CS3300: Operating Systems

P.Krishna Reddy
pkreddy@iiit.net

Outline

- **Introduction**
  - **What is an Operating System ?**
- Course topics and grading
- History, development and concepts of Oss (Stallings, 2.2 and 2.3)
- Different kinds of Computer Systems
  (Silberschatz, Chapter 1)
- Concept of virtual computer
  (Crowley, Chapter 1)

Questions

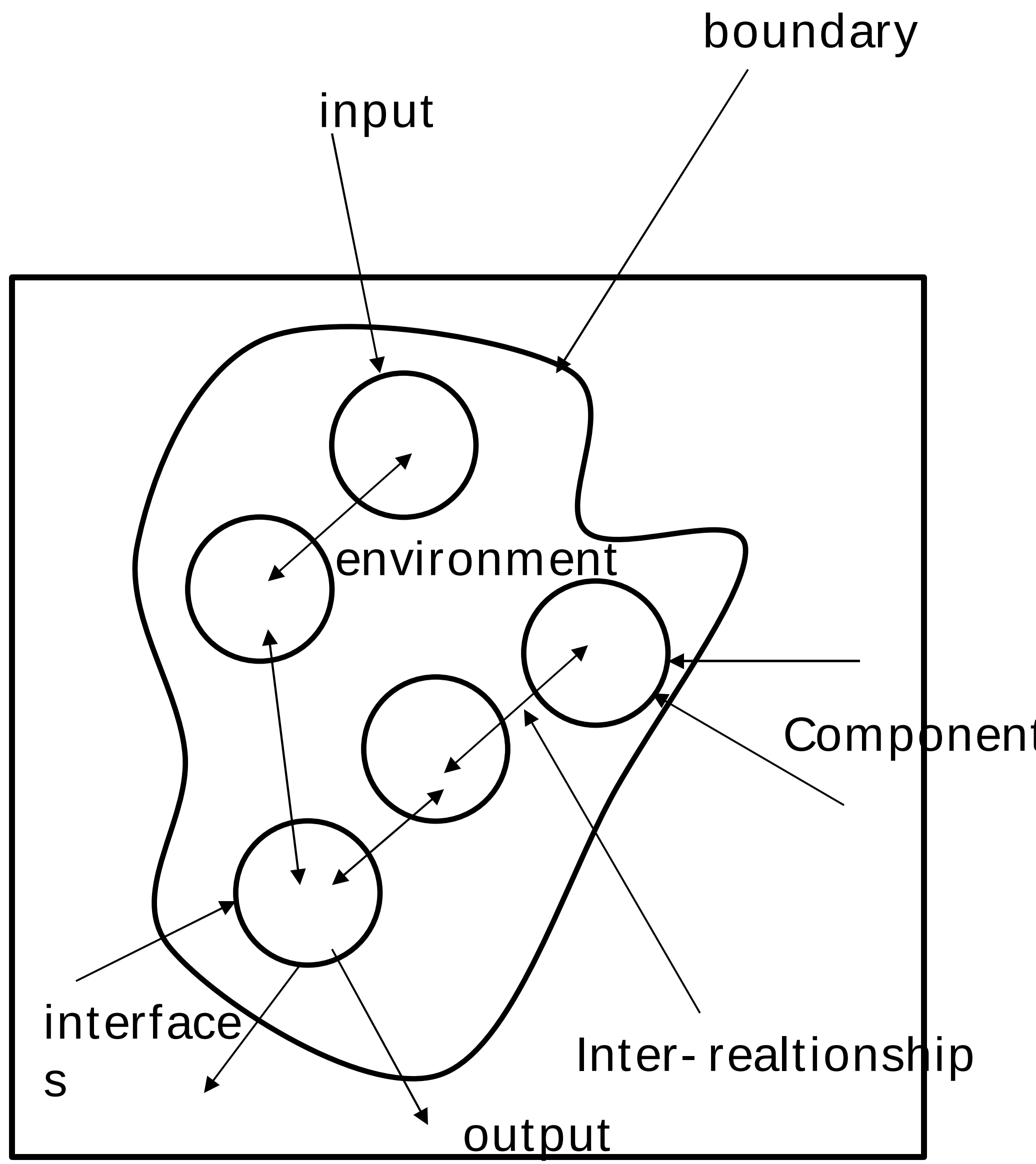
- What is a system ?
- What is an operating system ?
- What is a computer operating system ?

What is a system ?

- A system is an inter- related set of components with an identifiable boundary working together for some purpose.
- System can be natural of fabricated
  - Natural systems: human body or solar system
  - Fabricated systems: cycle, bus, computer, government, boat

System

- A system has nine characteristics
  - Components
  - Inter-related components
  - A boundary
  - A purpose
  - An environment
  - Interfaces
  - Input
  - Output
  - Constraints.



A general depiction of a system

Characteristics…

- Components:
  - A system is made up of components
  - A component is either an irreducible part or aggregation of parts that make- up a system. A component is also called a sub-system.
- Interrelated:
  - The components of interrelated
  - Dependence of one subsystem on one or more subsystems.

Characteristics…

- Boundary (Scope):
  - A system has a boundary, within which all of its components are contained and which establishes the limits of a separating the system from other systems.
- Purpose
  - The overall goal of function of a system. The system's reason for existing.

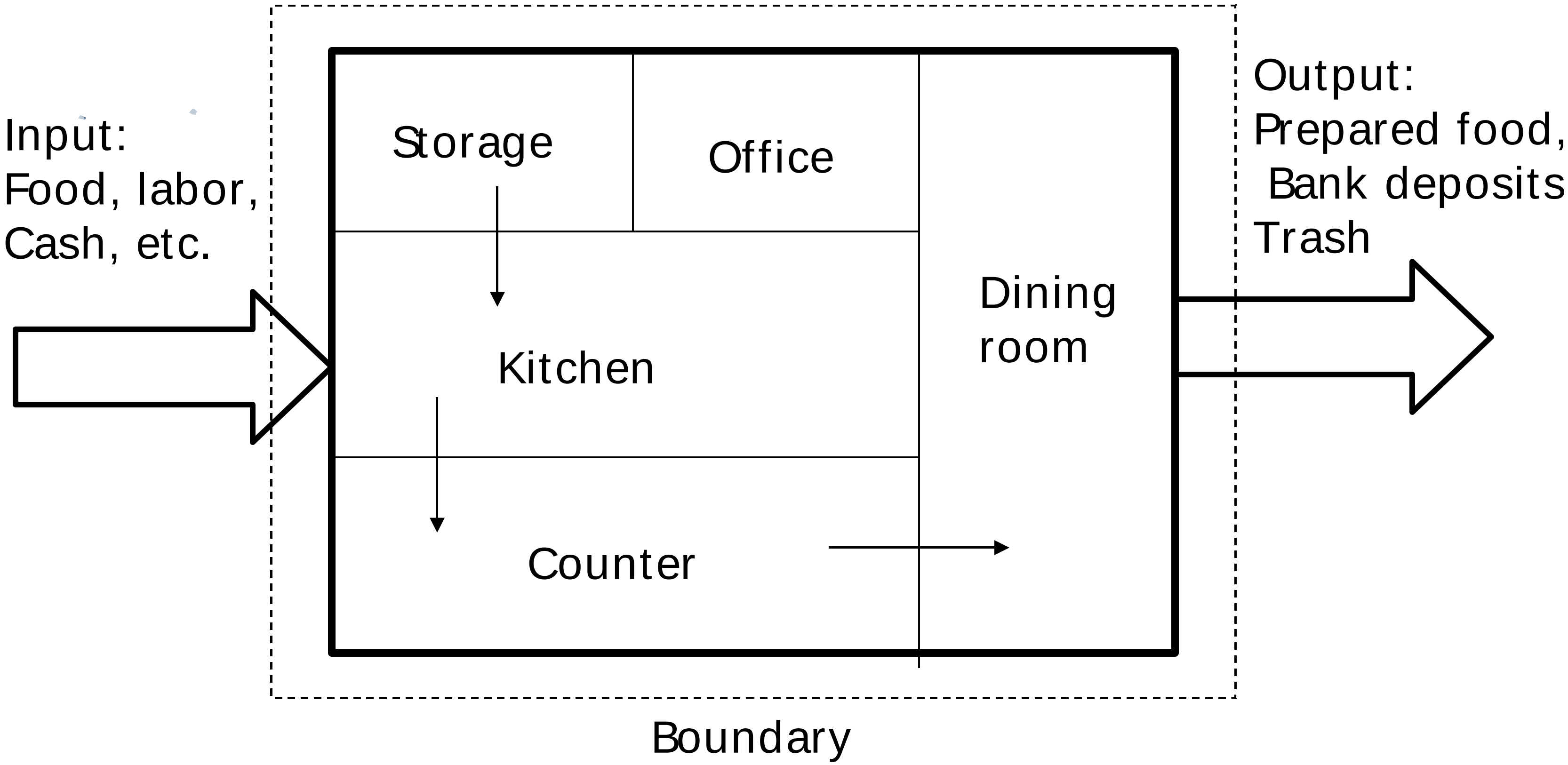
Characteristics…

- Environment
  - Everything external to the system that interacts with the system.
- Interface
  - Point of contact where a system meets its environment or subsystems meet each other.
- Constraint:
  - A limit what a system can accomplish: Capacity, speed or capabilities.

Characteristics…

- Input
  - Whatever a system takes from its environment in order to fulfill its purpose.
- Output:
  - Whatever a system returns to its environment in order to fulfill its purpose.

Example: A fast food restaurant as a system



Environment: Customers, food distributors, banks etc

→ Represents an inter-relationship

Constraints: Popular foods, Health dept., constraints of storage

Important System Concepts

- Decomposition
- Modularity
- Coupling
- Cohesion

Decomposition (Divide and Conquer)

- It deals with being able to break down a system into its components.
- Decomposition results in smaller and less complex pieces that are easier to understand than larger, complex pieces.
- Decomposing a system also allows to focus on one particular part of a system, making easier to think of how to modify that part independently of the entire system.

Modularity

- Modularity refers to dividing a system up into chunks or modules of a relatively uniform size.
- You can replace or add any other module (or a component) without effecting the rest of the system.
- It is a design strategy in which system is composed of relatively small and autonomous routines fit together.

Coupling

- Coupling is the extent to which subsystems are dependent on each other.
- Subsystems should be as independent as possible.
- If a subsystem fails and other subsystems are highly dependent on it, the others will either fail themselves or have problems in functioning.

Cohesion

- The extent to which a system or a subsystem performs a single function.

What is an operating system ?

- Operating system is a system.
- Operating system is a subsystem of any tool.
- Each tool constitutes machine part and operating part.
- The operating part of a tool is called operating system of that tool.
- The purpose of operating system is to facilitate the operation of the underlying machine or tool.
- For some tools, operating system may not exist.
  - Example: Pen.
- For a machine, the operating system abstracts the machine part in terms of simple services by hiding the details of the machine. The OS can provide services to users or other subsystems.
- Examples of typical operating systems:
  - Car operating system, Telephone operating system, TV operating system and so on.

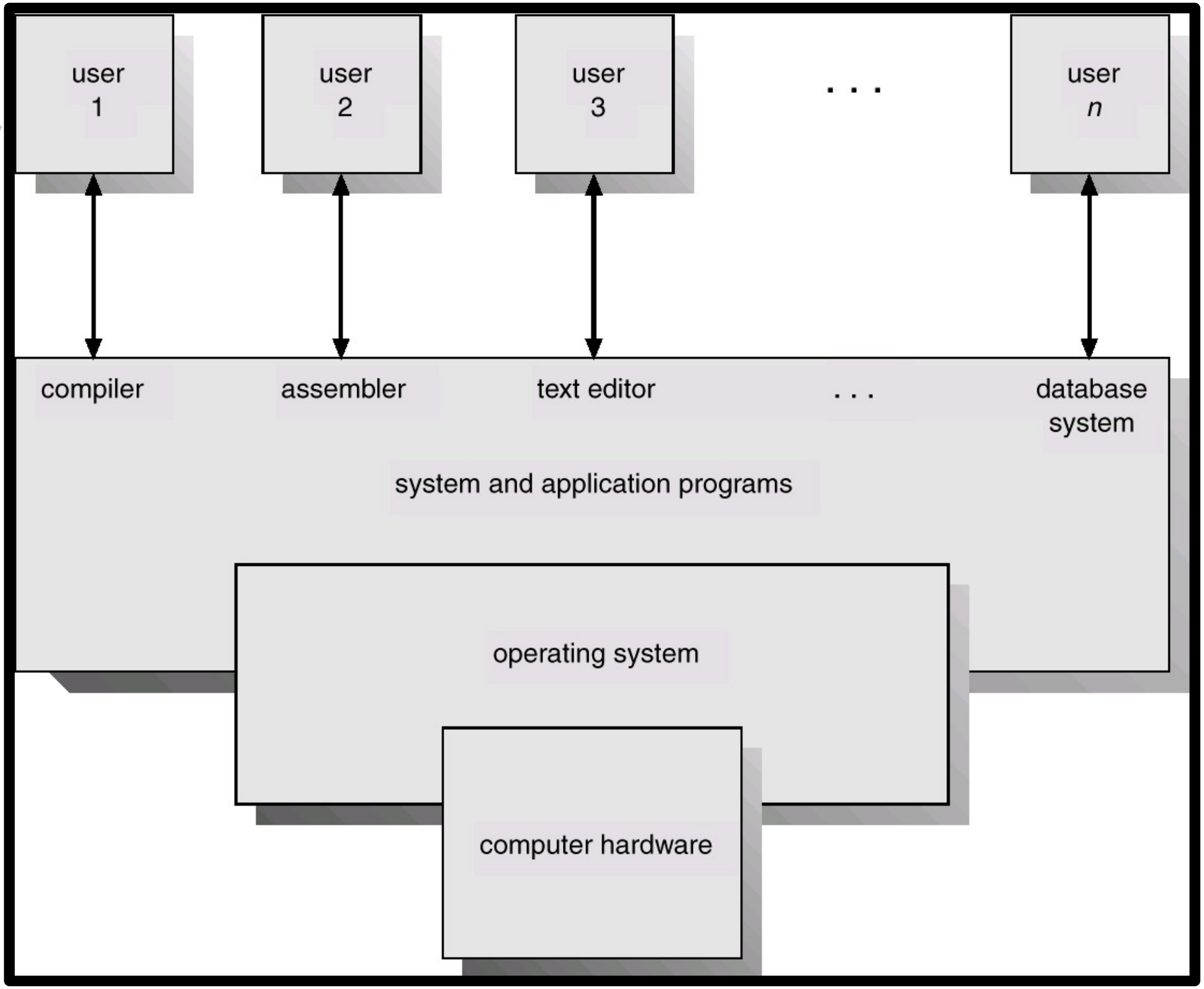
What is a computer operating system ?

- A computer is also a tool that contains machine part and operating part.
- The operating part of a computer is called Computer Operating System.
- For a computer, the operating system abstracts the underlying hardware in terms of simple services by hiding the details of the hardware. The OS can provide services to users or other subsystems.
- Examples of Computer operating systems:
  - WINDOWS NT, WINDOWS XP, Macintosh, UNIX, SOLARIS, LINUX and so on.
- In the rest of this course, operating system means computer operating system.

Computer System Components

1. **Hardware** – provides basic computing resources (CPU, memory, I/O devices).
2. **Operating system** – controls and coordinates the use of the hardware among the various application programs for the various users.
3. **Applications programs** – define the ways in which the system resources are used to solve the computing problems of the users (compilers, database systems, video games, business programs).
4. **Users** (people, machines, other computers).

Abstract View of System Components



What is an Operating System?...

- A program that acts as an intermediary between a user of a computer and the computer hardware.
- Operating system goals:
  - Make the computer system convenient to use.
  - Use the computer hardware in an efficient manner.

Operating System Definitions…

- **Resource allocator** – manages and allocates resources.
  - Resources: CPU time, Memory Space, file storage space, I/O devices and son on.
- **Control program** – controls the execution of user programs and operations of I/O devices .
- **Kernel** – the one program running at all times (all else being application programs).
- The two goals, efficiency and convenience are sometimes contradictory
- Much of OS theory is concentrates optimal use of resources.

Outline

- Introduction
  - What is an Operating System ?
- **Course topics and grading**
- History, development and concepts of Oss (Stallings, 2.2 and 2.3)
- Different kinds of Computer Systems

  (Silberschatz, Chapter 1)
- Concept of virtual computer

  (Crowley, Chapter 1)

Objectives

- The main objective is to understand the operational part of any computer.
- Understanding the general principles of OS design.
  - Focus on general-purpose, multi-user, uni-processor systems.
  - Emphasis on widely applicable concepts rather than any specific features of any specific OS.
- Understanding problems, solutions and design choices.
- Understanding the structure of specific OSs: UNIX, LINUX, WINDOWS2000

Course topics

- Introduction (6 hours)
- Process management (8)
- CPU scheduling (4)
- Process synchronization (6)
- Deadlocks (3)
- Memory management (3)
- Virtual memory (3)
- File systems (3)
- Protection and security (3)
- Overview of classical operating systems. (3)

References

- Text books:
  - Silberschatz, A, Galvin, P, Gagne, G. Operating System Concepts, Addison- Wesley (5th or latest edition).
- Other BOOKS::
  - Charles Crowley, Operating Systems: A design-oriented approach, Tata McGraw- Hill, 1997.
  - William Stallings, Operating systems, Prentice-Hall, 1998.
  - Tanenbaum, A., Modern Operating Systems, Prentice- Hall, second edition, 2000.

LAB WORK

- Experiments will be given on Linux.
- Sample experiments
  - Shell writing for MSDOS
  - Process Communication
    - Bounded buffer
      - Semaphores, shared memory based communication
      - Threads
  - Replace “ls” with lookup
  - Command line for /proc
  - Printer driver
  - Adding a new system call to linux
- The lab is very intensive. Please do not ask for the extension of deadline. Each experiment will be evaluated.

OUTCOME

- After completing the course, the students will understand
  - the fundamental concepts of several computer operating systems such as SOLARIS, LINUX, WINDOWS and MAC.
  - the solutions/ options to interesting problems which have been encountered by the designers of preceding operating systems.
  - the critical role of operation system in designing several computer based systems like database systems, expert systems, web based information systems, real-time systems, embedded systems and so on.

GRADING

- MIDTERM I: 15 %
- MIDTERM II: 15 %
- ENDSEM EXAM: 40 %
- LAB: 30%

Outline

- Introduction
  - What is an Operating System ?
- Course topics and grading
- **History, development and concepts of Oss (Stallings, 2.2 and 2.3)**
- Different kinds of Computer Systems
  (Silberschatz, Chapter 1)
- Concept of virtual computer
  (Crowley, Chapter 1)

Early systems (Serial processing)

- **1940-50:**
  - The programmer interacted directly with the computer hardware.
  - Display light, switches, printer, card reader.
  - No OS.
  - Error is displayed through lights.
- **Problems:**
  - Scheduling Users spend lots of time at the computer.
    - Signup sheet was used.
  - Job Setup time
    - Loading and compiling
      - Mounting and Un-mounting of tapes
      - Setting up of card desks
  - Libraries of functions, linkers, loaders, debuggers, and I/O driver routines were available for all the users.

Early Systems…

- Early computers were (physically) large machines run from a console.
- The programmer would operate the program directly from the console.
  - The program is loaded to the memory from panel of switches, paper tape, and from punched cards.
- As time went on, additional software and hardware were developed.
  - Card readers, line printers, and magnetic tape became common place.
  - Libraries, loaders, and common functions were created.
    - Software reusability.

Early Systems...

- The routines that performed I/O were especially became important.
- Device driver: A special subroutine was written for each I/O device.
  - A device driver knows how the buffers, flags, registers, control bits, and status bits for a particular device should be used.
  - Device driver is written once and called from the library.
- Later, compilers for FORTRAN, COBOL and other languages have appeared.
  - To prepare a FORTRAN program for execution:
    - Load the FORTRAN compiler
    - Mount compiler tape
    - Program would be read from the card reader.
    - Assembly program produced by the compiler would be linked to supporting library routines.
    - Finally, the binary code is ready to execute.
    - It would be loaded into the memory and debugged from the console.

Early Systems…

- Significant amount of setup time.
- Each job consisted of many separate steps:
  - Loading the FORTRAN compiler tape
  - Running the compiler
  - Unloading the compiler tape
  - Loading of assembler tape
  - Running assembler
  - Unloading the assemble tape
  - Loading the object program
  - Running the object program
- If error occurred during any step, you have to start over at the beginning.

Early Systems..

- The setup time was a real problem
- CPU is idle while tapes are being mounted or the programmer was operating the console.
- In the early days, few computers were available and they were expensive (millions of dollars).
  - + operational costs: power, cooling, programmers.
- Main question: how to increase the utilization of CPU ?

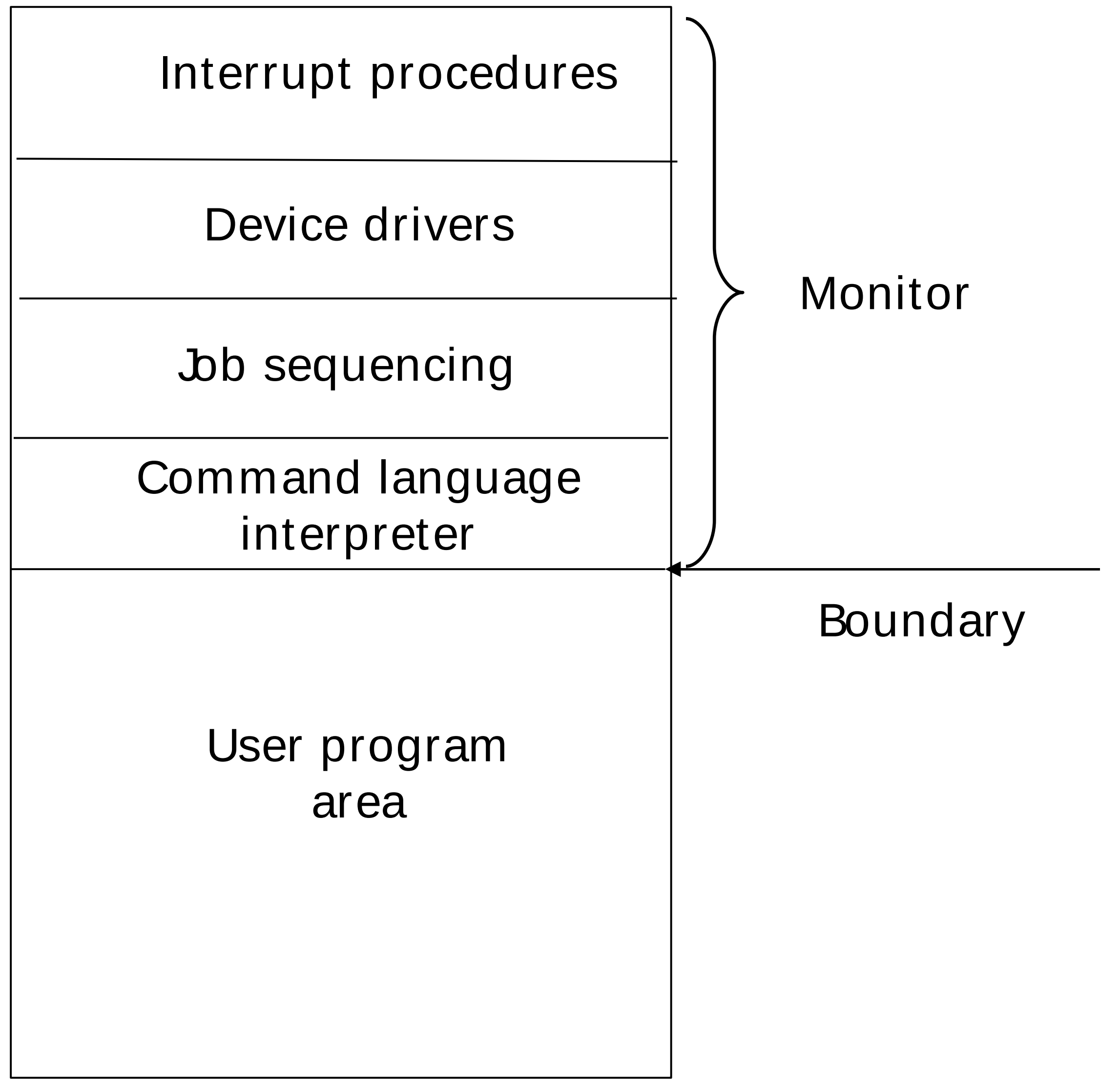
Early Systems…

- The solution was two fold.
- First, a professional computer operator was hired.
  - Once the program was finished, he operator could start next job.
  - The operator sets up the job, produces the dump, and starts the next job.
  - The set up time was reduced due to operator’s experience.
- Second, jobs with similar needs were batched together and run through the computer as a group.
  - For example, if there is a FORTRAN job, COBOL job, and FORTRAN job, two FORTRAN jobs were batched together.
- However, during transition time CPU sat idle.
- To overcome this idle time, people developed automatic job sequencing.
  - A first rudimentary OS was created
  - A small program called a **resident monitor** was developed.
  - The resident monitor always resided in memory.

Simple Batch Systems (early 1960s)

- In serial systems
  - Machines were very expensive
  - Wasting time was not acceptable.
- To improve usage, the concept of batch OS was developed.
- The main idea is the use of software known as monitor.
  - The user no longer has access to machine.
- The user submits the job (tape) to the operator.
- The operator batches the jobs together sequentially, places entire batch as an input device for use by the computer.

Memory Layout for a Simple Batch System



Resident monitor

Simple Batch Systems..

- At the beginning of any job, the corresponding subroutines and functions are loaded.
- The monitor reads the jobs one at a time from the input device.
- Algorithm:
  - The control is passed to the user's program.
    - Processor is fetching and executing user's instructions.
  - After completion, the control is returned to the monitor program
    - Processor is fetching and executing monitor instructions.

Features of Batch System

- The batch OS is simply a program. It relies on the ability of the processor to fetch instructions from various portions of main memory to seize and relinquish control.
- Hardware features:
  - **Memory protection**: While the user program is running, it must not alter the memory area containing the monitor.
  - If such is the case the processor hardware should detect the error and transfer control to monitor.
  - **Timer:** A timer is used to prevent the single job from monopolizing the system
  - **Privileged instructions**
    - Contains instructions that are only executed by monitor.
    - I/O instructions
    - If a program encounters them the control shifts through monitor..

Features of Batch System

- With batch OS the machine time alters between execution of user programs and execution of monitor.
- Two overheads
  - Machine time is consumed by the monitor.
  - Memory is consumed by the monitor.
- Still, they improved the performance over serial systems.

Problems with Batch System

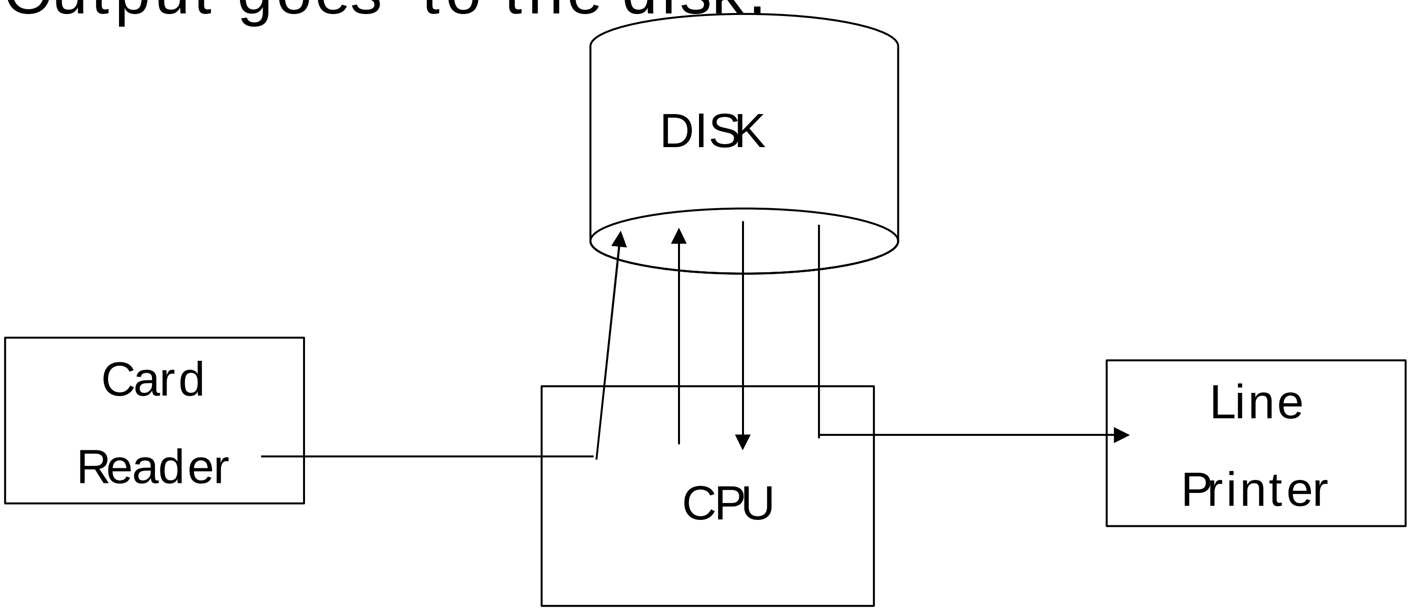
- CPU is idle
- Speed of mechanical devices is very slower than those of electronic devices.
- CPU works in a microsecond range
  - Thousands of instructions/ second
- A card reader may read 1200 cards per minute (20 cards per second)
- CPU speed has increased at a faster rate.
- Main perceived problem
  - Turn- around time: up to two days
  - CPU often underutilized
    - Most of the time was spent reading and writing from tape.

Resident monitor: summary

- Automatic job sequencing
  - Use of control cards
- Job control language
  - Commands
    - Mount this tape
    - Compile
    - Run
- OSs begin to be important.
  - IBM: Fortran monitor system
- Main perceived problems
  - Turn-around time
  - Inexpensive use of expensive hardware
  - CPU is still mostly idle.

Spooling

- The introduction of disk technology helped in this regard.
- Disk technology introduced the spooling (Simultaneous Peripheral Operations On- Line)
- Considers disk as a huge buffer.
- Input comes from the disk
- Output goes to the disk.

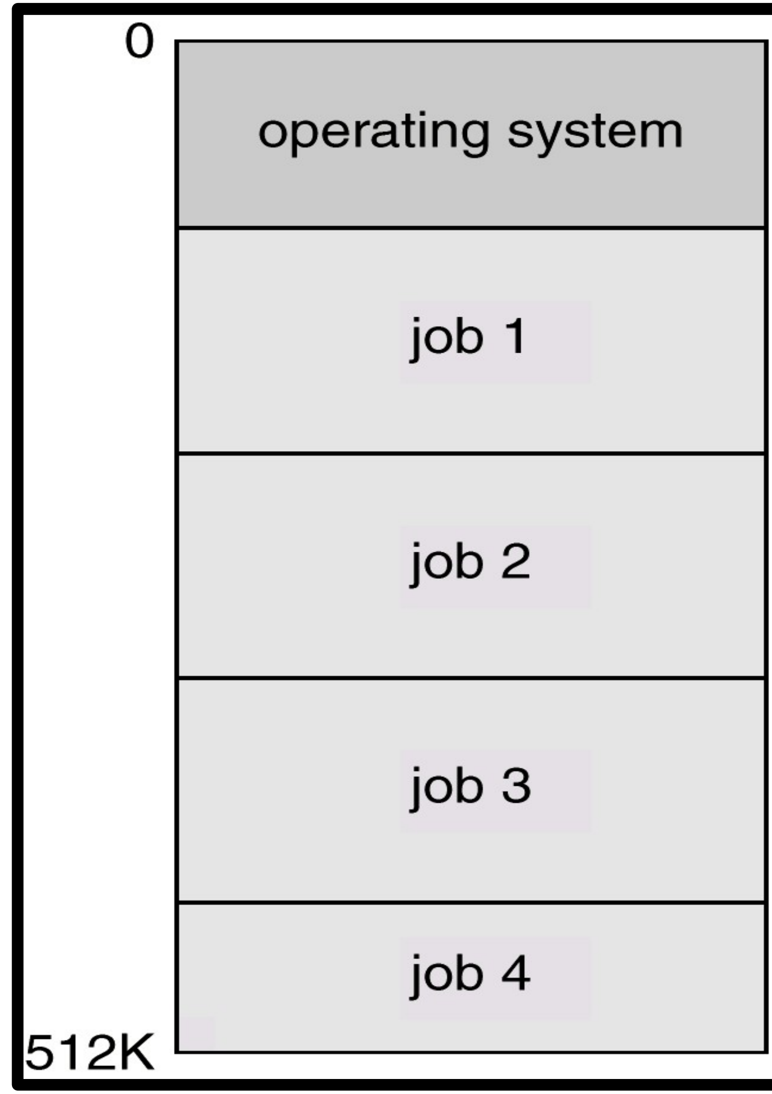


Advantage of Spooling

- Reading can be done in advance.
- Output can be stored on the disk.
- Spooling is also used for processing data at remote sites.
- Spooling overlaps the I/O of one job and computation of other jobs.
- Even printing and reading can overlap.
- Spooling can keep both the CPU and the I/O devices working at higher rates.

Multi- programmed Batched Systems (1960s)

- A single user can not keep either CPU or I/O busy.
- Multiprogramming increases CPU utilization by organizing jobs such that the CPU always has one to execute.
- The OS keeps several jobs in memory at a time and CPU is multiplexed among them



Multi- programmed Batch Systems

- If CPU is executing a job and requires a tape to be mounted
  - In a non multi- programmed system
    - CPU sits idle.
  - In a Multi- programmed system
    - CPU takes up another job.
- **Multiprogramming is the first instance when the OS started taking decisions.**
- Job scheduling is done by OS.
- Having several programs in the memory requires memory management.

OS Features Needed for Multiprogramming

- I/O routine supplied by the system.
- Memory management – the system must allocate the memory to several jobs.
- CPU scheduling – the system must choose among several jobs ready to run.
- Allocation of devices.

Time- Sharing Systems–Interactive Computing

- Programs could interact with user.
- Programs
  - Could wait for I/ O for arbitrary time
    - CPU switched to another job.
    - However, resident jobs took up valuable memory
      - Needed to be sapped out to disk
      - Virtual memory.
- Time- sharing systems were developed to provide interactive use of a computer system at a reasonable cost.

Time- Sharing Systems–Interactive Computing

- A TSS uses CPU scheduling and multi- programming to provide each user with a small portion of a time shared computer.
- A program that is loaded into a memory and is executing is commonly known as a process.
- In timesharing system, a process executes for only a short time.
  - I/ O is at people speeds, but OS can switch rapidly.
- A time- shared OS system allows the many users to share the computer simultaneously.
- It gives the impression that the user has own computer, whereas actually a computer is shared among many users.

Time- Sharing Systems…

- Multiprogramming and timesharing are the central themes of modern OSs.
- Multiprogramming and timesharing requires
  - Memory management and protection
  - Virtual memory: A program is bigger than physical memory
  - Online file systems.
  - Disk management
  - CPU scheduling
  - Process synchronization and communication
  - Deadlock detection

OS requirements (1960s and late 1960s)

- OS Research in 60s
  - MULTICS at MIT
  - Atlas (spooling, demand paging) at Manchester Univ.
- OS research (late 1960s)
  - Multiprogramming
    - Memory allocation and protection
    - I/O operations were responsibility of OS.
  - Interactive systems
    - Scheduling issues
    - Swapping and virtual memory.
  - Users wanted permanent files
    - Hierarchical directory systems.
- OSs in 1960s
  - Increased in size and complexity
  - Were not well understood
    - IBM: OS/360

UNIX (early 1970s)

- Originally developed at Bell labs for the PDP-7
  - Ken Thomson
  - Dennis Ritchie
- Smaller & Simpler
  - Process spawn and control
    - Each command creates a new process
  - Simple inter-process communication
  - Command interpreter not built in: runs as a user process
  - Files were streams of bytes.
  - Hierarchical file system
- Advantages
  - Written in a high-level language
  - Distributed in source form
  - Powerful OS primitives on an inexpensive platform

Personal Computers (1980s)

- Originally
  - Single user
  - Simplified OSs
    - No memory protection
    - MS-DOS
- Now run sophisticated OSs
  - Windows NT, Linux

Windowing Systems (1980s)

- Originally based on work at Xerox Parc
- Popularized by the Macintosh
- Characterized by
  - Graphical user interface
  - Mouse control

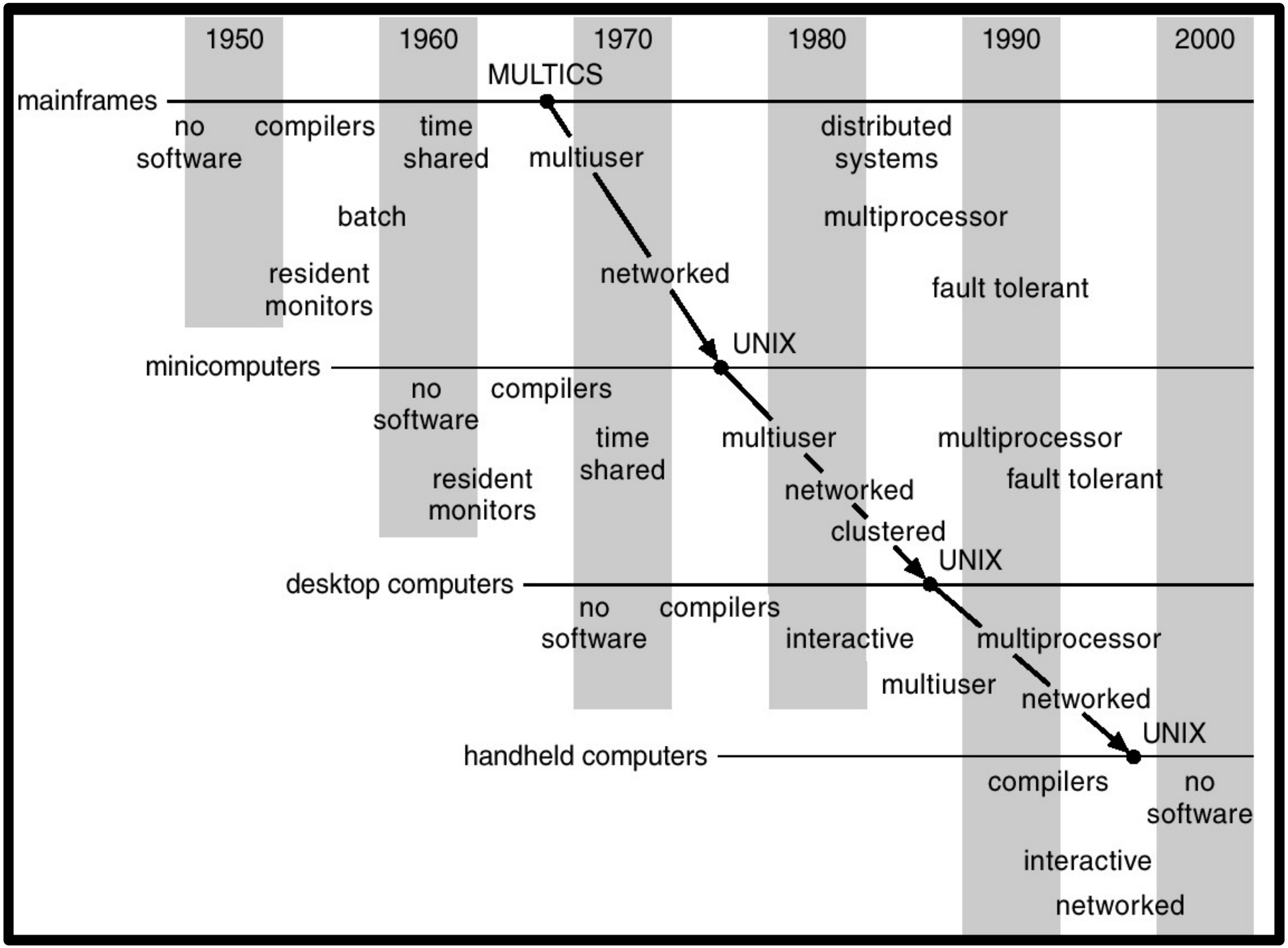
Networks of workstations (1990s)

- High-speed network connections
- Local & world-wide
- Client-server systems
  - File systems
  - Remote windowing systems
- Support a variety of node OSs
  - Unix, Windows XP, OS/2

Future

- Distributed systems
  - Network is invisible
- Micro-kernal and extensible OSs
  - Supports multiple OS flavors
    - E.g., Mach, Amoeba, WINDOWS XP
- Embedded services and network computers
  - Computer runs a very thin OS (Java Virtual machine).

Migration of Operating- System Concepts and Features



Outline

- Introduction
  - What is an Operating System ?
- Course topics and grading
- History, development and concepts of Oss (Stallings, 2.2 and 2.3)
- **Different kinds of Computer Systems**

  (Silberschatz, Chapter 1)
- Concept of virtual computer

  (Crowley, Chapter 1)

Different Kinds of Computer Systems

- Multiprocessor Systems
- Distributed Systems
- Real-time systems

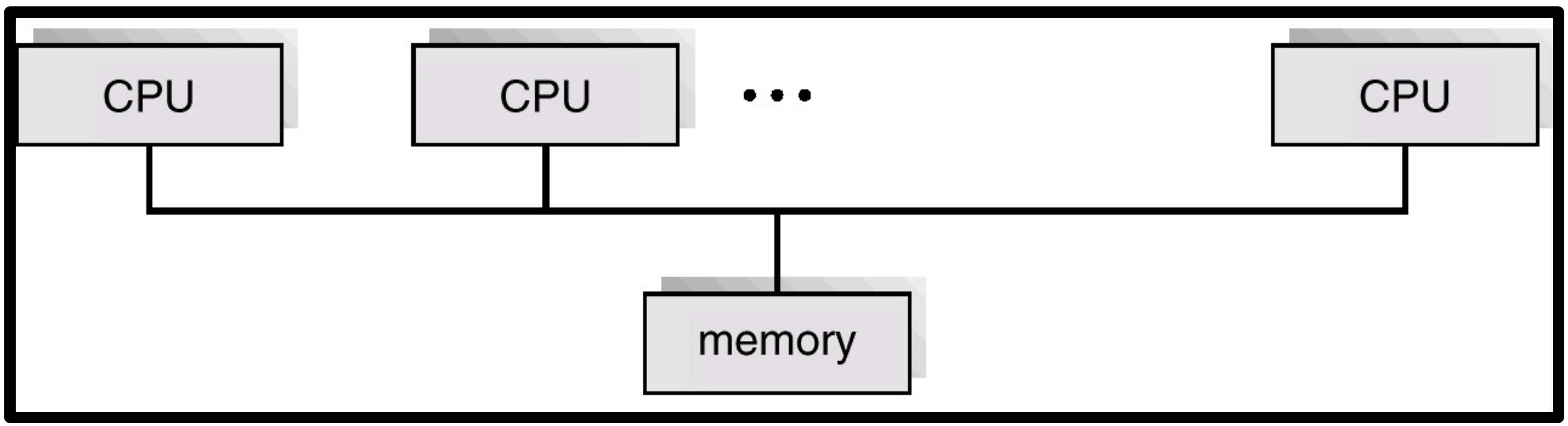
Multiprocessor Systems

- Multiprocessor systems with more than on CPU in close communication.
- *Tightly coupled system* – processors share memory and a clock; communication usually takes place through the shared memory.
- Advantages of multiprocessor system:
  - Increased *throughput: more processors more work*
  - Economical:
  - Increased reliability
    - graceful degradation: failure of one processor will not halt the system. Service is proportional to the level of surviving hardware.
    - Fault tolerant: Systems designed for graceful degradation.

Parallel Systems (Cont.)

- *Symmetric multiprocessing (SMP)*
  - Each processor runs and identical copy of the operating system.
  - Many processes can run at once without performance deterioration.
  - Most modern operating systems support SMP
- *Asymmetric multiprocessing*
  - Each processor is assigned a specific task; master processor schedules and allocated work to slave processors.
  - More common in extremely large systems

Symmetric Multiprocessing Architecture



Distributed Systems

- Distribute the computation among several physical processors.
- *Loosely coupled system* –
  - each processor has its own local memory;
  - processors communicate with one another through various communications lines
  - such as high-speed buses or telephone lines.
- Advantages of distributed systems.
  - Resources Sharing
    - Ex: Laser printer
  - Computation speed up – load sharing
  - Reliability
  - Communications
    - Ex: E-mail

Real- Time Systems

- Often used as a control device in a dedicated application such as controlling scientific experiments, medical imaging systems, industrial control systems, and some display systems.
- Well- defined fixed- time constraints.
  - A process must complete within the defined constraints or system will fail.
- Real- Time systems may be either *hard* or *soft* real-time.

Real- Time Systems (Cont.)

- Hard real- time:
  - Guarantees that critical tasks complete within time.
  - All the delays in the system are bounded.
  - Secondary storage limited or absent, data stored in short term memory, or read- only memory (ROM)
  - Conflicts with time- sharing systems, not supported by general- purpose operating systems.

- Soft real- time
  - Critical time tasks gets priority over other tasks, and retails that priority until it completes.
  - Limited utility in industrial control of robotics
  - Useful in applications (multimedia, virtual reality) requiring advanced operating- system features.

Outline

- Introduction
  - What is an Operating System ?
- Course topics and grading
- **History, development and concepts of Oss (Stallings, 2.2 and 2.3)**
- Different kinds of Computer Systems

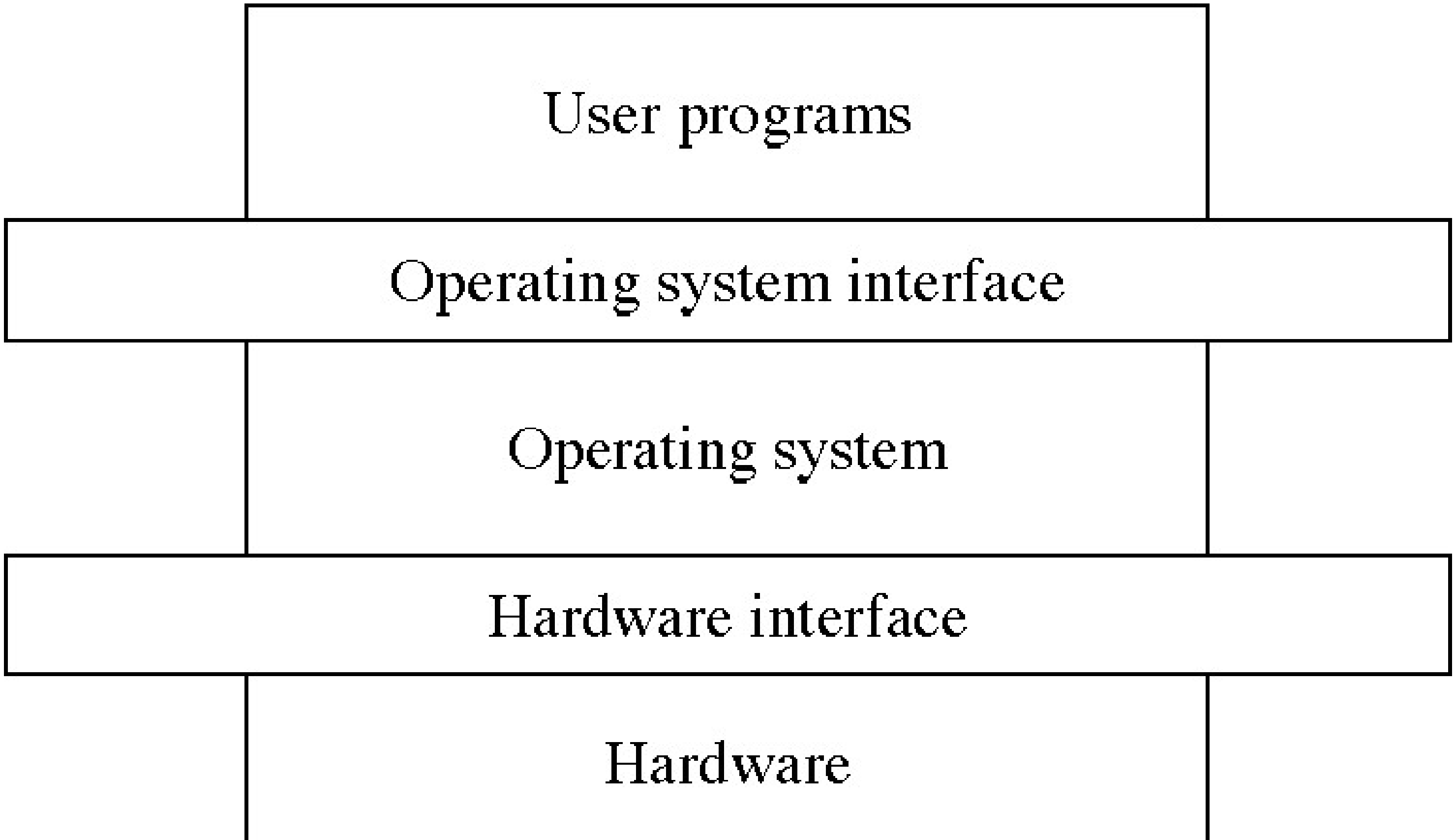
  (Silberschatz, Chapter 1)
- Concept of virtual computer

  (Crowley, Chapter 1)

Concept of virtual computer

- Multilevel implementation
  - also called layered
- Resources
  - Hardware: provided to the OS
  - Logical (virtual): created by the OS
- Resource management
  - transformation
  - multiplexing
    - time and space

Levels in a computer system



Design: Two- level implementation

- Two- level implementation
  - Lower level is a problem- specific language
  - Upper level solves the problem at hand
  - Lower level is reusable
- In operating systems
  - *mechanism*: lower level of basic functions, does not change
  - *policy*: upper level policy decisions, easy to change and experiment

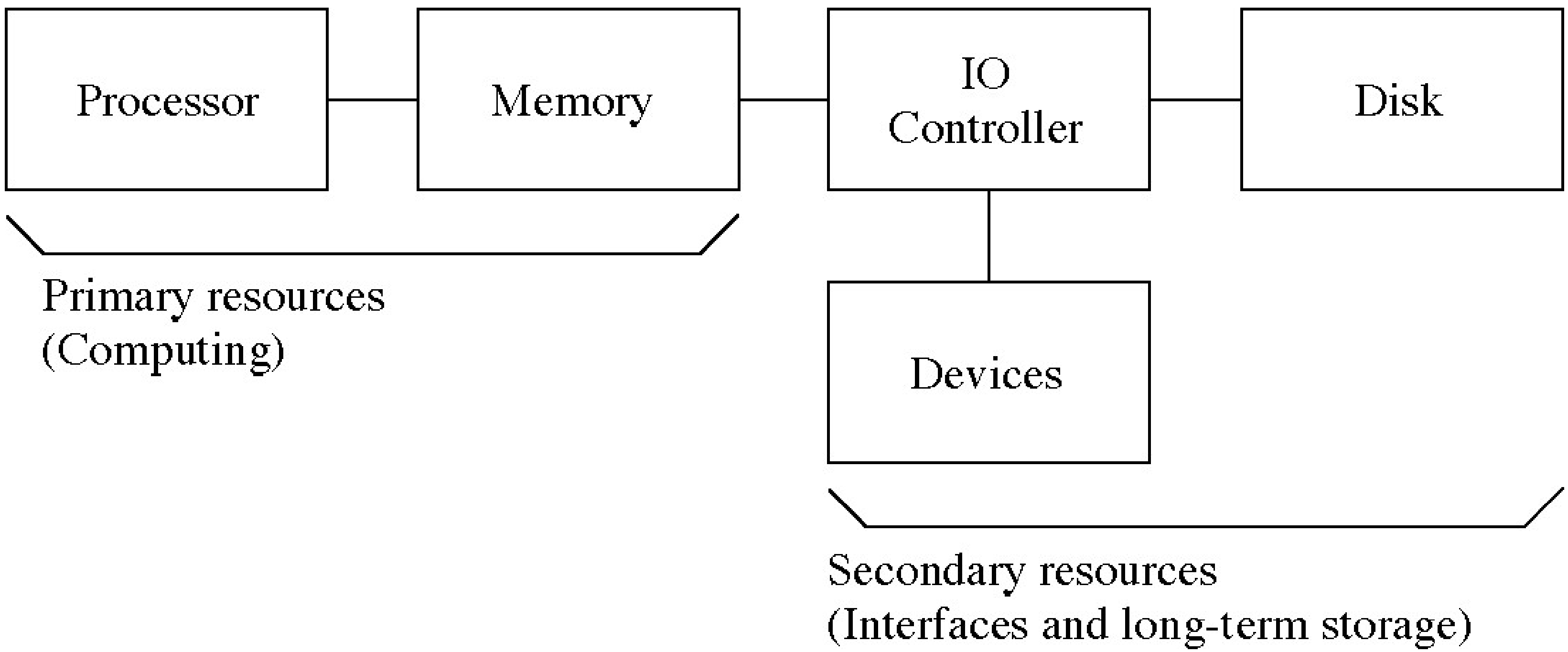
Operating system functions

- Resource manager
  - manage hardware and software resources
- Virtual machine manager
  - implement a virtual machine for processes to run in
  - a nicer environment than the bare hardware

Hardware resources

- *Processor*: execute instructions
- *Memory*: store programs and data
- *Input/ output (I/ O)controllers*: transfer to and from devices
- *Disk devices*: long- term storage
- *Other devices*: conversion between internal and external data representations

Hardware resources



Resource management functions

- *Transforming* physical resources to logical resources
  - Making the resources easier to use
- *Multiplexing* one physical resource to several logical resources
  - Creating multiple, logical copies of resources
- *Scheduling* physical and logical resources
  - Deciding who gets to use the resources

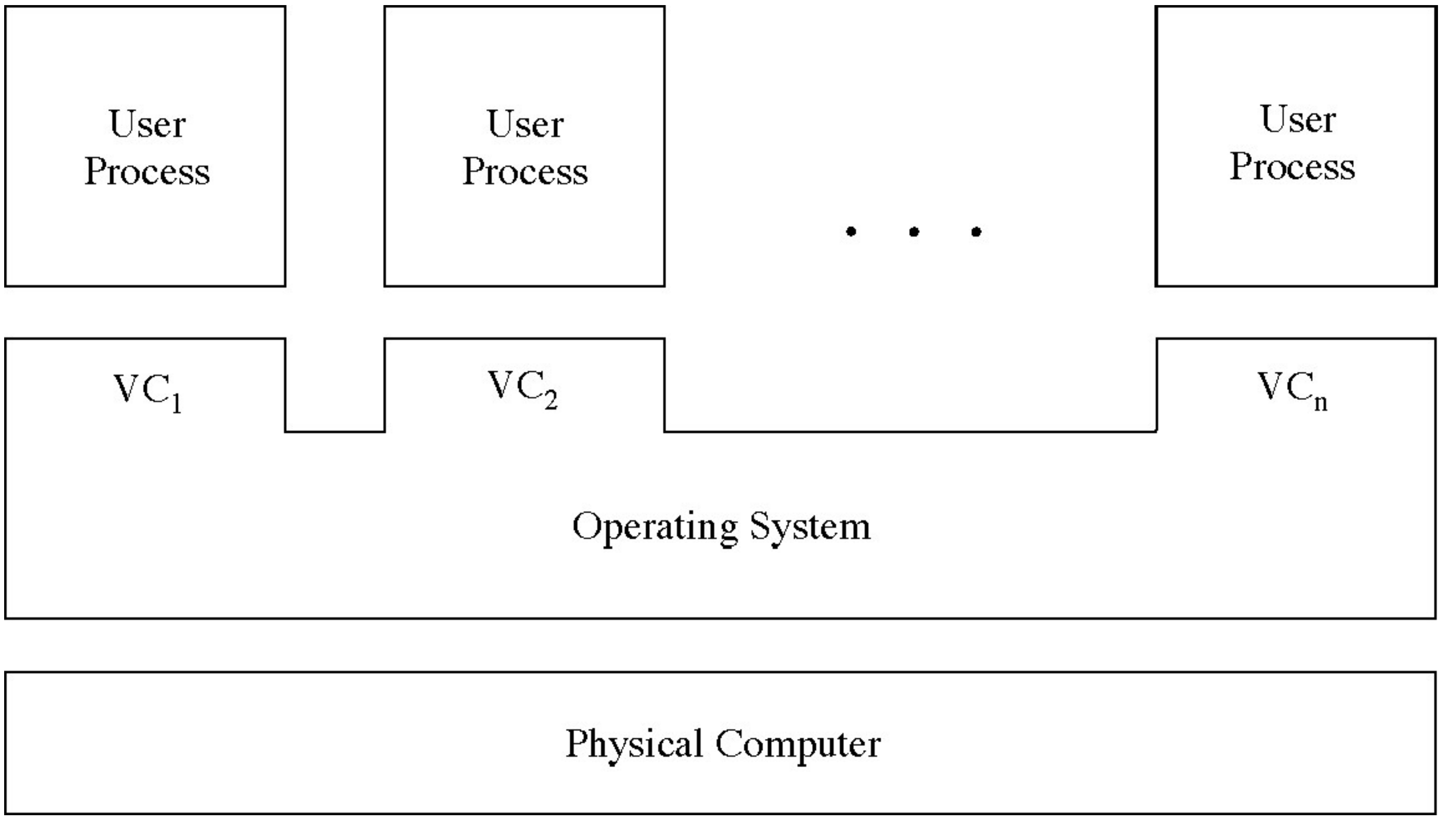
Types of multiplexing

- Time multiplexing
  - time- sharing
  - scheduling a serially- reusable resource among several users
- Space multiplexing
  - space- sharing
  - dividing a multiple- use resource up among several users

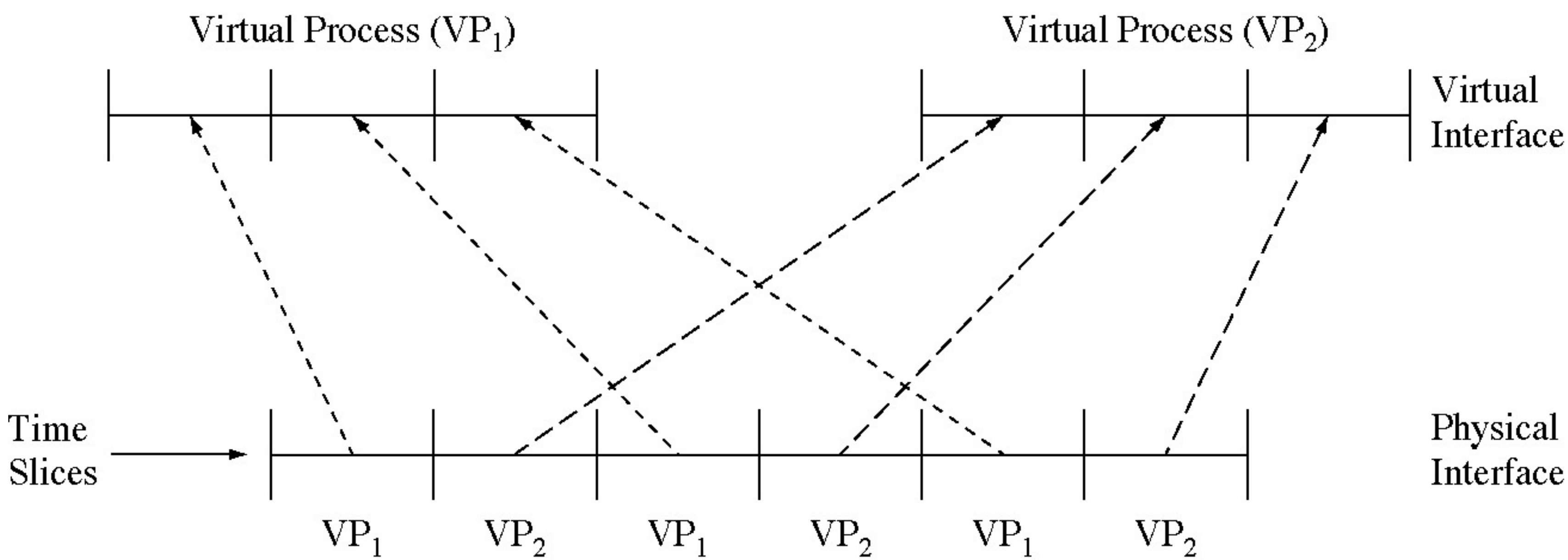
Virtual computers

- Processor virtualized to processes
  - mainly time- multiplexing
- Memory virtualized to address spaces
  - space and time multiplexing
- Disks virtualized to files
  - space- multiplexing
  - transforming

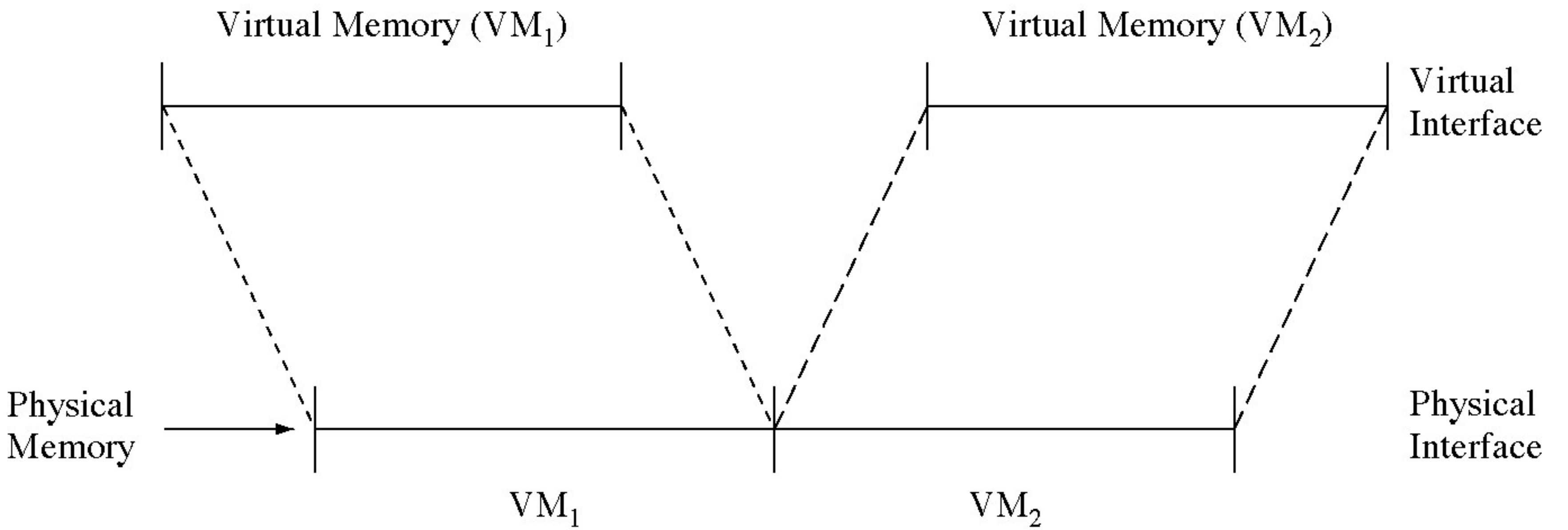
Multiple virtual computers



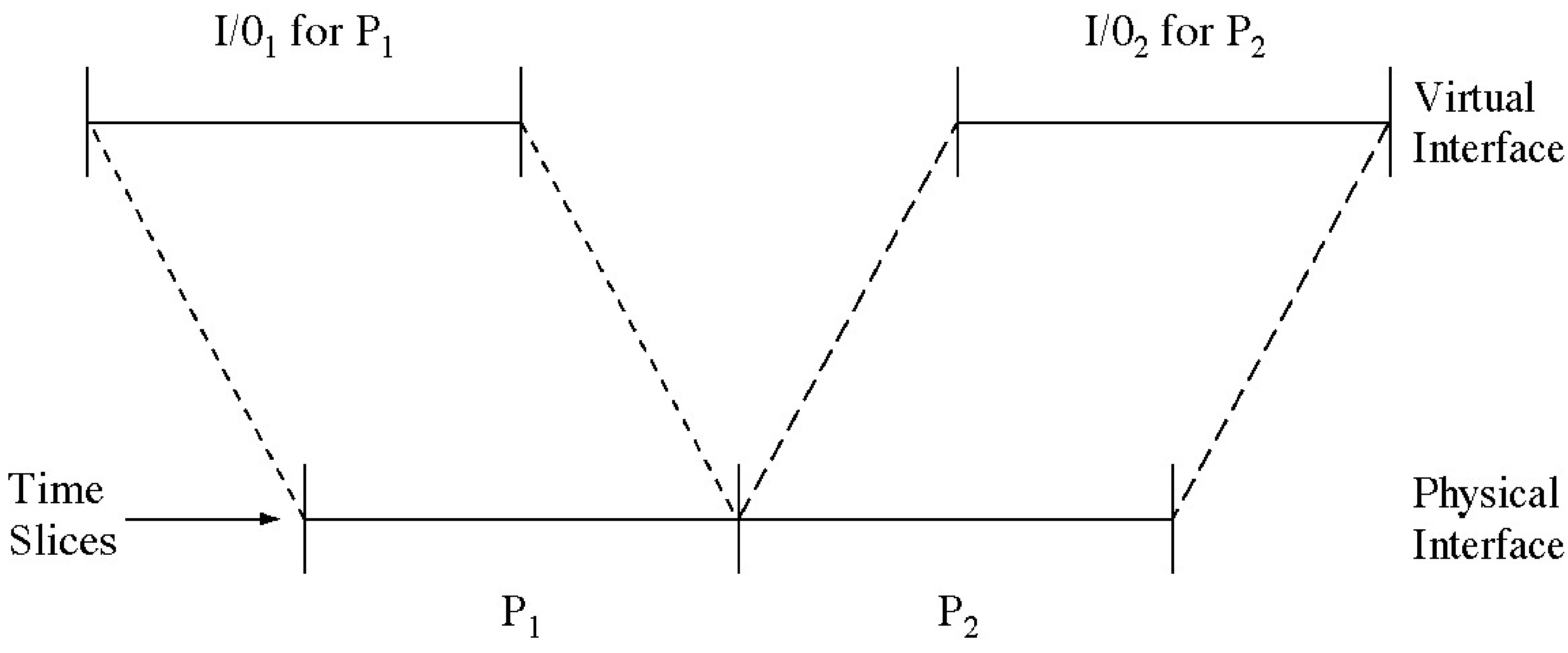
Time- multiplexing the processor



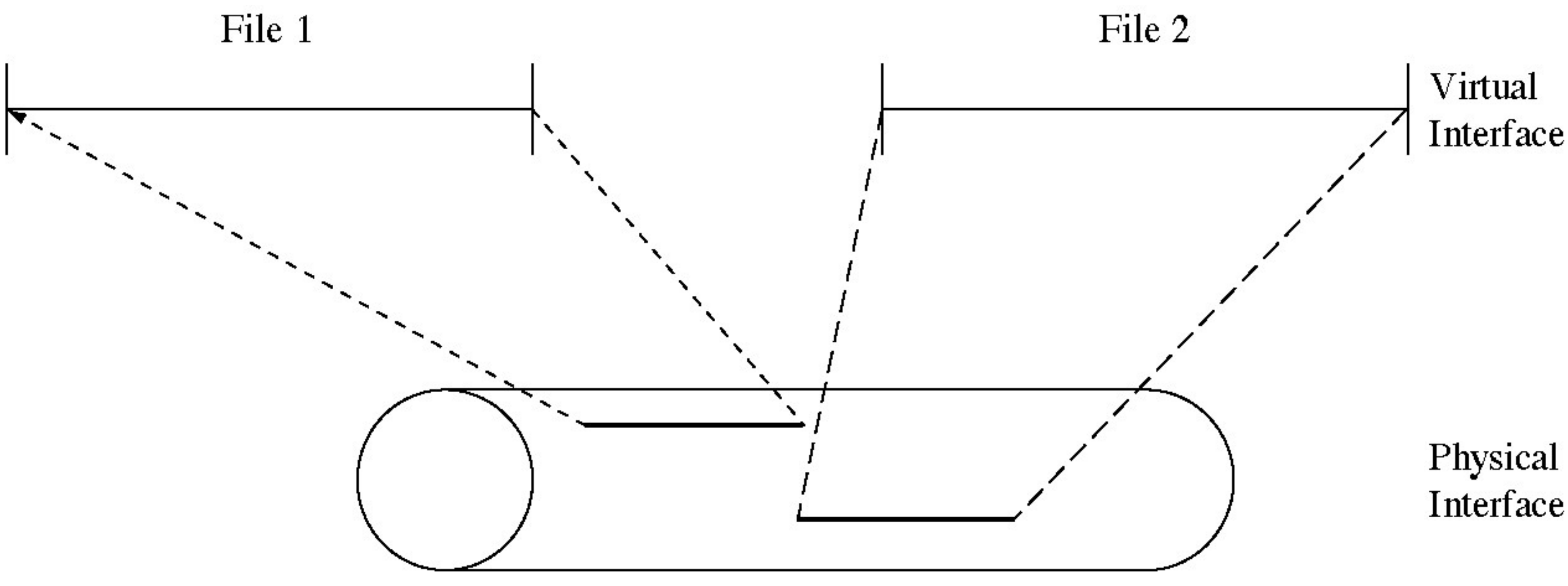
Space- multiplexing memory



Time- multiplexing I/ O devices



Space- multiplexing the disk



Do we need an OS?

- Not always
  - Some programs run “stand-alone”
- But they are very useful
  - Reusable functions
  - Easier to use than the bare hardware

Metric prefixes

- decisecond: 10^{-1} sec.
- centiecond: 10^{-2} sec.
- millisecond: 10^{-3} sec.
- microsecond: 10^{-6} sec.
- nanosecond: 10^{-9} sec.
- picosecond: 10^{-12} sec.
- femptosecond: 10^{-15} sec.
- attoseond: 10^{-18} sec.

- dekabyte: 10^{1} bytes
- hectobyte: 10^{2} bytes
- kilobyte: 10^{3} bytes
- megabyte: 10^{6} bytes
- gigabyte: 10^{9} bytes
- terabyte: 10^{12} bytes
- petabyte: 10^{15} bytes
- exabyte: 10^{18} bytes

Operating Systems and hardware

- Both have influence on each other
- To facilitate the use of hardware, operating systems were developed.
- As OSs were designed and used, it became obvious that changes in the design of hardware could simplify them.
- So in the next lecture
  - Review of Computer System Structures
    - I/O structure and storage hierarchy
    - Hardware protection
  - Operating System Structures
    - System components, services and system calls
    - Virtual machines
    - System design and implementation
- Reading: Silberscatz/galvin: Chapters 2&3.