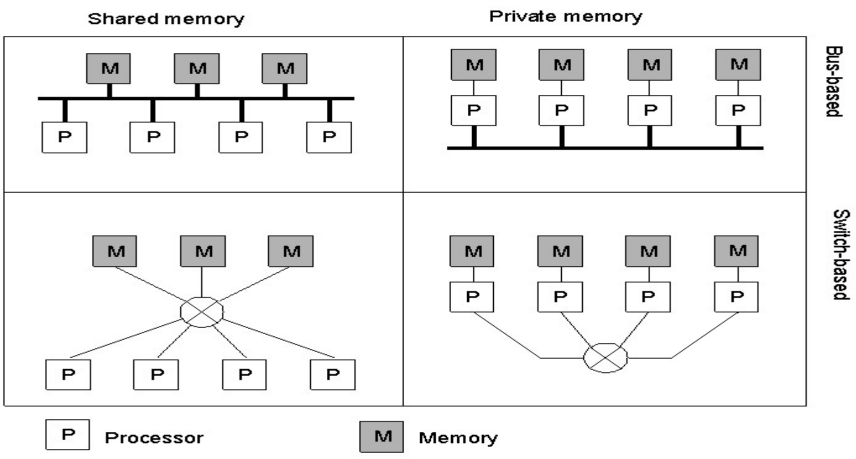
CSIE52400/CSIEM0140 Distributed Systems

Lecture 02 Basic Concepts

Shiow-yang Wu (吳秀陽)
Department of Computer Science and Information Engineering
National Dong Hwa University

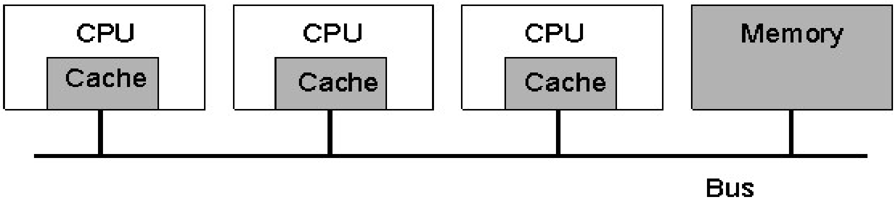
Hardware Concepts



Different basic organizations and memories in distributed computer systems

Multiprocessors (1)

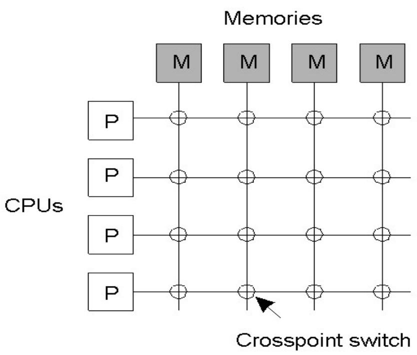
- A bus-based multiprocessor.



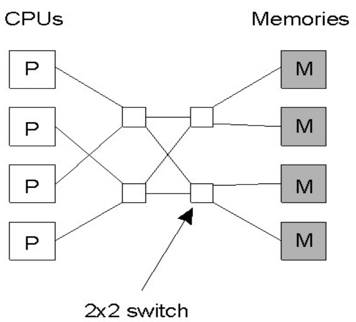
Problem: scalability

Multiprocessors (2)

a) A crossbar switch
b) An omega switching network



(a)

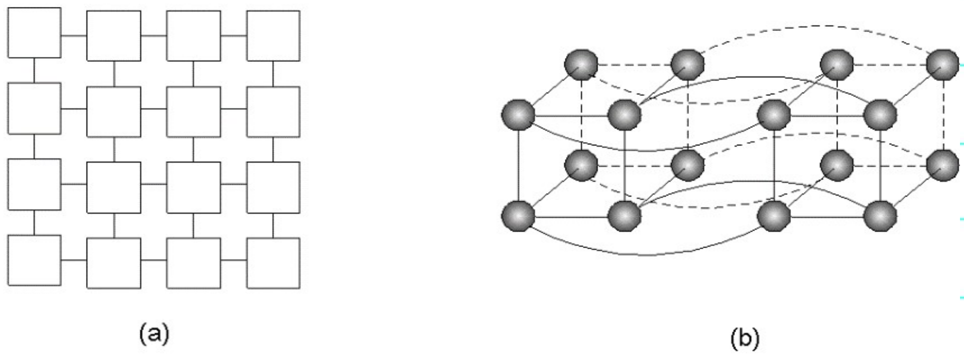


(b)

Problem: need n^2 crosspoint **Problem**: need several stages

Homogeneous Multicomputer Systems

- Multicomputer systems with identical nodes
  a) Grid
  b) Hypercube

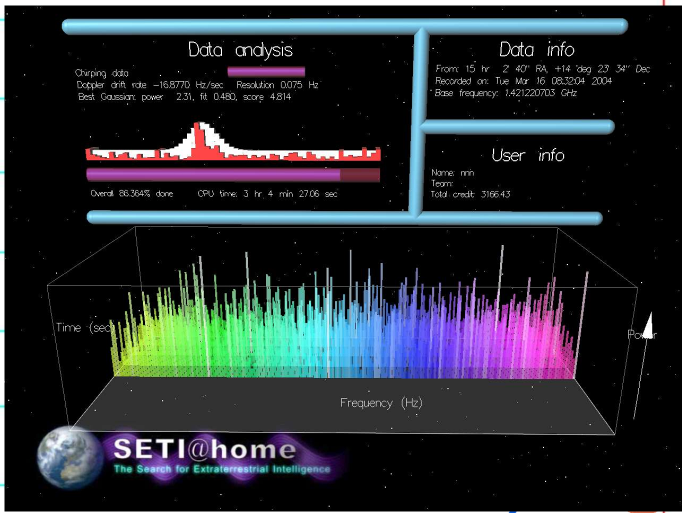


Heterogeneous Multicomputer Systems

- The nodes that form the system may vary widely.
- The interconnection network may be heterogeneous.
- Large scale multicomputer using existing networks and backbones.
- Examples: Grid computing, SETI@home
- Need sophisticated software for such systems.

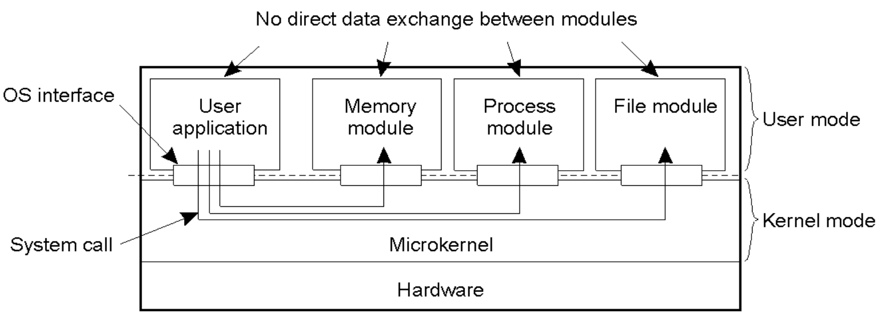
SETI@home

- **S**earch for **E**xtra-**T**errestrial **I**ntelligence
- Users participate by running a free program that downloads and analyzes radio telescope data.
- On 2020/03/31, the project stops sending out new work to users.

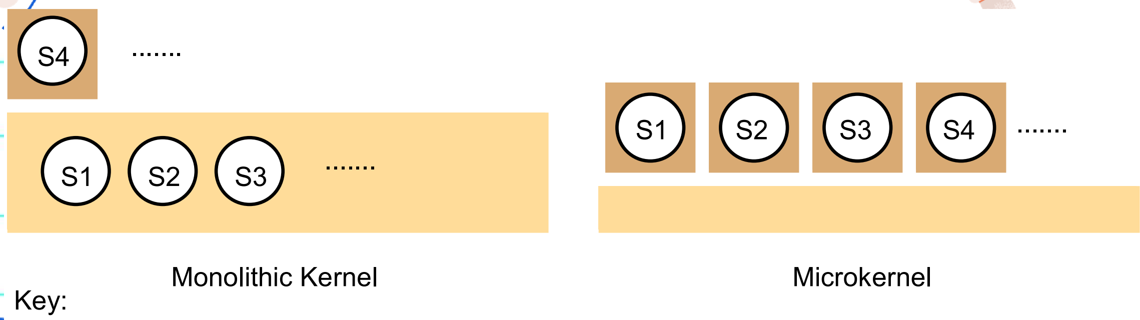


Uniprocessor Operating Systems

- Uniprocessor operating system
  - Implement a virtual machine
  - Two modes of operation: kernel mode vs. user mode
  - Almost all system code in kernel mode ⇒ hard to adapt
- A microkernel approach:



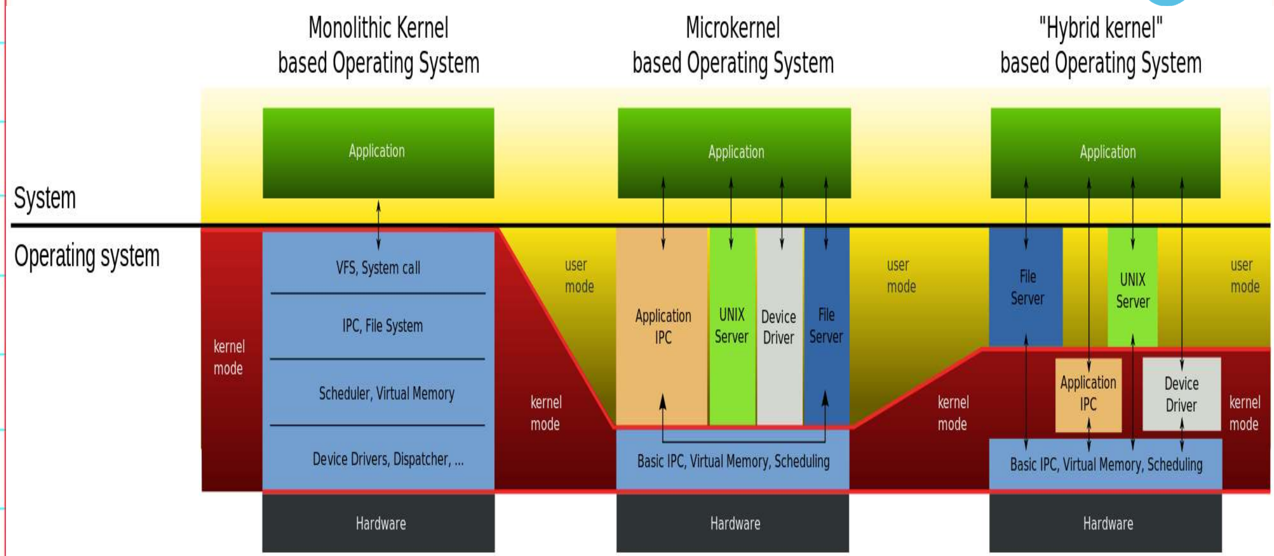
Monolithic Kernel and Microkernel



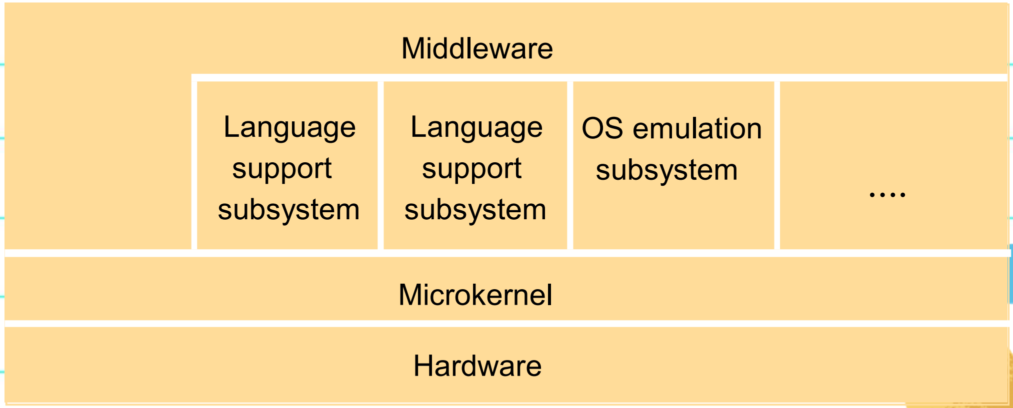
Key:

Server: ◯ Kernel code and data: Dynamically loaded server program:

Structures of OSs



The Role of the Microkernel



The microkernel supports middleware via subsystems.

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

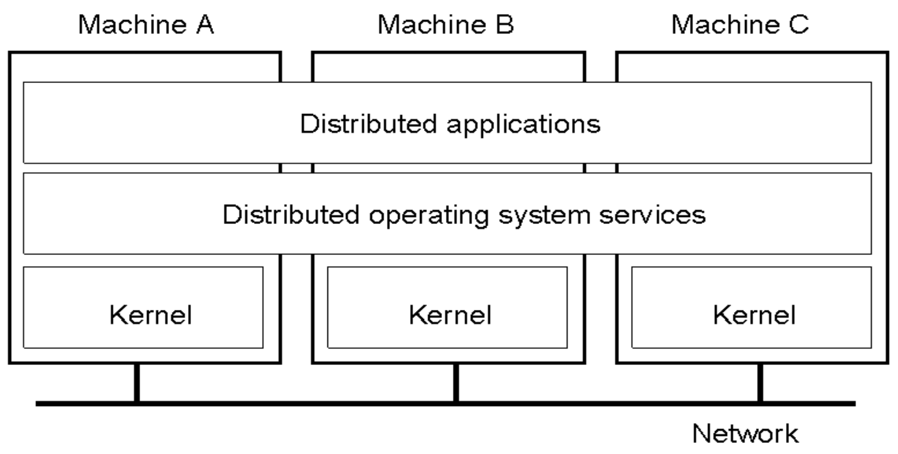
System	Description	Main Goal
DOS	Tightly-coupled operating system for multi-processors and homogeneous multicomputers	Hide and manage hardware resources
NOS	Loosely-coupled operating system for heterogeneous multicomputers (LAN and WAN)	Offer local services to remote clients
Middleware	Additional layer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

Multiprocessor Operating Systems

- Need to handle multiple CPUs and synchronize concurrent access
- Synchronization primitives: semaphore, monitor, locks, …
- Locks are low level primitives.
- Semaphores are error-prone.
- Monitors are easier to deal with.
- . . .

Multicomputer Operating Systems

- General structure of a multicomputer operating system



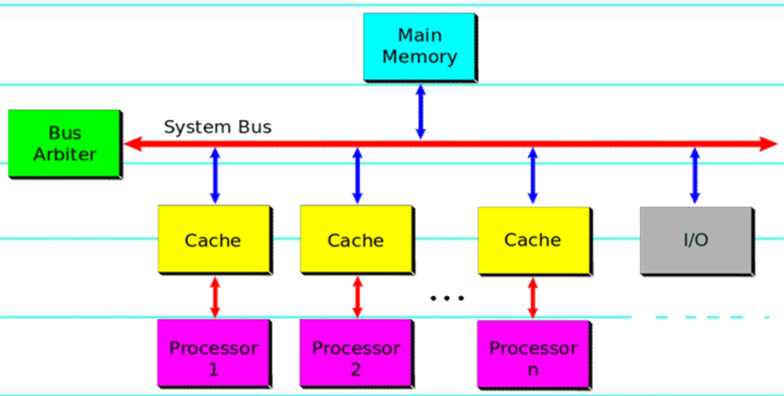
Types of Multiprocessor OS

- Symmetric multiprocessor
- Asymmetric multiprocessor
- Shared memory multiprocessor
- Distributed memory multiprocessor
- Uniform memory access multiprocessor
- Non uniform memory access multiprocessor

Symmetric Multiprocessor

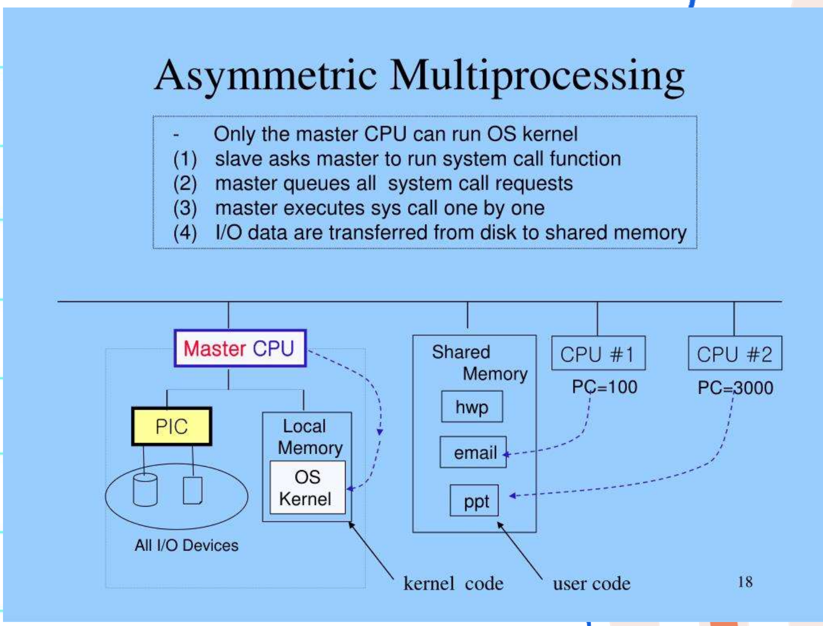
- One OS can use all the CPUs (allowing several tasks to be performed simultaneously).

SMP - Symmetric Multiprocessor System



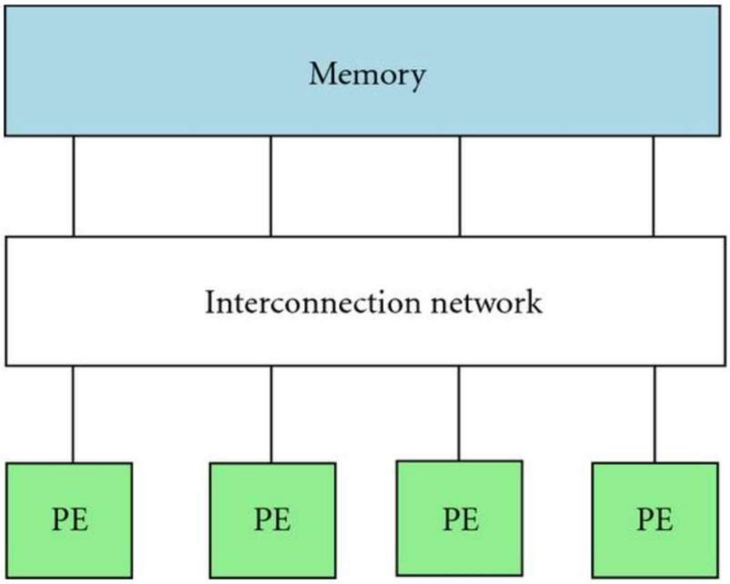
Asymmetric Multiprocessor

- Processors play different roles(mostly master-slave)



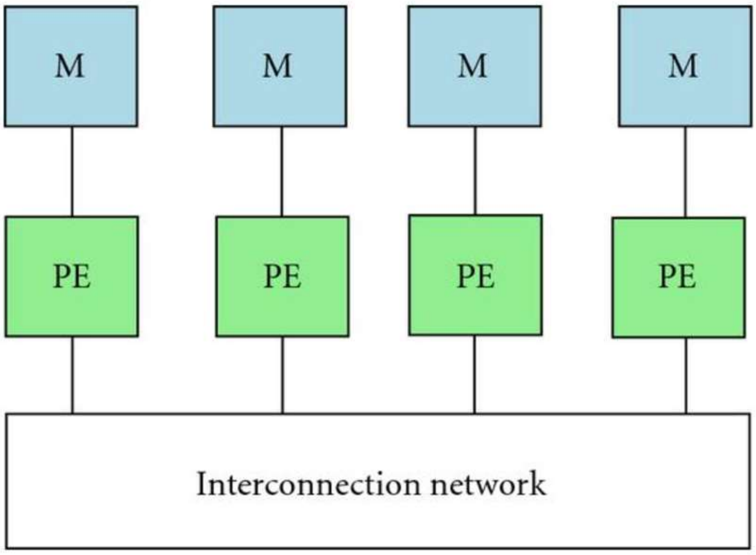
Shared Memory Multiprocessor

- A shared memory to all processors



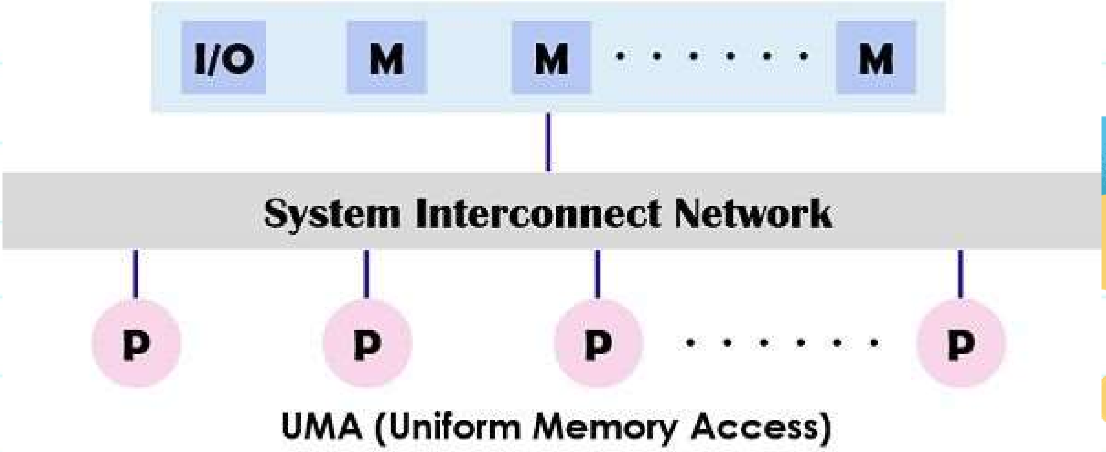
Distributed Memory Multiprocessor

- Each processor has its own memory.



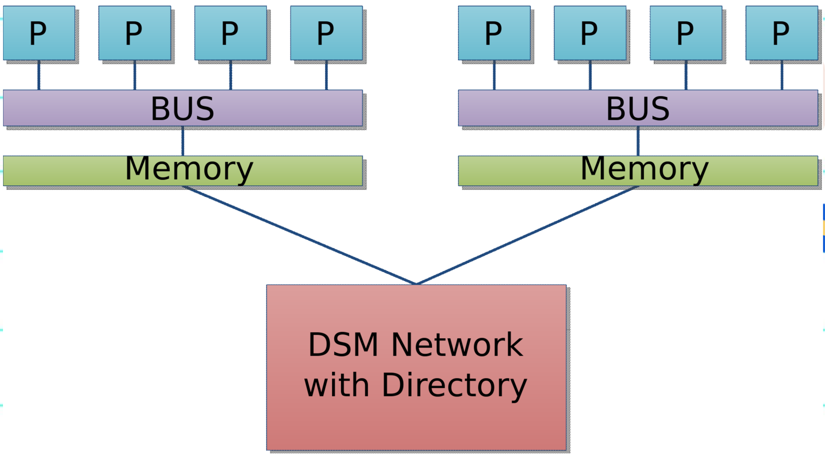
UMA(Uniform Memory Access)

- Access to all memory at the same speed for all processors



NUMA(Non-Uniform Memory Access)

- Access to some parts of memory is faster for some processors than other.



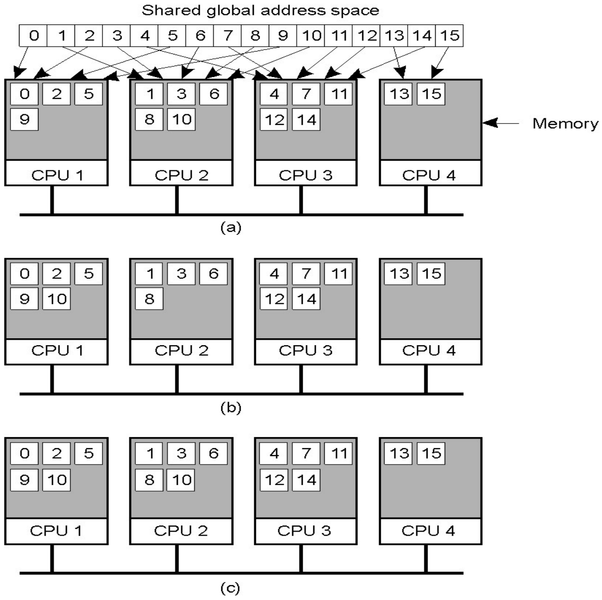
Distributed Shared Memory Systems

Page-based DSM:

a) Pages of address space distributed among four machines

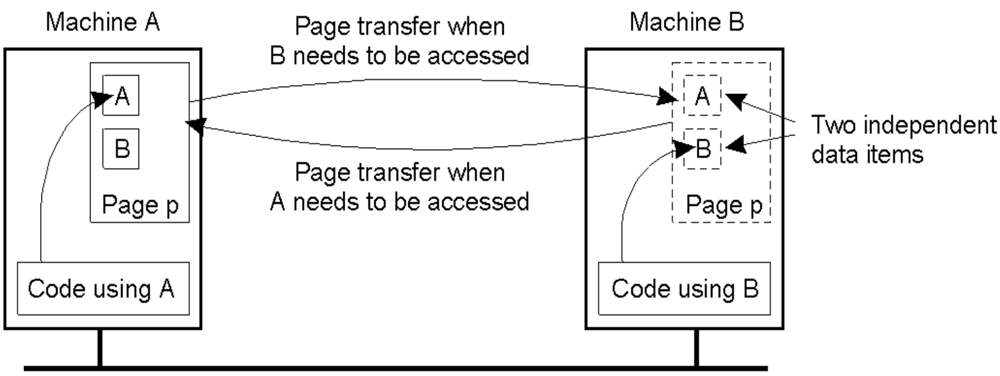
b) Situation after CPU 1 references page 10

c) Situation if page 10 is read only and replication is used



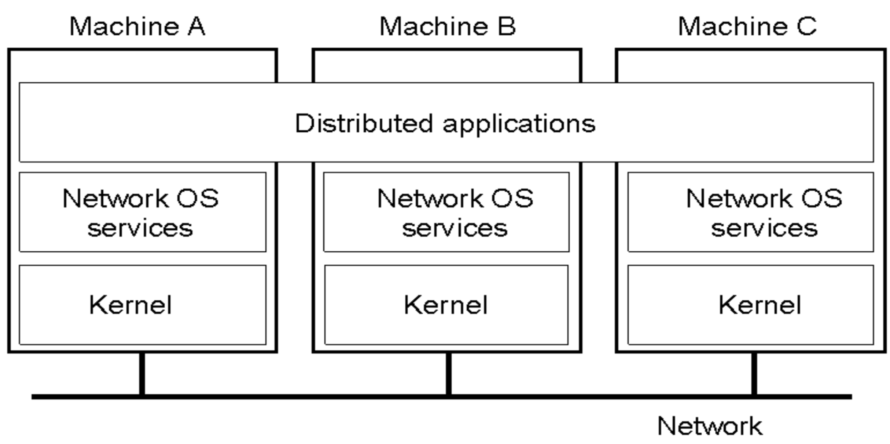
Distributed Shared Memory Systems

- Page size issue
- False sharing of a page between two independent processes.



Network Operating System

- General structure of a network operating system.
- Only primitive services such as `rlogin`, `rcp`, ...

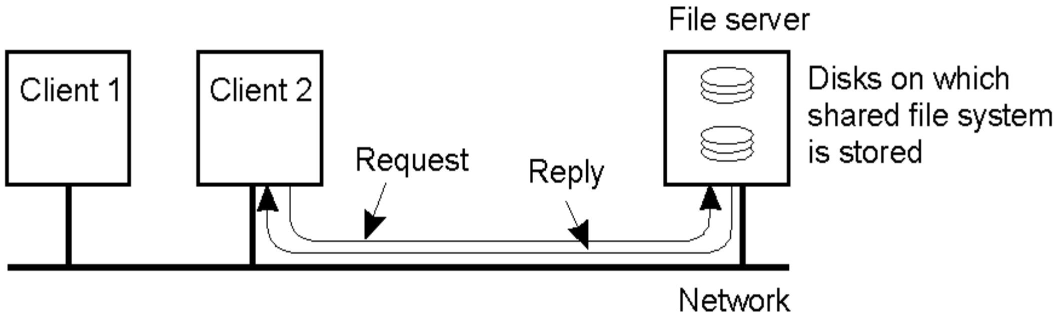


NOS by Example

- So what is a NOS? Probably easiest to describe by example.
- Suppose you have files on CSIE, but you want to work at home on your laptop. What do you do?
- Using things like FTP is inconvenient. There are things like NFS(Network File System) and CIFS(Common Internet File System). These are NOS capabilities.
- NOS does not provide true distribution transparency, but does provide convenience.

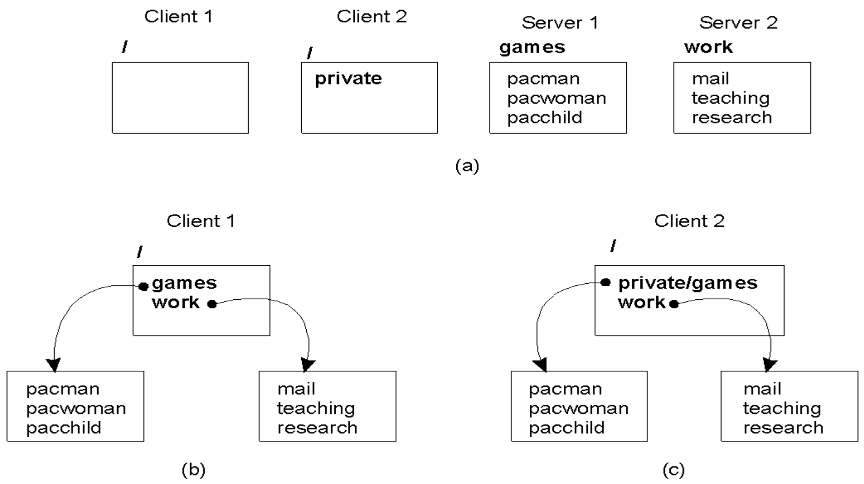
Shared Servers on NOS (1)

- Can provide better services with shared servers.
- Example: Two clients and a file server in a network operating system.

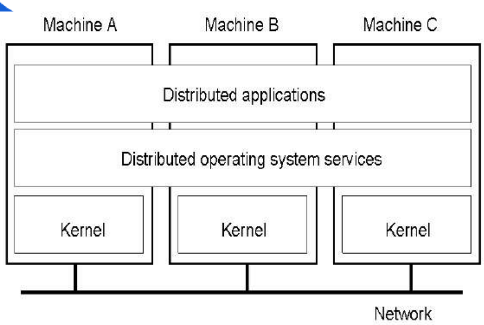


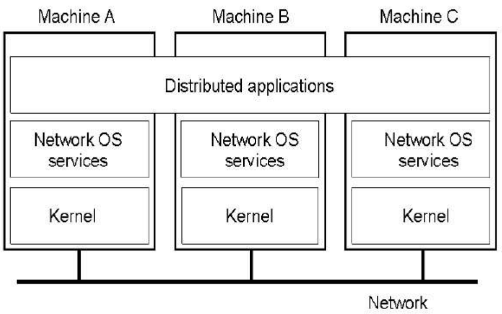
Shared Servers on NOS (2)

- Different clients may mount the servers in different places.



DOS vs. NOS





- User is not aware of the multiple CPUs.
- Each machine runs a part of the Distributed Operating System.
- The system is fault-tolerant.

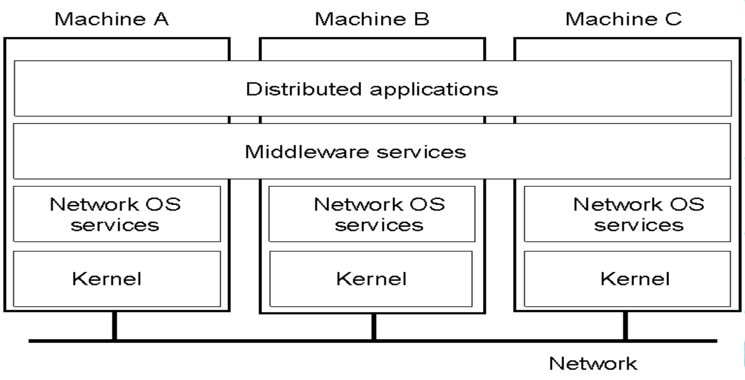
- User is aware of the existence of multiple CPUs.
- Each machine has its own private Operating System.
- The system is not fault-tolerant.

DOS vs. NOS

- What is good/bad about DOS?
  - Transparency
  - Hard to make work right
  - Lack of support for applications
  - Problems are often socio-technological.
- What is good/bad about NOS?
  - Simple.
  - Decoupled, easy to add/remove.
  - Lack of transparency.
  - No integration for your apps

Positioning Middleware

- Neither a DOS or a NOS qualifies as a true distributed system according to our definition.
- **Solution**: Structure a distributed system as **middleware**.



Middleware Models

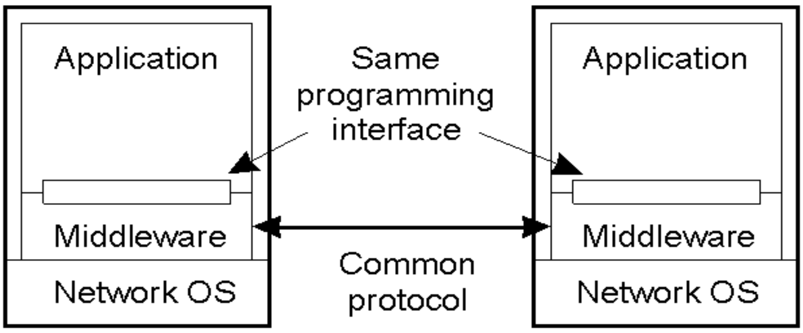
- Middleware models describe the way of distribution and communication.
- Treating everything as a file (from UNIX)
- Based on distributed file systems
- Based on Remote Procedure Calls (RPCs)
- Based on distributed objects
- Based on distributed documents

Middleware Services

- High-level communication facilities
- Naming
- Persistence
- Distributed transactions
- Security
- Replication
- Location management
- Data management

Middleware and Openness

- 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.



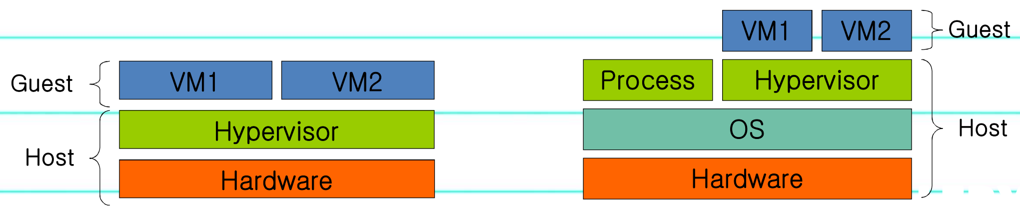
Virtualization

- Run multiple operating systems and user applications on the same hardware
  - E.g., run both Windows and Linux on the same laptop
- How is it different from **dual-boot**?
  - Both OSes run **simultaneously**
- The OSes are completely **isolated** from each other

Virtualization - Hypervisor

- **Hypervisor** (or **VMM** – Virtual Machine Monitor) is a software layer that allows several **virtual machines** to **run** on a **physical machine**
- The physical OS and hardware are called the **Host**
- The virtual machine OS and applications are called the **Guest**

Type 1 (bare-metal) **Type 2 (hosted)**



VMware ESX, Microsoft Hyper-V, Xen

VMware Workstation, Microsoft Virtual PC, Sun VirtualBox, QEMU, KVM

Comparison of 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