Introduction

Operating Systems
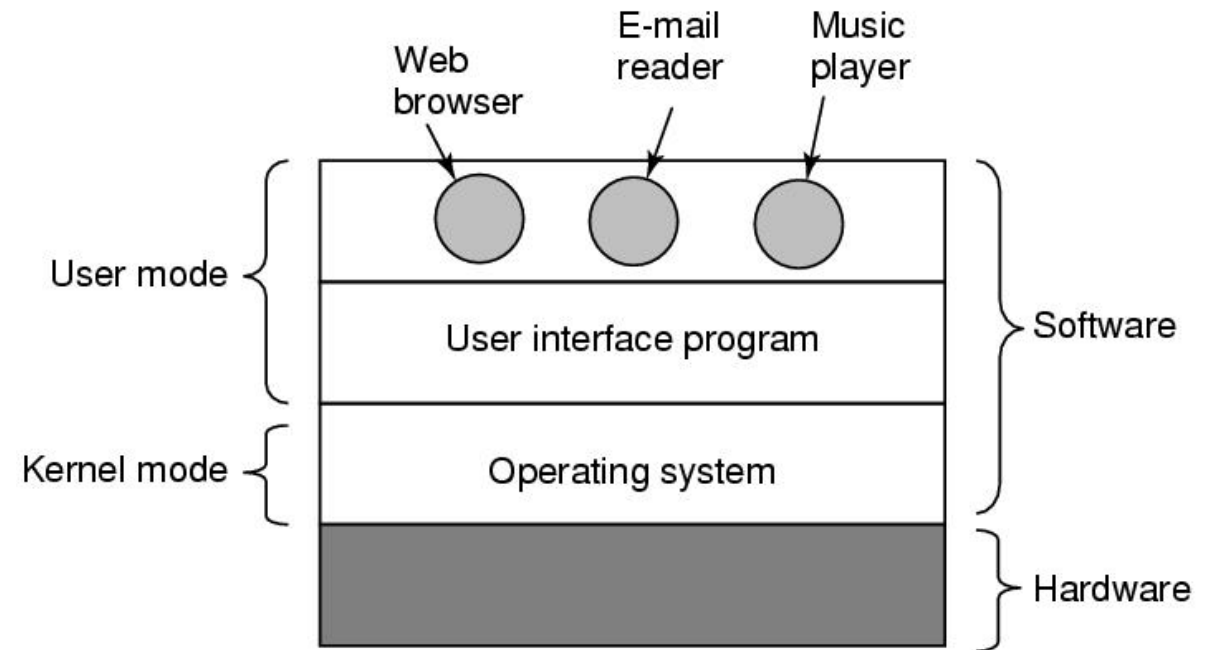
Based on Tanenbaum, Modern Operating Systems 3 e

What is an Operating System?

- Anyone?

- Give a few names of an OS?
  - Desktops?
  - Smart phones?
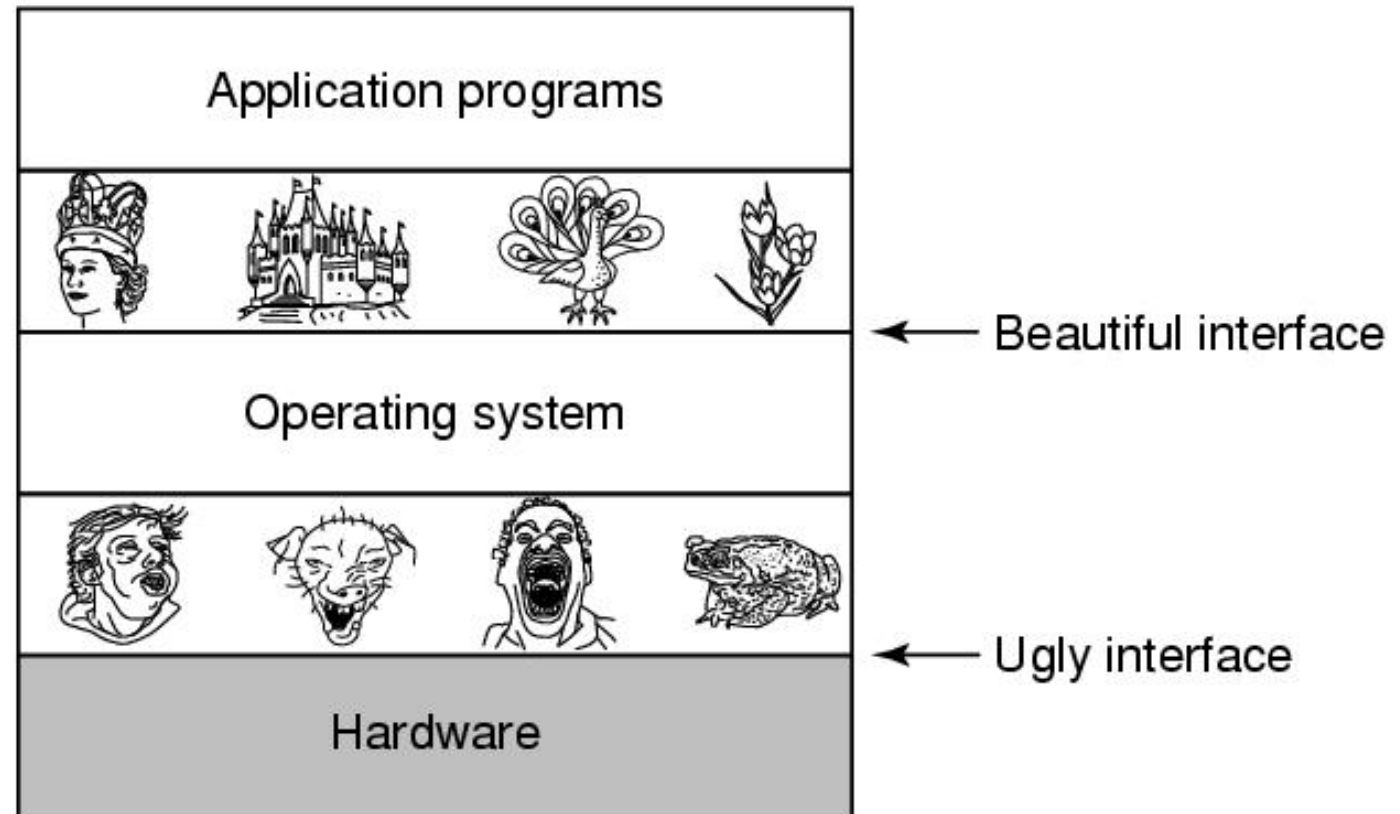
What is an Operating System?

- A modern computer consists of:
  - One or more processors
  - Main memory
  - Disks
  - Printers
  - Various input/output devices
- Managing all these components requires a layer of software
  - the **operating system**

Why learn OS?

- Great programmers understand Operating System
  - http://www.youtube.com/watch?v=-3Rt2_9d7Jg
- OS is the foundation of all software! (and it's hard to code!)
- Academic research in OS is very influential
  - What do Microsoft, Apple, VMware, or even Google sell?

Based on Tanenbaum, Modern Operating Systems 3 e

The Operating System as an Extended Machine

Operating systems turn ugly hardware into beautiful abstractions.

The Operating System as a Resource Manager

- Allow multiple programs to run at the same time

- Manage and protect memory, I/O devices, and other resources

- Includes multiplexing (sharing) resources in two different ways:
    - In time
    - In space

History of Operating Systems

Generations of Operating Systems:

1. (1945–55) Vacuum Tubes
   - Punch cards and machine language programming

2. (1955–65) Transistors and Batch Systems
   - Mainframes (IBM 1401, IBM 7094), FORTRAN compiler

3. (1965–1980) ICs and Multiprogramming
   - IBM System/360, OS/360, multiprogramming, timesharing, MULTICS

4. (1980–Present) Personal Computers
   - IBM PC, DOS, Macintosh, Windows, Linux

5. (1990–Present) Mobile Computers
   - PDAs, Symbian OS, Blackberry OS, Smartphones, iOS, Android
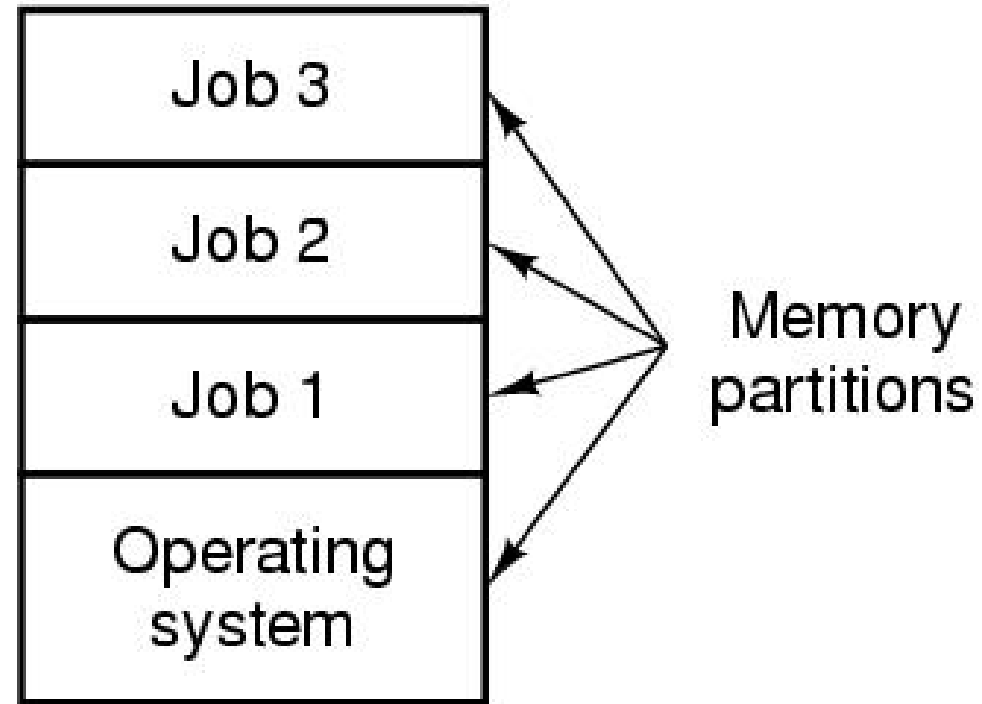
Transistors and Batch Systems
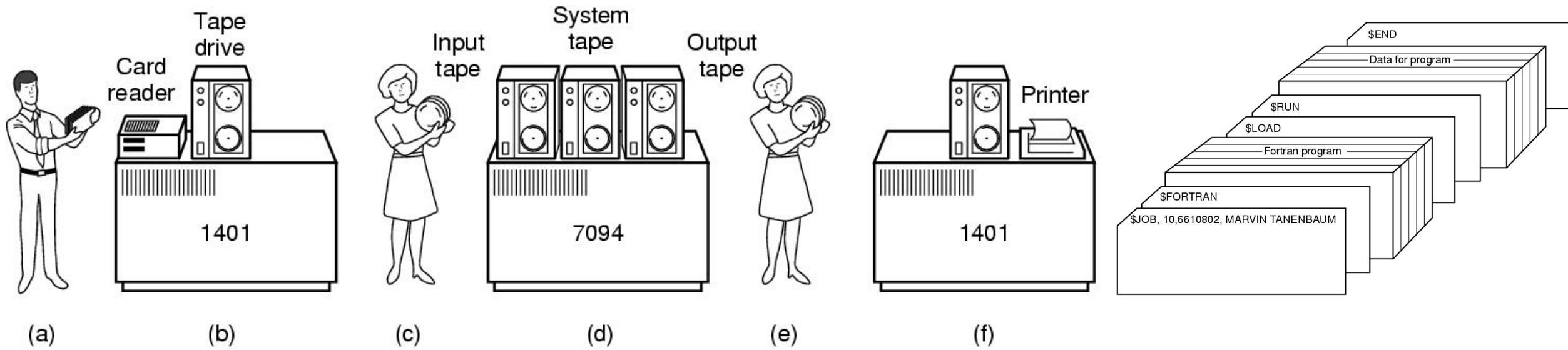
a) Programmers bring cards to 1401.

b) 1401 reads batch of jobs onto tape.

c) Operator carries input tape to 7094.

d) 7094 does computing.

e) Operator carries output tape to 1401.
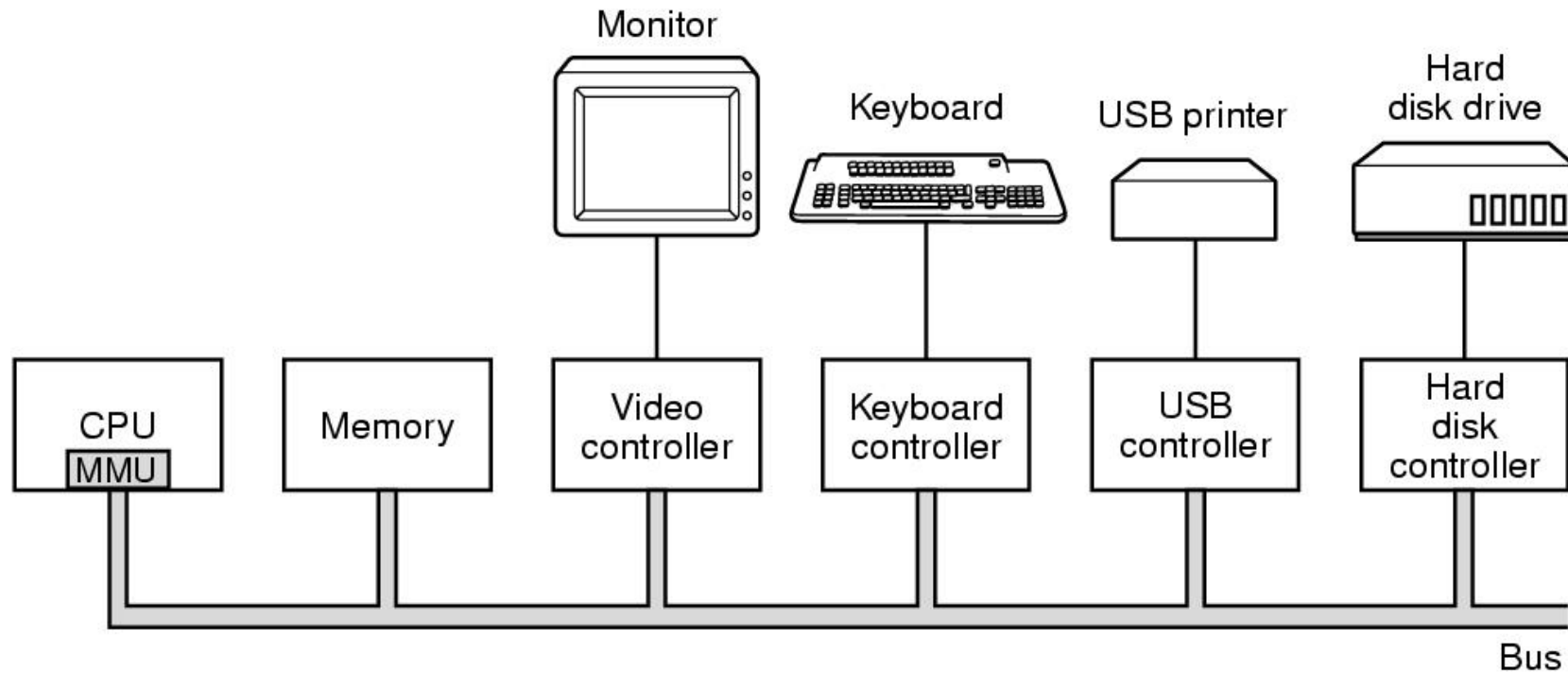
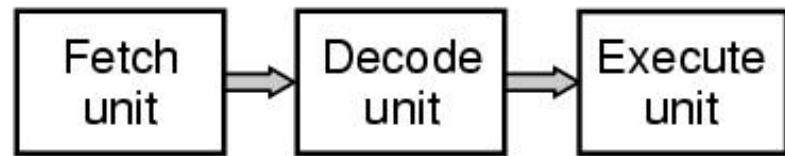f) 1401 prints output.

ICs and Multiprogramming

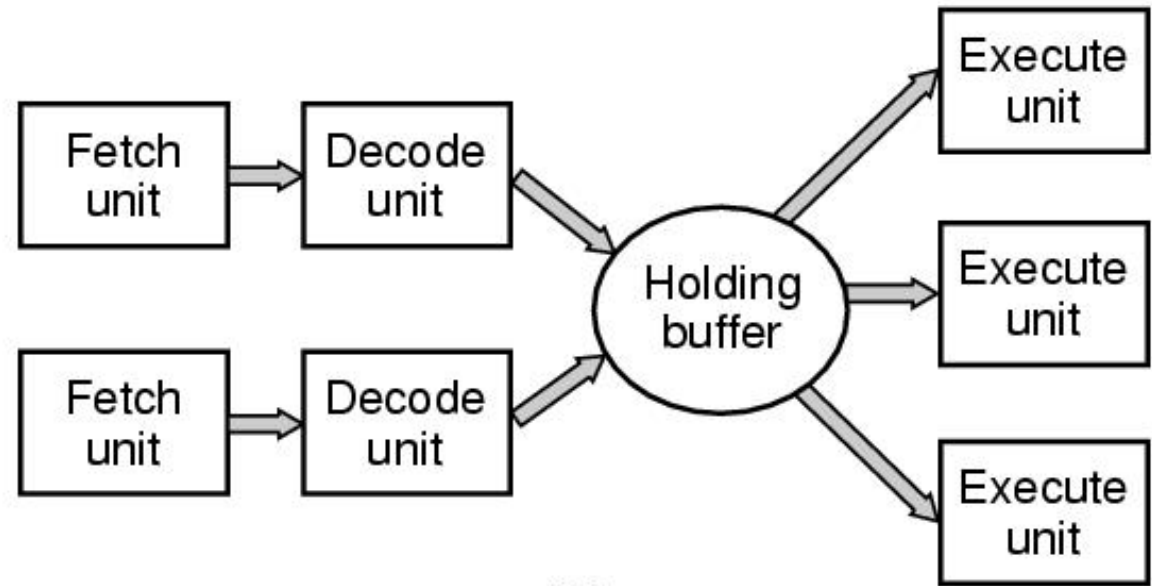A multiprogramming system with three jobs in memory.

Computer Hardware Review

Some of the components of a simple personal computer.

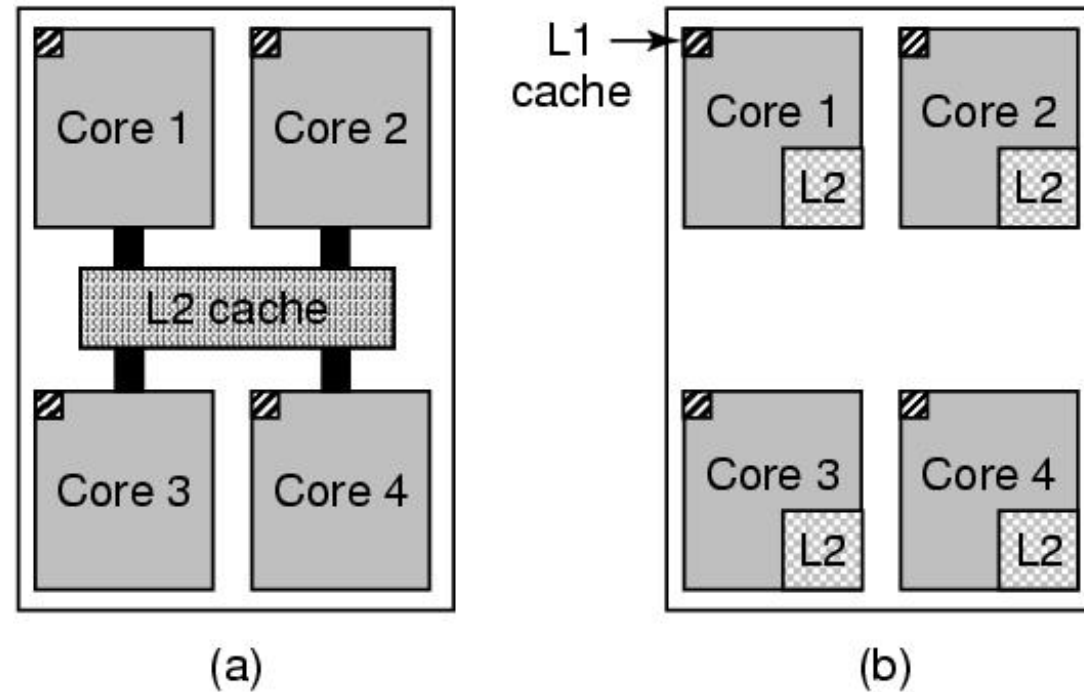# CPU Pipelining

(a) A three-stage pipeline. (b) A superscalar CPU.

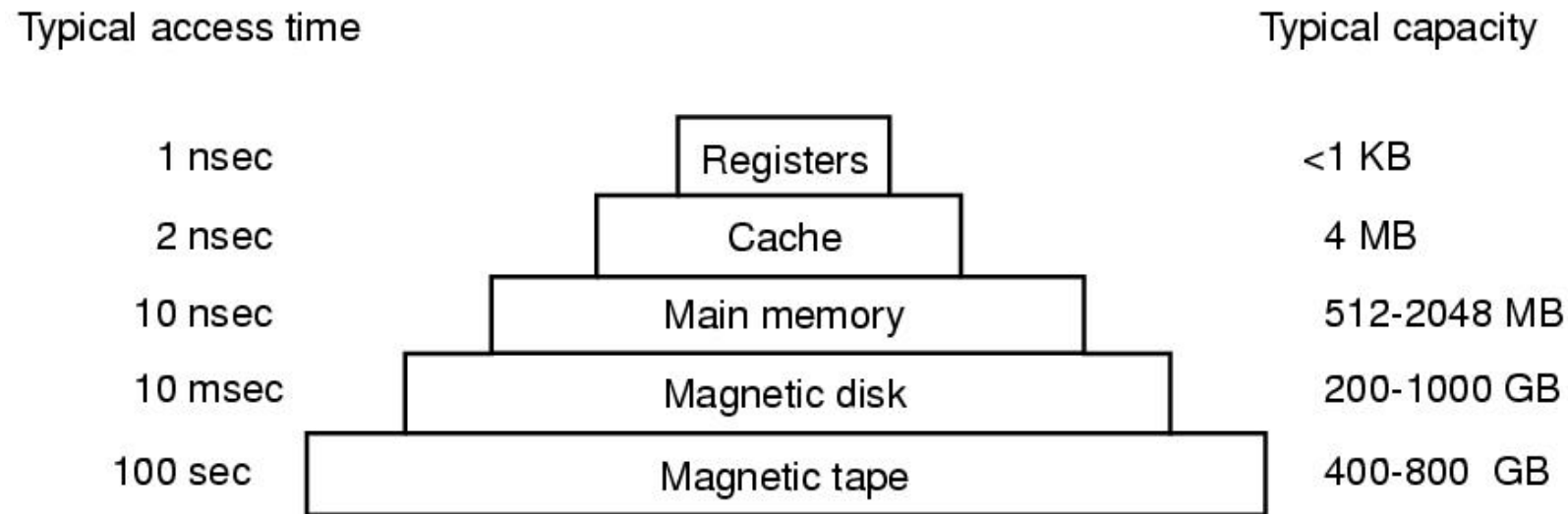Based on Tanenbaum, Modern Operating Systems 3 e

Multithreaded and Multicore Chips

(a) A quad-core chip with a shared L2 cache.

(b) A quad-core chip with separate L2 caches.

Memory

Typical access time | | Typical capacity

| Typical access time		Typical capacity
1 nsec	Registers	<1 KB
2 nsec	Cache	4 MB
10 nsec	Main memory	512-2048 MB
10 msec	Magnetic disk	200-1000 GB
100 sec	Magnetic tape	400-800 GB

- Questions when dealing with cache:
  - When to put a new item into the cache.
  - Which cache line to put the new item in.
  - Which item to remove from the cache when a slot is needed.
  - Where to put a newly evicted item in the larger memory.

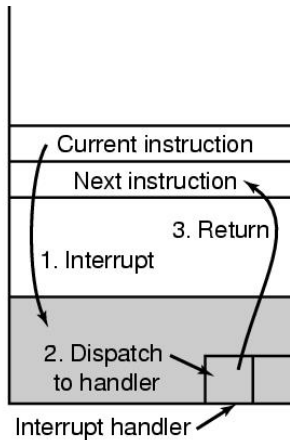# Disks
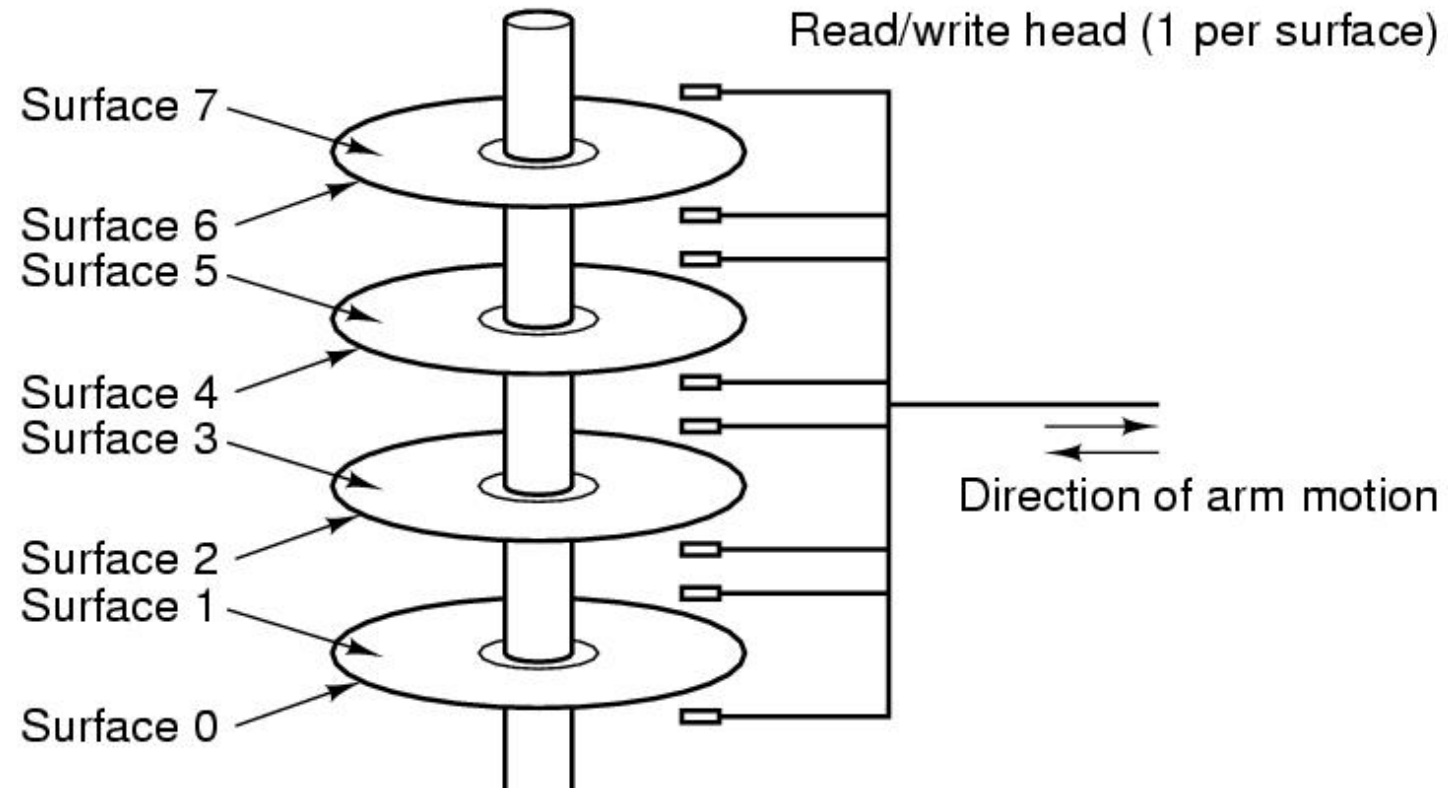
Structure of a disk drive.

I/O Devices
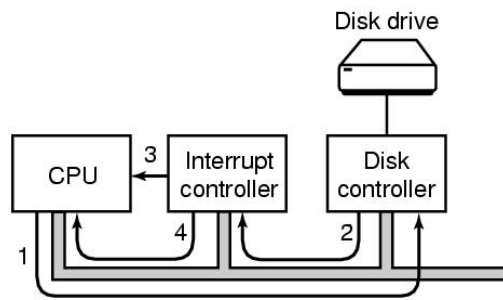
- (a) The steps in starting an I/O device and getting an interrupt.
- (b) Interrupt processing involves taking the interrupt, running the interrupt handler, and returning to the user program.

Buses

Core1   Core2
Cache   Cache
Shared cache
GPU Cores   PCIe   Graphics
DDR3 Memory — Memory controllers   DDR3 Memory

DMI

PCIe slot   Platform   SATA
PCIe slot   Controller   USB 2.0 ports
PCIe slot   Hub   USB 3.0 ports
PCIe slot   PCIe Gigabit Ethernet

More PCIe devices
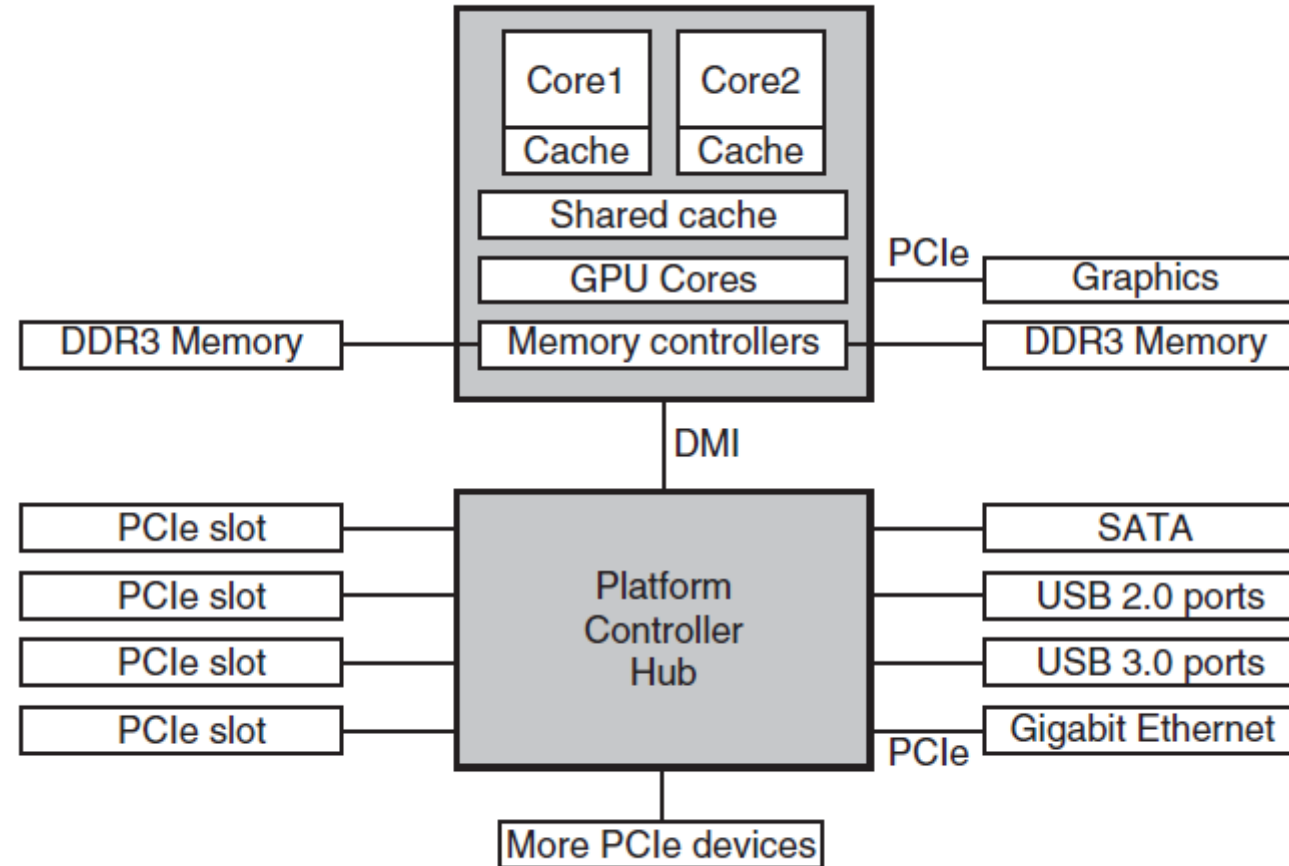
The structure of a large x86 system.

Types of Operating Systems

- Mainframe operating systems

- Server operating systems

- Multiprocessor operating systems

- Personal computer operating systems

- Handheld operating systems

- Embedded operating systems

- Sensor node operating systems

- Real-time operating systems

- Smart card operating systems

Operating System Concepts

- Processes

- Address spaces (Memory)

- Files

- Input/Output

- Protection

- The shell

- Development in computing over time
  - Large memories
  - Protection hardware
  - Disks
  - Virtual memory

System Calls
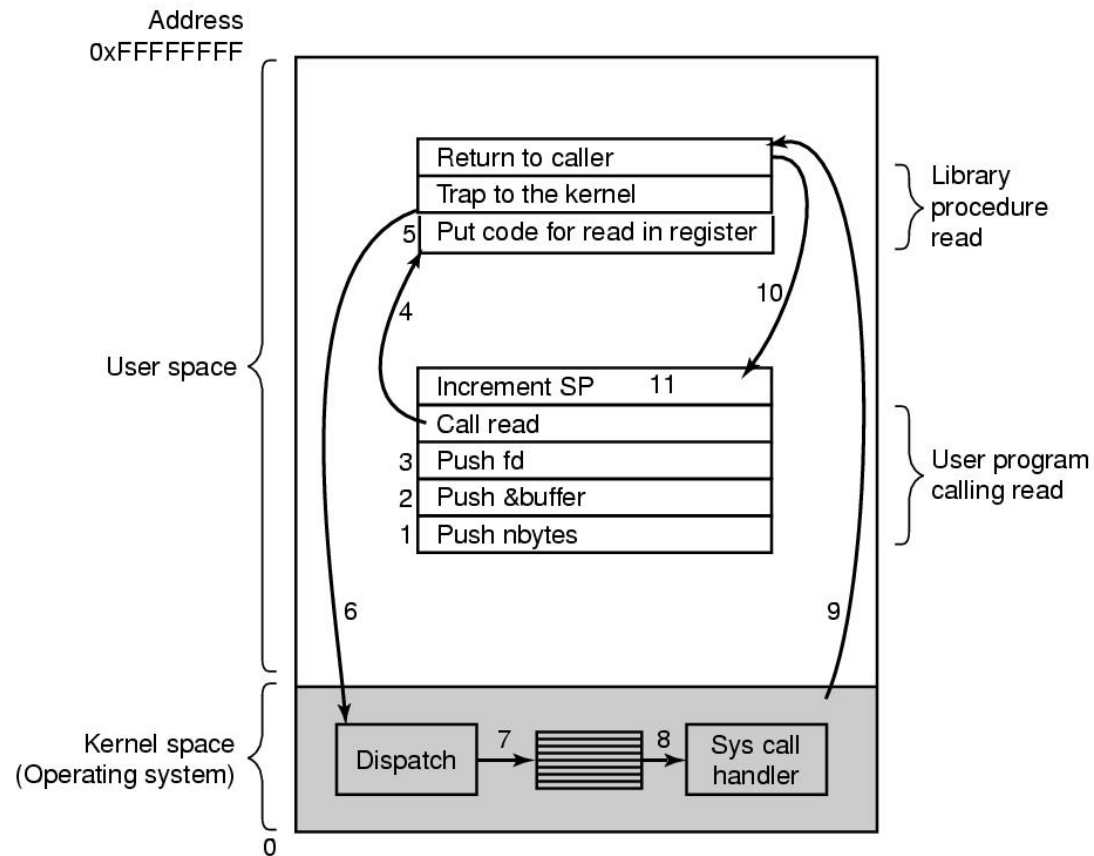
- The 11 steps in making the system call
  - read(fd, buffer, nbytes).

System Calls

Process management

Call	Description
pid = fork( )	Create a child process identical to the parent
pid = waitpid(pid, &statloc, options)	Wait for a child to terminate
s = execve(name, argv, environp)	Replace a process' core image
exit(status)	Terminate process execution and return status

File management

Call	Description
fd = open(file, how, ...)	Open a file for reading, writing, or both
s = close(fd)	Close an open file
n = read(fd, buffer, nbytes)	Read data from a file into a buffer
n = write(fd, buffer, nbytes)	Write data from a buffer into a file
position = lseek(fd, offset, whence)	Move the file pointer
s = stat(name, &buf)	Get a file's status information

Based on Tanenbaum, Modern Operating Systems 3 e

System Calls

Directory- and file-system management

Call	Description
s = mkdir(name, mode)	Create a new directory
s = rmdir(name)	Remove an empty directory
s = link(name1, name2)	Create a new entry, name2, pointing to name1
s = unlink(name)	Remove a directory entry
s = mount(special, name, flag)	Mount a file system
s = umount(special)	Unmount a file system

Miscellaneous

| Call	Description
s = chdir(dirname)	Change the working directory
s = chmod(name, mode)	Change a file's protection bits
s = kill(pid, signal)	Send a signal to a process
seconds = time(&seconds)	Get the elapsed time since Jan. 1, 1970

Windows Win32 API

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

Operating Systems Structures

- Monolithic systems

- Layered systems

- Microkernels

- Client-server systems

- Virtual machines

- Exokernels

Monolithic systems

Main procedure

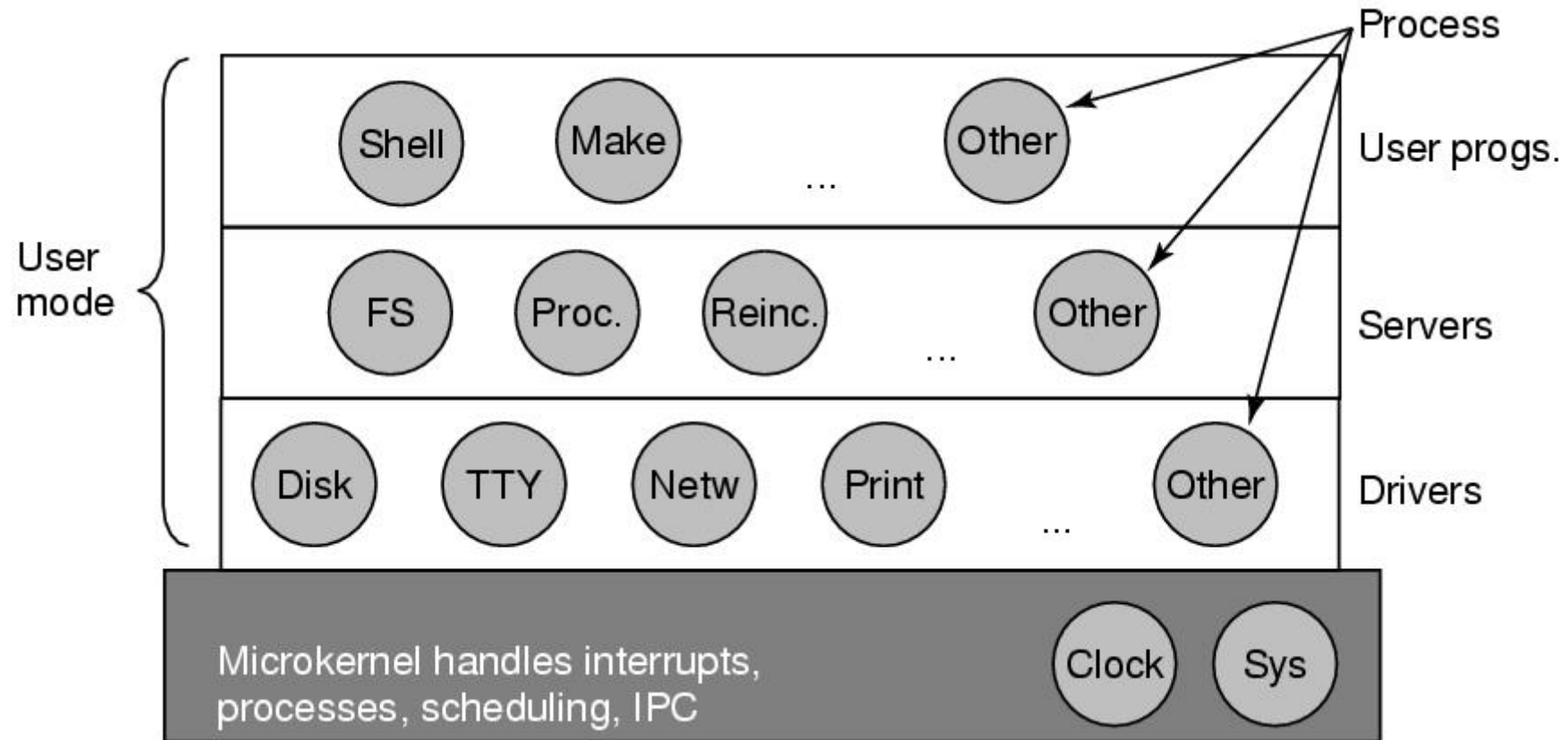Service procedures

Utility procedures
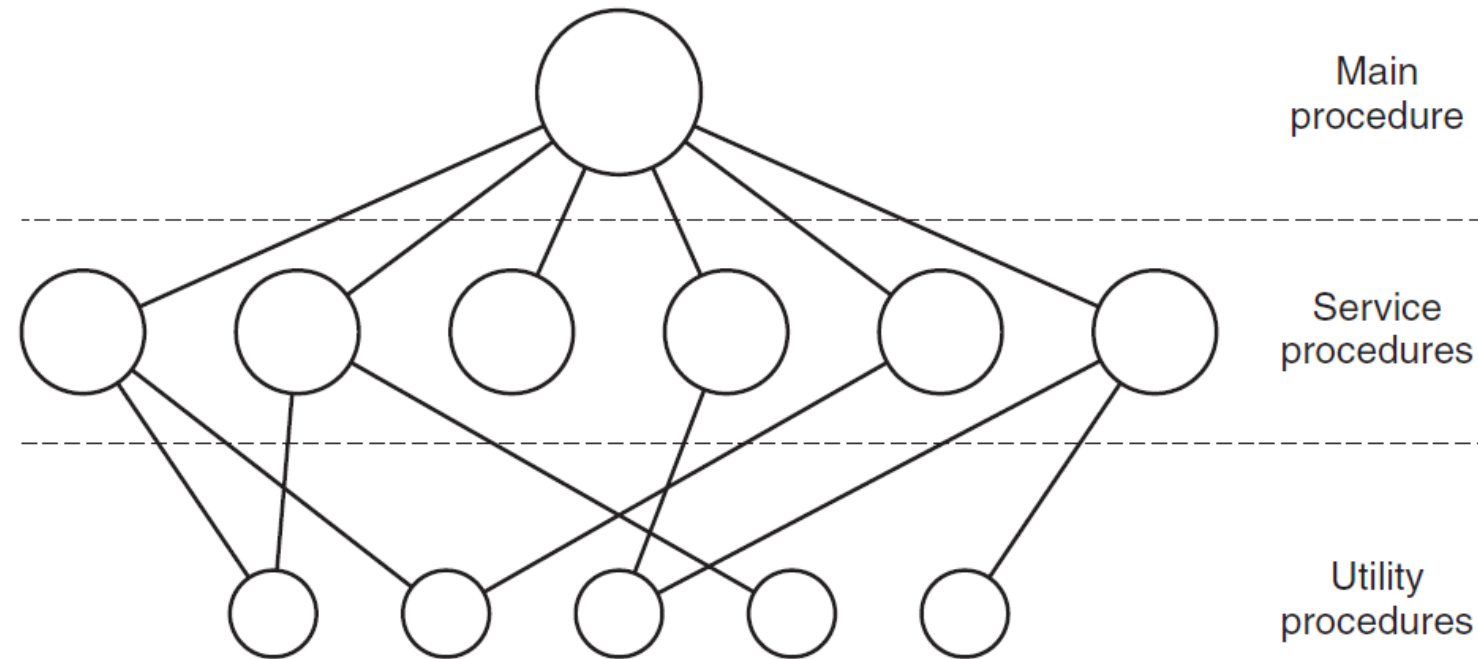
- A main program that invokes the requested service procedure.
- A set of service procedures that carry out the system calls.
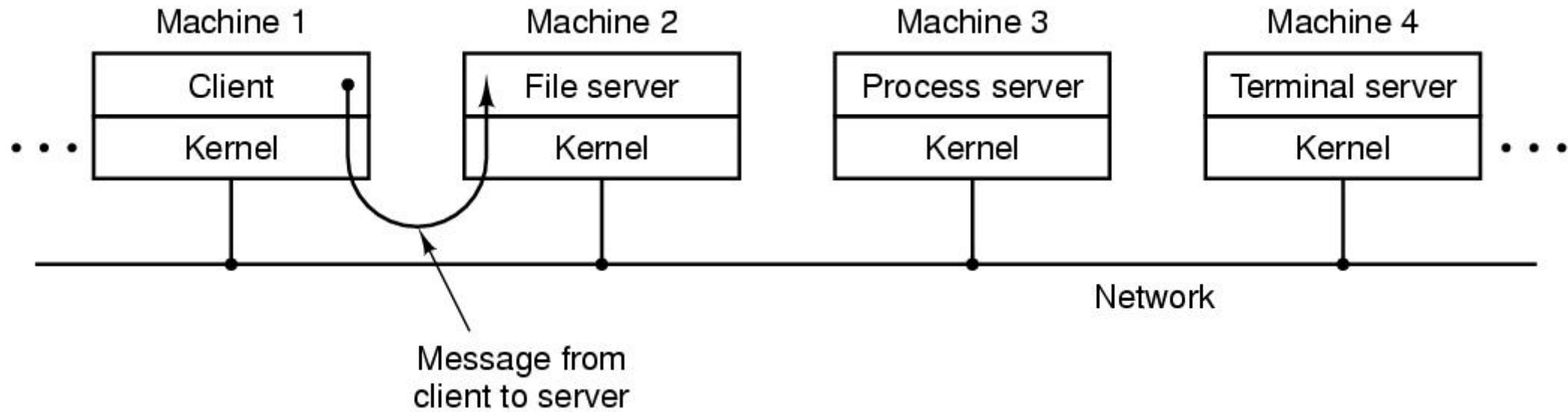- A set of utility procedures that help the service procedures.

Layered Systems

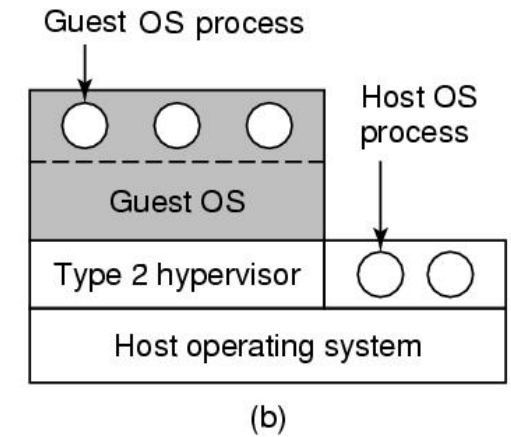| Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and drum management
0	Processor allocation and multiprogramming

Microkernels

Based on Tanenbaum, Modern Operating Systems 3 e

Client-Server Model

Machine 1 | Machine 2 | Machine 3 | Machine 4

Machine 1: Client / Kernel
Machine 2: File server / Kernel
Machine 3: Process server / Kernel
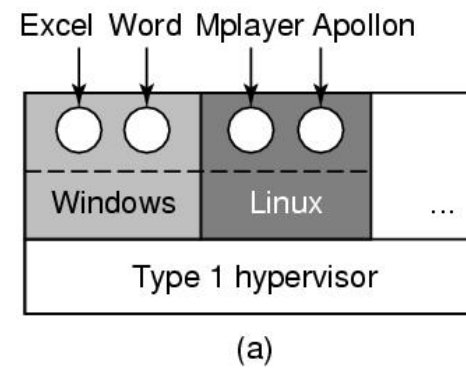Machine 4: Terminal server / Kernel

Network

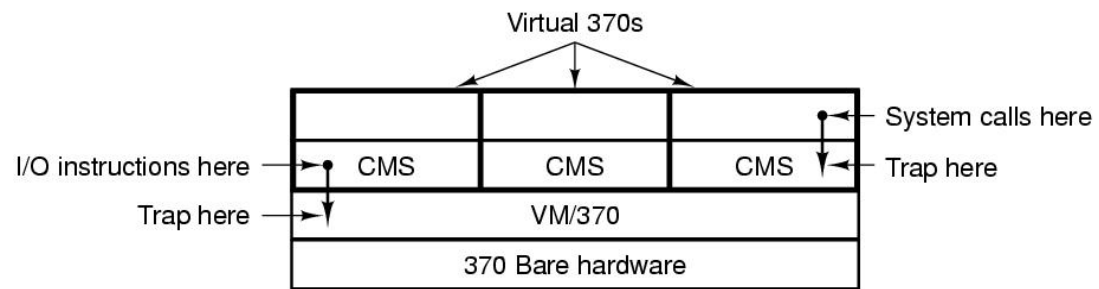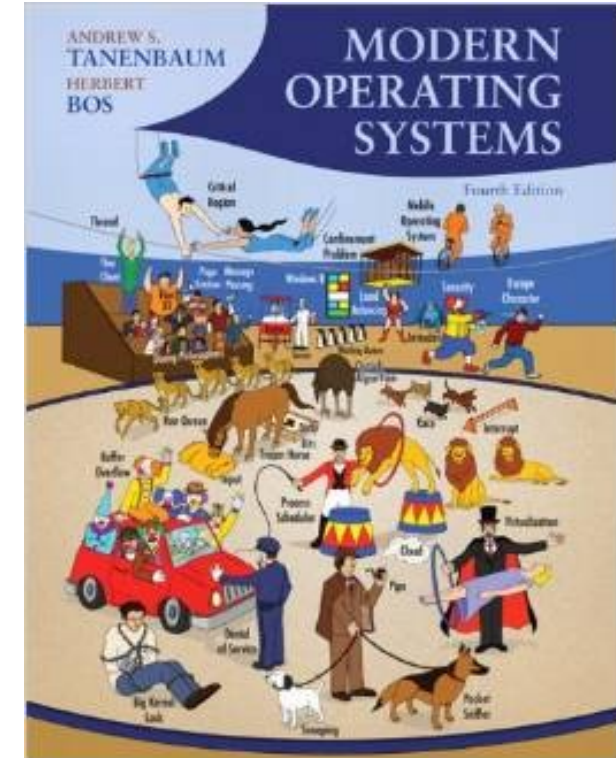Message from client to server

Virtual Machines

Suggested Textbook

- Modern Operating Systems, 4th Edition
- Andrew S. Tanenbaum and Herbert Bos

Course Details

- LMS:
  - http://lms.tech.sjp.ac.lk/course/view.php?id=127
- Previous Course Page:
  - https://academic.nimal.info/
- Contact:
  - nimal@sjp.ac.lk
  - 076 980 4524