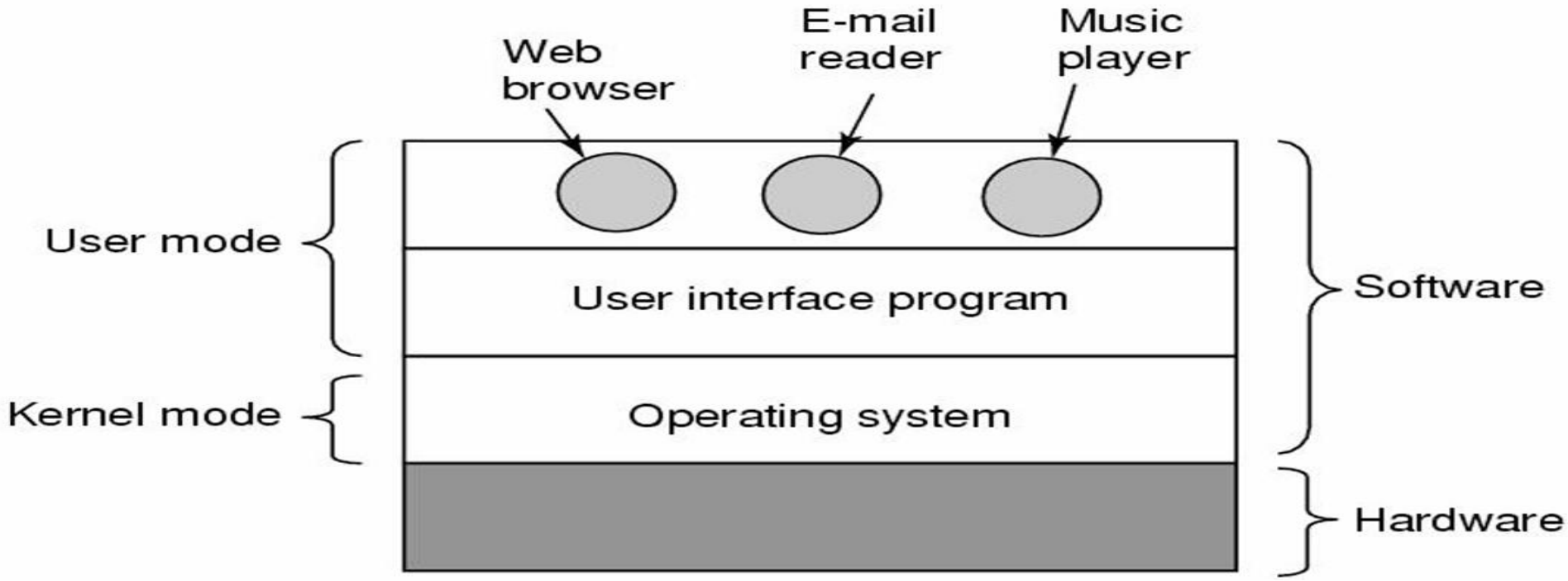
Applied Operating System

Chapter 1: Introduction

Prepared By:
Amit K. Shrivastava
Asst. Professor
Nepal College Of Information Technology

1.1 Introduction

- An operating system acts as an intermediary between the user of a computer and the computer hardware. The purpose of an Operating system is to provide an environment in which a user can execute programs in a convenient and efficient manner.
- An operating system is software that manages the computer hardware. The hardware must provide appropriate mechanisms to ensure the correct operation of the computer system and to prevent user programs from interfering with the proper operation of the system.



What is Applied Operating System?

- AOS refers to the practical implementation and utilization of operating system concepts, principles, and technologies in real-world systems and applications.
- Applied operating systems involve understanding how operating system concepts such as process management, memory management, file systems, networking, security, and user interface design.

1.2 Goals of an Operating System: The major goals of OS are to

- Provide a user-friendly interface: This allows users to interact with the computer hardware in a simple and intuitive way
- Efficiently manage system resources: The OS manages the allocation and utilization of system resources such as CPU, memory, storage, and I/O devices to ensure optimal performance and prevent conflicts.
- Execute programs: The OS creates an environment where programs can run reliably and securely, providing the necessary services and resources for their execution.
- Facilitate multitasking: The OS allows multiple programs to run concurrently
- Provide security: The OS protects the system and its data from unauthorized access, malicious attacks, and accidental errors.
- Ensure reliability: The OS strives to maintain system stability and prevent crashes or data loss. It includes features like error detection, recovery mechanisms

History/Generation of Operating System:

- Operating systems have been evolving through the years. Since operating systems have historically been closely tied to the architecture of the computers on which they run, we will look at successive generations of computers to see what their operating systems were like.

0th Generation (Before 1940s): known as Pre-Computer Era

- No actual "computers" yet, but mechanical devices like the abacus, tally sticks, Napier's bones, and other early calculation devices existed.

The First Generation (1945- 55) Vacuum Tubes and Plug boards

- First of all calculating engines is built and mechanical relays were used but were very slow, with cycle times measured in seconds. Relays were later replaced by vacuum tubes. All programming was done in absolute machine language, often by wiring up plugboards to control the machine's basic functions. By the early 1950s, the routine had improved somewhat with the introduction of punched cards. It was now possible to write programs on cards and read them in instead of using plugboards;

History of Operating System(contd..):

The Second Generation (1955-65) Transistors and Batch Systems

- The introduction of the transistor in the mid-1950s changed the picture radically. These machines, now called mainframes, and to run a job a programmer would first write the program on paper (in FORTRAN or possibly even in assembly language), then punch it on cards. But it was time consuming and costly. The solution generally adopted was the batch system. The idea behind it was to collect a tray full of jobs in the input room and then read them onto a magnetic tape using a small (relatively) inexpensive computer, such as the IBM 1401.

The Third Generation (1965-1980) ICs and Multiprogramming

- The 360 developed by IBM was the first major computer line to use (small-scale) Integrated Circuits (ICs), thus providing a major price/ performance advantage over the second-generation machines, which were built up from individual transistors. And the most important advantage was multiprogramming. Here, the memory was partitioned into several pieces, with a different job in each partition, while one job was waiting for the I/O to complete, another job could be using the CPU.

History of Operating System(contd..):

The Fourth Generation (1980Present) Personal Computers

▪With the development of LSI (Large Scale Integration) circuits chips containing thousands of transistors on a square centimeter of silicon, the age of personal computer dawned. First kildall wrote a disk base operating system called CP/M(Control Program for Microcomputers) for Intel in 1974, then in early 1980's DOS(Disk Operating System) was invented and after that Microsoft revised it and renamed MS-DOS(Microsoft Disk Operating System). All these operating systems were all based users typing in commands from the keyboard after that GUI(Graphical User Interface) was invented, complete with windows, icons, menus, and mouse. After that different version of windows and Unix came in to light.

Operating System Types:

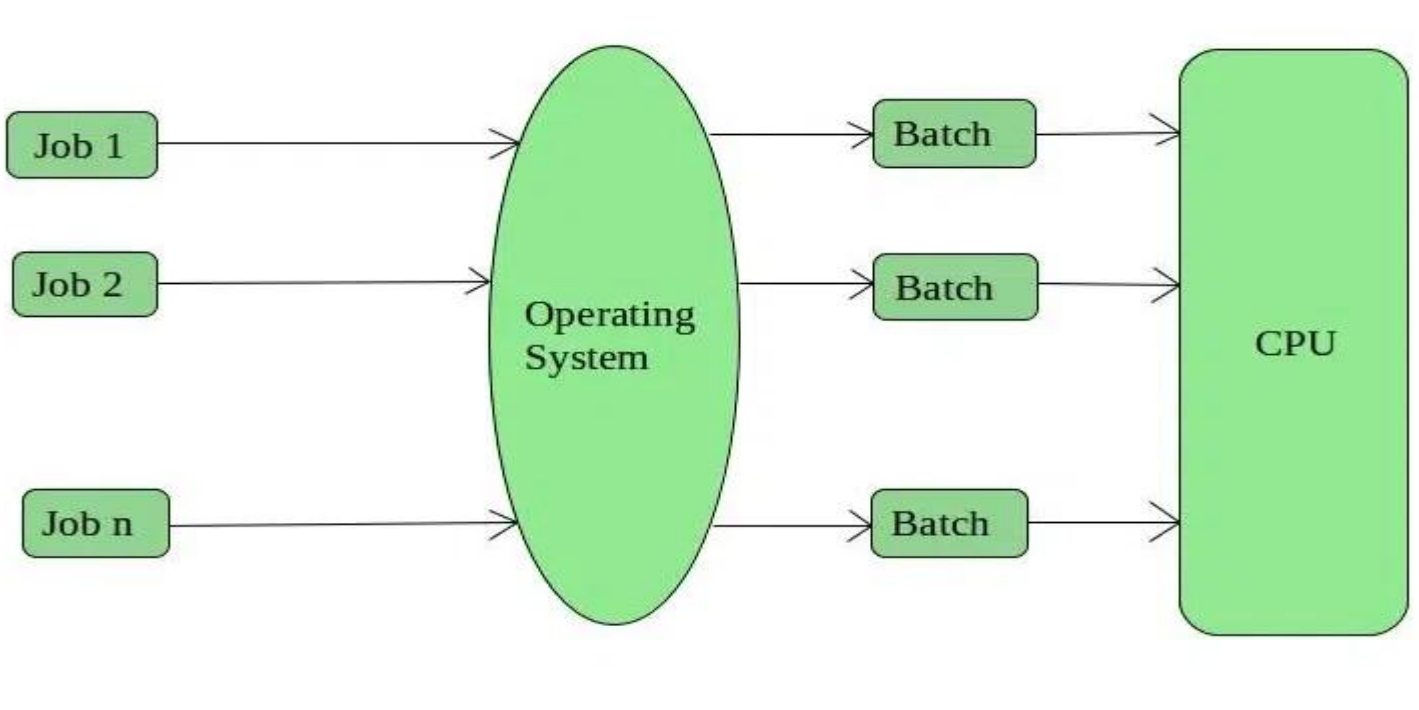
- Batch Systems
- Time-Sharing Systems
- Personal-Computer Systems
- Parallel Systems
- Real Time Systems
- Distributed Systems
- Multiprocessing Systems
- Multiprogramming Systems

Batch Systems

- Jobs with similar needs are batched together and run through the computer as a group by an operator or automatic job sequencer. Performance is increased by attempting to keep CPU and I/O devices busy at all times through buffering, off-line operation, spooling, and multiprogramming. Batch is good for executing large jobs that need little interaction; it can be submitted and picked up later. The problems with Batch Systems are following.

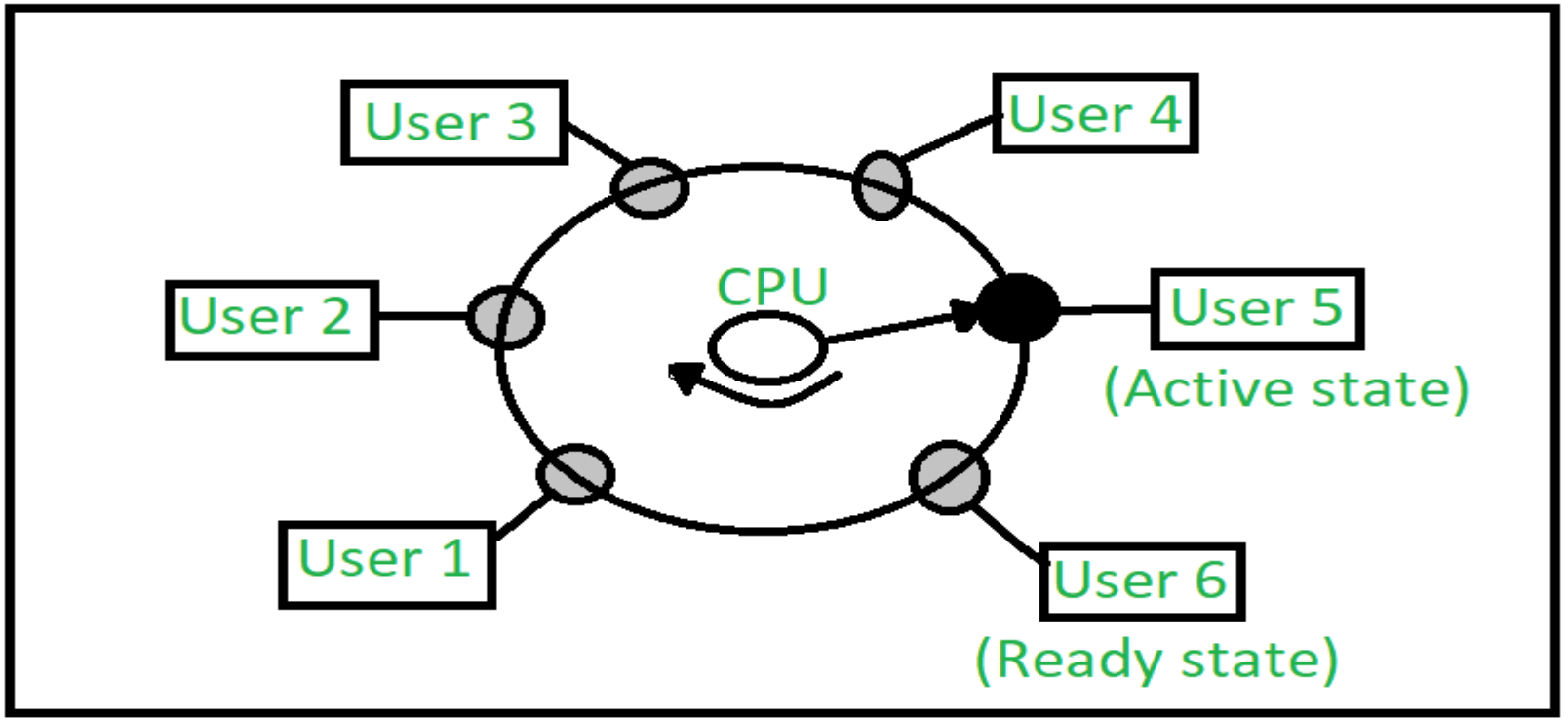
➢ Lack of interaction between the user and job.

➢ Difficult to provide the desired priority.



Time-Sharing Systems

▪ This systems uses CPU scheduling and multiprogramming to provide economical interactive use of a system. The CPU switches rapidly from one user to another. Instead of having a job defined by spooled card images, each program reads its next control card from the terminal, and output is normally printed immediately to the screen. Advantages of Timesharing operating systems are - Provide advantage of quick response, Avoids duplication of software, Reduces CPU idle time.



Personal-Computer Systems

- A Personal Computer(PC) is a small, relatively inexpensive computer designed for an individual user. All are based on the microprocessor technology that enables manufacturers to put an entire CPU on one chip. At home, the most popular use for personal computers is for playing games. Businesses use personal computers for word-processing, accounting, desktop publishing, and for running spreadsheet and database management applications. The goals of these operating systems is not only maximizing CPU and peripheral utilization, but also maximizing user convenience and responsiveness.

Parallel Systems

- Parallel operating systems are used to interface multiple networked computers to complete tasks in parallel. The architecture of the software is often a UNIX-based platform, which allows it to coordinate distributed loads between multiple computers in a network. Parallel operating systems are able to use software to manage all of the different resources of the computers running in parallel, such as memory, caches, storage space, and processing power. Parallel operating systems also allow a user to directly interface with all of the computers in the network. Its one advantage is increased throughput.

Real-Time Systems

- Often used in a dedicated application, this system reads information from sensors and must respond within a fixed amount of time to ensure correct performance. In this Response Time is already fixed. Means time to Display the Results after Possessing has fixed by the Processor or CPU. Real Time System is used at those Places in which we Requires higher and Timely Response.

Distributed Systems

- This system distributes computation among several physical processors. The processors do not share memory or a clock. Instead, each processor has its own local memory. They communicate with each other through various communication lines, such as a high-speed bus or telephone line. The advantages of distributed systems are- With resource sharing facility user at one site may be able to use the resources available at another, Speedup the exchange of data with one another via electronic mail, If one site fails in a distributed system, the remaining sites can potentially continue operating, Better service to the customers, Reduction of the load on the host computer.

Multiprocessing systems

The term multiprocessing is introduced around modern times when they started to use more than one processor in a single computer (Remember the terms like dual-core, quad-core, octa-core processor). These processors share some things in common like memory, peripherals, etc. By sharing the memory and peripherals they are able to execute different tasks simultaneously. In general, Multiprogramming takes place in a system where it has a single processor.

Multitasking: single processor or sometimes multiprocessor Multiprocessing: multiprocessor.

Multiprogramming systems

The concept of multiprogramming is that more than one program that is to be executed by the processor is loaded into the memory. Say we have 2 programs Loaded into the memory. The first program that is loaded is getting executed. At one point of time, it requires input from the user or waiting for some data. During the waiting time, the CPU is idle. Instead of wasting time, the CPU will now begin to execute the second program. Meanwhile, the first program once it receives the required data, will again get the CPU time and get executed blocking or pausing the execution of the second program. After the completion of the first program, the second program is executed from where it was paused. The concept was introduced to maximize CPU usage.

❖Operating System as Resource Manager:

• Operating system is collection of software which is close to hardware. We can view operating system as a resource – hardware and software collector. A system has many hardware and software that may be required to solve the problem, cpu time, memory space, file storage space, i/o device etc. the operating system acts as manager of these resources.

• Modern computer consists of process, memories, times, disks, network, printer and wide varieties of other devices. The tab of the operating system is to provide for an orderly and controlled allocation of the process, memories and I/O devices among the various programs completing for them.

• An operating system is a control program, a control program manages the execution of user program to prevent errors and improve use of computer. It is especially concerned with the operation and control of I/O devices. When a computer has multiple users the operating system manages and protects the memory I/O devices. The operating system keeps in trace that who is using which resource to grant resource required amount for usage and to mediate conflicting required different programs and users.

❖Operating System as Extended Machine:

• The program that hides the truth about the hardware from the programmer and presents a nice, simple view of named files that can be read and written is, of course, the operating system. Just as the operating system shields the programmer from the disk hardware and presents a simple file-oriented interface, it also conceals a lot of unpleasant business concerning interrupts, timers, memory management, and other low-level features. In each case, the abstraction offered by the operating system is simpler and easier to use than that offered by the underlying hardware.

• In this view, the function of the operating system is to present the user with the equivalent of an **extended machine** or **virtual machine** that is easier to program than the underlying hardware. To summarize it in a nutshell, the operating system provides a variety of services that programs can obtain using special instructions called system calls.

Computer System Organization and Architecture:

Computer Architecture:

Computer architecture is the blueprint for the design and functionality of a computer system. It includes:

- **Instruction Set Architecture (ISA)**: This defines the set of instructions that the computer can execute. It includes the data types, registers, addressing modes, and the instruction set itself.
- **Microarchitecture**: This describes how a particular processor will implement the ISA. It involves the design of the processor's data paths, control unit, and the implementation of the instruction set.
- **System Design**: This encompasses all the hardware components within a computer system, including the CPU, memory, and I/O devices, and how they interact.

Computer Organization

Computer organization focuses on the operational units and their interconnections that realize the architectural specifications. Key components include:

•**Central Processing Unit (CPU)**: The brain of the computer, responsible for executing instructions. It includes:

- **Arithmetic Logic Unit (ALU)**: Performs arithmetic and logical operations.
- **Control Unit (CU)**: Directs the operation of the processor.
- **Registers**: Small, fast storage locations within the CPU.

•**Memory Hierarchy**: Organizes memory into levels (e.g., cache, main memory, secondary storage) to optimize performance and cost.

•**Input/Output (I/O) Systems**: Manages data exchange between the computer and external devices like keyboards, monitors, and printers.

•**Bus Systems**: Communication pathways that connect different components of the computer, such as the data bus, address bus, and control bus.

Operating-System Structures

System Components: Common System Components are

- Process Management
- Main Memory Management
- Secondary-Storage Management
- I/O System Management
- File Management
- Protection System
- Networking
- Command-Interpreter System

Process Management

- A process is a program in execution. A process needs certain resources, including CPU time, memory, files, and I/O devices, to accomplish its task.
- The operating system is responsible for the following activities in connection with process management.
  - Process creation and deletion.
  - process suspension and resumption.
  - Provision of mechanisms for:
    - * process synchronization
    - * process communication

Main-Memory Management

• Memory is a large array of words or bytes, each with its own address. It is a repository of quickly accessible data shared by the CPU and I/O devices.
• Main memory is a volatile storage device. It loses its contents in the case of system failure.
• The operating system is responsible for the following activities in connections with memory management:
– Keep track of which parts of memory are currently being used and by whom.
– Decide which processes to load when memory space becomes available.
– Allocate and deallocate memory space as needed.

Secondary-Storage Management

• Since main memory (primary storage) is volatile and too small to accommodate all data and programs permanently, the computer system must provide secondary storage to back up main memory.
• Most modern computer systems use disks as the principle on-line storage medium, for both programs and data.
• The operating system is responsible for the following activities in connection with disk management:
  – Free space management
  – Storage allocation
  – Disk scheduling

I/O System Management

• The I/O system consists of:
  – A buffer-caching system
  – A general device-driver interface
  – Drivers for specific hardware devices

File Management

• A file is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data.
• The operating system is responsible for the following activities in connections with file management:
  – File creation and deletion.
  – Directory creation and deletion.
  – Support of primitives for manipulating files and directories.
  – Mapping files onto secondary storage.
  – File backup on stable (nonvolatile) storage media.

Protection System

• *Protection refers to a mechanism for controlling access by* programs, processes, or users to both system and user resources.
• The protection mechanism must:
  – distinguish between authorized and unauthorized usage.
  – specify the controls to be imposed.
  – provide a means of enforcement.

Networking (Distributed Systems)

• A *distributed system is a collection processors that do not share* memory or a clock. Each processor has its own local memory.
• The processors in the system are connected through a communication network.
• A distributed system provides user access to various system resources.
• Access to a shared resource allows:
  – Computation speed-up
  – Increased data availability
  – Enhanced reliability

Command-Interpreter System

• Many commands are given to the operating system by control statements which deal with:
  – process creation and management
  – I/O handling
  – secondary-storage management
  – main-memory management
  – file-system access
  – protection
  – networking

Operating System Services

• Program execution – system capability to load a program into memory and to run it.
• I/O operations – since user programs cannot execute I/O operations directly, the operating system must provide some means to perform I/O.
• File-system manipulation – program capability to read, write, create, and delete files.
• Communications – exchange of information between processes executing either on the same computer or on different systems tied together by a network. Implemented via *shared memory or message passing.*
• Error detection – ensure correct computing by detecting errors in the CPU and memory hardware, in I/O devices, or in user.

Additional Operating System Functions

Additional functions exist not for helping the user, but rather for ensuring efficient system operations.
• Resource allocation – allocating resources to multiple users or multiple jobs running at the same time.
• Accounting – keep track of and record which users use how much and what kinds of computer resources for account billing or for accumulating usage statistics.
• Protection – ensuring that all access to system resources is controlled.

Kernel Data Structures:

Kernel data structures are essential components of an operating system's kernel, which is the core part responsible for managing system resources and facilitating communication between hardware and software. Some key kernel data structures are:

1. Process Control Block (PCB)

• **Purpose**: Stores information about a process.

•**Contents**: Process ID, process state, CPU registers, memory management information, accounting information, and I/O status.

2. File Descriptor Table

•**Purpose**: Manages open files for each process.

•**Contents**: Pointers to file control blocks, which contain details about the files.

3. Inode Table

•**Purpose**: Manages file metadata in Unix-like systems.

•**Contents**: File attributes, such as size, permissions, timestamps, and pointers to data blocks.

4. Page Table

•**Purpose**: Manages virtual memory.

•**Contents**: Mappings between virtual addresses and physical memory addresses.

Kernel data Structure(contd..)

5. Scheduler Queue

•**Purpose**: Manages processes ready to execute.
•**Contents**: Linked lists or other structures containing PCBs of processes in the ready state.

6. Interrupt Vector Table

•**Purpose**: Manages interrupt handling.
•**Contents**: Pointers to interrupt service routines for different interrupt types.

7. Buffer Cache

•**Purpose**: Manages frequently accessed disk blocks.
•**Contents**: Cache entries containing disk block data and metadata.

8. Device Queue

•**Purpose**: Manages I/O requests for devices.
•**Contents**: Linked lists or other structures containing I/O request blocks.

User and Operating-System Interface

The user interface (UI) is the bridge that connects a human user to OS. It includes:

- **Graphical User Interface (GUI)**: This is the most common type of UI, featuring windows, icons, menus, and pointers (WIMP). Examples include Windows, macOS, and many Linux distributions.
- **Command-Line Interface (CLI)**: This interface allows users to type commands directly to the operating system. It's powerful for advanced users and is used in systems like Linux terminal and Windows Command Prompt.
- **Touch Interface**: Found on mobile devices and some laptops, this interface allows users to interact with the system through touch gestures.

Operating-System Interface

The Operating-System Interface refers to the way the operating system manages hardware and software resources and provides services for computer programs. It includes:

- **System Calls**: These are the programming interface through which applications request services from the operating system. Examples include file operations, process control, and communication.

Operating-System Interface(contd..)

- **Kernel**: The core part of the operating system that manages system resources, such as CPU, memory, and I/O devices. It acts as a bridge between applications and the hardware.
- **Device Drivers**: These are specialized programs that allow the operating system to communicate with hardware devices like printers, graphics cards, and network adapters.

Interaction Between User and OS

When a user interacts with the UI, their actions are translated into system calls that the operating system processes. For example, clicking a file icon in a GUI sends a system call to the OS to open the file. The OS then uses the appropriate device drivers to access the file on the storage device and display its contents to the user.

System Calls

- System calls provide the interface between a running program and the operating system.
  - Generally available as assembly-language instructions.
  - Languages defined to replace assembly language for systems programming allow system calls to be made directly.
- Three general methods are used to pass parameters between a running program and the operating system.
  - Pass parameters in registers.
  - Store the parameters in a table in memory, and the table address is passed as a parameter in a register.
  - Push (store) the parameters onto the stack by the program, and pop off the stack by operating system.

System Calls(contd..)

- System calls can be grouped roughly into six major categories: process control, file manipulation, device manipulation, information maintenance, communications, and protection.

- **Process control**
  - end, abort
  - load, execute
  - create process, terminate process
  - get process attributes, set process attributes
  - wait for time
  - wait event, signal event
  - allocate and free memory
- **File management**
  - create file, delete file
  - open, close
  - read, write, reposition
  - get file attributes, set file attributes

System Calls(contd..)

- **Device management**
  - request device, release device
  - read, write, reposition
  - get device attributes, set device attributes
  - logically attach or detach devices
- **Information maintenance**
  - get time or date, set time or date
  - get system data, set system data
  - get process, file, or device attributes
  - set process, file, or device attributes
- **Communications**
  - create, delete communication connection
  - send, receive messages
  - transfer status information
  - attach or detach remote devices

System Programs

• System programs provide a convenient environment for program development and execution. The can be divided into:

- File manipulation
- Status information
- File modification
- Programming language support
- Program loading and execution
- Communications
- Application programs

• Most users' view of the operation system is defined by system programs, not the actual system calls.

System Programs

• System programs provide a convenient environment for program development and execution. The can be divided into:

- File manipulation
- Status information
- File modification
- Programming language support
- Program loading and execution
- Communications
- Application programs

• Most users' view of the operation system is defined by system programs, not the actual system calls.

System Structure – Simple Approach

- MS-DOS – written to provide the most functionality in the least space
  - not divided into modules
  - Although MS-DOS has some structure, its interfaces and levels of functionality are not well separated.

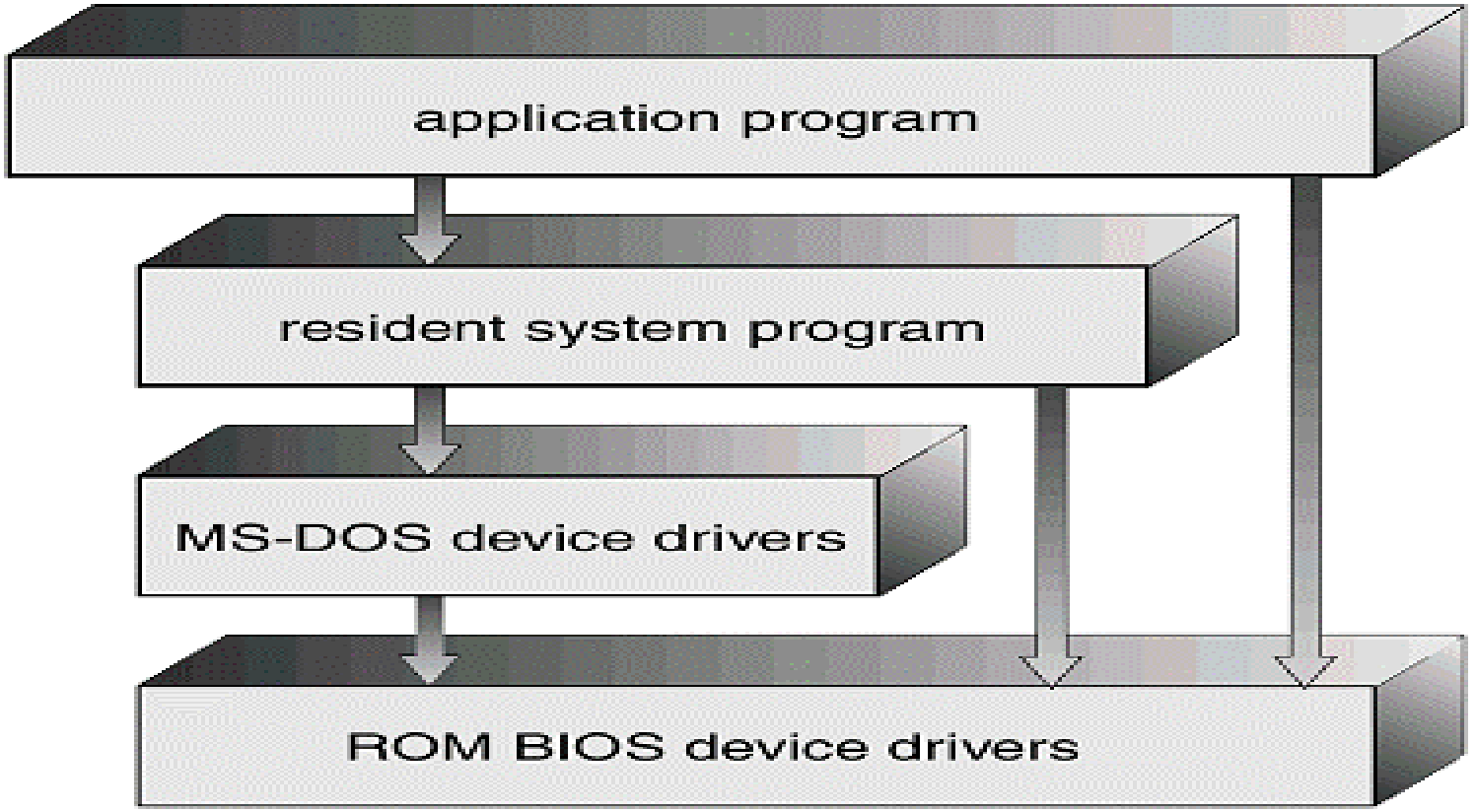


Fig: MS-DOS Layer Structure

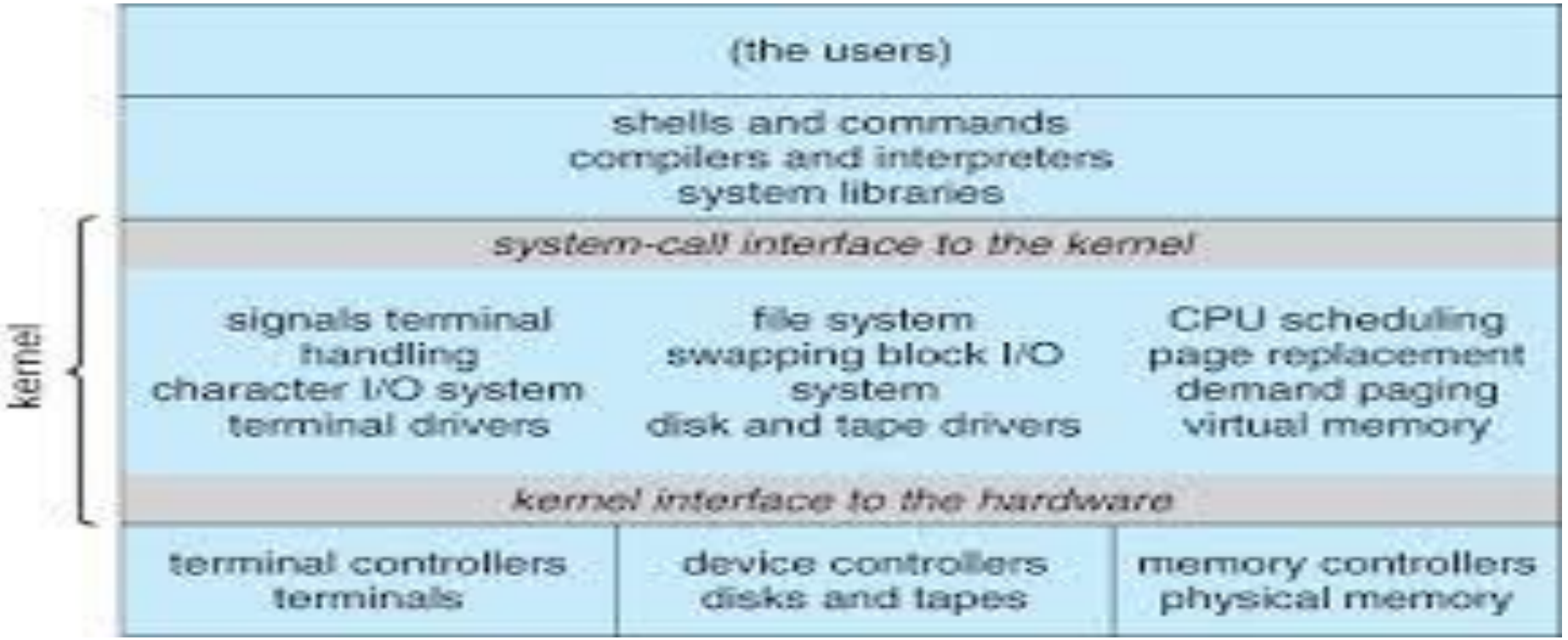
System Structure – Simple Approach (Cont.)

• UNIX – limited by hardware functionality, the original UNIX operating system had limited structuring. The UNIX OS consists of two separable parts.

– Systems programs

– The kernel

* Consists of everything below the system-call interface and above the physical hardware.
* Provides the file system, CPU scheduling, memory management, and other operating-system functions; a large number of functions for one level.



System Structure – Layered Approach

• The operating system is divided into a number of layers (levels), each built on top of lower layers. The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.

• With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers.

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

Structure of the THE operating system.

Layered Systems(contd..)

- A further generalization of the layering concept was present in the MULTICS system. Instead of layers, MULTICS was organized as a series of concentric rings, with the inner ones being more privileged than the outer ones. When a procedure in an outer ring wanted to call a procedure in an inner ring, it had to make the equivalent of a system call, that is, a TRAP instruction whose parameters were carefully checked for validity before allowing the call to proceed.

Virtual Machines:

• A *virtual machine takes the layered approach to its logical* conclusion. It treats hardware and the operating system kernel as though they were all hardware.

• A virtual machine provides an interface *identical to the underlying* bare hardware.

• The operating system creates the illusion of multiple processes, each executing on its own processor with its own (virtual) memory.

• The resources of the physical computer are shared to create the virtual machines.

– CPU scheduling can create the appearance that users have their own processor.

– Spooling and a file system can provide virtual card readers and virtual line printers.

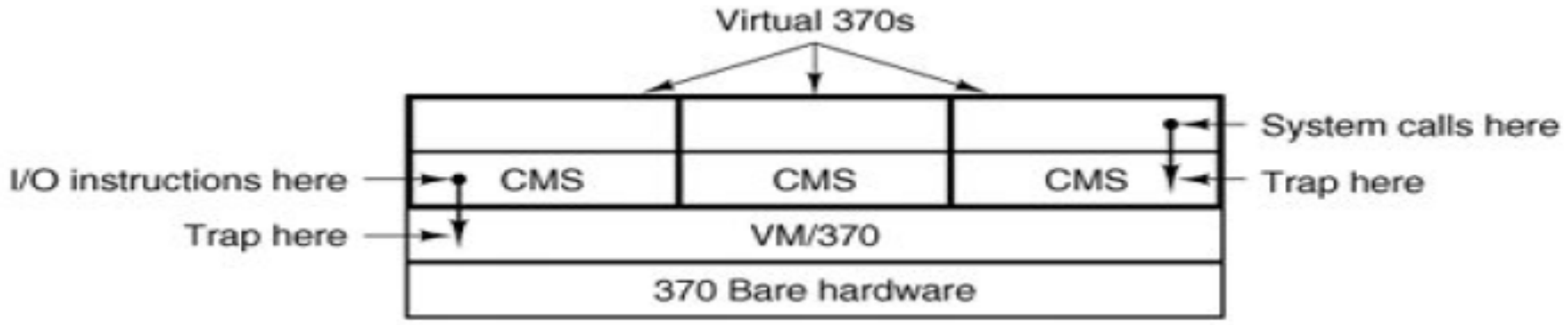
– A normal user time-sharing terminal serves as the virtual machine operator's console.

Advantages/Disadvantages of Virtual Machines

• The virtual-machine concept provides complete protection of system resources since each virtual machine is isolated from all other virtual machines. This isolation, however, permits no direct sharing of resources.

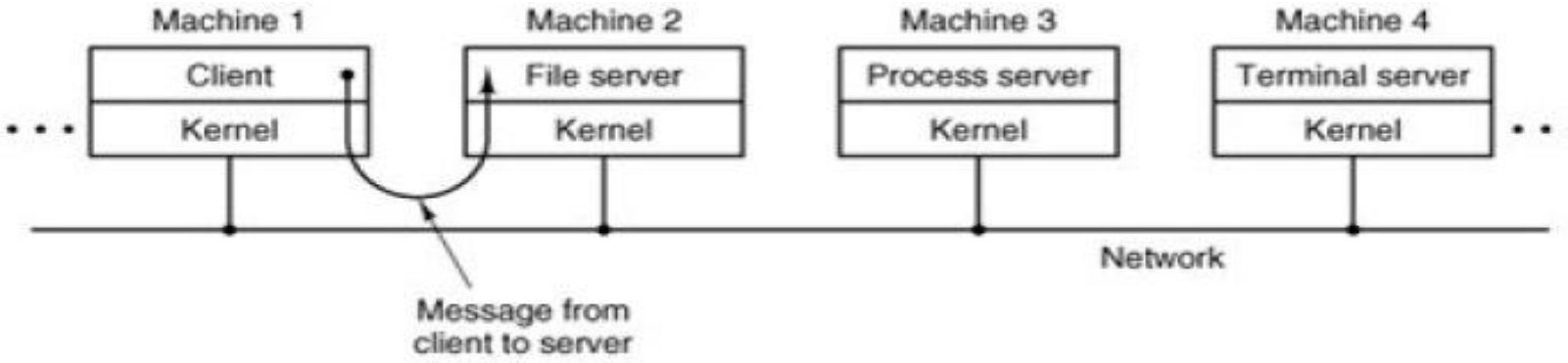
• A virtual-machine system is a perfect vehicle for operating systems research and development. System development is done on the virtual machine, instead of on a physical machine and so does not disrupt normal system operation.

• The virtual machine concept is difficult to implement due to the effort required to provide an *exact duplicate to the underlying* machine.



Client-Server Model:

- This concepts is based on of two classes of processes, the servers, each of which provides some service, and the clients, which use these services. This model is known as client-server model. Communication between clients and servers is often by message passing. To obtain a service, a client process constructs a message saying what it wants and sends it to the appropriate service. The service then does the work and sends back the answer. An obvious generalization of this idea is to have the clients and servers run on different computers, connected by a local or wide-area network.



System Design Goals

• User goals – operating system should be convenient to use, easy to learn, reliable, safe, and fast.

• System goals – operating system should be easy to design, implement, and maintain, as well as flexible, reliable, error-free, and efficient.

Mechanisms and Policies

• Mechanisms determine how to do something, policies decide what will be done.

• The separation of policy from mechanism is a very important principle, it allows maximum flexibility if policy decisions are to be changed later.