Introduction

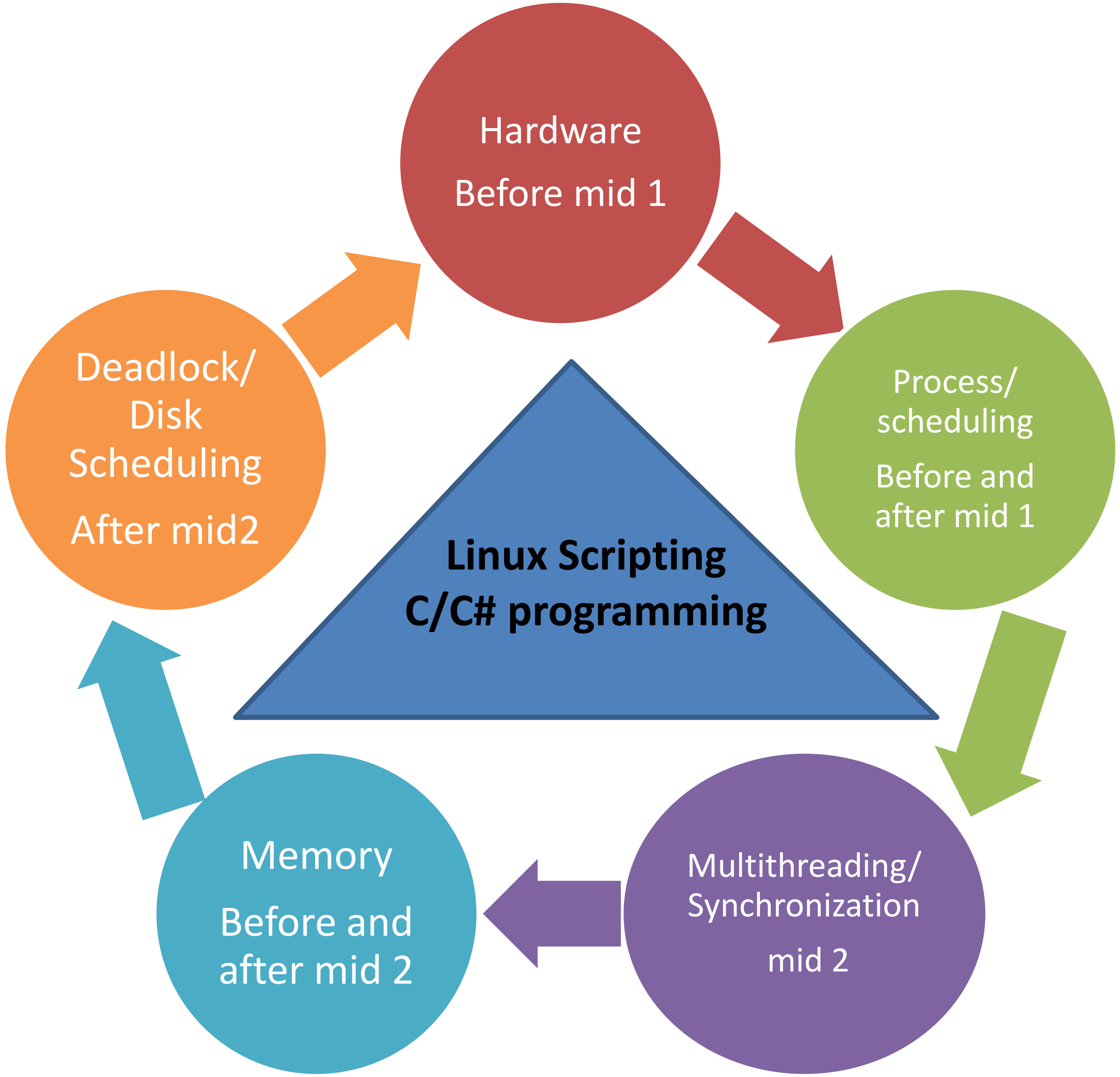
Operating Systems (CS-220)
Spring 2021, FAST NUCES

Google Classroom Code: **dpjhurs**

COURSE SUPERVISOR: ANAUM HAMID

anaum.hamid@nu.edu.pk

Course Outline



Evaluation Instruments and Marks Distributions

- Assignments = 20 marks
- Project = 10 marks
- Midterm = 30 marks (15 for each)
- Final Term = 50 marks
- Total = 100 Marks

Course Recourses

TEXTBOOK:

- Operating Systems Concepts, 10th edition, by Abraham Silberschatz, Peter Baer Galvin, and Greg Gagne.

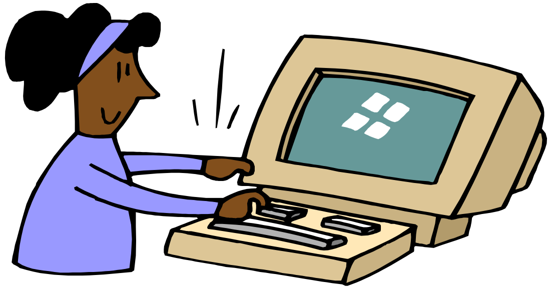
REFERENCE BOOKS:

- Operating Systems – Internals and Design Principles, 8th edition, by William Stallings.

Introduction

Road Map.

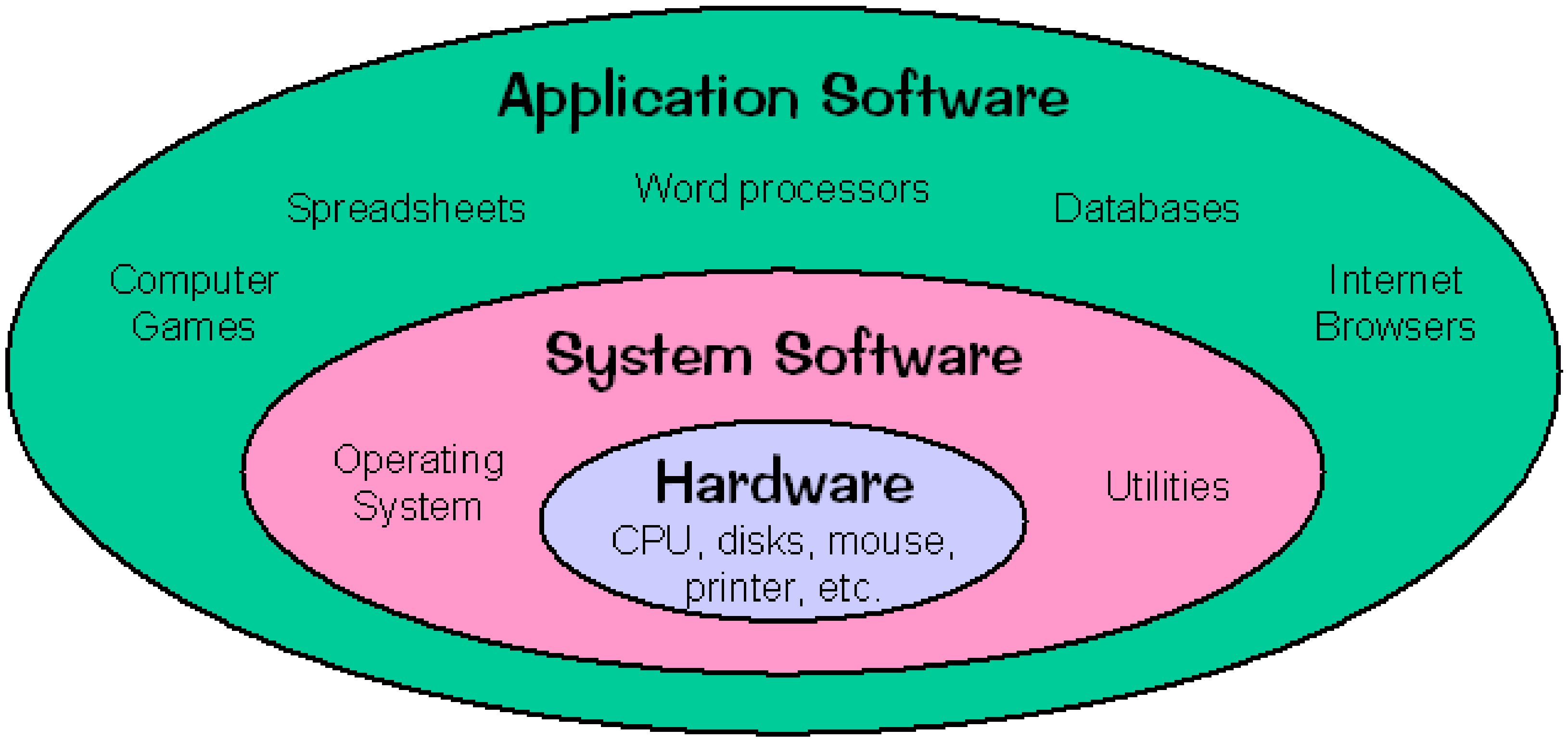
1. What is an Operating System?
2. CS structures.
3. Interrupt
4. I/O
5. Memory
6. CS Architecture
7. Clustered Systems
8. OS Structure + Operations
9. Computing Environments

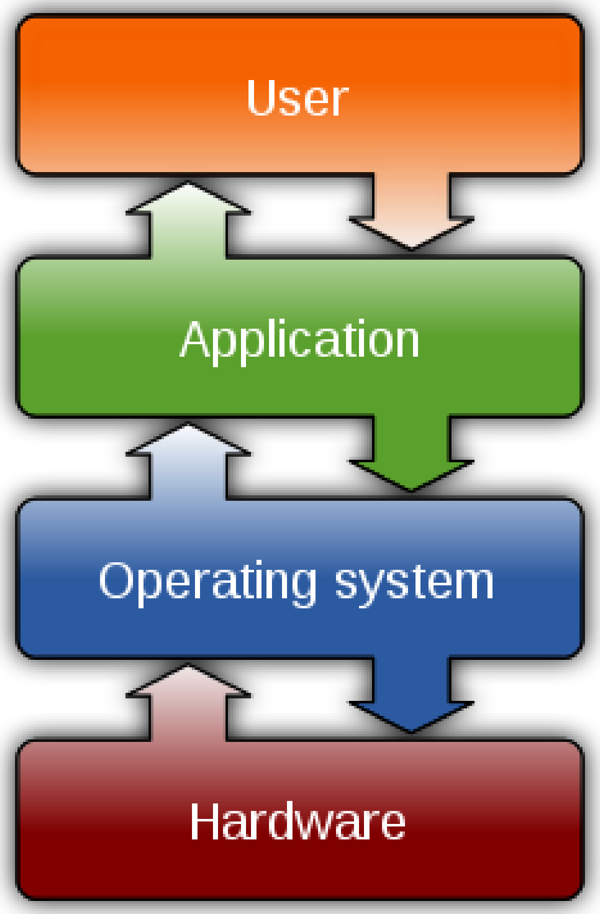


Operating System: Overview

- **Operating system (OS):** a **system software** that exploits the **hardware resources** to provide a set of services to system users.
- The OS manages the processor(s), memory and input/output (I/O) devices on behalf of its users.
- ***Note:*** it is important to have some fundamental understanding of basic data structures, computer organization and a high-level programming language, such as C or Java, before examining topics related to operating systems.

Operating System: Overview (Cont.)





Relationship between application software and system software

Computer System Structure

- Computer system can be divided into four components:

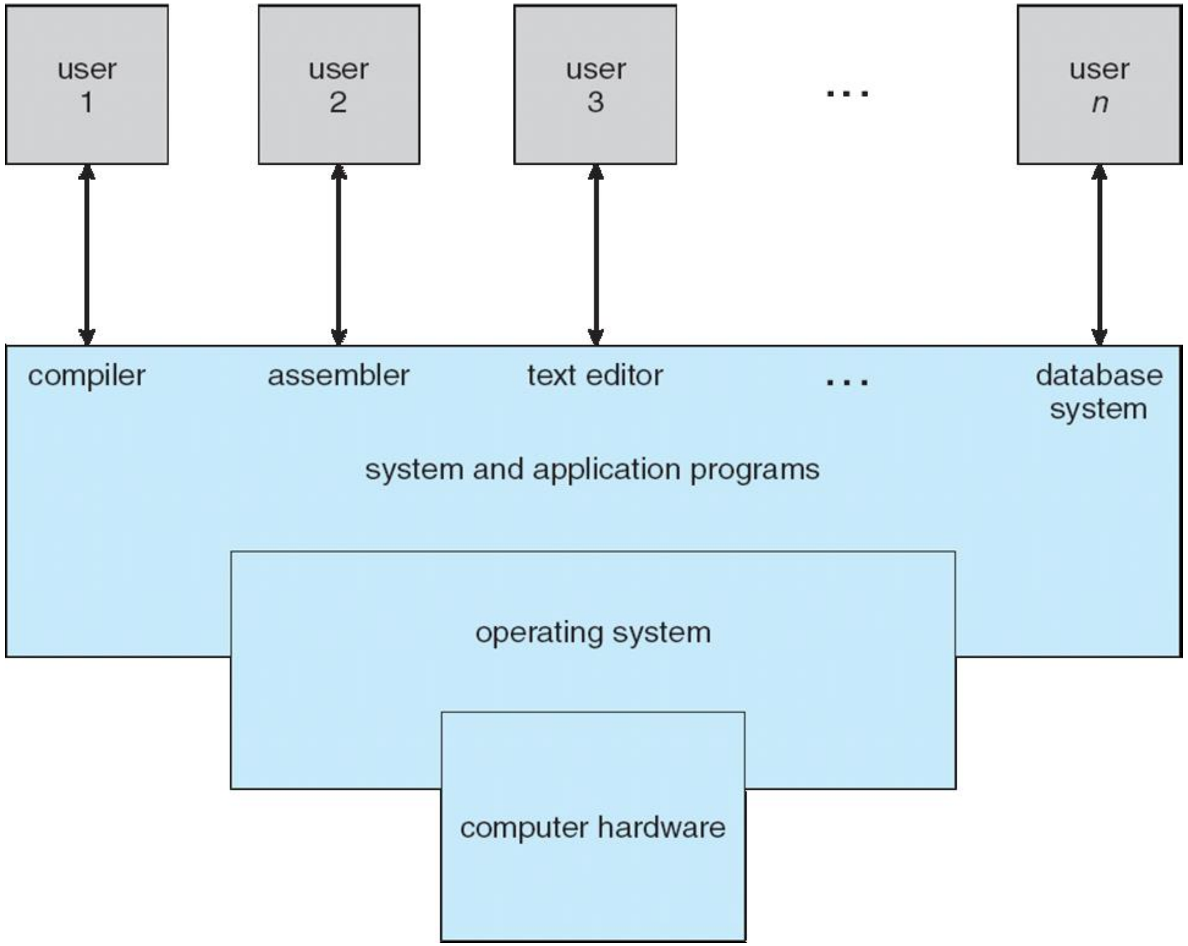
    1. Hardware – provides basic computing resources

    2. Operating system

    3. Application programs

    4. Users

Four Components of a Computer System



Operating System Definition

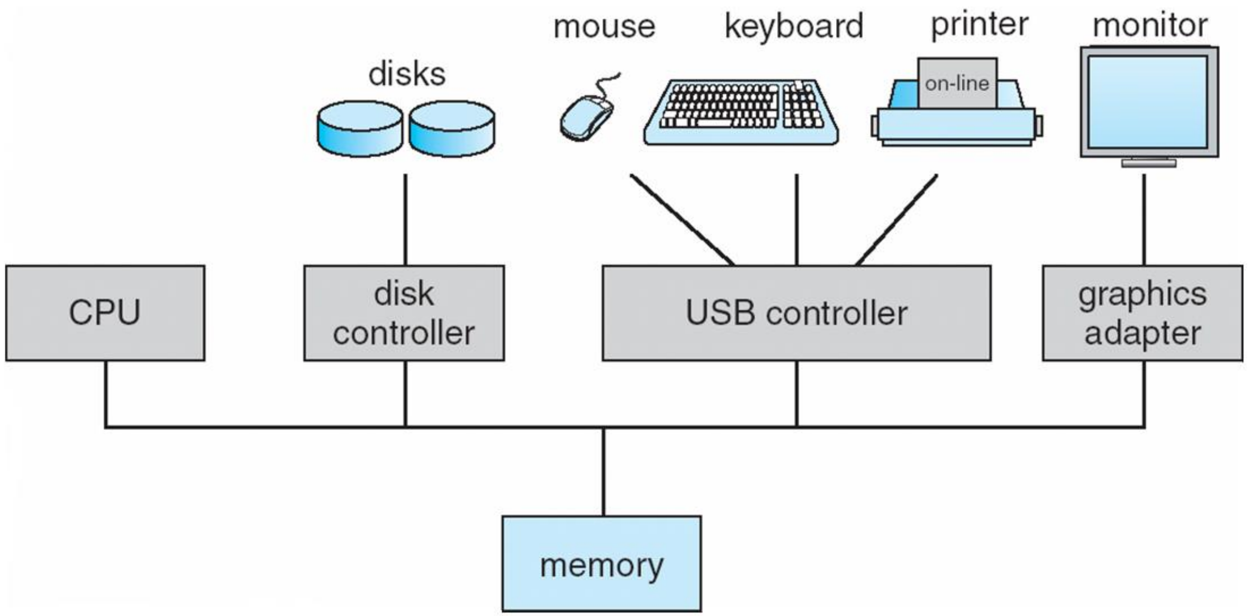
- OS is a **resource allocator**
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- OS is a **control program**
  - Controls execution of programs to prevent errors and improper use of the computer

Computer Startup

- **bootstrap program** is loaded at power-up or reboot
  - Typically stored in ROM or EPROM, generally known as **firmware**
  - Initializes all aspects of system
  - Loads operating system kernel and starts execution

Computer System Organization

- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles



Computer-System Operation

- I/O devices and the CPU can execute concurrently
- Each device controller is in charge of a particular device type
- Each device controller has a local buffer
- CPU moves data from/to main memory to/from local buffers
- I/O is from the device to local buffer of controller
- Device controller informs CPU that it has finished its operation by causing an **INTERRUPT**

Interrupts

- **Interrupt:** a mechanism by which other modules (I/O, memory) may interrupt the ***"normal sequencing of the processor"***.
- Most common classes of Interrupts are:

I/O	**Generated by an I/O controller, to signal normal completion of an operation or to signal a variety of error conditions.**
Program	**Generated by a condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, and reference outside a user's allowed memory space.**
Timer	**Generated by a timer within the processor.**
Hardware failure	**Generated by a failure, such as memory error.**

Common Functions of Interrupts

- Interrupt transfers control to the interrupt service routine generally, through the **interrupt vector**, which contains the addresses of all the service routines
- Interrupt architecture must save the address of the interrupted instruction
- A **trap** or **exception** is a software-generated interrupt caused either by an error or a user request
- An operating system is **interrupt driven**

Interrupt Handling

1. The operating system preserves the state of the CPU by storing registers and the program counter

2. Determines which type of interrupt has occurred:
   1. **polling**
   2. **vectored** interrupt system

3. Separate segments of code determine what action should be taken for each type of interrupt

Multiple Interrupts

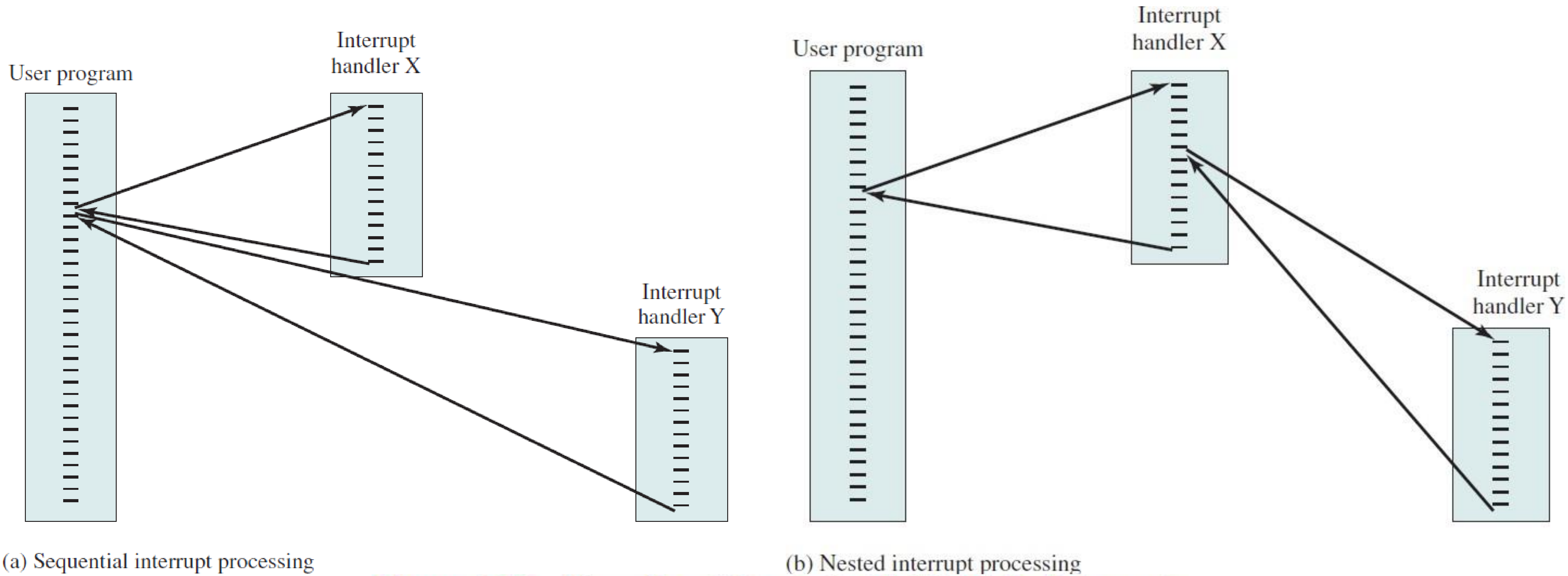


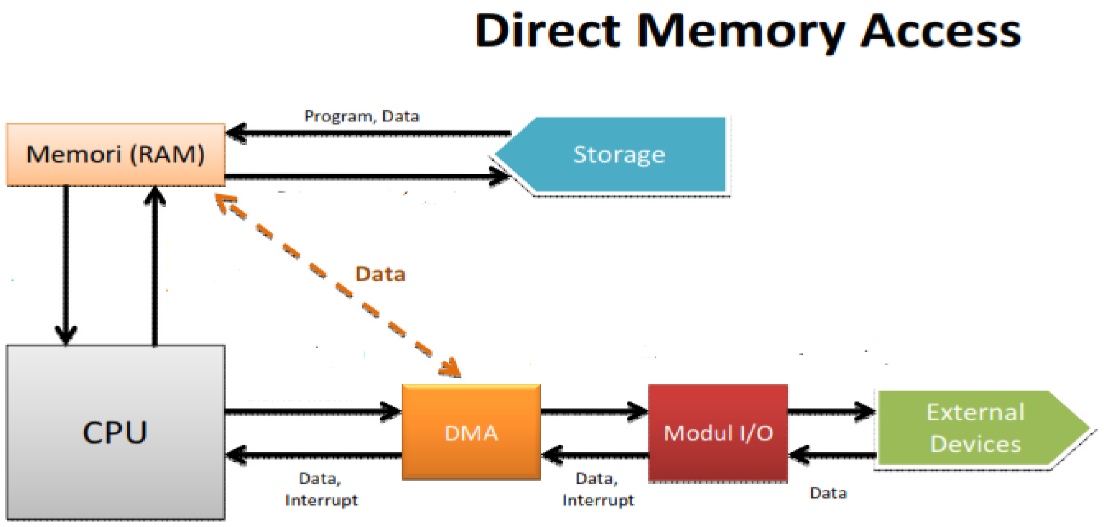
Figure 1.12 Transfer of Control with Multiple Interrupts

I/O Structure

1. After I/O starts, control returns to user program only upon I/O completion
   - Wait instruction idles the CPU until the next interrupt
   - Wait loop (contention for memory access)
   - At most one I/O request is outstanding at a time, no simultaneous I/O processing.
2. After I/O starts, control returns to user program without waiting for I/O completion
   - **System call** – request to the OS to allow user to wait for I/O completion
   - **Device-status table** contains entry for each I/O device indicating its type, address, and state

Direct Memory Access Structure

- Used for high-speed I/O devices able to transmit information at close to memory speeds



- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention

- Only one interrupt is generated per block, rather than the one interrupt per byte

Memory

Storage Structure

1. Main memory – only large storage media that the CPU can access directly
2. Secondary storage – extension of main memory that provides large **nonvolatile** storage capacity.
3. Magnetic disks – rigid metal or glass platters covered with magnetic recording material
   - Disk surface is logically divided into **tracks**, which are subdivided into **sectors**
   - The **disk controller** determines the logical interaction between the device and the computer

Storage Hierarchy

Storage systems organized in hierarchy

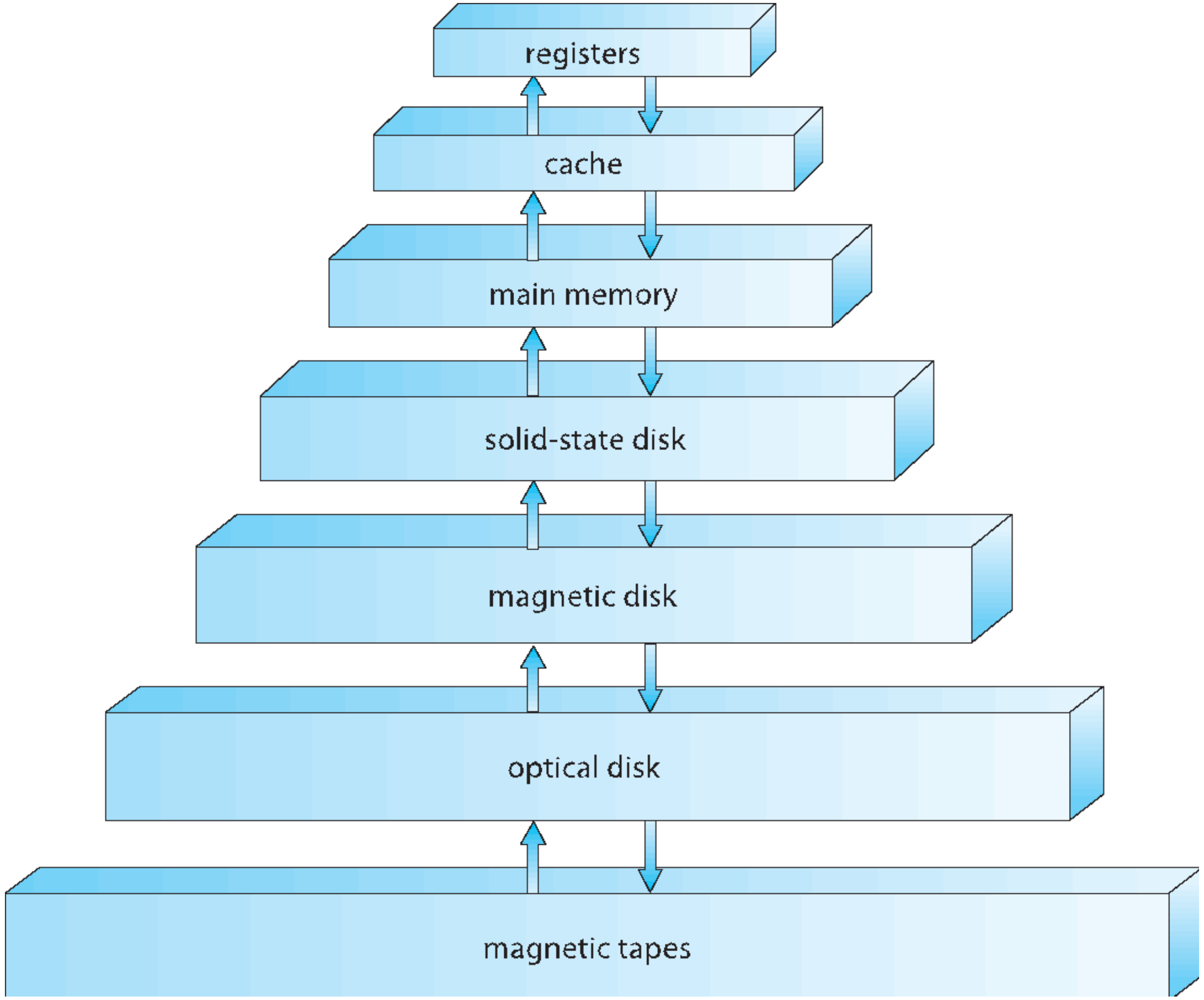
- Speed
- Cost
- Volatility

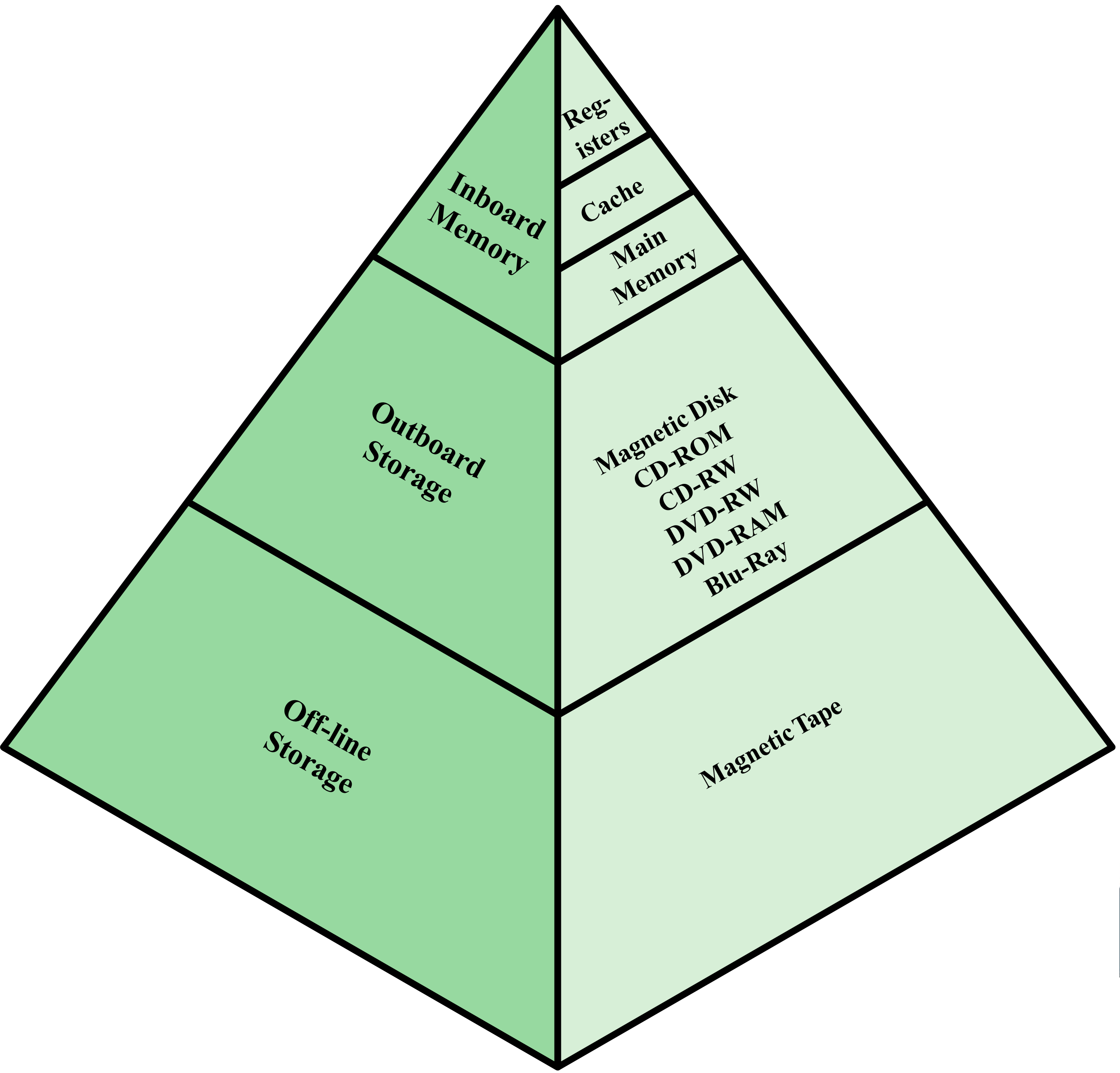
1. Caching – copying information into faster storage system; main memory can be viewed as a cache for secondary storage

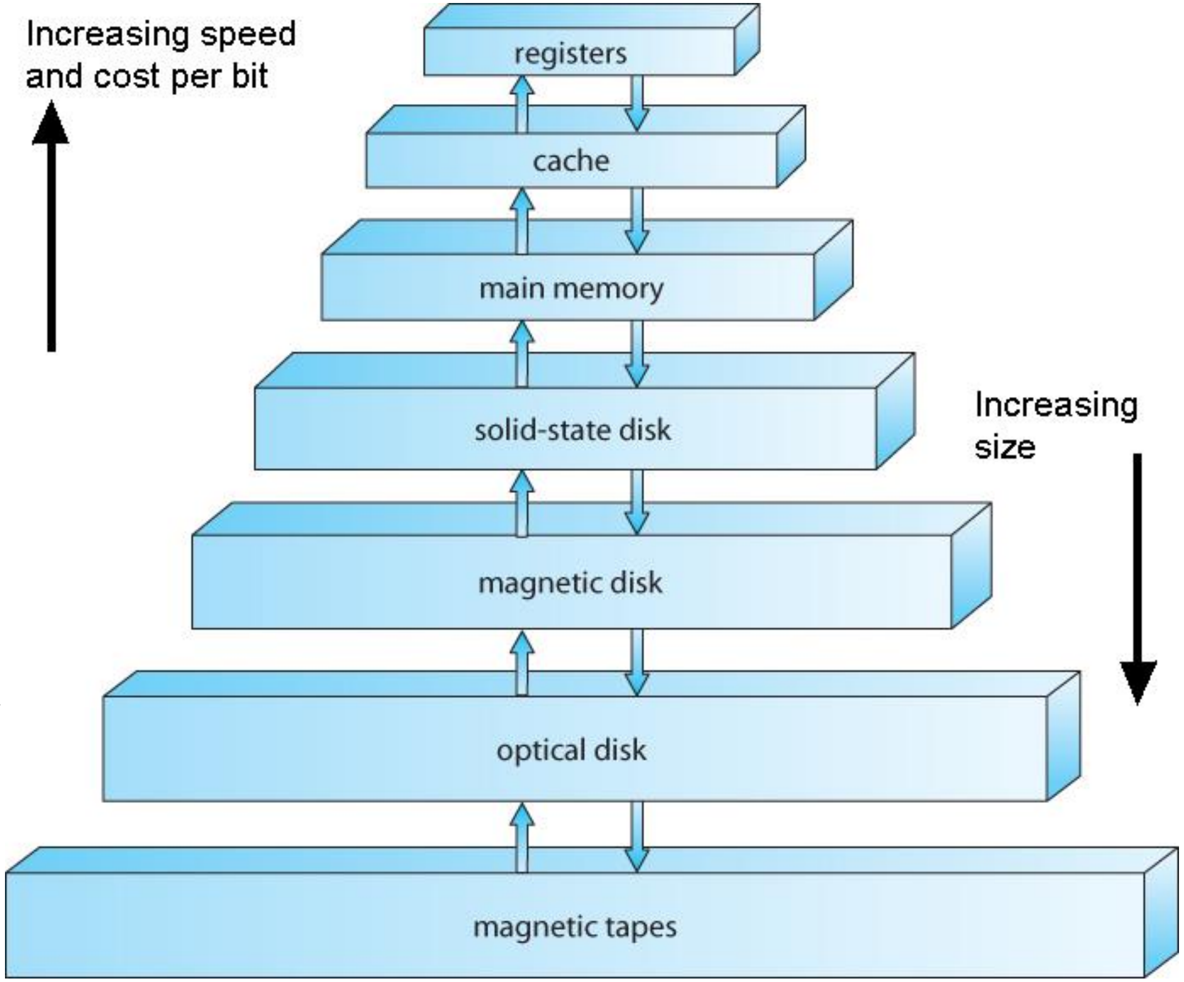
2. Device Driver for each device controller to manage I/O

- Provides uniform interface between controller and kernel

Storage-Device Hierarchy




Reg-
isters
Cache
Main
Memory
Inboard
Memory
Outboard
Storage
Magnetic Disk
CD-ROM
CD-RW
DVD-RW
DVD-RAM
Blu-Ray
Off-line
Storage
Magnetic Tape


Increasing speed
and cost per bit
registers
cache
main memory
solid-state disk
magnetic disk
optical disk
magnetic tapes
Increasing
size

Caching

1. Important principle, performed at many levels in a computer (in hardware, operating system, software).

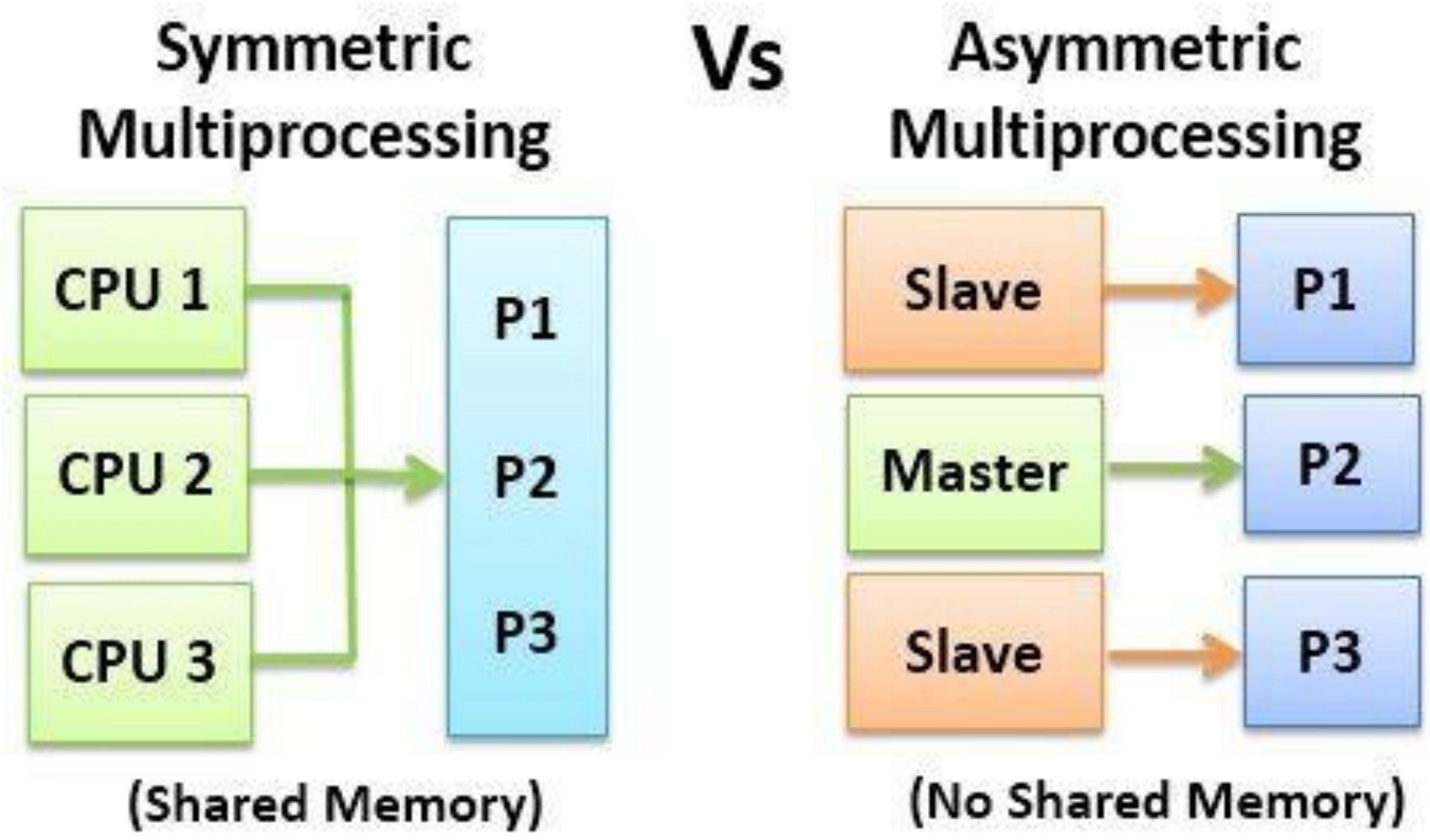
2. Faster storage (cache) checked first to determine if information is there
   - If it is, information used directly from the cache (fast)
   - If not, data copied to cache and used there

Computing Architectures

Computer-System Architecture

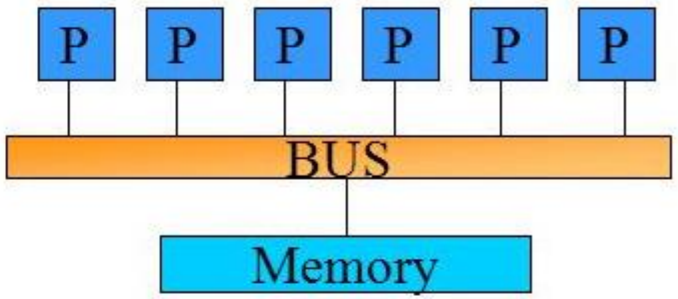
- **Multiprocessor** systems growing in use and importance
  - Also known as **parallel systems**, **tightly-coupled systems**
  - Advantages include:
    1. **Increased throughput**
    2. **Economy of scale**
    3. **Increased reliability – graceful degradation** or **fault tolerance**
  - Two types:
    1. **Asymmetric Multiprocessing**
    2. **Symmetric Multiprocessing**

Symmetric vs. Asymmetric Multiprocessing Architecture



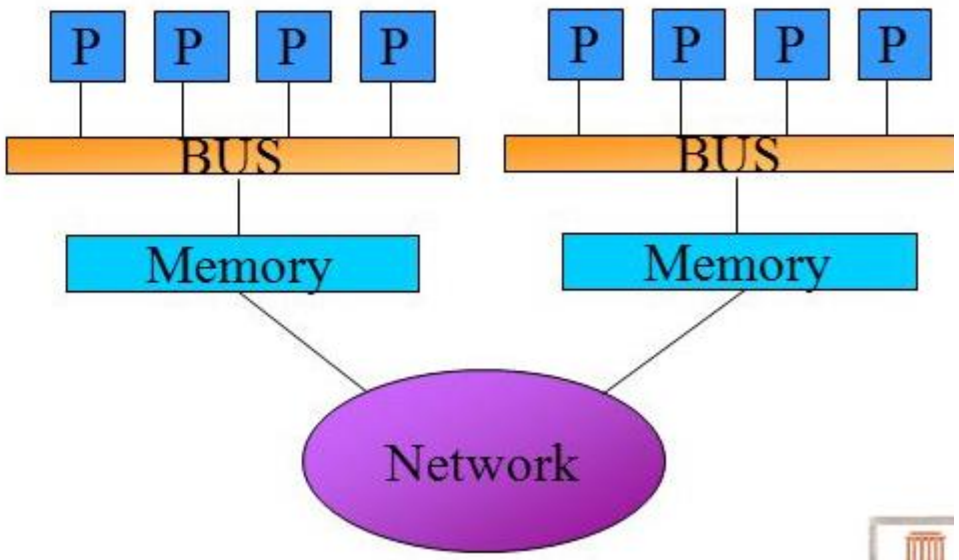
A Dual-Core Design

- **UMA** and **NUMA** architecture variations
- Multi-chip and **multicore**



Uniform memory access (UMA): Each processor has uniform access to memory. Also known as symmetric multiprocessors, or SMPs (Sun E10000)

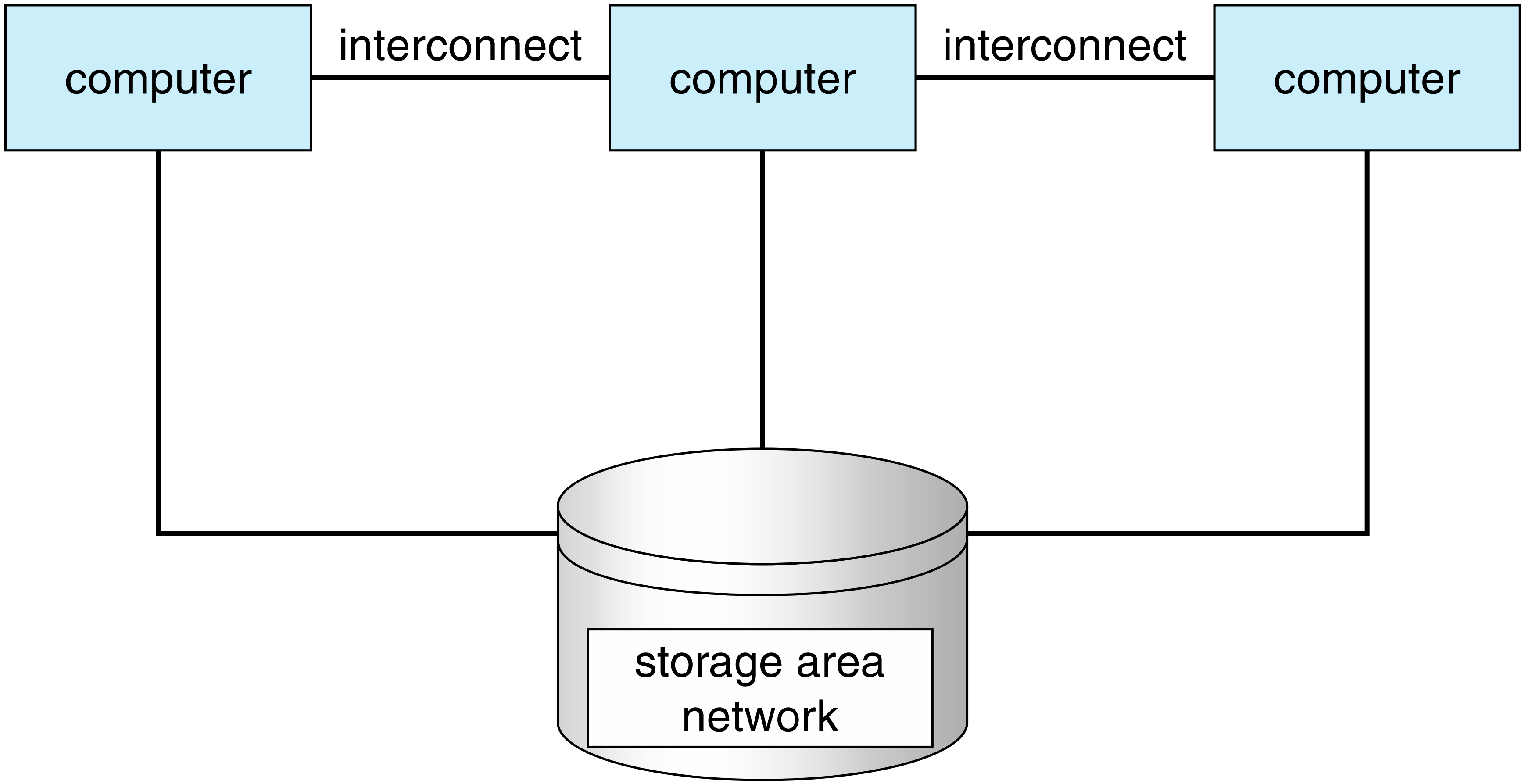
Non-uniform memory access (NUMA): Time for memory access depends on location of data. Local access is faster than non-local access. Easier to scale than SMPs (SGI Origin)



Clustered Systems

- Like multiprocessor systems, but multiple systems working together
  - Usually sharing storage via a **storage-area network (SAN)**
  - Provides a **high-availability** service which survives failures
    - **Asymmetric clustering** has one machine in hot-standby mode
    - **Symmetric clustering** has multiple nodes running applications, monitoring each other

Clustered Systems

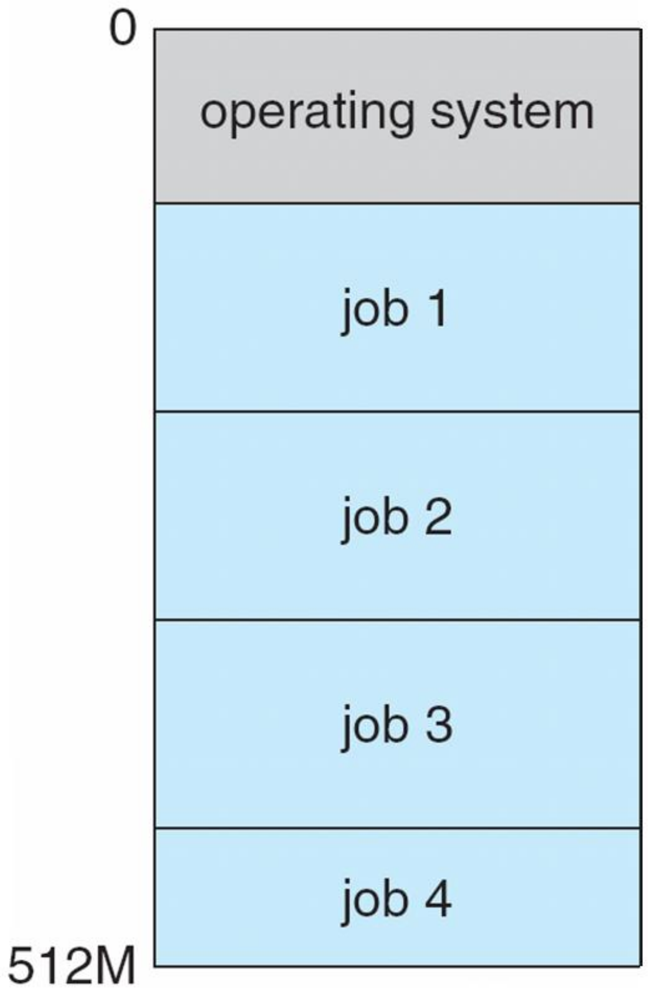


Operating System Structures + Operations

Operating System Structure

- **Multiprogramming** needed for efficiency
  - Single user cannot always keep CPU and I/O devices busy
  - Multiprogramming organizes jobs (code and data) so CPU always has one to execute
  - A subset of total jobs in system is kept in memory
  - One job selected and run via **job scheduling**
  - When it must wait (for I/O for example), OS switches to another job
- **Timesharing** (**multitasking**) is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating **interactive** computing
  - **Response time** should be < 1 second
  - Each user has at least one program executing in memory ⇨**process**
  - If several jobs ready to run at the same time ⇨ **CPU scheduling**
  - If processes don't fit in memory, **swapping** moves them in and out to run
  - **Virtual memory** allows execution of processes not completely in memory

Memory Layout for Multiprogram System



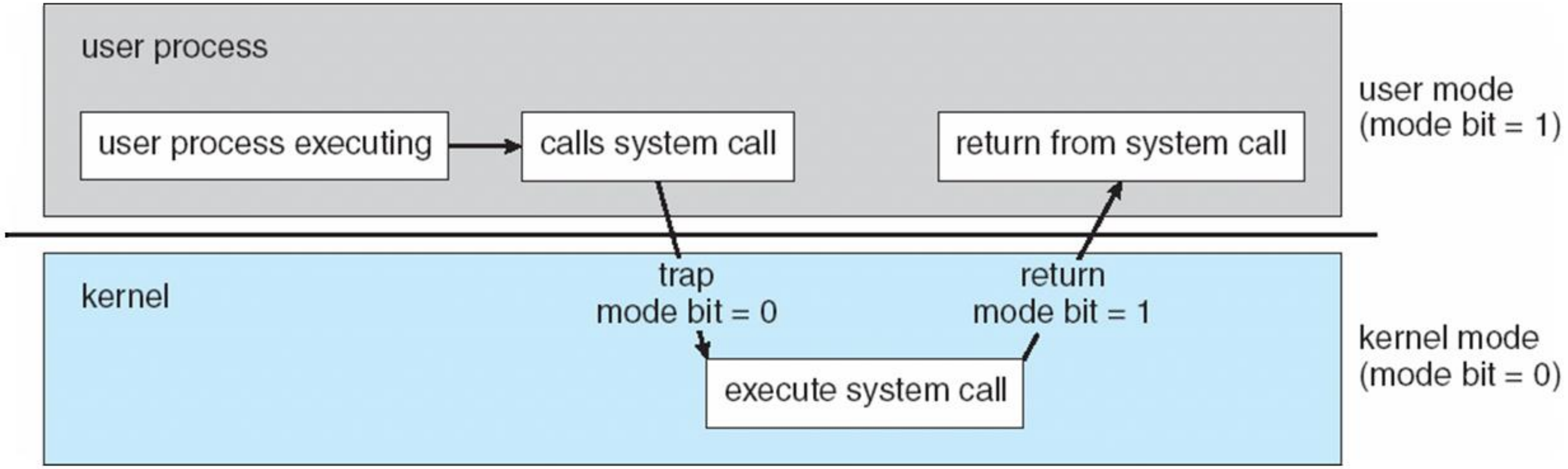
Operating-System Operations

- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as **privileged**, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user
- Increasingly CPUs support multi-mode operations
  - i.e. **virtual machine manager** (**VMM**) mode for guest **VMs.**

- "The one program running at all times on the computer" is the **kernel**.

Transition from User to Kernel Mode

- Timer to prevent infinite loop / process hogging resources
  - Set interrupt after specific period
  - Operating system decrements counter
  - When counter zero generate an interrupt
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time



Process Management

- A process is a program in execution. It is a unit of work within the system. Program is a ***passive entity;*** process is an ***active entity***.

- Process needs resources to accomplish its task
  - CPU, memory, I/O, files
  - Initialization data

- **Program counter (PC):** Contains the address of an instruction to be fetched

- Single-threaded process has one **program counter** specifying location of next instruction to execute

- Multi-threaded process has one program counter per thread

Process Management Activities

The operating system is responsible for the following activities in connection with process management:

- Creating and deleting both user and system processes
- Suspending and resuming processes
- process synchronization
- process communication
- deadlock handling

Memory Management

- Memory management activities
  - Keeping track of which parts of memory are currently being used and by whom
  - Deciding which processes (or parts thereof) and data to move into and out of memory
  - Allocating and deallocating memory space as needed

Storage Management

- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
  - OS activities include
    - Creating and deleting files and directories
    - Primitives to manipulate files and dirs
    - Mapping files onto secondary storage
    - Backup files onto stable (non-volatile) storage media

Mass-Storage Management

- OS activities
  - Free-space management
  - Storage allocation
  - Disk scheduling
- Some storage need not be fast
  - Tertiary storage includes optical storage, magnetic tape
  - Still must be managed – by OS or applications
  - Varies between WORM (write-once, read-many-times) and RW (read-write)

I/O Subsystem

- I/O subsystem responsible for
  - Memory management of I/O
  - caching (storing parts of data in faster storage for performance)
  - spooling (the overlapping of output of one job with input of other jobs)

Protection and Security

- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service

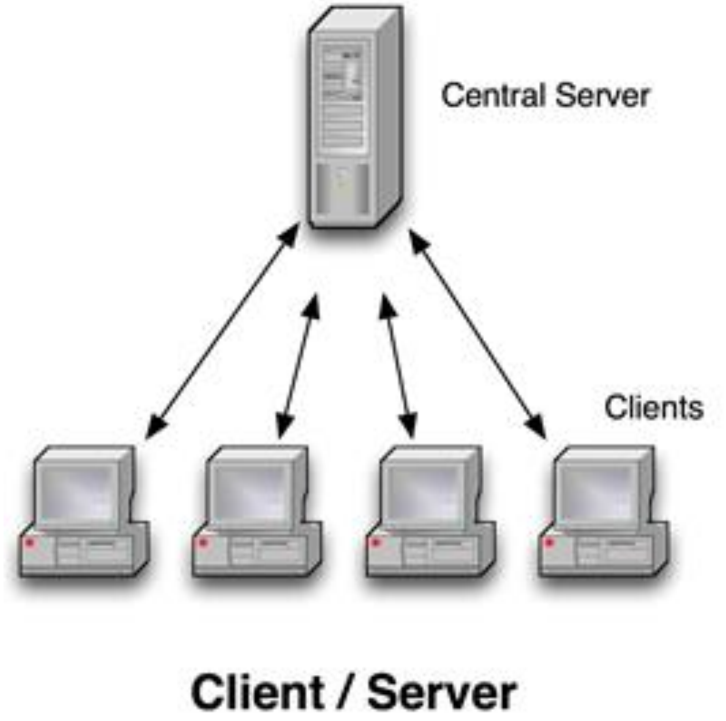
Computing Environments

Computing Environments -Distributed

- Distributed
  - Collection of separate, possibly heterogeneous, systems networked together
    - **Network** is a communications path,
      - **Local Area Network** (**LAN**)
      - **Wide Area Network** (**WAN**)
  - **Network Operating System** provides features between systems across network

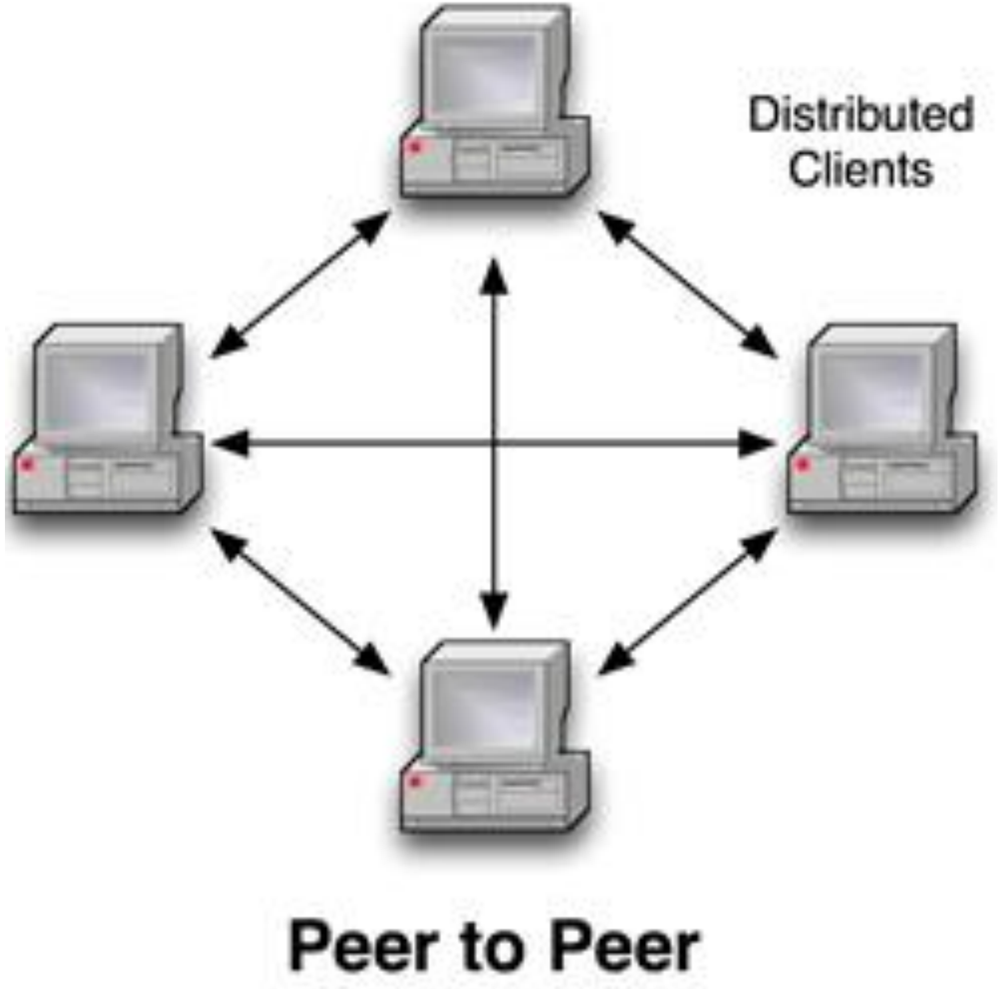
Computing Environments – Client-Server

- Client-Server Computing
  - Dumb terminals supplanted by smart PCs
  - Many systems now **servers**, responding to requests generated by **clients**

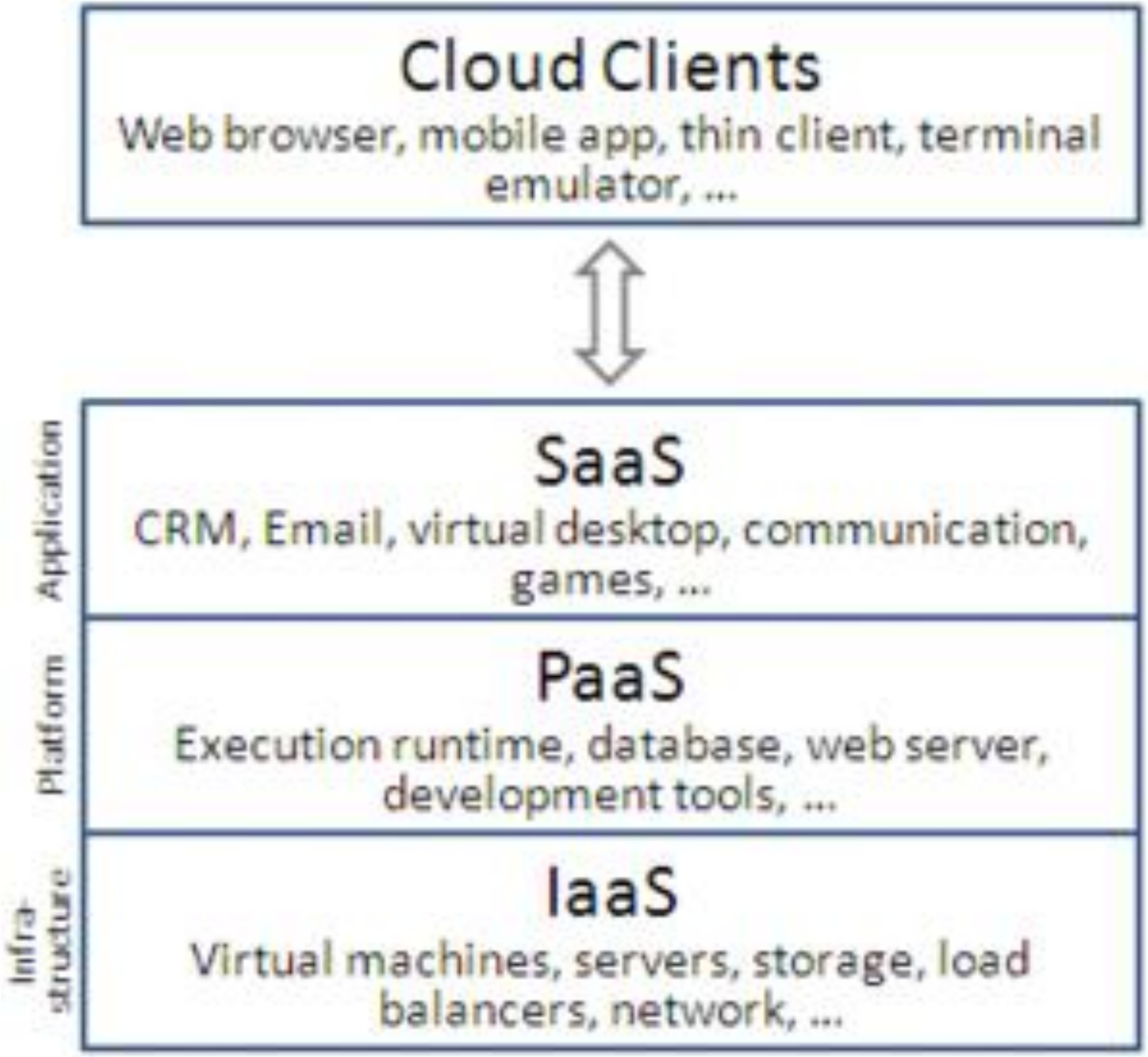


Computing Environments - Peer-to-Peer

- Another model of distributed system
- P2P does not distinguish clients and servers



Computing Environments – Cloud Computing



Real-Time Embedded Systems

- Real-time embedded systems most prevalent form of computers
  - Vary considerable, special purpose, limited purpose OS, **real-time OS**
  - Use expanding
- Many other special computing environments as well
  - Some have OSes, some perform tasks without an OS
- Real-time OS has well-defined fixed time constraints
  - Processing ***must*** be done within constraint
  - Correct operation only if constraints met

Thank You!

Chapter 1 Completed