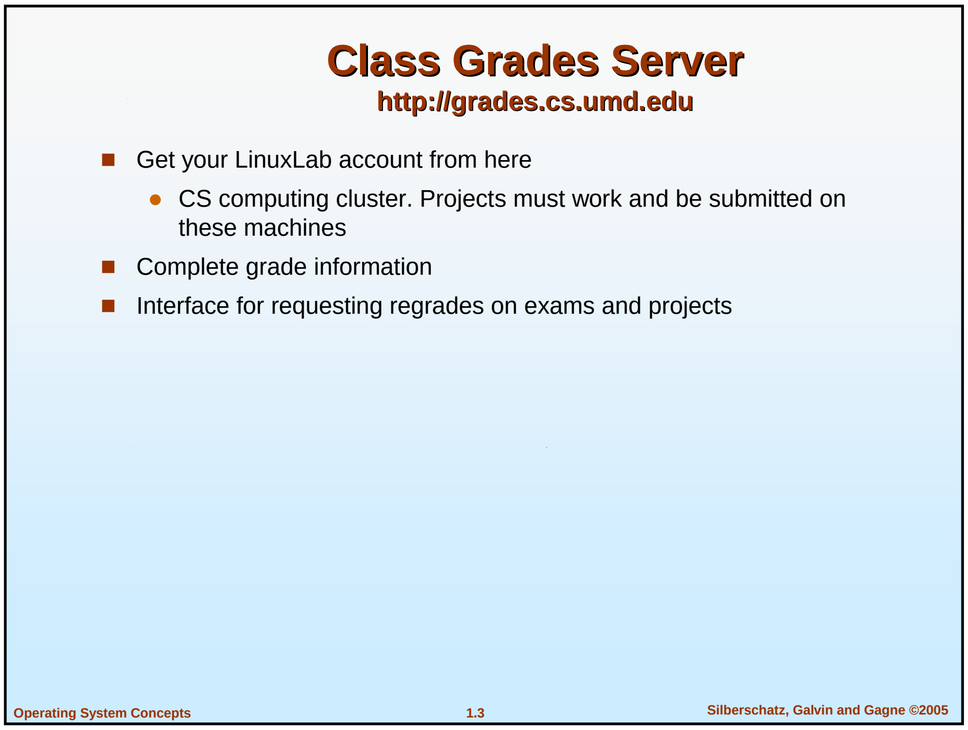
CSMC 412

Operating Systems

Prof. Ashok K Agrawala

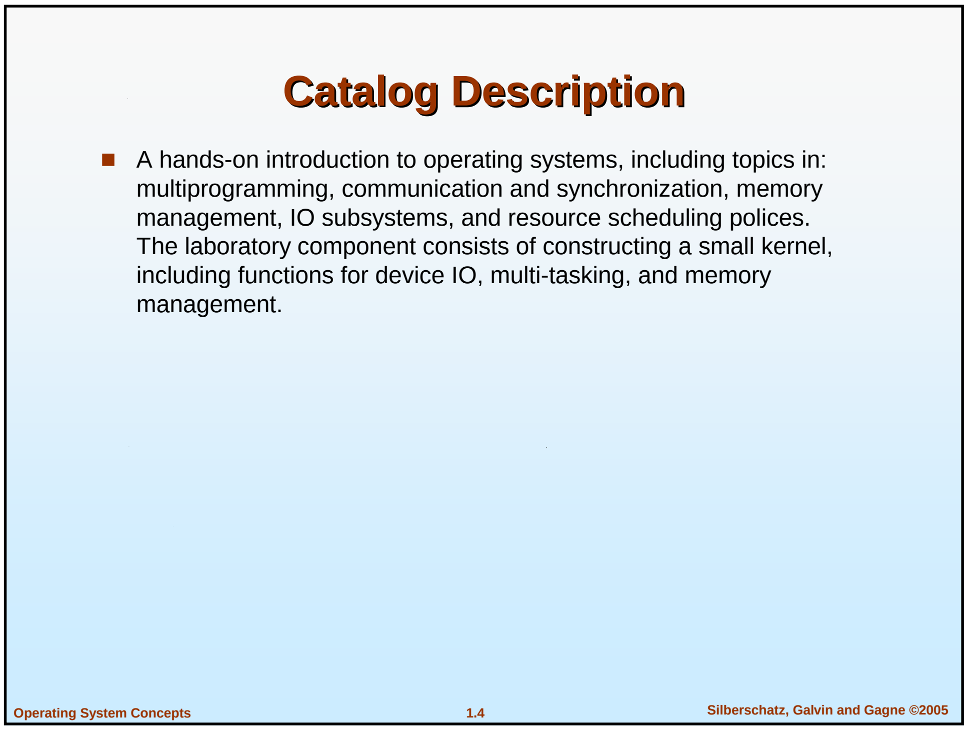
Today

- Review Syllabus
  - read the warning about the size of the project
  - Preferred 7th edition of the book. If you have 6th, it may be OK.
- Program #0 Handout (posted on the web)
  - its due in just over one week
  - purpose is to get familiar with the simulator/compiler/debugger
- Class Grades Server
  - Grades.cs.umd.edu
- Discussion Sections
  - will focus on the project and meet only once a week (W)
    - Probably will have three times per week; you can attend the section of your choice. See web page for details.
- Reading
  - Chapter 1
  - Chapter 2



Class Grades Server
http://grades.cs.umd.edu

- Get your LinuxLab account from here
  - CS computing cluster. Projects must work and be submitted on these machines
- Complete grade information
- Interface for requesting regrades on exams and projects



Catalog Description

- A hands-on introduction to operating systems, including topics in: multiprogramming, communication and synchronization, memory management, IO subsystems, and resource scheduling polices. The laboratory component consists of constructing a small kernel, including functions for device IO, multi-tasking, and memory management.

Prerequisites

- CMSC 311
- CMSC 330

Topics Covered

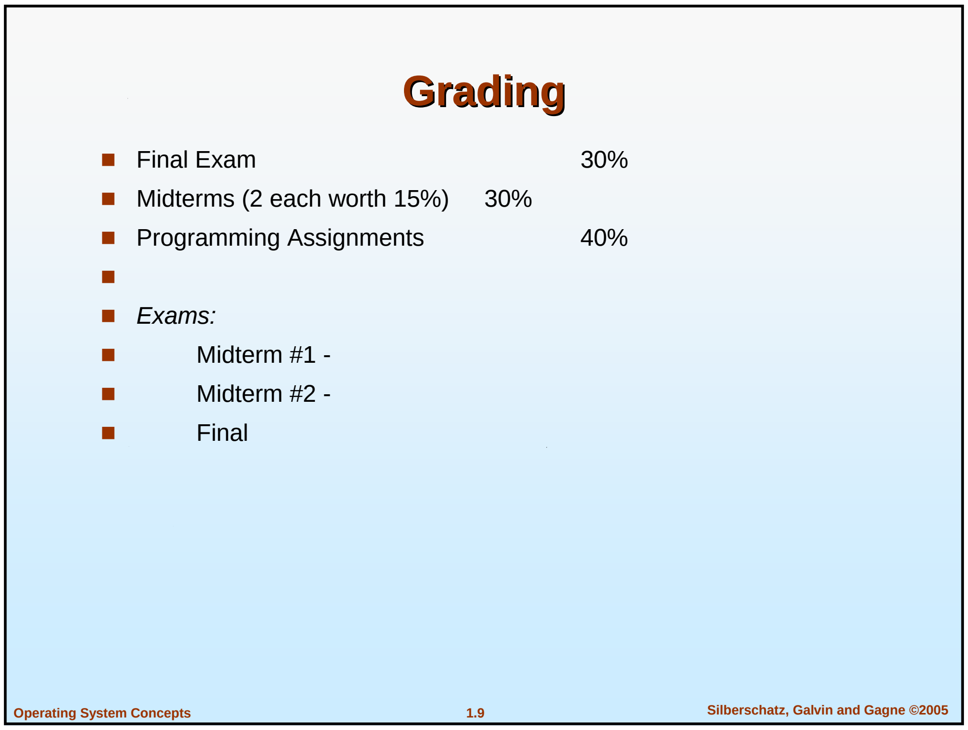
- • Introduction to Operating Systems (1 week)
- • The Cyclone Programming Language (0.5 weeks)
- • Concurrent Processes (2 weeks)
- • Kernel implementation techniques (1 week)
- • CPU scheduling (1 week)
- • Memory Management (2 weeks)
- • File and I/O Systems (2 weeks)
- • Security and Protection (1 week)
- • Networking and Distributed Systems (2 weeks)
- • Objects and Naming (1 week)
- • Window and Display Services (0.5 weeks)

Text

- Required
  - *Operating System Concepts* 7th Edition, Siberschatz, Galvin and Gagne, John Wiley 2005.

Programming Projects:

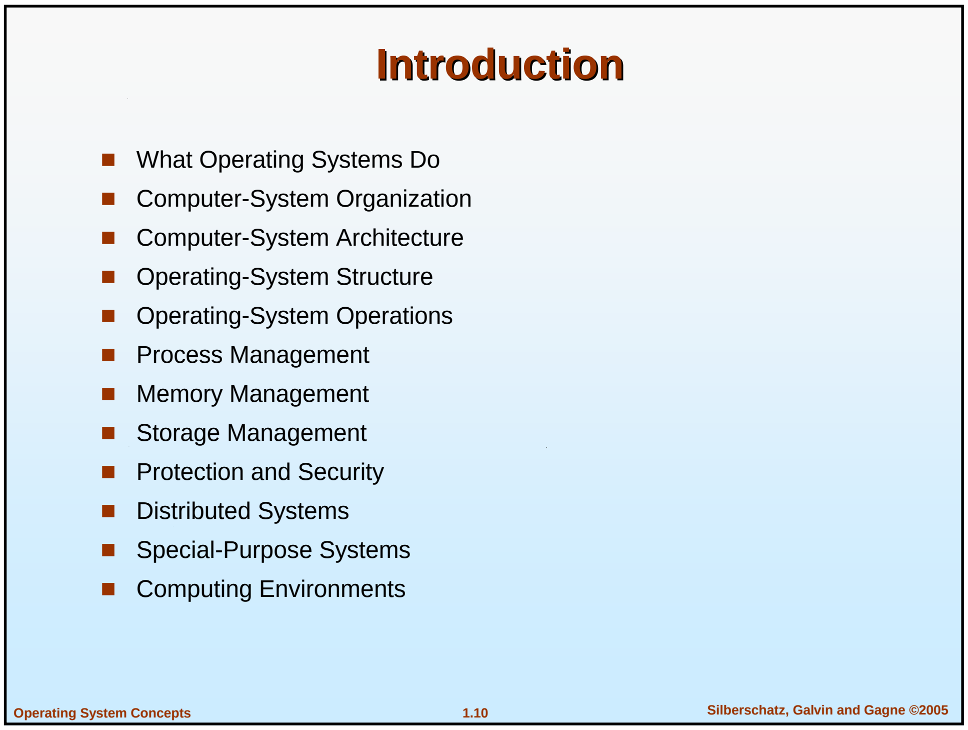
- Understanding operating system concepts is a hands-on activity. This class will include several substantial programming projects that will require students to read and understand provided code, write new modules, and debug the resulting system. The programming assignments will be time consuming and students taking this class should plan their class schedules accordingly.
- The instructor reserves the right to fail, regardless of overall numeric score, students who do not submit a good faith attempt to complete all programming assignments.



Grading

Final Exam	30%
Midterms (2 each worth 15%)	30%
Programming Assignments	40%

-
- *Exams:*
- Midterm #1 -
- Midterm #2 -
- Final



Introduction

- What Operating Systems Do
- Computer-System Organization
- Computer-System Architecture
- Operating-System Structure
- Operating-System Operations
- Process Management
- Memory Management
- Storage Management
- Protection and Security
- Distributed Systems
- Special-Purpose Systems
- Computing Environments

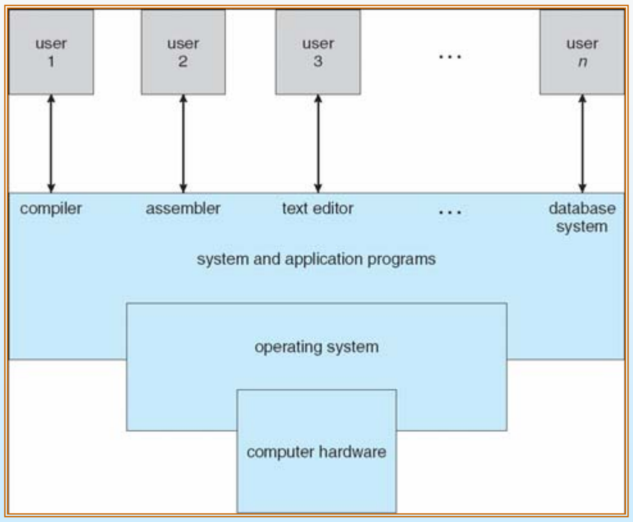
Objectives

- To provide a grand tour of the major operating systems components
- To provide coverage of basic computer system organization

What Operating Systems Do

- Computer system can be divided into four components
  - Hardware
    - CPU, memory, I/O devices
  - Operating system
    - Controls and coordinates use of hardware among various applications and users
  - Application programs
    - Word processors, compilers, web browsers
  - Users

Four Components of a Computer System



User View of a Computer

- Varies according to the interface being used
- Most systems designed for one user monopolizing its resources
  - OS maximizes the work (or play) user is performing
  - OS designed mostly for ease of use, not for resource utilization
- Some users interface to mainframe or minicomputer
  - OS is designed to maximize resource use (CPU, memory, I/O)
- Some users set at workstations connected to networks of servers
  - Dedicated and shared resources
  - OS compromises between individual usability and resource utilization
- Handheld systems have OS designed for individual usability
- Embedded systems designed to run without user intervention

System View of a Computer

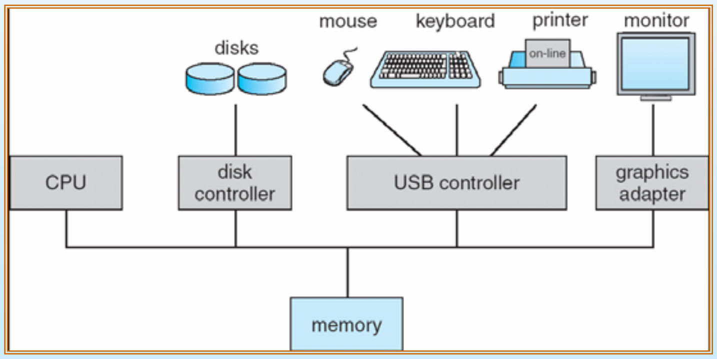
- OS is program most involved with the hardware
- OS is a **resource allocator**
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- OS is a **control program**
  - Controls execution of programs to prevent errors and improper use of the computer

Defining Operating Systems

- No universally accepted definition
- “Everything a vendor ships when you order an operating system” is good approximation
  - But varies wildly
- “The one program running at all times on the computer” is the one generally used in this course
  - This is the **kernel**
  - Everything else is either a system program (ships with the operating system) or an application program

Computer System Organization

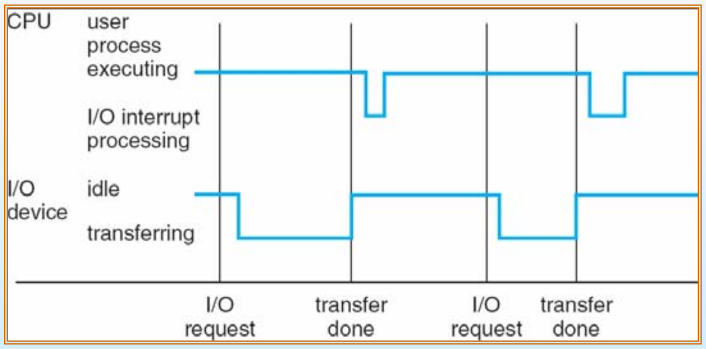
- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles



Computer Startup and Execution

- **bootstrap program** is loaded at power-up or reboot
  - Typically stored in ROM or EEPROM, generally known as **firmware**
  - Initializates all aspects of system
  - Loads operating system kernel and starts execution
- Kernel runs, waits for event to occur
  - **Interrupt** from either hardware or software
    - Hardware sends trigger on bus at any time
    - Software triggers interrupt by **system call**
    - Stops current kernel execution, transfers execution to fixed location
      - Interrupt service routine executes and resumes kernel where interrupted
      - Usually a service routine for each device / function
        - **Interrupt vector** dispatches interrupt to appropriate routine

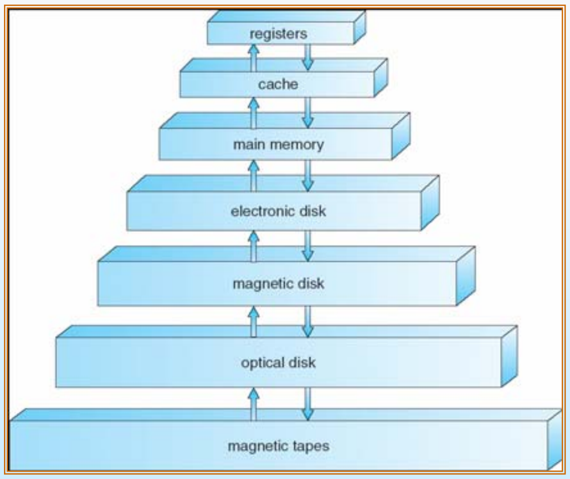
Interrupt Timeline



Storage Structure

- Programs must be in main memory (RAM) to execute
- **Von-Neumann** architecture
  - Load instruction from memory into **instruction register**
  - Operands fetched from memory to internal registers
  - Stores instructions and data in main memory
  - Result may be written back to main memory
- Main memory usually not large enough to hold all programs and data
- Main memory is *volatile* – loses contents on power loss
- **Secondary storage** holds large quantities of data, permanently
  - Actually, a hierarchy of storage varying by speed, cost, size and volatility

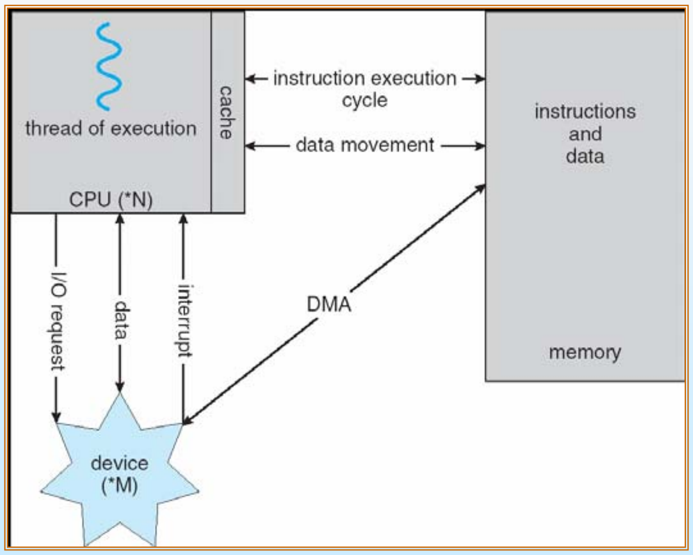
Storage-Device Hierarchy



I/O Structure

- Storage is one of many types of I/O devices
- Each device connected to a controller
  - Some controllers provide a bus for one or more devices (i.e. SCSI)
  - Device driver for each device controller
    - Knows details of controller
    - Provides uniform interface to kernel
- I/O operation
  - Device driver loads controller registers appropriately
  - Controller examines registers, executes I/O
  - Controller interrupts to signal device driver that I/O completed
  - High overhead for moving bulk data (i.e. disk I/O)
- **Direct Memory Access (DMA)**
  - Device controller transfers block of data to/from main memory
  - Interrupts when block transfer completed

How a Modern Computer System Works

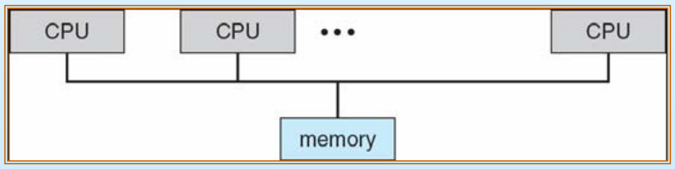


Computer-System Architecture

- Single-processor system
  - From PDAs to mainframes
  - Almost all have special-purpose processors for graphics, I/O
    - Not considered multiprocessor
- Multi-processor systems
  - Increase throughput
  - Economy of scale
  - Increased reliability
    - Some are **fault tolerant**
  - **Asymmetric multiprocessing**
    - Each processor assigned a specific task
  - **Symmetric multiprocessing (SMP)** most common
    - All processors perform tasks within the OS

Symmetric Multiprocessing Architecture

- Requires careful I/O management
- Virtually all modern OSes support SMP
- Multi-core CPU chips becoming coming – multiple compute cores on one chip
- Blade servers include chassis that hold multiple blades
  - Blades are uni- or multi-CPU, each running its own OS instance



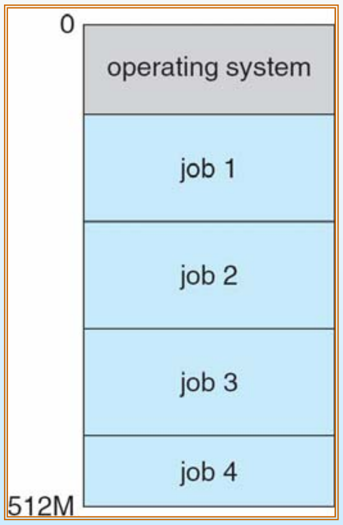
Clustered Systems

- Composed of two or more individual systems coupled together via a LAN or interconnect
- Provides high-availability by moving applications between nodes (computers in the cluster) if a node fails
- **Asymmetric clustering** has one node active and the other monitoring and waiting
- **Symmetric clustering** has all nodes active, able to take one more programs if one fails
- Clusters cannot allow multiple nodes to access the same data unless a D**istributed Lock Manager (DLM)** plays traffic cop
- Clusters can include dozens of nodes, but typically only two or a few
  - Need shared storage, usually provided by a **Storage Area Network (SAN)**

Operating System Structure

- OS provides a structure in which programs execute
- **Multiprogramming** needed for efficiency
  - Singer user cannot keep CPU and I/O devices busy at all times
  - 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 has to 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 Multiprogrammed System

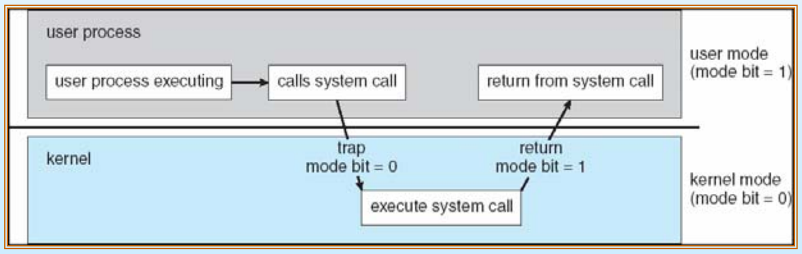


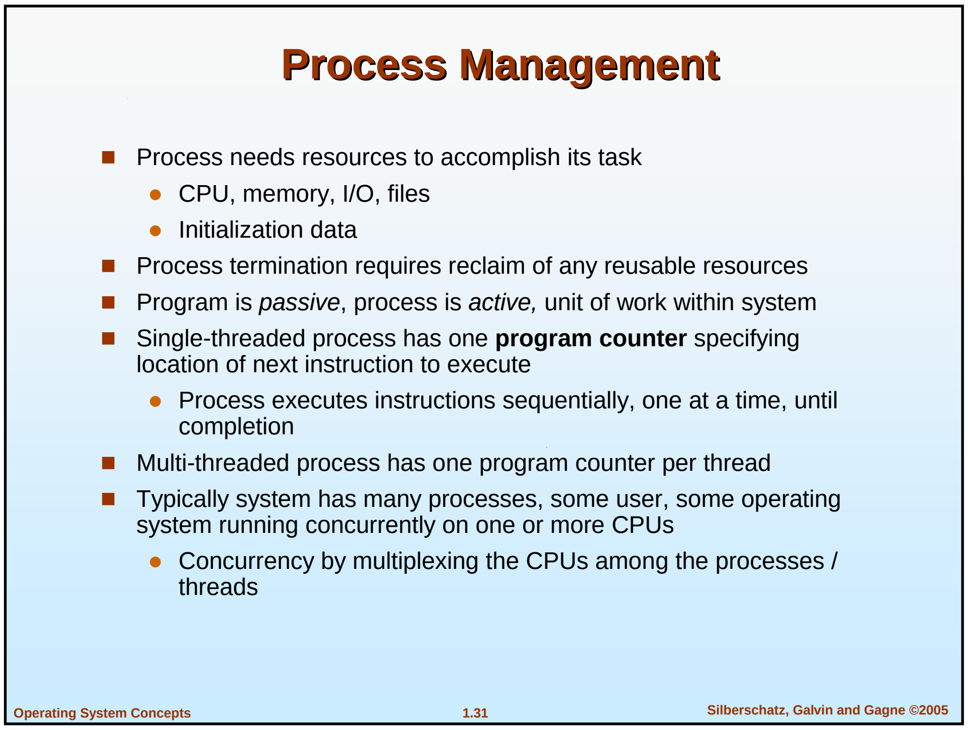
Operating-System Operations

- Interrupt driven by hardware
- Software error or request creates **exception** or **trap**
  - Division by zero, request for operating system service
- Other process problems include infinite loop, processes modifying each other or the operating system
- **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

Transition from User to Kernel Mode

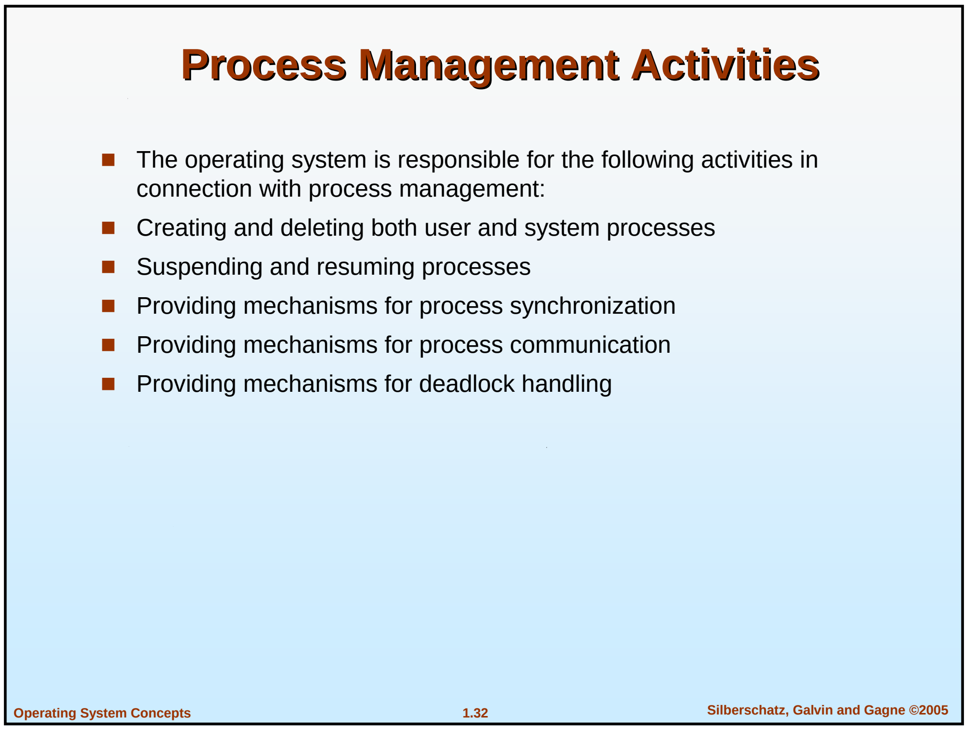
- Timer to prevent infinite loop / process hogging resources
  - Set interrupt after specific period
  - Operating system decrements counter
  - When counter zero
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time





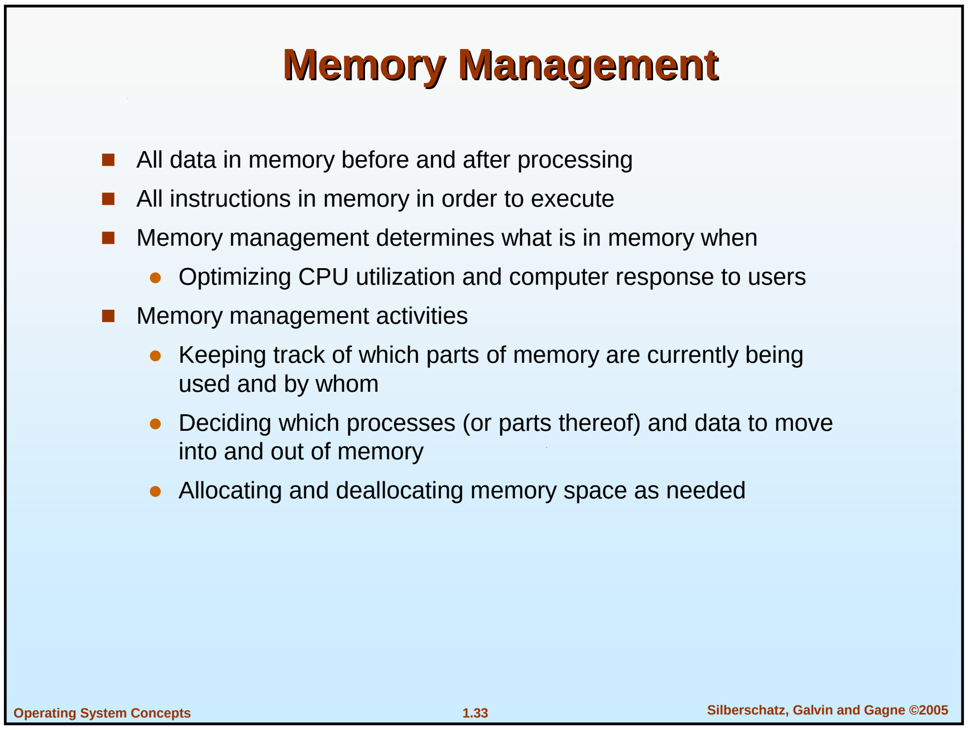
Process Management

- Process needs resources to accomplish its task
  - CPU, memory, I/O, files
  - Initialization data
- Process termination requires reclaim of any reusable resources
- Program is *passive*, process is *active,* unit of work within system
- Single-threaded process has one **program counter** specifying location of next instruction to execute
  - Process executes instructions sequentially, one at a time, until completion
- Multi-threaded process has one program counter per thread
- Typically system has many processes, some user, some operating system running concurrently on one or more CPUs
  - Concurrency by multiplexing the CPUs among the processes / threads



Process Management Activities

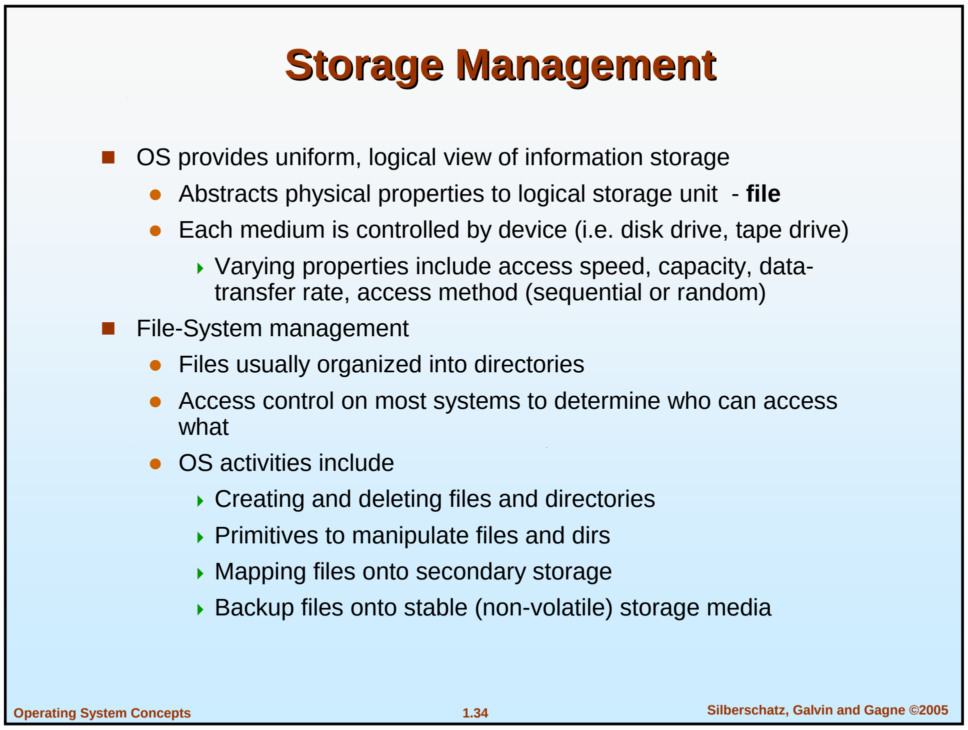
- The operating system is responsible for the following activities in connection with process management:
- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling



Memory Management

- All data in memory before and after processing
- All instructions in memory in order to execute
- Memory management determines what is in memory when
  - Optimizing CPU utilization and computer response to users
- 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

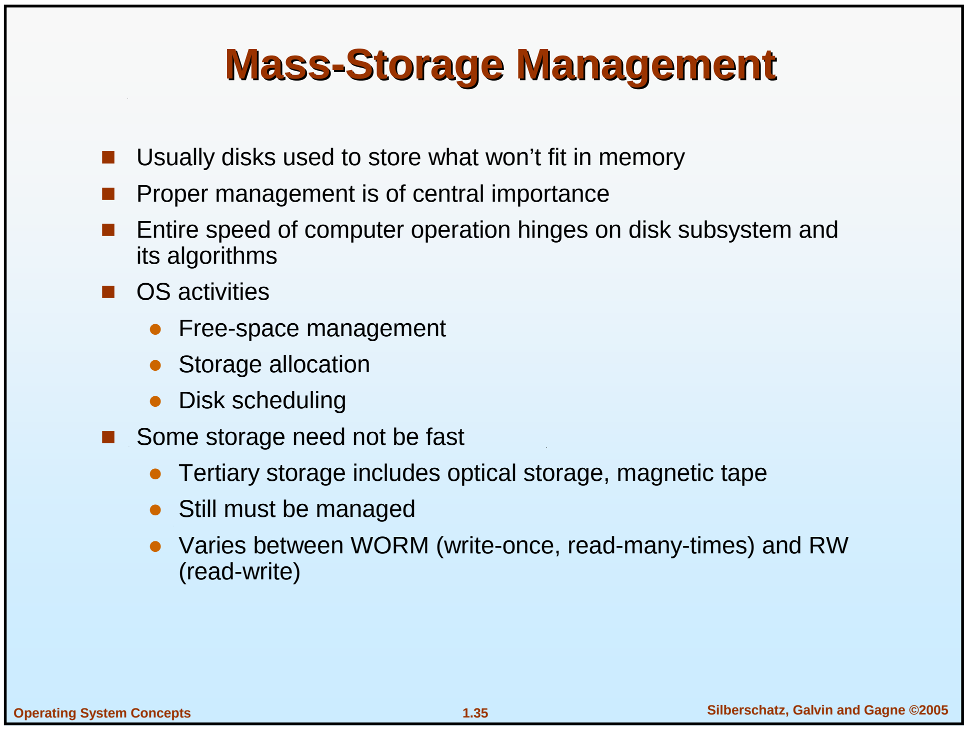
Operating System Concepts 1.33 Silberschatz, Galvin and Gagne ©2005



Storage Management

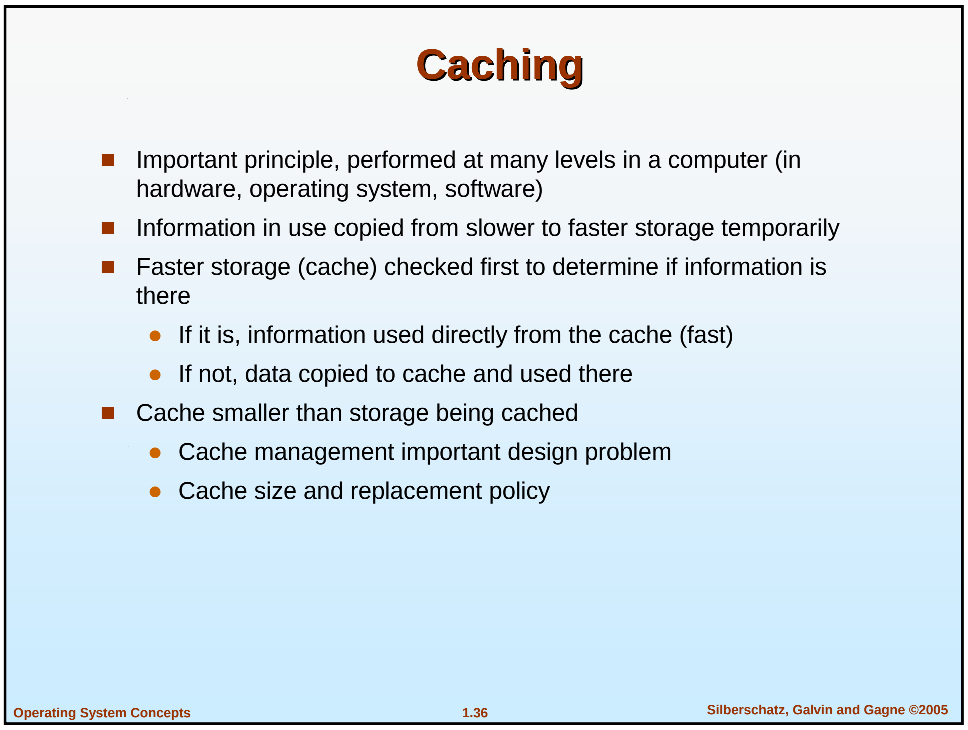
- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - **file**
  - Each medium is controlled by device (i.e. disk drive, tape drive)
    - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- 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

Operating System Concepts 1.34 Silberschatz, Galvin and Gagne ©2005



Mass-Storage Management

- Usually disks used to store what won't fit in memory
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystem and its algorithms
- 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
  - Varies between WORM (write-once, read-many-times) and RW (read-write)

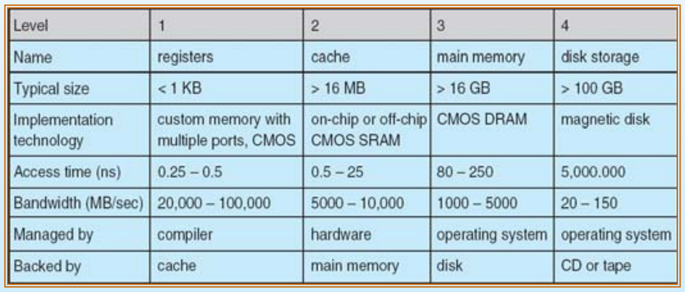


Caching

- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- 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
- Cache smaller than storage being cached
  - Cache management important design problem
  - Cache size and replacement policy

Performance of Various Levels of Storage

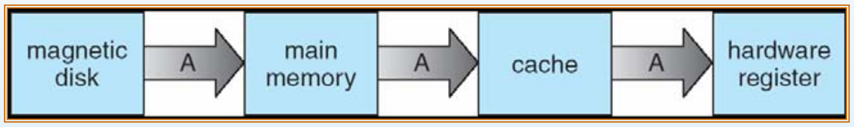
- Movement between levels of storage hierarchy can be explicit or implicit



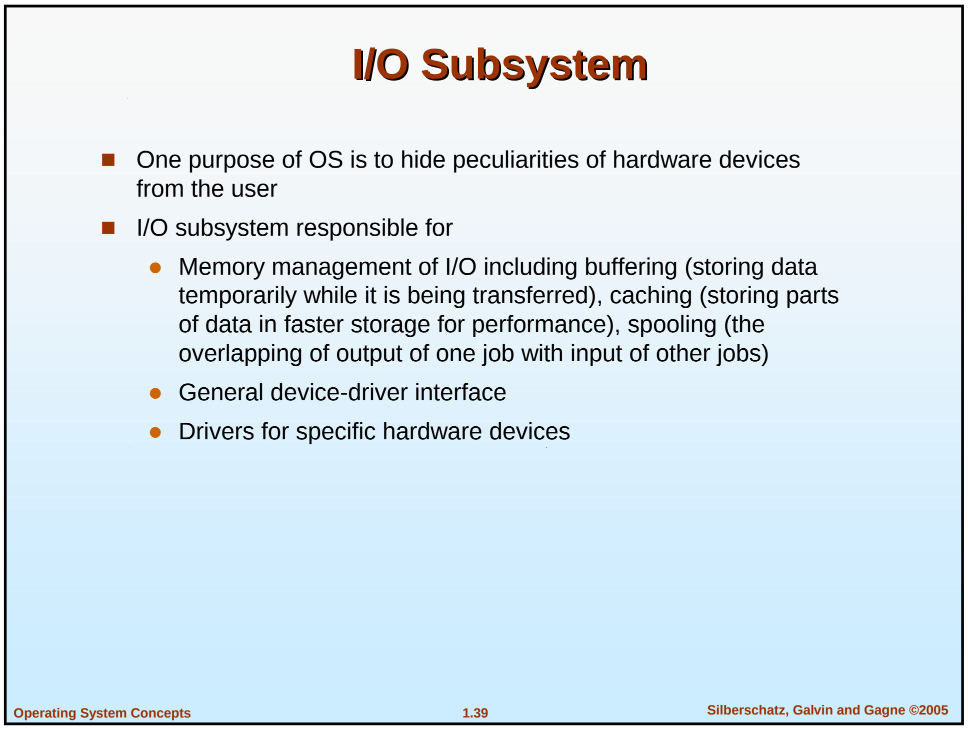
Level	1	2	3	4
Name	registers	cache	main memory	disk storage
Typical size	< 1 KB	> 16 MB	> 16 GB	> 100 GB
Implementation technology	custom memory with multiple ports, CMOS	on-chip or off-chip CMOS SRAM	CMOS DRAM	magnetic disk
Access time (ns)	0.25 – 0.5	0.5 – 25	80 – 250	5,000.000
Bandwidth (MB/sec)	20,000 – 100,000	5000 – 10,000	1000 – 5000	20 – 150
Managed by	compiler	hardware	operating system	operating system
Backed by	cache	main memory	disk	CD or tape

Migration of Integer A from Disk to Register

- Multitasking environments must be careful to use most recent value, not matter where it is stored in the storage hierarchy

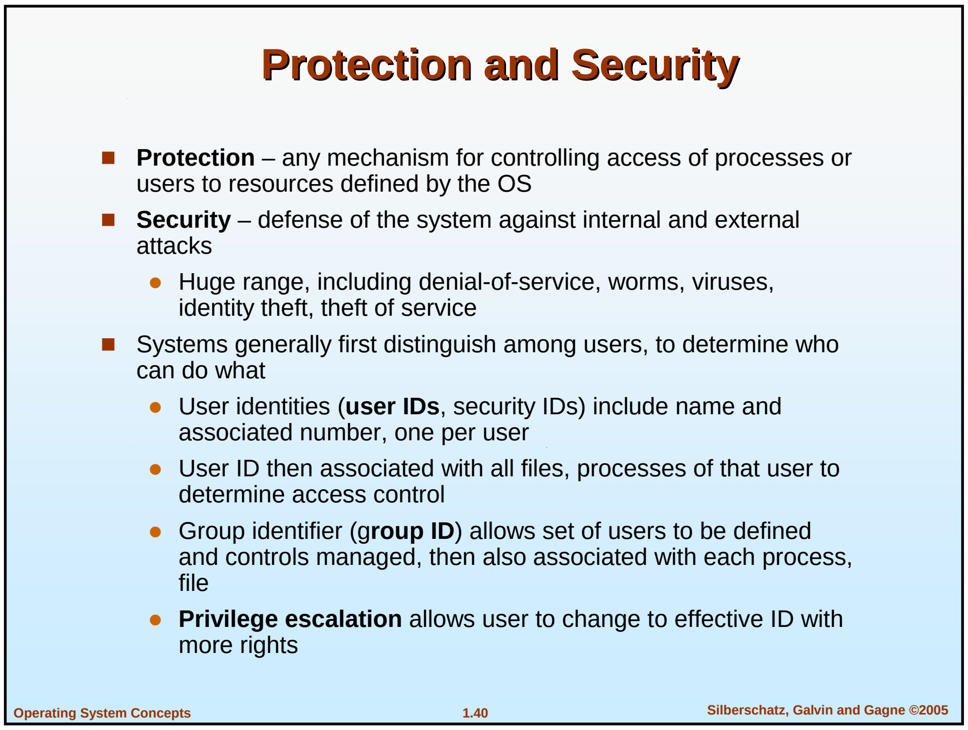


- Multiprocessor environment must provide cache coherency in hardware such that all CPUs have the most recent value in their cache
- Distributed environment situation even more complex
  - Several copies of a datum can exist
  - Various solutions covered in Chapter 17



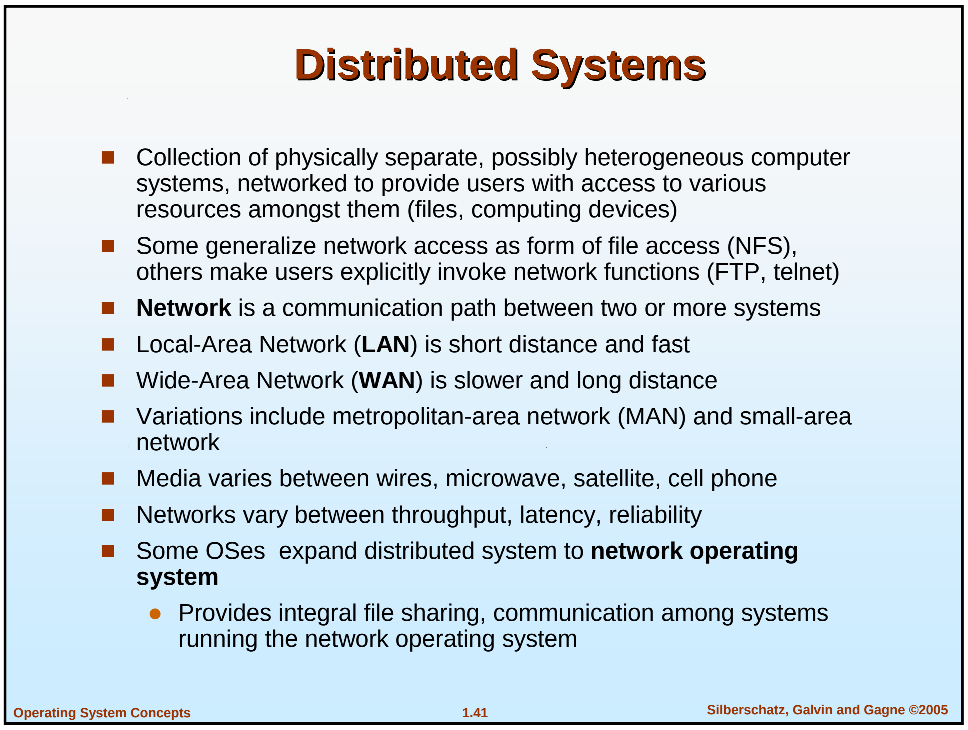
I/O Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user
- I/O subsystem responsible for
  - Memory management of I/O including buffering (storing data temporarily while it is being transferred), caching (storing parts of data in faster storage for performance), spooling (the overlapping of output of one job with input of other jobs)
  - General device-driver interface
  - Drivers for specific hardware devices



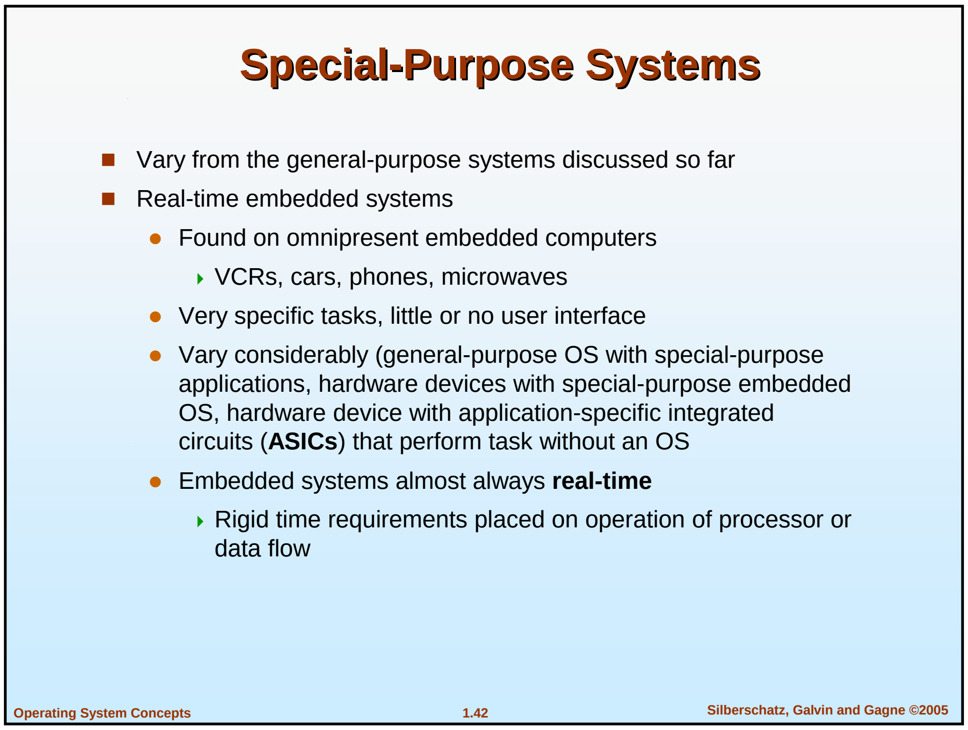
Protection and Security

- **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
- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control
  - Group identifier (g**roup ID**) allows set of users to be defined and controls managed, then also associated with each process, file
  - **Privilege escalation** allows user to change to effective ID with more rights



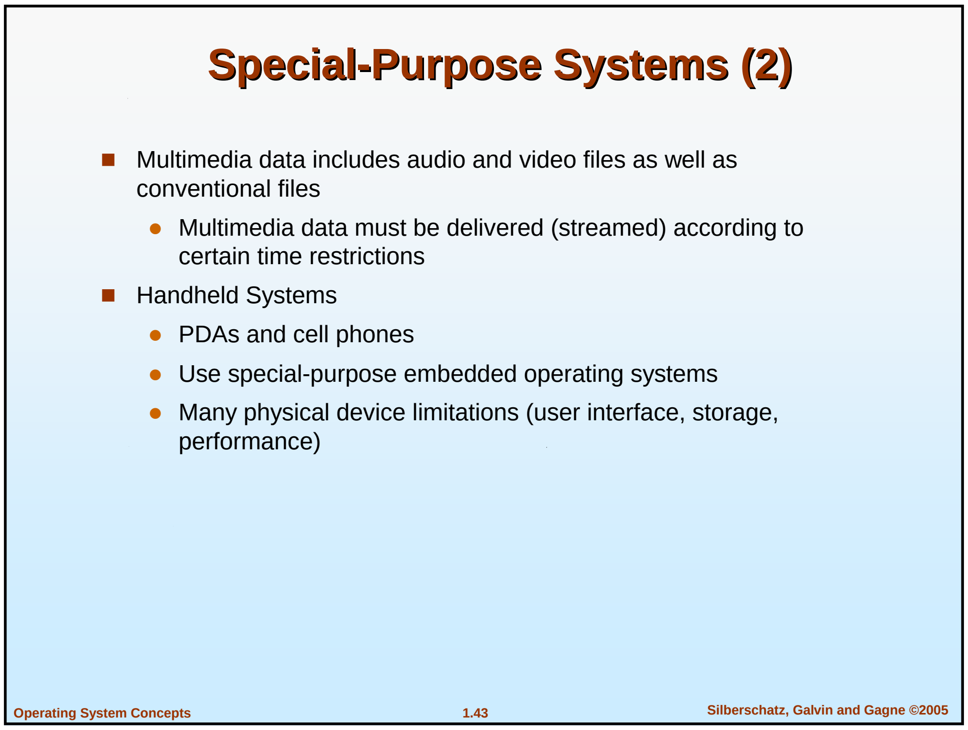
Distributed Systems

- Collection of physically separate, possibly heterogeneous computer systems, networked to provide users with access to various resources amongst them (files, computing devices)
- Some generalize network access as form of file access (NFS), others make users explicitly invoke network functions (FTP, telnet)
- **Network** is a communication path between two or more systems
- Local-Area Network (**LAN**) is short distance and fast
- Wide-Area Network (**WAN**) is slower and long distance
- Variations include metropolitan-area network (MAN) and small-area network
- Media varies between wires, microwave, satellite, cell phone
- Networks vary between throughput, latency, reliability
- Some OSes expand distributed system to **network operating system**
  - Provides integral file sharing, communication among systems running the network operating system



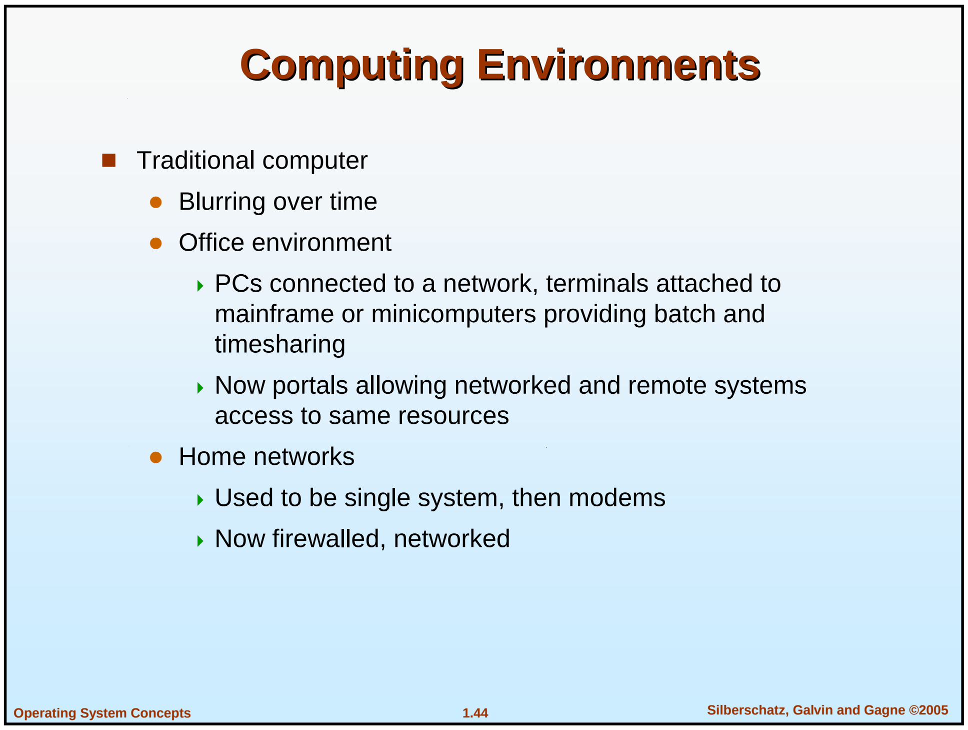
Special-Purpose Systems

- Vary from the general-purpose systems discussed so far
- Real-time embedded systems
  - Found on omnipresent embedded computers
    - VCRs, cars, phones, microwaves
  - Very specific tasks, little or no user interface
  - Vary considerably (general-purpose OS with special-purpose applications, hardware devices with special-purpose embedded OS, hardware device with application-specific integrated circuits (**ASICs**) that perform task without an OS
  - Embedded systems almost always **real-time**
    - Rigid time requirements placed on operation of processor or data flow



Special-Purpose Systems (2)

- Multimedia data includes audio and video files as well as conventional files
  - Multimedia data must be delivered (streamed) according to certain time restrictions
- Handheld Systems
  - PDAs and cell phones
  - Use special-purpose embedded operating systems
  - Many physical device limitations (user interface, storage, performance)

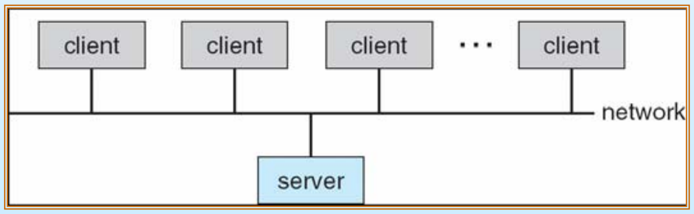


Computing Environments

- Traditional computer
  - Blurring over time
  - Office environment
    - PCs connected to a network, terminals attached to mainframe or minicomputers providing batch and timesharing
    - Now portals allowing networked and remote systems access to same resources
  - Home networks
    - Used to be single system, then modems
    - Now firewalled, networked

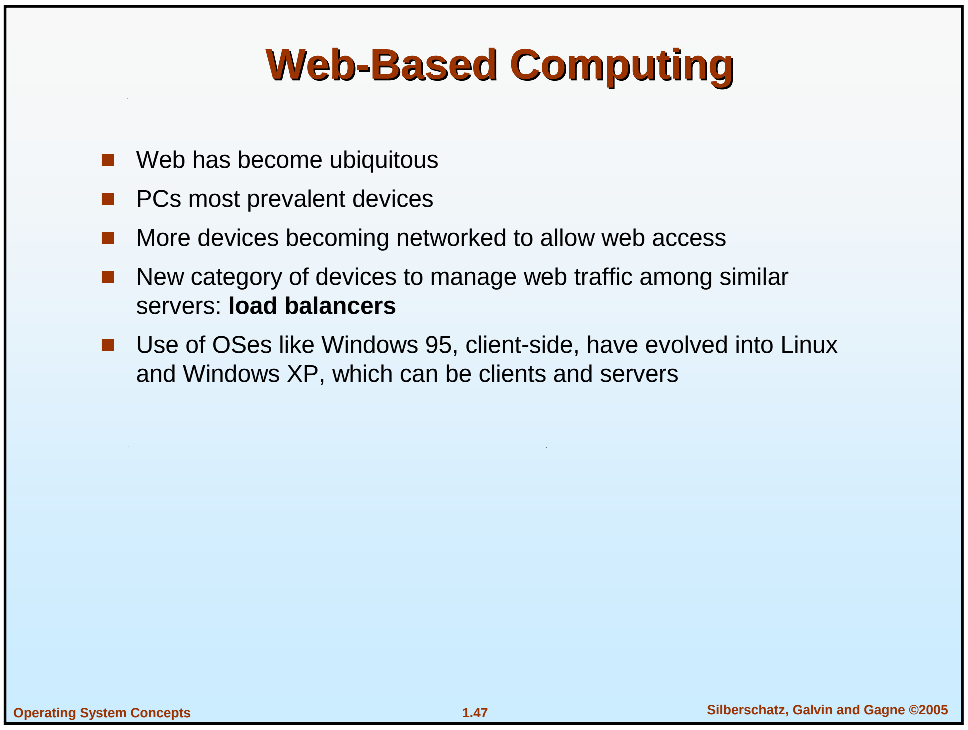
Computing Environments (Cont.)

- Client-Server Computing
  - Dumb terminals supplanted by smart PCs
  - Many systems now **servers**, responding to requests generated by **clients**
    - **Compute-server** provides an interface to client to request services (i.e. database)
    - **File-server** provides interface for clients to store and retrieve files



Peer-to-Peer Computing

- Another model of distributed system
- P2P does not distinguish clients and servers
  - Instead all nodes are considered peers
  - May each act as client, server or both
  - Node must join P2P network
    - Registers its service with central lookup service on network, or
    - Broadcast request for service and respond to requests for service via *discovery protocol*
  - Examples include *Napster* and *Gnutella*



Web-Based Computing

- Web has become ubiquitous
- PCs most prevalent devices
- More devices becoming networked to allow web access
- New category of devices to manage web traffic among similar servers: **load balancers**
- Use of OSes like Windows 95, client-side, have evolved into Linux and Windows XP, which can be clients and servers