

# Distributed System

---

Prepared by: Thoại Nam

Lectured by: Phạm Trần Vũ



# Resources

---

## □ Textbook:

- Distributed Systems: Concepts and Design, Edition 4, © Addison Wesley 2005, by George Coulouris, Jean Dillimore and Tim Kindberg



# Introduction

---

- ❑ Distributed Systems
- ❑ Hardware & software
- ❑ Transparency
- ❑ Scalability
- ❑ Distributed OS



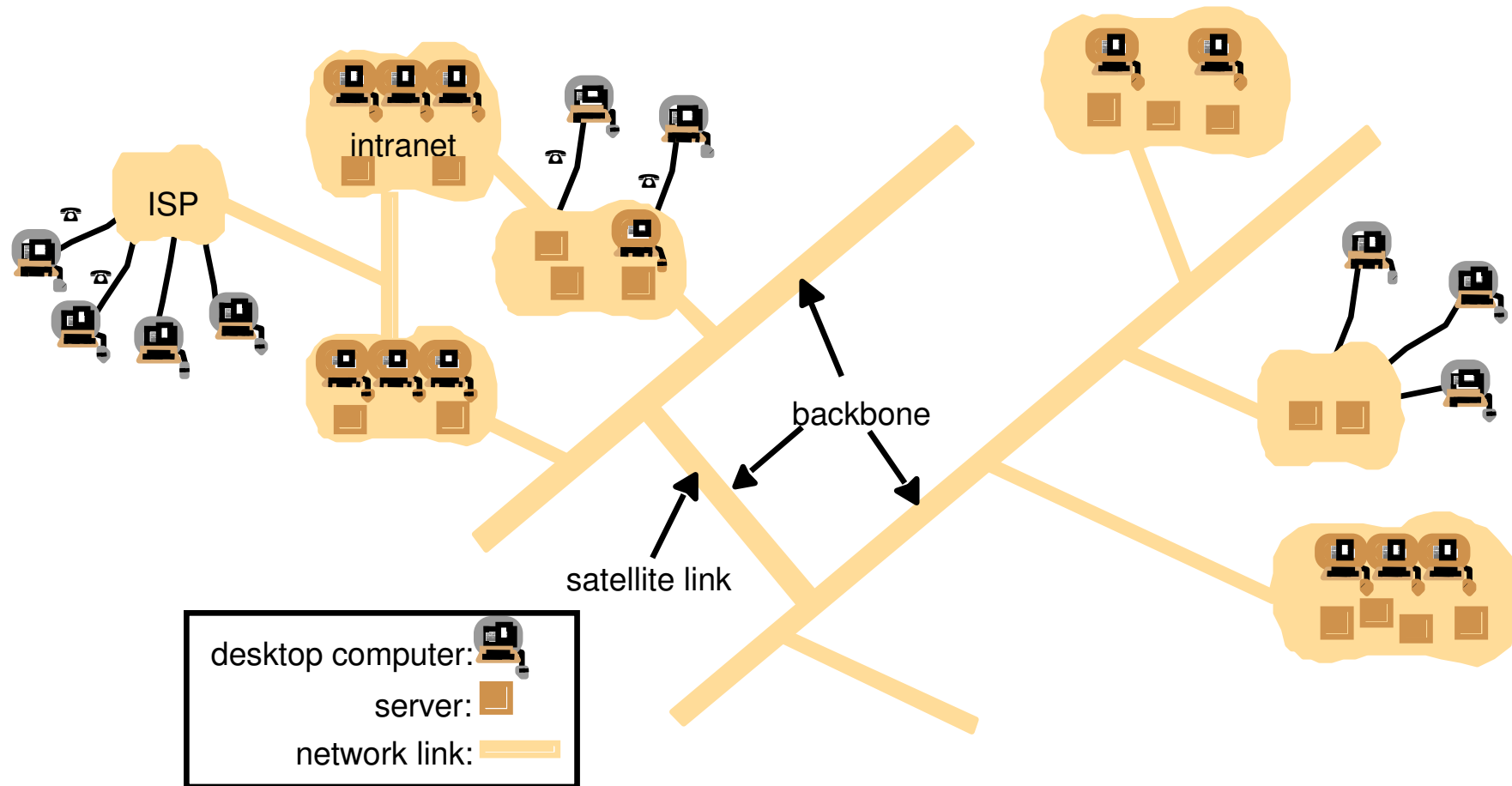
# Definition of a Distributed System

---

- A distributed system:
  - Multiple connected CPUs working together
  - A collection of independent computers that appears to its users as a single coherent system
- Examples: parallel machines, networked machines

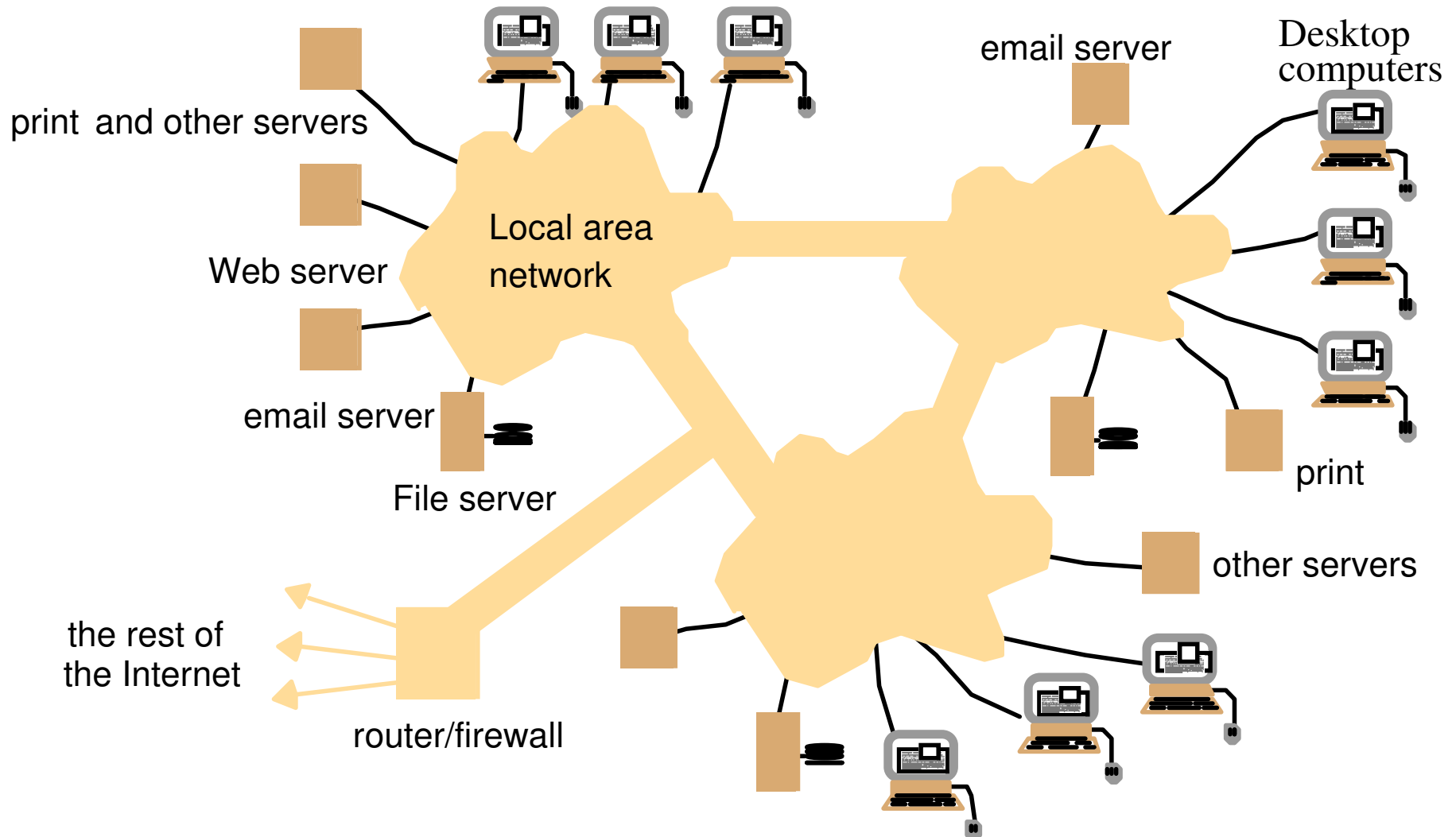


# Distributed System Examples: The Internet



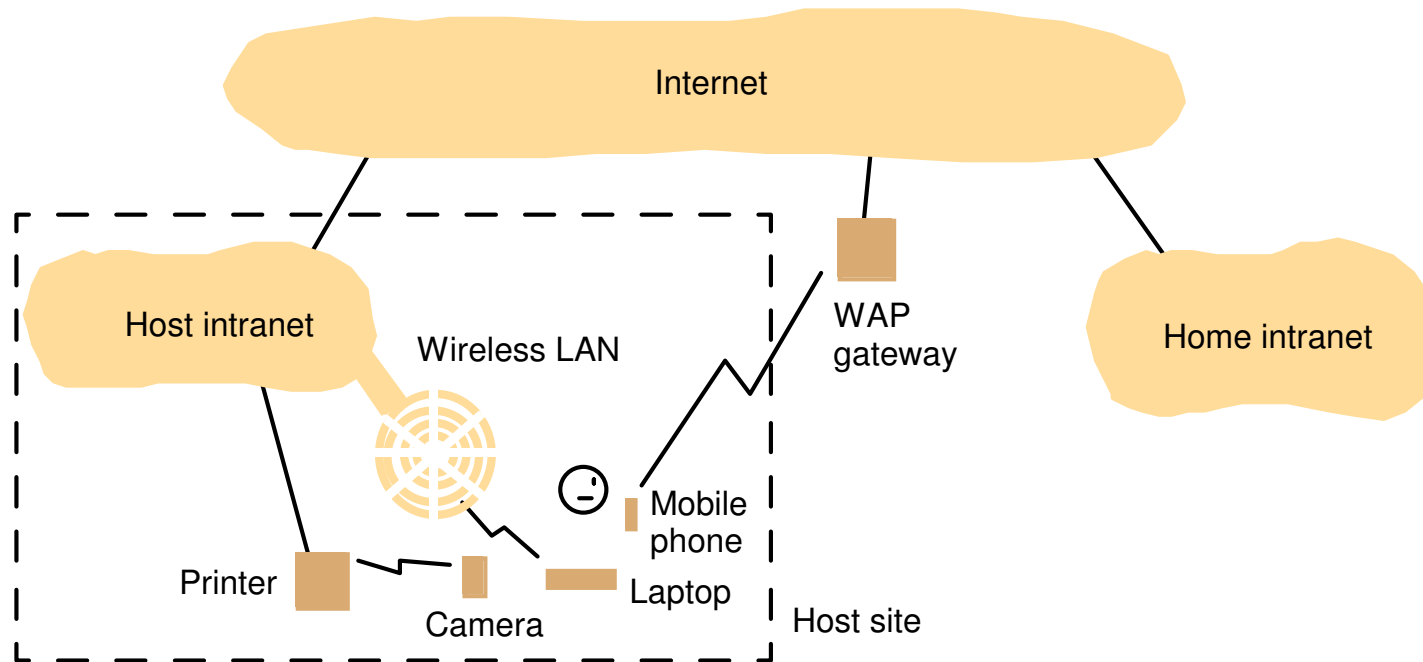


# Distributed Systems Examples: Intranets





# Distributed System Examples: Systems of Portable and Handheld Devices





# Computers vs. Web Servers in the Internet

---

<i>Date</i>	<i>Computers</i>	<i>Web servers</i>	<i>Percentage</i>
1993, July	1,776,000	130	0.008
1995, July	6,642,000	23,500	0.4
1997, July	19,540,000	1,203,096	6
1999, July	56,218,000	6,598,697	12
2001, July	125,888,197	31,299,592	25
		42,298,371	

---





# Potentials and Challenges

---

## □ Potentials

- Communication and resource sharing possible
- Economics – price-performance ratio
- Reliability, scalability
- Potential for incremental growth

## □ Challenges

- Heterogeneity of resources
- Distribution-aware PLs, OSs and applications
- Design of scalable systems
- Network connectivity essential
- Security and privacy
- Failure handling
- Concurrency



# 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
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 Problems

---

<b>Concept</b>	<b>Example</b>
Centralized services	A single server for all users
Centralized data	A single on-line telephone book
Centralized algorithms	Doing routing based on complete information

Examples of scalability limitations.



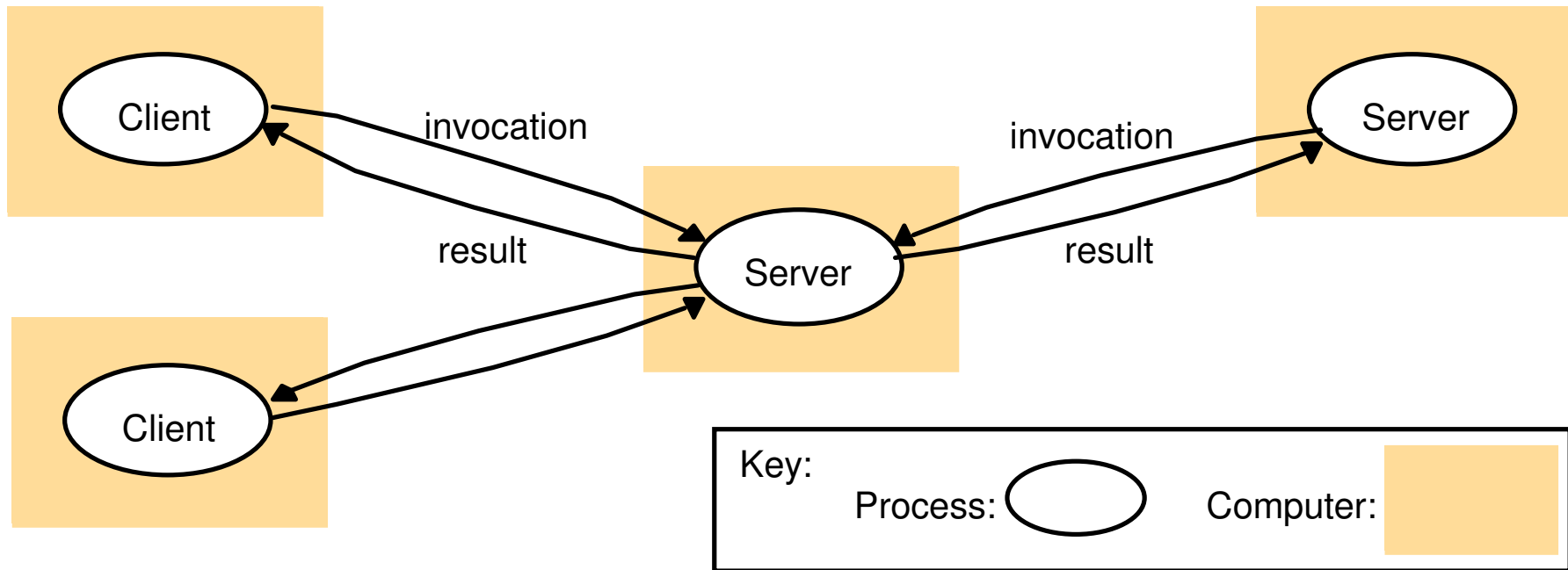
# Distributed Systems Models

---

- ❑ Minicomputer model
  - Each user has local machine
  - Local processing but can fetch remote data (files, databases)
- ❑ Workstation model
  - Processing can also migrate
- ❑ Client-server Model
  - User has local workstation
  - Powerful workstations serve as servers (file, print, DB servers)
- ❑ Peer-to-Peer
  - No distinction in the roles of servers and clients
- ❑ Processor pool model
  - Terminals are Xterms or diskless terminals
  - Pool of backend processors handle processing

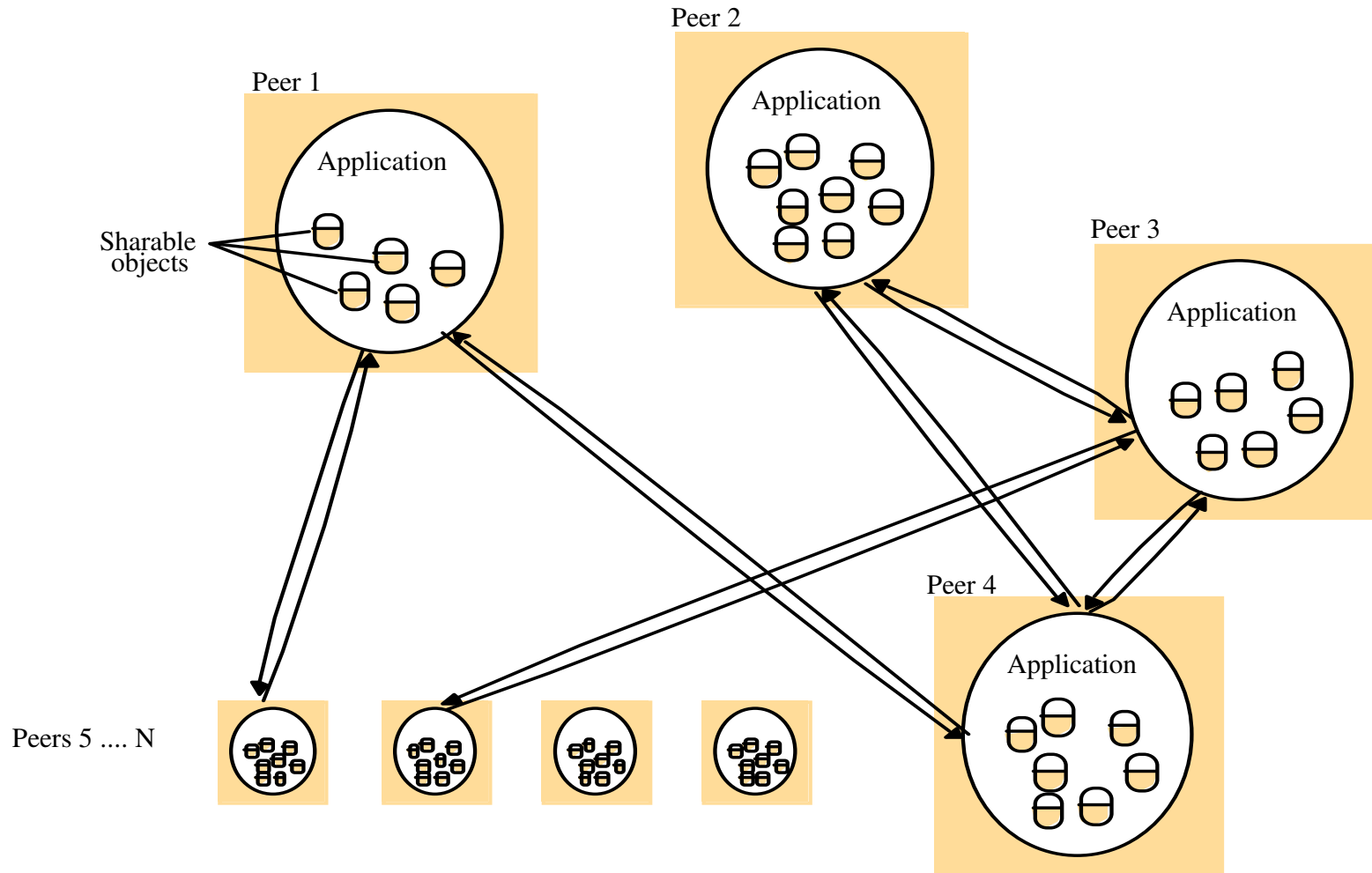


# Client-Server Model





# Peer-to-Peer Model





# Uniprocessor Operating Systems

---

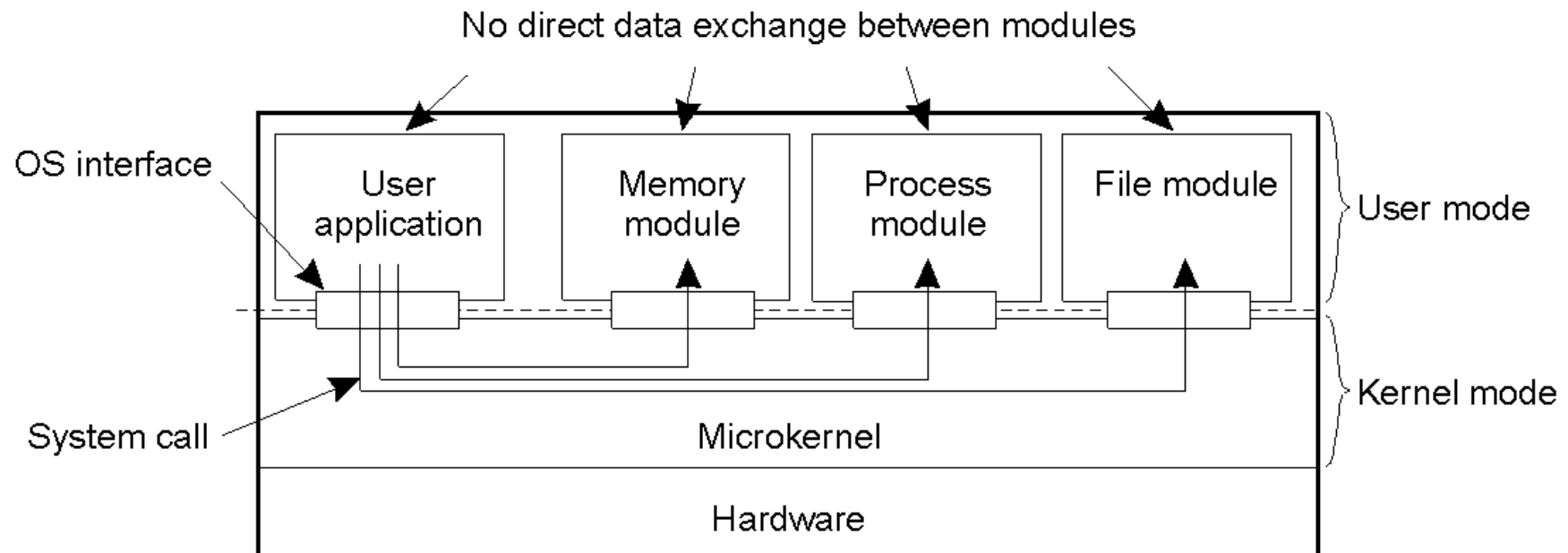
- ❑ An OS acts as a resource manager or an arbitrator
  - Manages CPU, I/O devices, memory
- ❑ OS provides a virtual interface that is easier to use than hardware
- ❑ Structure of uniprocessor operating systems
  - Monolithic (e.g., MS-DOS, early UNIX)
    - » One large kernel that handles everything
  - Layered design
    - » Functionality is decomposed into N layers
    - » Each layer uses services of layer N-1 and implements new service(s) for layer N+1



# Uniprocessor Operating Systems

## Microkernel architecture

- Small kernel
- user-level servers implement additional functionality







# Distributed Operating System

---

- ❑ Manages resources in a distributed system
  - Seamlessly and transparently to the user
- ❑ Looks to the user like a centralized OS
  - But operates on multiple independent CPUs
- ❑ Provides transparency
  - Location, migration, concurrency, replication,...
- ❑ Presents users with a virtual uniprocessor



# Types of Distributed OSs

<b>System</b>	<b>Description</b>	<b>Main Goal</b>
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



# Multiprocessor Operating Systems

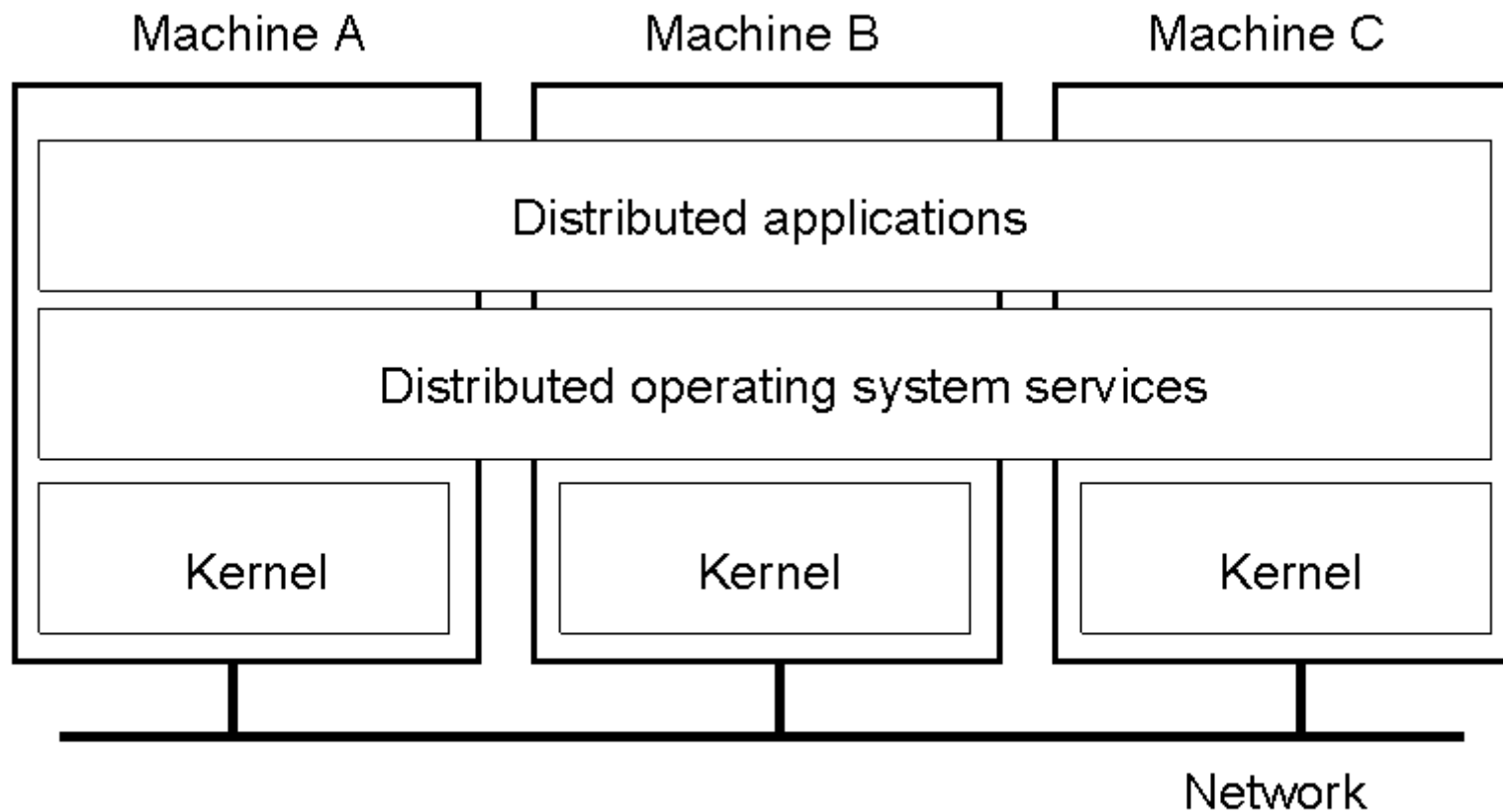
---

- ❑ Like a uniprocessor operating system
- ❑ Manages multiple CPUs transparently to the user
- ❑ Each processor has its own hardware cache
  - Maintain consistency of cached data



# Multicomputer Operating Systems

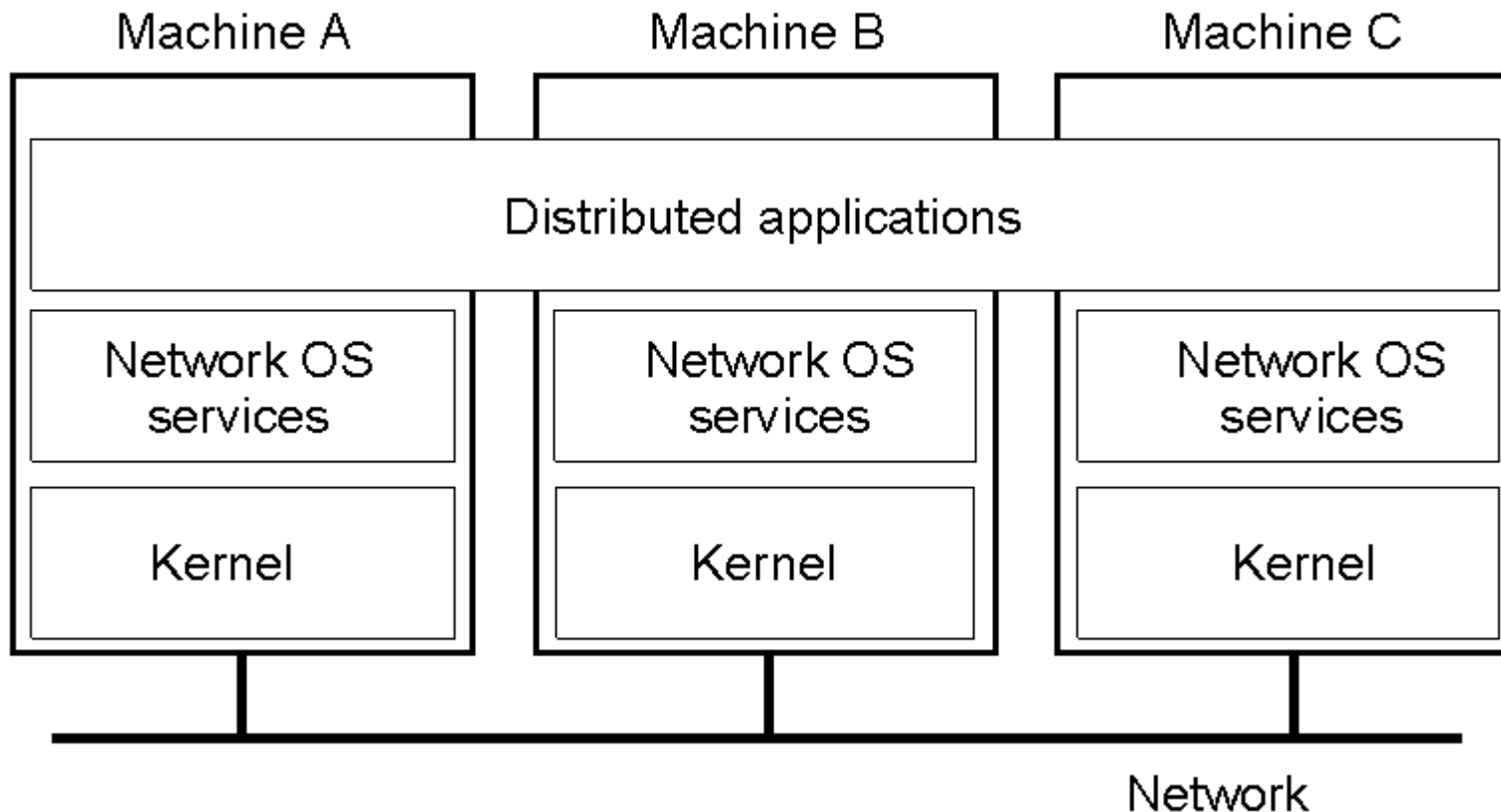
---





# Network Operating System (1)

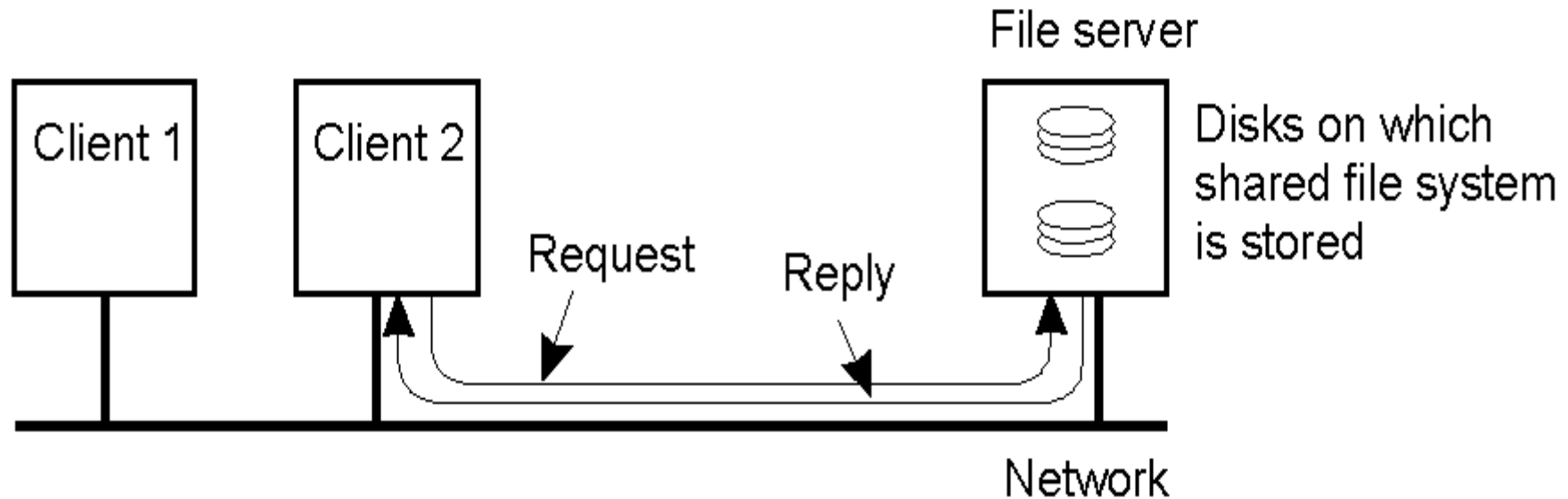
---





## Network Operating System (2)

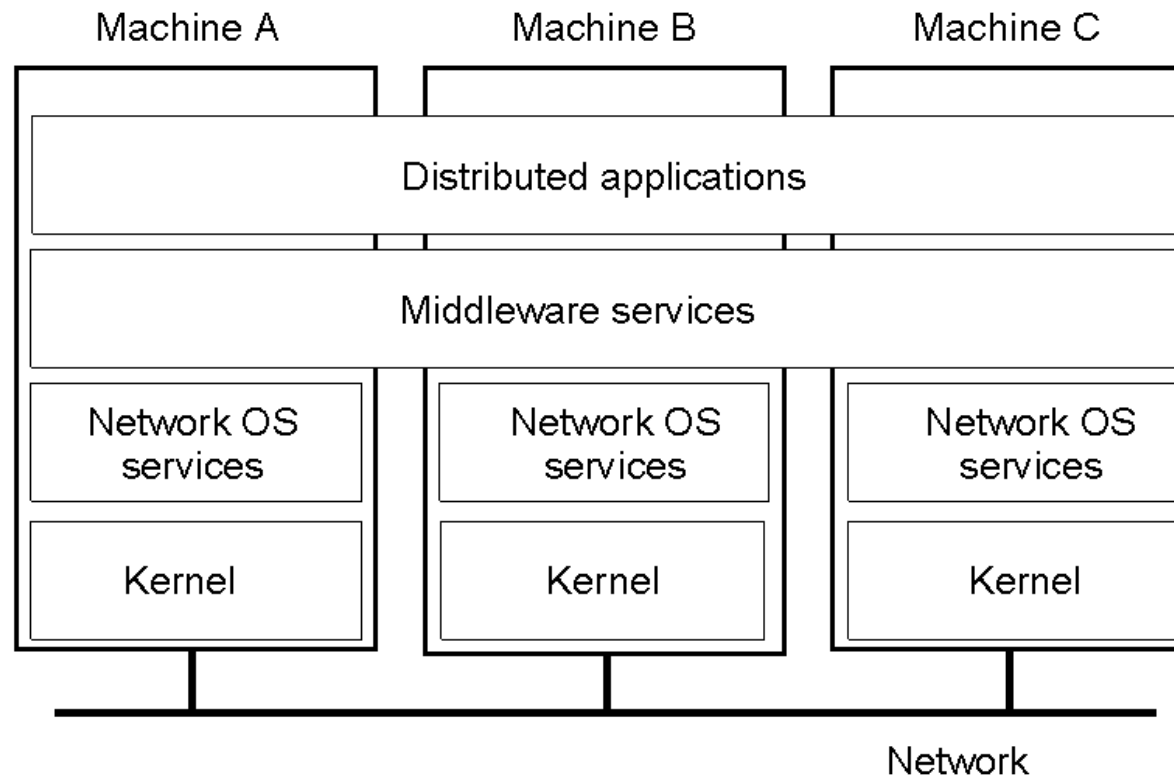
- Employs a client-server model
  - Minimal OS kernel
  - Additional functionality as user processes





# Middleware-based Systems

- General structure of a distributed system as middleware.





# 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