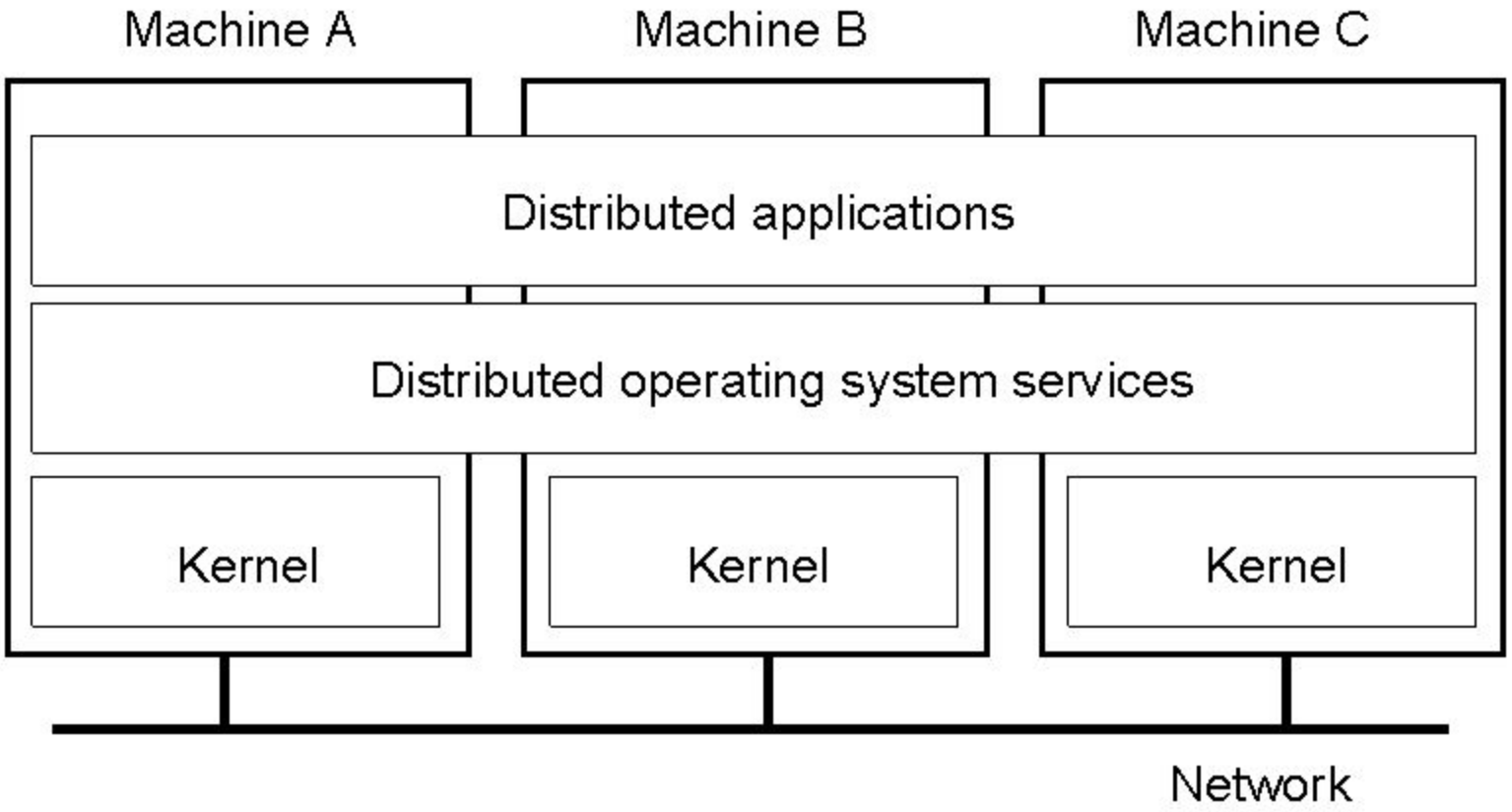
- DOS (Distributed Operating Systems)
- NOS (Network Operating Systems)
- Middleware

Definition of a Distributed System (2)



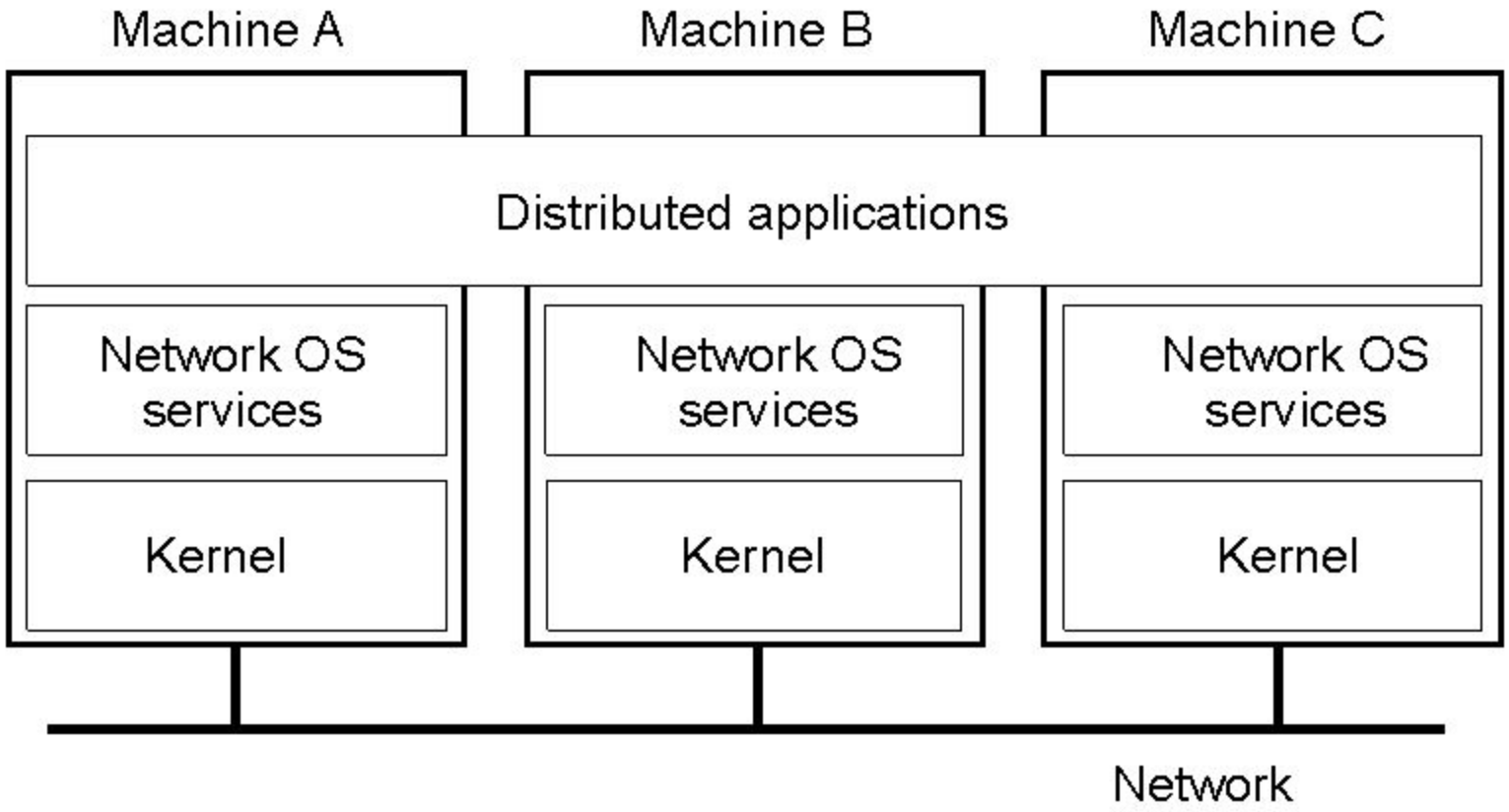
A distributed system organized as middleware.
Note that the middleware layer extends over multiple machines.

Multicomputer Operating Systems (1)



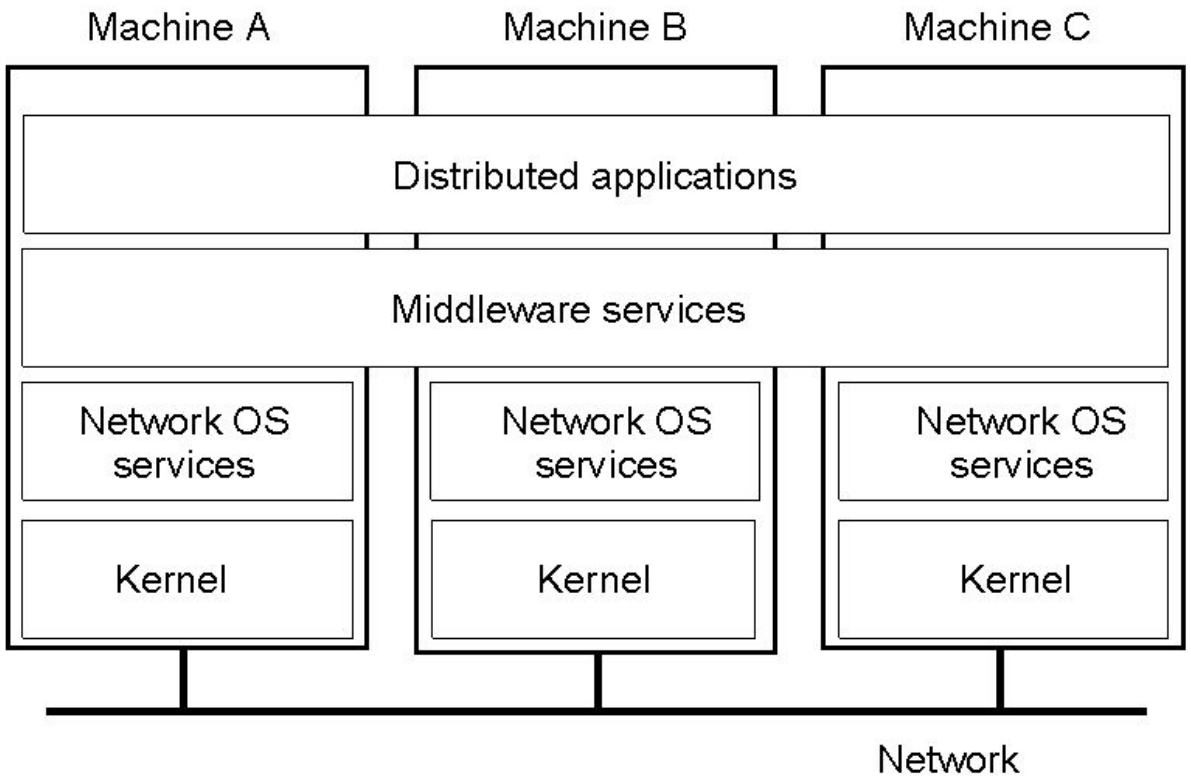
General structure of a multicomputer operating system

Network Operating System (1)



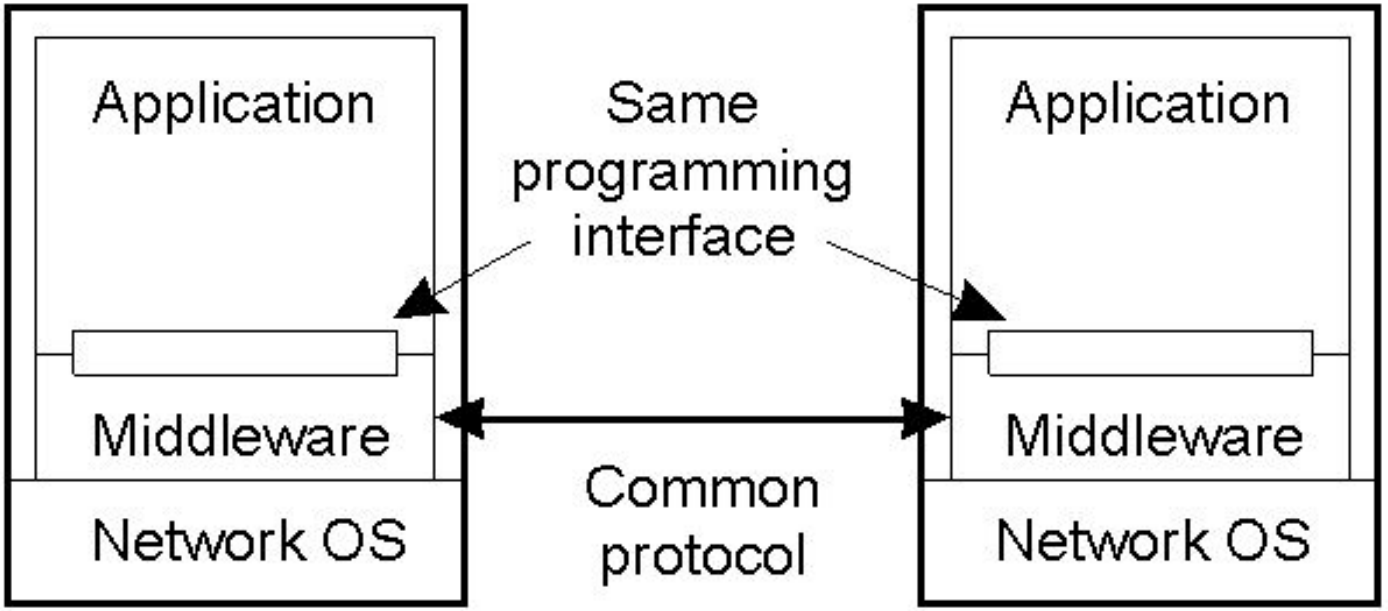
General structure of a network operating system.

Positioning Middleware



General structure of a distributed system as middleware.

Middleware



In an open middleware-based distributed system, the protocols used by each middleware layer should be the same, as well as the interfaces they offer to applications.

Comparison between Systems

Item	Distributed OS		Network OS	Middleware-based OS
	Multiproc.	Multicomp.		
Degree of transparency	Very High	High	Low	High
Same OS on all nodes	Yes	Yes	No	No
Number of copies of OS	1	N	N	N
Basis for communication	Shared memory	Messages	Files	Model specific
Resource management	Global, central	Global, distributed	Per node	Per node
Scalability	No	Moderately	Yes	Varies
Openness	Closed	Closed	Open	Open

A comparison between multiprocessor operating systems, multicomputer operating systems, network operating systems, and middleware based distributed systems.

Network Operating system :(supports 2 tier architecture)

Only follow RPC,RMI No application server involved

1)Windows NT (2000,XP)

2)Ubuntu

3)Linux

4) Windows 2003, Windows 2000 Advanced Server , Windows Server 2003

Distributed Operating system: (supports 3 tier architecture)

There are numerous technologies and standards used to construct distributed computations, including some which are specially designed and optimized for that purpose, such as Remote Procedure Calls (RPC) or Remote Method Invocation (RMI) or .NET Remoting.

1). Windows 2003 + application server(IIS&PWS)

2)Ubuntu

3)Linux(apache server)

http://www.cs.wichita.edu/~chang/lecture/cs843/homework/dist-os.html

Amoeba

MAC

Thank You