Unit 1: Introduction to Operating Systems

Computer Hardware Review

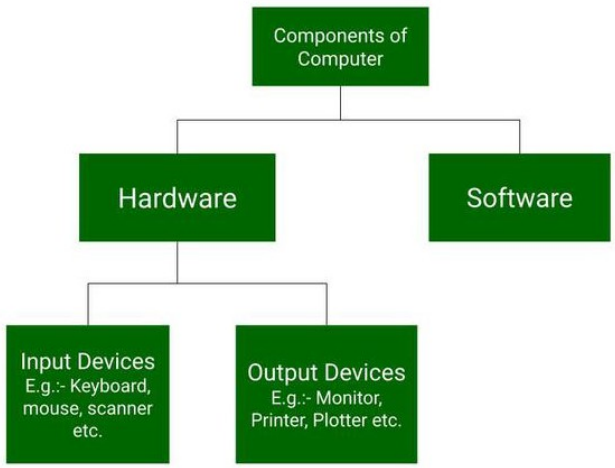
Computer hardware refers to the tangible, physical components of a computer system that you can see, touch, and manipulate. These parts work together to execute the instructions provided by software, allowing the system to process input and deliver the appropriate output. Key physical components include the **cabinet** (also known as the **chassis**), which houses most of the internal hardware like the **CPU (Central Processing Unit)**, **RAM (Random Access Memory)**, and other essential circuits. The **CPU**, often considered the brain of the computer, is responsible for executing instructions and performing calculations, while **RAM** serves as temporary storage for data currently in use, allowing for quick access and retrieval.

Other visible components include the **monitor**, which displays the graphical output, and input devices like the **mouse** and **keyboard**, which allow users to interact with the system. When a user inputs data, for example by typing on the keyboard or clicking with the mouse, the instructions are processed by the internal components, and the resulting output is displayed on the monitor or through other output devices like a printer or speakers.

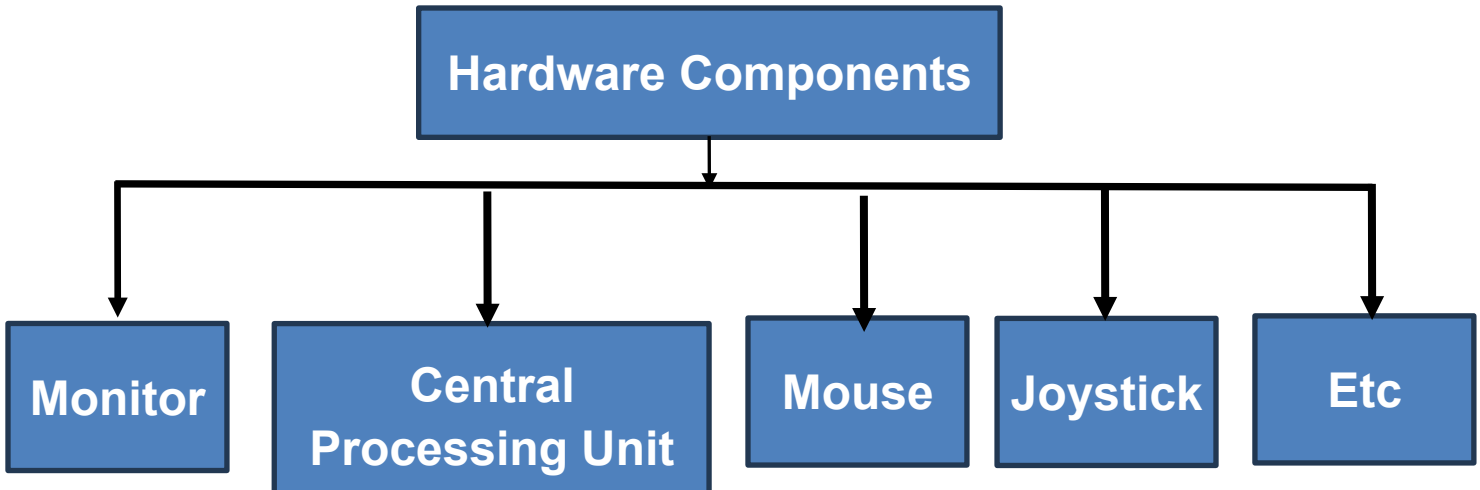
In a broader sense, a computer system is made up of two main elements:

- **Hardware**: The physical devices and machinery of the computer.
- **Software**: The programs, applications, and operating systems that instruct the hardware on how to perform specific tasks.

Together, hardware and software form a complete and functional computer system capable of performing a vast range of operations, from simple calculations to complex data analysis.

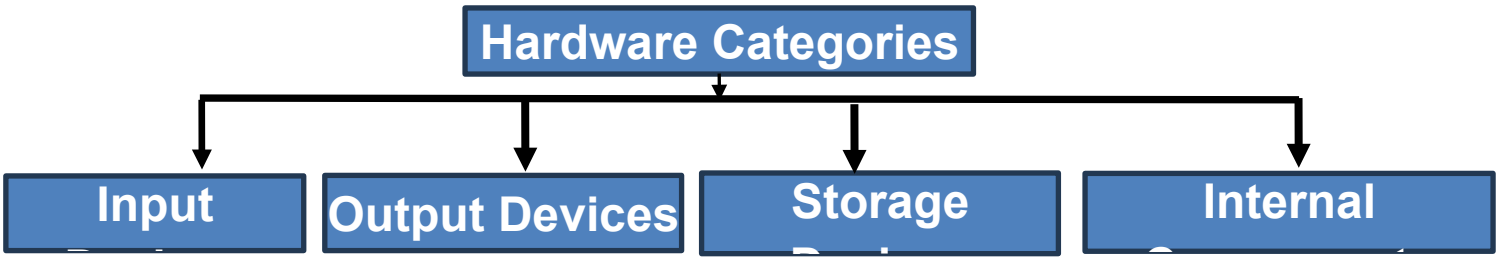


This version provides more context on the roles of the individual hardware components and how they interact with the software to create a functioning computer system.



Hardware Categories

- Input Devices: Devices that allow interaction with the computer (e.g., keyboard, mouse, scanner, light pen, bar code reader).
- Output Devices: Devices that display the output in human-readable form (e.g., monitor, printer, speaker).
- Storage Devices: Components for data storage (e.g., hard drives).
- Internal Components: Includes CPU, motherboard, RAM, power supply, cooling fans.



Computer Hardware Review

Internal Components

1. **CPU (Central Processing Unit):** Its also known as the heart of the computer. It consists of three units, generally known as the control unit, Arithmetic Logical Unit (ALU), and the memory unit.

2. **Mother Board:** It contain the main circuit board inside a computer and contains most of the electronic components together. All the components of the computer are directly or indirectly connected to the motherboard. It includes RAM slots, controllers, system chipsets etc.

Internal Components

3. **RAM (Random Access Memory):** It is also known as temporary or volatile memory. It holds the program and data, which are currently in process or processing.

4. **Power Supply**

All of a computer system's parts are powered by a power source. Typically, a power cord is used to connect a computer tower to an electrical outlet.

5. **Cooling Fan**

   A computer's system to prevent overheating uses cooling fans. To aid customers who use their computers intensively, such as when streaming video or playing games, many computers contain more than one cooling fan.

6. **Hard Drive**

   On a computer system, files, programs, and other types of information are stored on hard drives, which are data storage devices. They utilise hard drives, which are magnetically coated discs used to store digital versions of information. A computer technician can suspect a corrupt hard disk when a hard drive dies.

Computer System

- Computer system consists of hardware components that have been carefully chosen so that they work well together and software components or programs that run in the computer.
- The main software component is itself an operating system (OS) that manages and provides services to other programs that can be run in the computer.
- In its most basic form, a computer system is a programmable electronic device that can accept input; store data; and retrieve, process and output information.

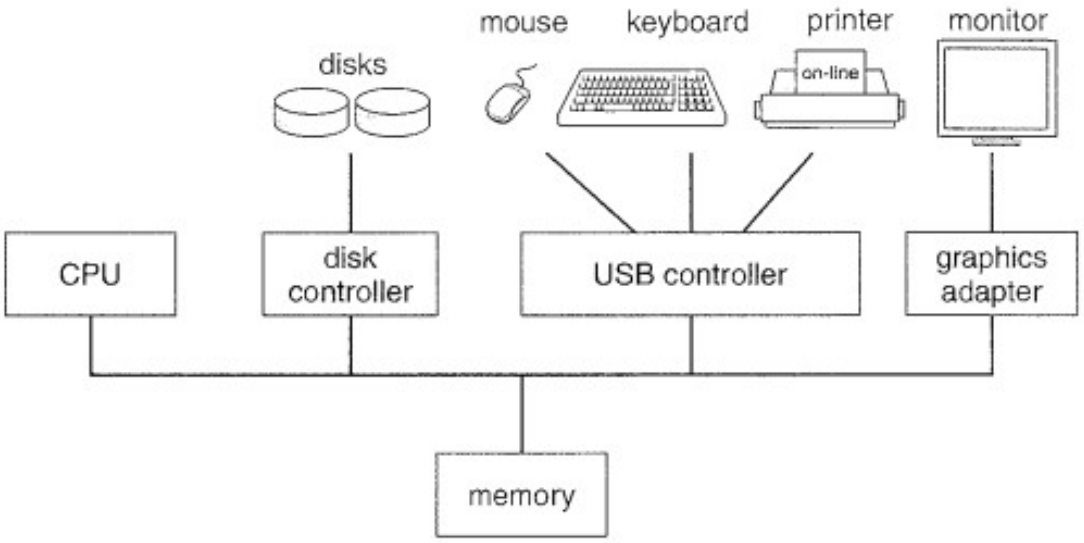


Fig 12. A modern computer system.

Introduction to Operating System: Definition

- An Operating System (OS) is a system software which is a collection of software that manages computer hardware resources and provides common services for computer programs. The operating system is the most important type of system software in a computer system.
- An operating system is like a bridge between your computer's hardware and the programs you use. It makes sure that your computer's memory, CPU, and storage are used effectively while running.
- Operating System is a fully integrated set of specialized programs that handle all the operations of the computer. It controls and monitors the execution of all other programs that reside in the computer, which also includes application programs and other system software of the computer. Examples of Operating Systems are Windows, Linux, Mac OS, etc.

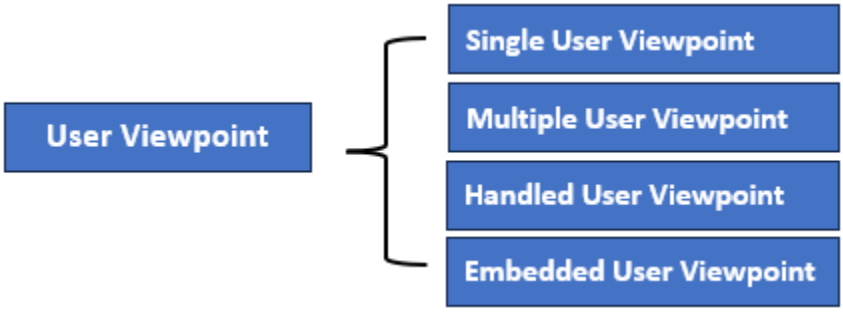
Operating System: View

An operating system can be defined or observed in two ways

- User View

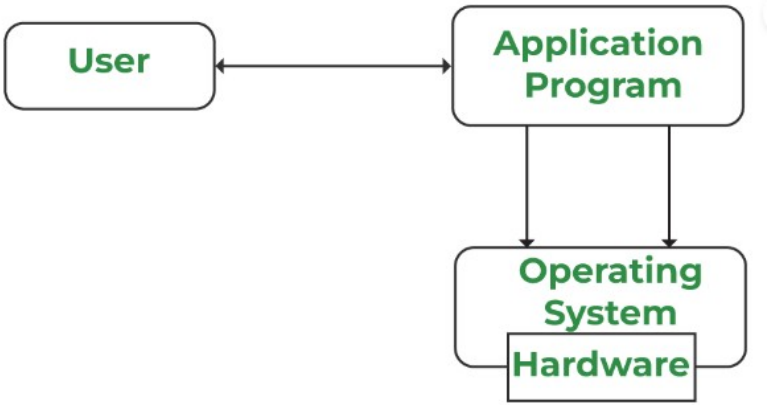
- System View

- User View: Focuses on how users interact with application programs. Types include single-user, multiple-user, handled user, and embedded user viewpoints.
- System View: Focuses on how hardware interacts with the operating system to manage resources and ensure maximum performance.



Operating System: User Viewpoint

- **Single User Viewpoint**: These systems are designed for a single user experience and meet the needs of a single user
- The performance is not given focus as the multiple user systems.



Multiple User Viewpoint

- These systems consists one mainframe computer and many users on their computers trying to interact with their kernels over the mainframe to each other.
- In such systems, memory allocation by the CPU must be done effectively to give a good user experience.
- The client-server architecture is another good example where many clients may interact through a remote server

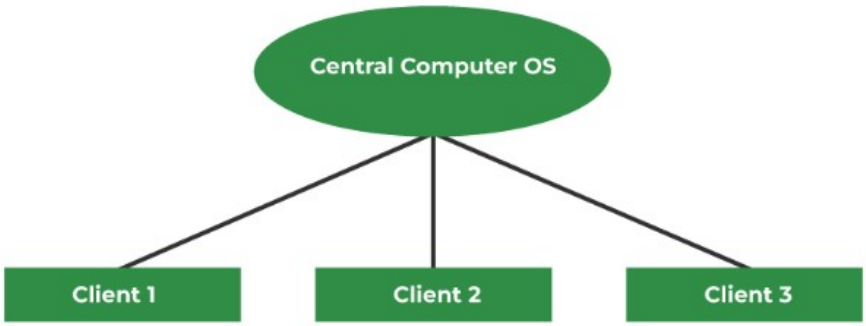


Fig 14. Multiple User Viewpoint

Handled User Viewpoint

- These systems are lies under touchscreen era that comes with best handheld technology ever. Smartphones interact via wireless devices to perform numerous operations,
- Such operating system is a great example of creating a device focused on the user's point of view.

Embedded User Viewpoint

- Systems in which remote control used to turn on or off the tv is all part of an embedded system in which the electronic device communicates with another program where the user viewpoint is limited and allows the user to engage with the application.

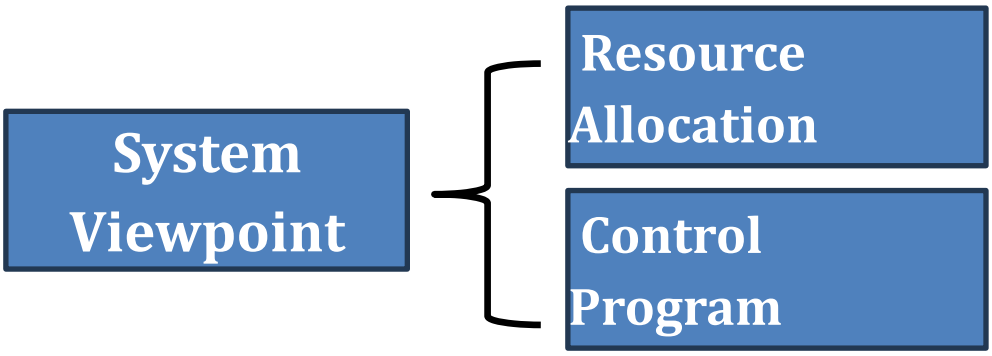
Operating System: View

An operating system can be defined or observed in two ways

- User View
- System View

System View

- A computer system comprises various sources, such as hardware and software, which must be managed effectively. The operating system manages the resources, decides between competing demands, controls the program execution, etc.
- According to this point of view, the operating system's purpose is to maximize performance. The operating system is responsible for managing hardware resources and allocating them to programs and users to ensure maximum performance.



From a system viewpoint, the hardware interacts with the operating system than with the user. The hardware and the operating system interact for a variety of reasons, including:

Resource Allocation

- The hardware contains several resources like registers, caches, RAM, ROM, CPUs, I/O interaction, etc. These are all resources that the operating system needs when an application program demands them.

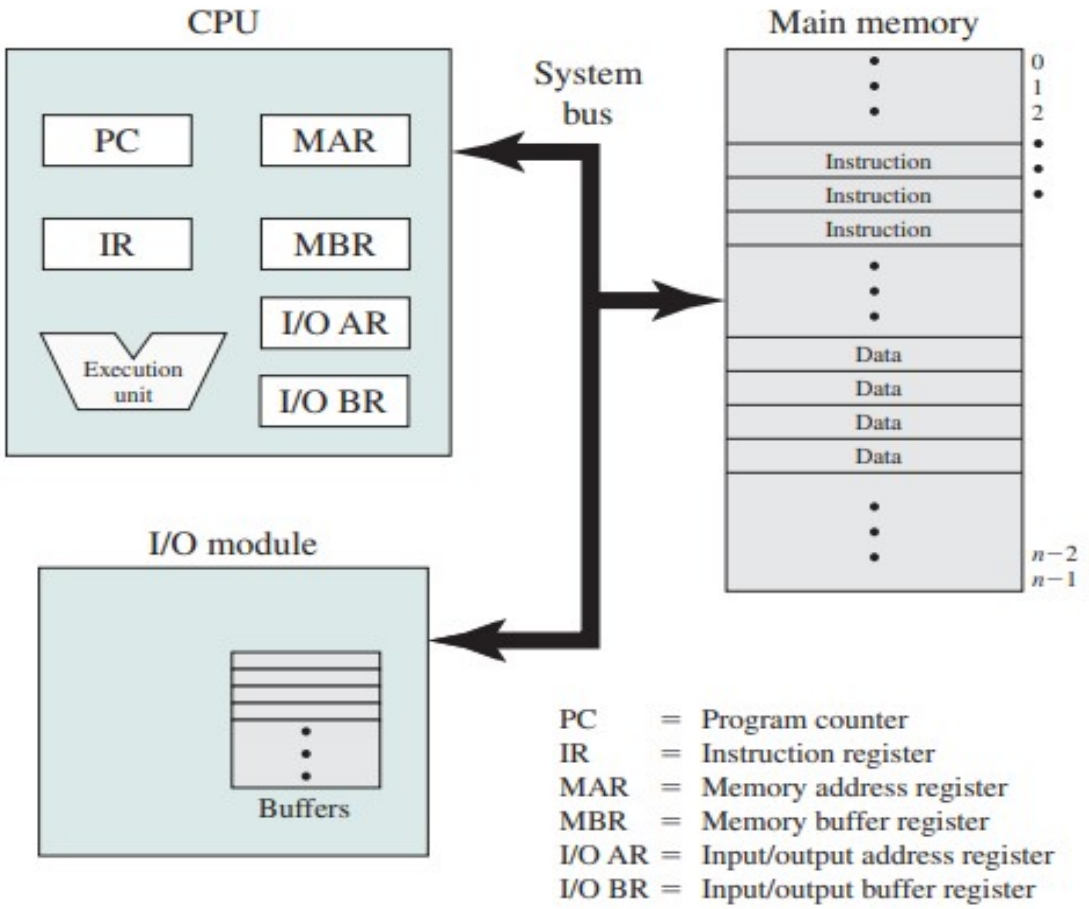
- Only the operating system can allocate resources with several tactics and strategies to maximize its processing and memory space. The operating system uses a variety of strategies to get the most out of the hardware resources, including paging, virtual memory, caching, and so on.

Control Program

- The control program controls how input and output devices (hardware) interact with the operating system.

- The user may request an action that can only be done with I/O devices; in this case, the operating system must also have proper communication, control, detect, and handle such devices.

Computer Components:
Top Level View



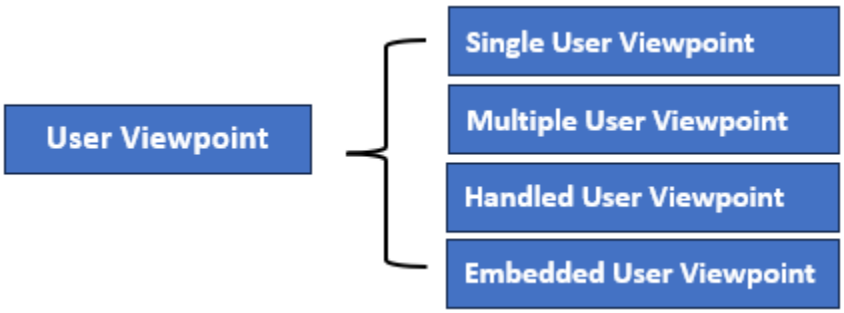
---

1. **History**
2. **Types of Operating System**
3. **Functions of Operating System**

Operating System: History

- An operating system is a type of software that acts as an interface between the user and the hardware.
- It is responsible for handling various critical functions of the computer or any other machine. Various tasks that are handled by OS.

Evolution (Generation) of Operating System



Operating System: Generations

The First Generation (1940 – 1950)

- An OS was not included in the creation of the first electrical computer.
- Early computer users had complete control over the device and wrote programs in pure machine language for every task.
- During the computer generation, a programmer can merely execute and solve basic mathematical calculations. an operating system is not needed for these computations.

The Second Generation (1955 – 1965)

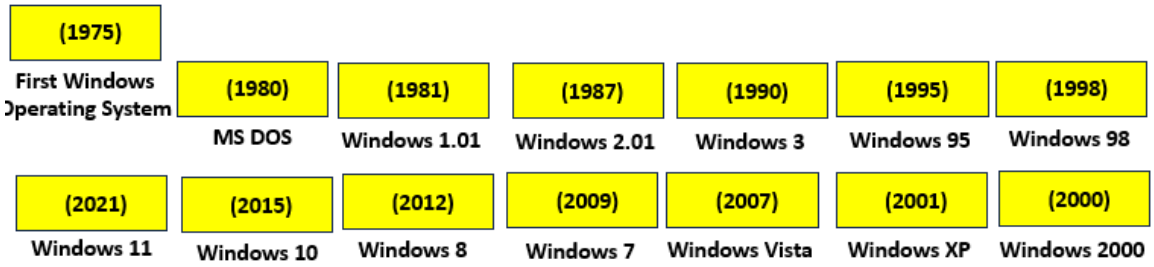
- GMOSIS, the first operating system (OS) was developed in the early 1950s.
- General Motors has created the operating system for the IBM Computer.
- The second-generation operating system was built on a single-stream batch processing system
- It gathers all related jobs into groups or batches and then submits them to the operating system using a punch card to finish all of them.

The Third Generation (1965 – 1980)

- In 2nd Generation, the operating system cleans up after each work is finished before reading and starting the subsequent job on a punch card
- Operating system designers were able to create a new operating system in the late 1960s that was capable of multiprogramming—the simultaneous execution of several tasks in a single computer program.

The Fourth Generation (1980 – Present Day)

- The evolution of the personal computer is linked to the fourth generation of operating systems. Nonetheless, the third-generation minicomputers and the personal computer have many similarities
- At that time, minicomputers were only slightly more expensive than personal computers, which were highly expensive.
- The development of Microsoft and the Windows operating system was a significant influence in the creation of personal computers.



Operating System: Types

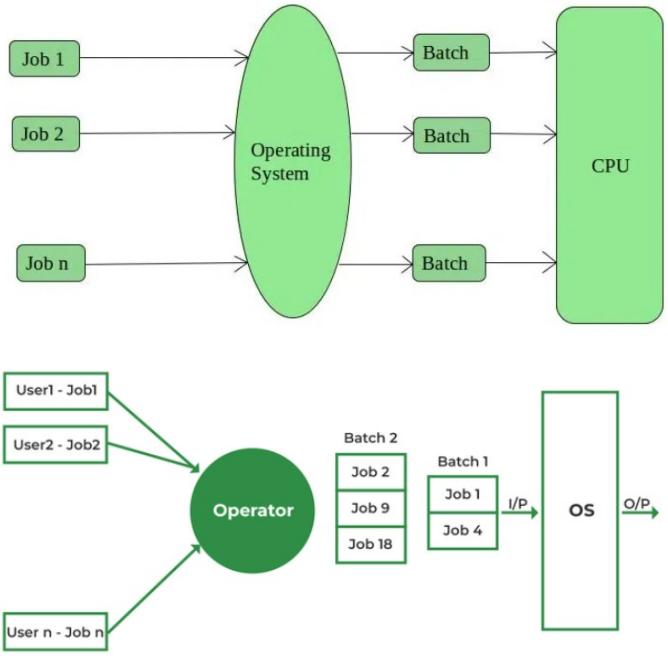
An OS performs all the basic tasks like managing files, processes, and memory. Thus, OS acts as the manager of all the resources, i.e. resource manager and becomes an interface between the user and the machine. It is one of the most required software that is present in the device.

There are several types of Operating Systems:

- Batch Operating System
- Multi-Programming System
- Multi-Processing System
- Multi-Tasking Operating System
- Time-Sharing Operating System
- Distributed Operating System
- Network Operating System
- Real-Time Operating System

Operating System: Batch Processing OS

The batch-processing operating system was very popular in the **1970s**. In batch operating system the jobs were performed in batches. This means Jobs having similar requirements are grouped and executed as a group to speed up processing. Users using batch operating systems do not interact with the computer directly. It is the responsibility of the operator to sort jobs with similar needs



Operating System: Batch Processing OS

Advantages of Batch processing OS

- Multiple users can share the batch systems.
- The idle time for the batch system is very less.
- It is easy to manage large work repeatedly in batch systems.
- Some examples are Payroll Systems, Bank Statements, etc.

Disadvantages of Batch processing OS

- The computer operators should be well known with batch systems.
- Batch systems are hard to debug and sometimes costly.
- The other jobs will have to wait for an unknown time if any job fails.
- In batch operating system the processing time for jobs is commonly difficult to accurately predict while they are in the queue.
- It is difficult to accurately predict the exact time required for a job to complete while it is in the queue.

Operating System: Multi-Programming OS

Before Multiprogramming, there were single tasking operating systems like MS DOS that used to allow only one program to be loaded at a time and run. These systems were not efficient as CPU was not used efficiently. E.g., in

a single tasking system if the current program waits for some input/output to finish, the CPU is not used. The idea of multiprogramming is to assign CPUs to other processes while the current process might not be finished.

- Multiprogramming OS is an ability of an operating system that executes more than one program using a single processor machine.
- In Multi-programming, More than one task or program or jobs are present inside the main memory at one point of time.

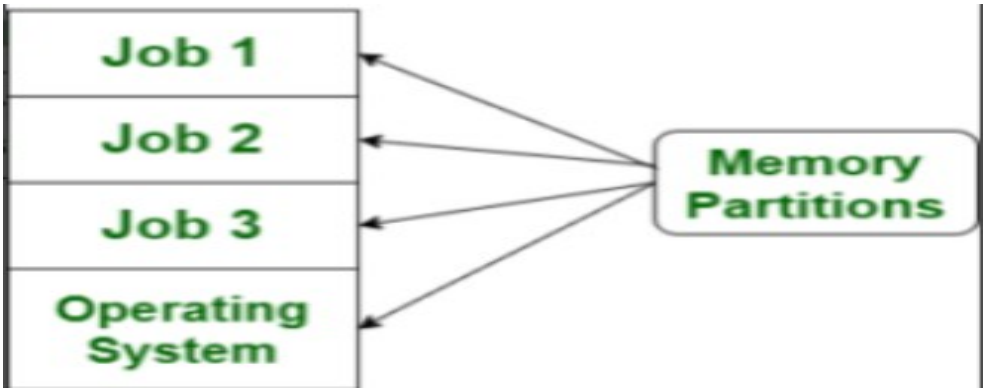


Fig 3. Multi-Programming OS

Operating System: Multi-Programming OS

Job

In systems using multiprogramming a program loaded to memory and ready to execute is called a job.

States: In a multiprogramming system, a job can be in one of three states.

- **Running**: The job is currently executing on the CPU. At any time, at most one job can be in this state.
- **Ready**: The job is ready to run but currently not selected to do so.
- **Waiting**: The job is blocked from running on the CPU while waiting for an I/O request to be completed.

Operating System: Multi-Programming OS

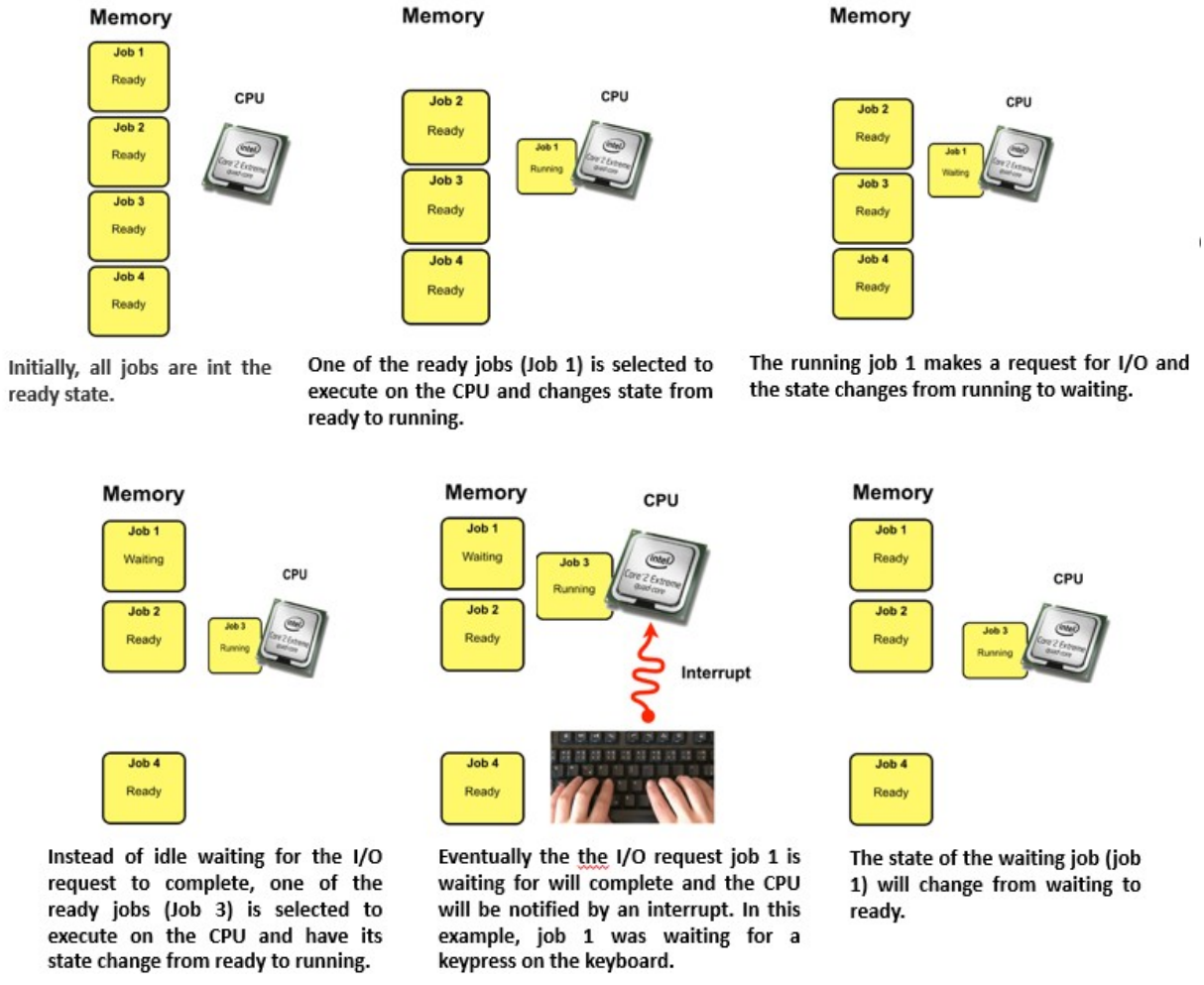
State Transitions

In a multiprogramming system, the following state transitions are possible

From	To	Description
Running	Waiting	When a running job requests I/O, the job changes state from running to waiting.
Running	Ready	When an I/O requests completes, the running job changes state from running to ready.
Waiting	Ready	When an I/O requests completes, the job waiting for the request to complete changes state from waiting to ready.
Ready	Running	When an I/O requests completes, one of the ready jobs are selected to run on the CPU and changes state from ready to running.

Fig 4. State Transition in Multi-Programming System

Operating System: Multi-Programming OS



Operating System: Multi-Programming OS

Advantages of Multiprogramming

- Need Single CPU for implementation.
- Context switch between process.
- Switching happens when current process undergoes waiting state.
- CPU idle time is reduced.
- High resource utilization.
- High Performance.

Disadvantages of Multiprogramming

- Prior knowledge of scheduling algorithms (An algorithm that decides which next process will get hold of the CPU) is required.
- If it has a large number of jobs, then long-term jobs will have to require a long wait.
- Memory management is needed in the operating system because all types of tasks are stored in the main memory.
- Using multiprogramming up to a larger extent can cause a heat-up issue.

Operating System: Multi-Processing System

- A multiprocessing operating system is defined as a type of operating system that makes use multiple central processing units within a single system to improve performance.
- It enables a system to support more than one processor and divide the tasks among them.
- Every process requires a CPU for its execution. So, this allows multiple processes to execute parallelly on different processing units.
- All available processors are connected to peripheral devices, computer buses, physical memory, and clocks.
- The main aim of the multi-processing operating system is to increase the speed of execution of the system.
- For example, UNIX, LINUX, and Solaris are the most widely used multi-processing operating system.

Operating System: Multi-Processing System

Working of Multi-Processing OS

- In a multiprocessing operating system, the workload is divided among the multiple processors or cores.
- Each processor handles a specific task, which allows for improved performance and faster execution.
- After the completion of the task, the results from each processor are compiled to produce a single output.
- The operating system manages the allocation of resources and ensures that each processor is assigned a task it can handle efficiently.
- This results in better resource utilization and optimized system performance.

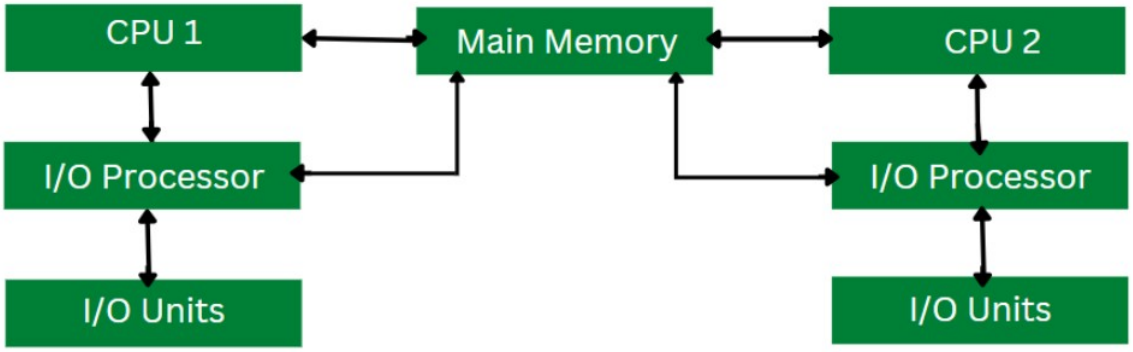


Fig 6. Multi-Processing OS

Operating System: Multi-Processing System

Advantages of Multiprocessing OS

- **Increased reliability:** Due to the multiprocessing system, processing tasks can be distributed among several processors. This increases reliability as if one processor fails; the task can be given to another processor for completion.
- **Increased throughout:** As several processors increase, more work can be done in less
- **The economy of Scale:** As multiprocessors systems share peripherals, secondary storage devices, and power supplies, they are relatively cheaper than single-processor systems.

Disadvantages of Multiprocessing operating System

- Multiprocessing Operating Systems are complex and require specialized knowledge.
- The cost of a Multiprocessing Operating system can be high because of the need for specialized hardware resources.
- They may face compatibility issues with software that is not designed to work with multiprocessing operating systems.
- Achieve Synchronization between multiple processors in a multiprocessing operating system is a challenging task.

Operating System: Multi-Tasking OS

- The term "multitasking" is commonly used in modern computer systems.
- It is an advancement of multiprogramming systems, which enables the execution of several programs concurrently.
- A multitasking operating system enables a user to perform multiple computer tasks simultaneously. These tasks are referred to as processes, and they share processing resources such as the CPU.
- The operating system maintains a record of the progress of each process and enables users to switch between them without losing any data.

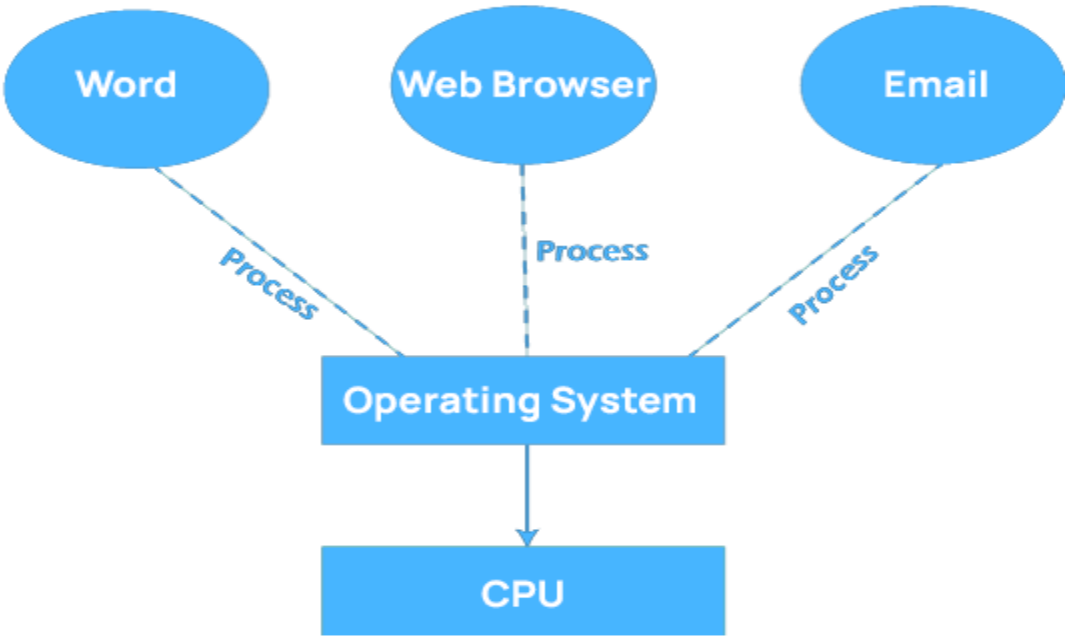


Fig 7. Multi-Tasking OS

- Early operating system could execute various programs at the same time, although multitasking was not fully supported.
- As a result, a single software could consume the entire CPU of the computer while completing a certain activity.
- Basic operating system functions, such as file copying, prevented the user from completing other tasks, such as opening and closing windows.
- Fortunately, because modern operating systems have complete multitasking capability, numerous programs can run concurrently without interfering with one other.
- In addition, many operating system processes can run at the same time.

Features of Multi-Tasking Operating System

- **Time Sharing:** Many processes are allocated with resources of computer in respective time slots, processors time is shared with multiple processes.
- **Context Switching:** Context switching is a process of saving the context of one process and loading the context of another process. In simpler terms it is loading another process when the prior process has finished its execution.
- **Multi-Threading:** Multithreading is the ability of a program or an operating system to enable more than one user at a time without requiring multiple copies of the program running on the computer.
- **Hardware Interrupt:** When a process or an event requires urgent attention, hardware or software will signal with an interrupt. It informs the processor that a high-priority task has arisen that necessitates interrupting the running process.

Types of Multi-Tasking OS

Pre-emptive Multi-Tasking Operating System:

- In pre-emptive multitasking, the operating system can initiate a context switching from the running process to another process.

- In other words, the operating system allows stopping the execution of the currently running process and allocating the CPU to some other process.
- The OS uses some criteria to decide for how long a process should execute before allowing another process to use the operating system.
- The mechanism of taking control of the operating system from one process and giving it to another process is called pre-emption. Here are some Examples UNIX, Windows 95, Windows NT operating system.

Types of Multi-Tasking OS

Non-pre-emptive Multi-Tasking Operating System:

- Non-pre-emptive Multi-Tasking Operating System is also known as cooperative multitasking, this operating system never initiates context switching from the running process to another process.
- A context switch occurs only when the processes voluntarily yield control periodically or when idle or logically blocked to allow multiple applications to execute simultaneously.
- Also, in this multitasking, all the processes cooperate for the scheduling scheme to work. Example – Macintosh OS version 8.0-9.2.2 and Windows 3.x operating system.

Advantages of Multitasking Operating System

- Multitasking operating systems allow multiple applications to run concurrently without affecting CPU performance, making them suitable for multiple users working simultaneously.
- Multitasking operating systems possess a robust virtual memory system that eliminates long wait times for program execution by shifting applications to virtual memory if necessary.
- Additionally, to prevent CPU wait times, all jobs in Multitasking Operating System are given a predetermined duration.
- Multitasking operating systems can efficiently manage computer resources such as I/O devices, RAM, hard disks, and CPUs.
- Users can concurrently run several programs, such as internet browsers, games, Microsoft Excel, PowerPoint, and other utilities in a Multitasking Operating System.

Disadvantages of Multitasking Operating System

- Due to the slower pace of the processors in a Multitasking Operating System, the computer system may perform slowly and experience longer response times when running multiple programs. To fix this issue, additional processing power may be required.
- Running numerous programs concurrently in a Multitasking Operating System can overload the main memory, leading to slow system performance and increased reaction times since the CPU cannot allocate sufficient time for each program.
- In a multitasking operating system, multiple processors work simultaneously to complete tasks, resulting in increased CPU heat generation.

Operating System: Time Sharing OS

- Multi-programmed, batched systems provide an environment where various system resources were used effectively, but it did not provide for user interaction with computer systems. Time-sharing is a logical extension of multi-programming. A time-sharing operating system i design allows multiple users (processes) to concurrently share the same system resources, such as the CPU, memory, and peripherals. It enables each user or process to have the illusion of having dedicated access to the system while effectively sharing resources in a time-sliced manner.

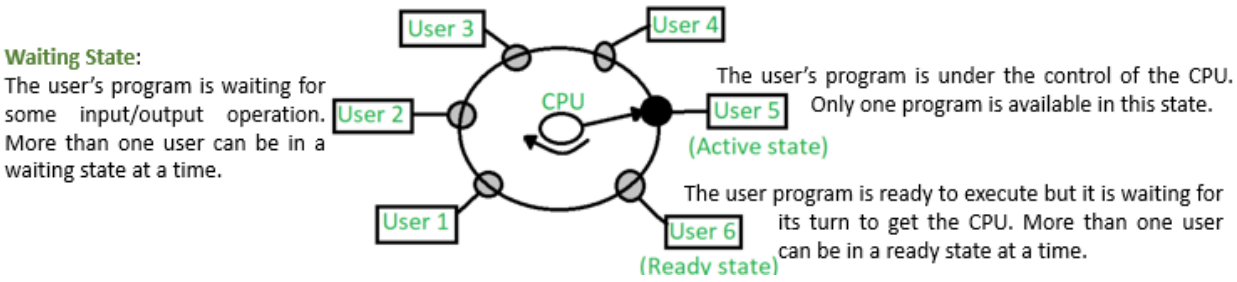


Fig 8. Time Shared OS

- Time-shared operating system uses CPU scheduling and multi-programming to provide each user with a small portion of a shared computer at once.
- Each user has at least one separate program in memory.
- A program is loaded into memory and executes, it performs a short period of time either before completion or to complete I/O.
- This short period of time during which the user gets the attention of the CPU is known as time slice, time slot, or quantum. It is typically of the order of 10 to 100 milliseconds.
- An alarm clock mechanism to send an interrupt signal to the CPU after every time slice.

Advantages of Time-Sharing OS

- Each task gets an equal opportunity.
- Fewer chances of duplication of software.
- CPU idle time can be reduced.

Disadvantages of Time-Sharing OS

- Reliability problem.
- One must have to take of the security and integrity of user programs and data.
- Data communication problem.

Operating System: Distributed OS

A distributed operating system is one in which several computer systems connected through a single communication channel. Moreover, these systems have their individual processors and memory. Furthermore, these processors communicate through high-speed buses or telephone lines. These individual systems that connect through a single channel are considered as a single unit. We can also call them loosely coupled systems. The individual components or systems of the network are nodes.

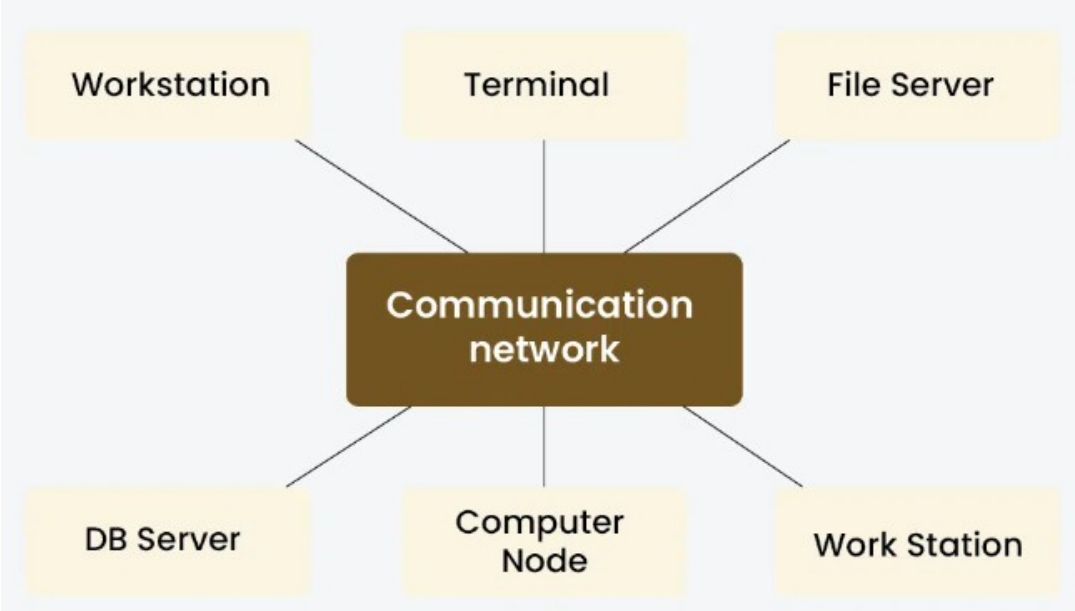


Fig 9. Distributed OS

Types of Distributed OS

- **Client-Server Systems**

In a client-server system within a distributed operating system, clients request services or resources from servers over a network. Clients initiate communication, send requests, and handle user interfaces, while servers listen for requests, perform tasks, and manage resources.

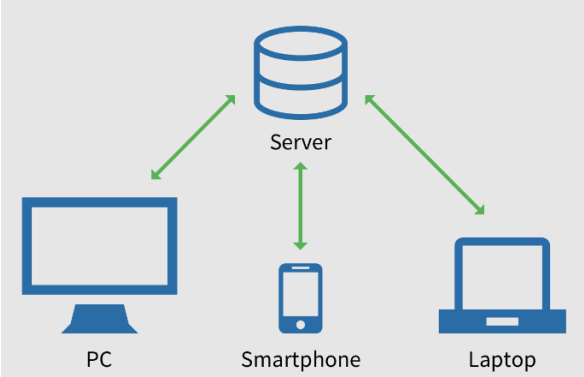


Fig 10. Client-Server OS

Peer-to-Peer(P2P) Systems

In peer-to-peer (P2P) systems, interconnected nodes directly communicate and collaborate without centralized control. Each node can act as both a client and a server, sharing resources and services with other nodes. P2P systems enable decentralized resource sharing, self-organization, and fault tolerance.

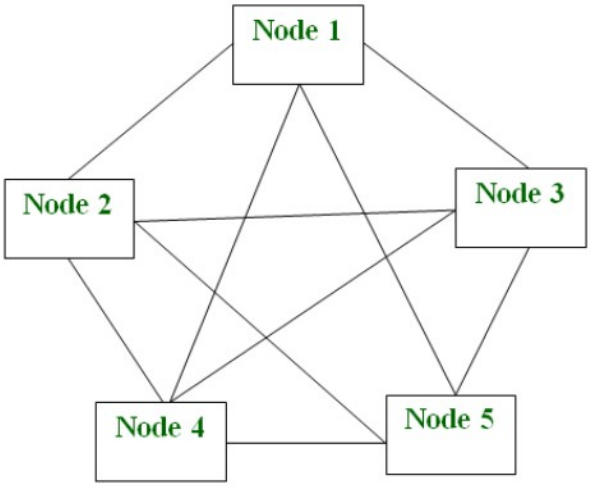
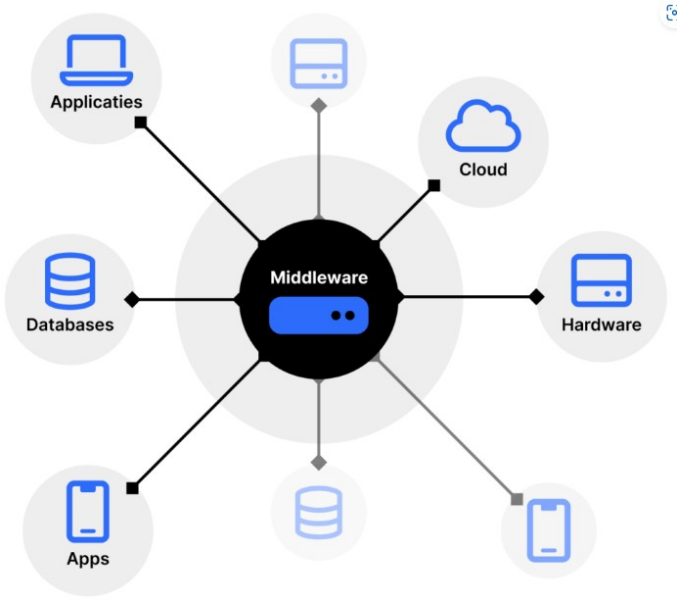


Fig 11. Peer-to-Peer OS

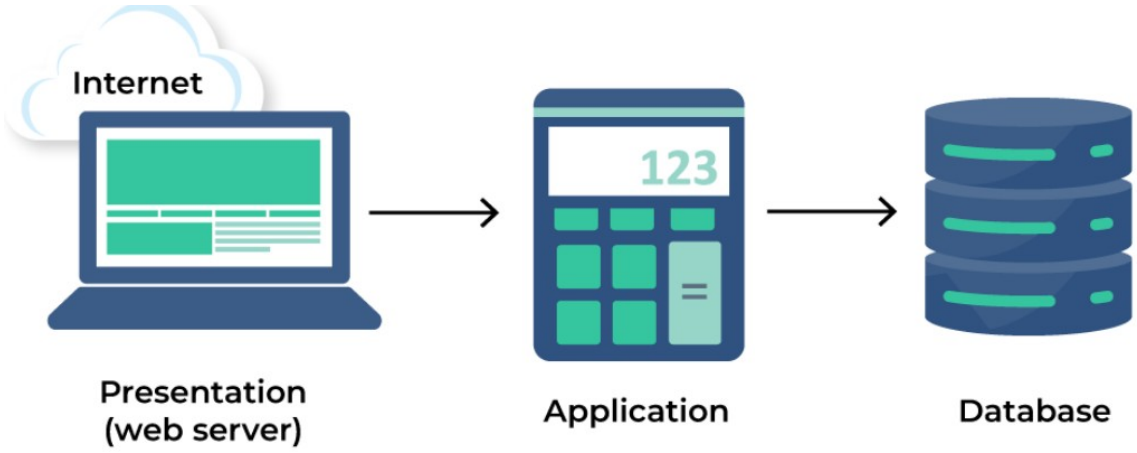
Middleware

Middleware acts as a bridge between different software applications or components, enabling communication and interaction across distributed systems. It abstracts complexities of network communication, providing services like message passing, remote procedure calls (RPC), and object management.



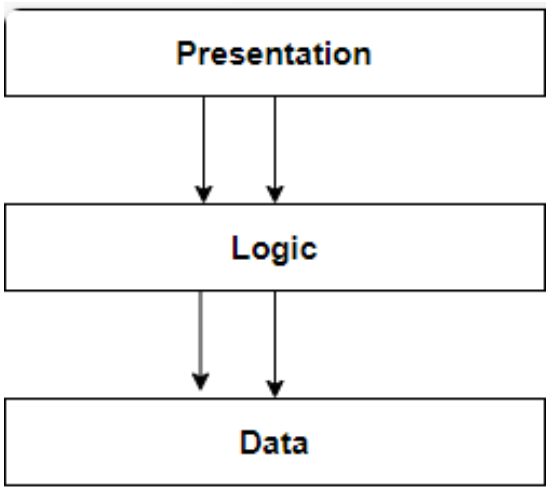
Three-Tier OS

In a distributed operating system, “Tier” refer to the physical separation of components. the three-tier architecture divides tasks into presentation, logic, and data layers. The presentation tier, comprising client machines or devices, handles user interaction. The logic tier, distributed across multiple nodes or servers, executes processing logic and coordinates system functions.



N-Tier OS

In an N-tier architecture, applications are structured into multiple tiers or layers beyond the traditional three-tier model. Each tier performs specific functions, such as presentation, logic, data processing, and data storage, with the flexibility to add more tiers as needed. In a distributed operating system, this architecture enables complex applications to be divided into modular components distributed across multiple nodes or servers.



Examples of Distributed OS

- **Solaris:** The SUN multiprocessor workstations are the intended use for it.
- **OSF/1:** The Open Foundation Software Company designed it, and it works with Unix.
- **Micros:** All nodes in the system are assigned work by the MICROS operating system, which also guarantees a balanced data load.
- **DYNIX:** It is created for computers with many processors, known as Symmetry.
- **Locus:** It can be viewed simultaneously from both local and distant files without any location restrictions.
- **Mach:** It permits the features of multitasking and multithreading.

Advantages of Distributed OS

- It can increase data availability throughout the system by sharing all resources (CPU, disk, network interface, nodes, computers, and so on) between sites.
- Because all data is replicated across all sites, it reduces the probability of data corruption because users can access data from another operating site if one site fails.
- Data transfer from one site to another is accelerated by it.
- Since it may be accessible from both local and remote sites, it is an open system.
- It facilitates a reduction in the time needed to process data.
- Most distributed systems are composed of multiple nodes that work together to provide fault tolerance. Even if one machine malfunctions, the system still functions.

Disadvantages of Distributed OS

- Which tasks need to be completed, when they need to be completed, and where they need to be completed must be determined by the system. The restrictions of a scheduler might result in unpredictable runtimes and unused hardware.
- Since the nodes and connections in DOS need to be secured, it is challenging to establish sufficient security.
- Comparing a DOS-connected database to a single-user system, the latter is easier to maintain and less complex.
- Compared to other systems, the underlying software is incredibly sophisticated and poorly understood.
- Compiling, analyzing, displaying, and keeping track of hardware utilization metrics for large clusters may be quite challenging.

Operating System: Network OS

- Since an operating system is that the operating system is the interface between the computer hardware and the user. In daily life, we use the operating system on our devices which provides a good GUI, and many more features.
- Similarly, a network operating system(NOS) is software that connects multiple devices and computers on the network and allows them to share resources on the network. Let’s see what are the functions of the network operating system.

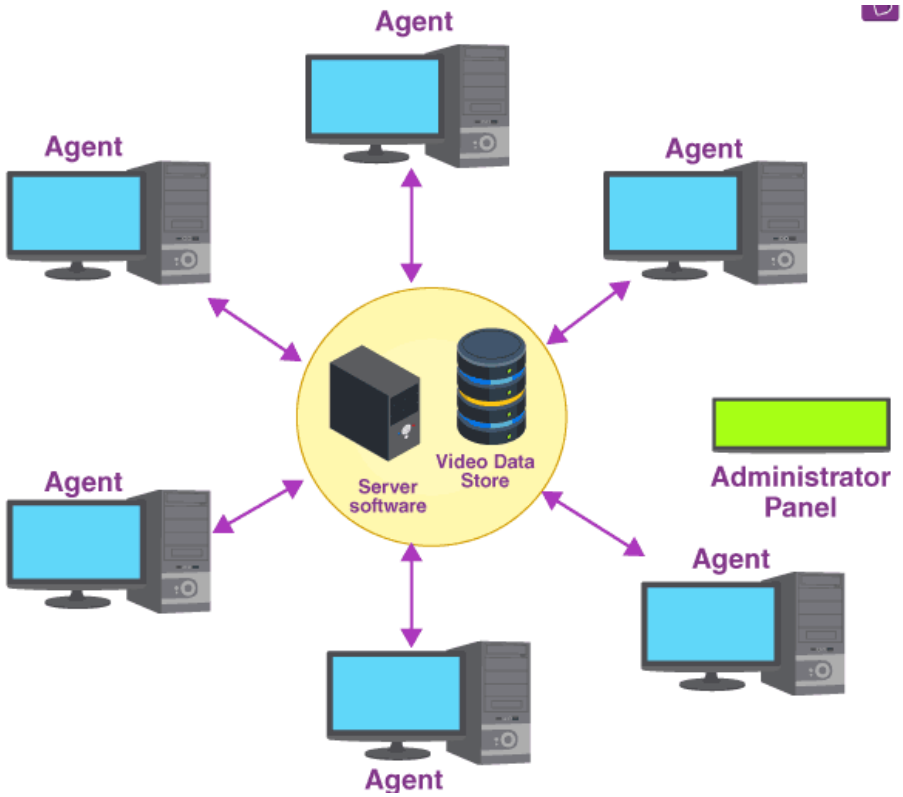


Fig 15. Network OS

Functions of the Network OS

- Creating and managing user accounts on the network.
- Controlling access to resources on the network.
- Provide communication services between the devices on the network.
- Monitor and troubleshoot the network.
- Configuring and Managing the resources on the network.

Types of Network OS

- **Peer to Peer**

Peer-to-peer network operating systems allow the sharing of resources and files with small-sized networks and having fewer resources. In general, peer-to-peer network operating systems are used on LAN.

- **Client/server**

Client-server network operating systems provide users access to resources through the central server. This NOS is too expensive to implement and maintain. This operating system is good for the big networks which provide many services.

Advantages of Network Operating Systems

- Highly stable due to central server.
- Provide good security.
- Upgradation of new technology and hardware can be easily implemented in the network.

- Provide remote access to servers from different locations.

Disadvantages of Network Operating Systems

- Depend on the central location to perform the operations.
- High cost to buying server.
- Regular updating and maintenance are required.

Examples of Network OS

- Microsoft Windows Server, UNIX/Linux, Artisoft's LANtastic, Banyan's VINES

Operating System: Real Time OS

Real-time OS are used in environments where a large number of events, mostly external to the computer system, must be accepted and processed in a short time or within certain deadlines. such applications are industrial control, telephone switching equipment, flight control, and real-time simulations. With a Real Time OS, the processing time is measured in tenths of seconds. This system is time-bound and has a fixed deadline. The processing in this type of system must occur within the specified constraints. Otherwise, This will lead to system failure.

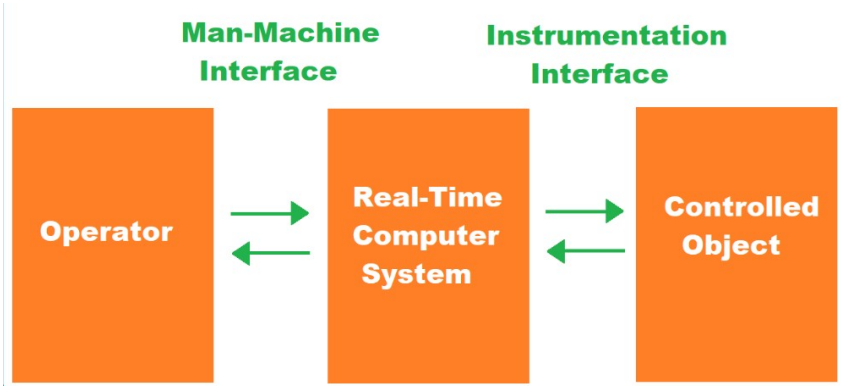


Fig 16. Real Time OS

Types of Real Time OS

Hard Real-Time OS

These operating systems guarantee that critical tasks are completed within a range of time.

For example, a robot is hired to weld a car body. If the robot welds too early or too late, the car cannot be sold, so it is a hard real-time system that requires complete car welding by the robot hardly on time., scientific experiments, medical imaging systems, robots, air traffic control systems, etc.

Soft real-time OS

This operating system provides some relaxation in the time limit.

For example – Multimedia systems, digital audio systems, etc. Explicit, programmer-defined, and controlled processes are encountered in real-time systems. A separate process is changed by handling a single external event. The process is activated upon the occurrence of the related event signaled by an interrupt.

Firm Real-time Operating System

Real Time OS of this type have to follow deadlines as well. In spite of its small impact, missing a deadline can have unintended consequences, including a reduction in the quality of the product.

Example: Multimedia applications.

Deterministic Real-Time OS

Consistency is the main key in this type of real-time operating system. It ensures that all the task and processes execute with predictable timing all the time, which make it more suitable for applications in which timing accuracy is very important. Examples: INTEGRITY, PikeOS.

Advantages of Real-Time OS

- Maximum utilization of devices and systems. Thus, more output from all the resources.
- Time assigned for shifting tasks in these systems is very less. For example, in older systems, it takes about 10 microseconds. Shifting one task to another and in the latest systems, it takes 3 microseconds.
- Focus on running applications and less importance to applications that are in the queue.
- Since the size of programs is small, RTOS can also be embedded systems like in transport and others.
- These types of systems are error-free.
- Memory allocation is best managed in these types of systems.

Dis-advantages of Real-Time OS

- Very few tasks run simultaneously, and their concentration is very less on few applications to avoid errors.
- Sometimes the system resources are not so good, and they are expensive as well.
- The algorithms are very complex and difficult for the designer to write on.
- It needs specific device drivers and interrupts signals to respond earliest to interrupts.
- It is not good to set thread priority as these systems are very less prone to switching tasks.
- Real-Time OS performs minimal task switching.

Operating System: Functions

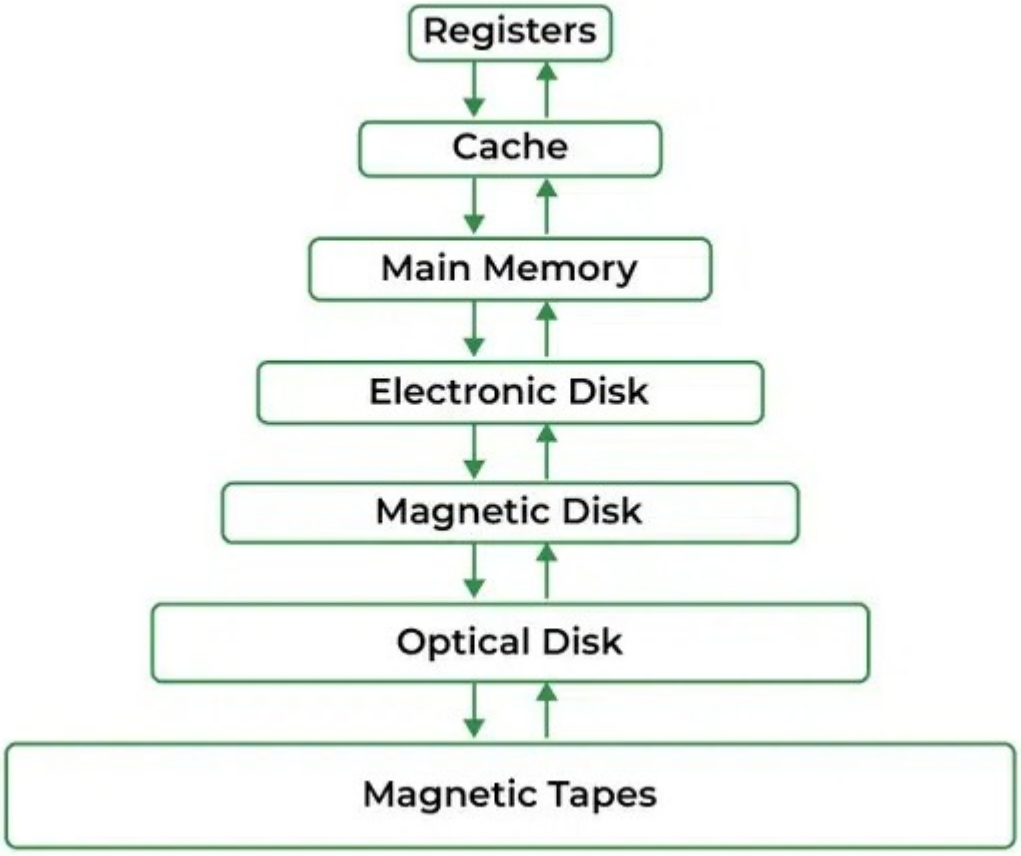


Fig: **Memory Management**

Operating System handles the following responsibilities

- It is the management of the main or primary memory. Whatever program is executed, it must be present in the main memory.
- It is used for achieving better concurrency, system performance, and memory utilization.
- Main memory is a quick storage area that may be accessed directly by the CPU.
- When the program is completed, the memory region is released and can be used by other programs.
- Therefore, there can be more than one program present at a time. Hence, it is required to manage the memory.

Operating System: Functions

- OS Allocates and deallocates the memory.
- OS Keeps a record of which part of primary memory is used by whom and how much.
- OS Distributes the memory while multiprocessing.
- In multiprogramming, the operating system selects which processes acquire memory when and how much memory they get.

- Memory management moves processes from primary memory to secondary memory and vice versa. It also keeps track of available memory, memory allocation, and unallocated.
- It keeps track of the status of each memory location, whether it is allocated or free.
- It enables computer systems to run programs that require more main memory than the amount of free main memory available on the system. This is achieved by moving data between primary and secondary memory.
- It addresses the system's primary memory by providing abstractions such that the programs running on the system perceive a large memory is allocated to them.
- It is the job of memory management to protect the memory allocated to all the processes from being corrupted by other processes. If this is not done, the computer may exhibit unexpected/faulty behavior.
- Memory management enables sharing of memory spaces among processes, with the help of which, multiple programs can reside at the same memory location (although only one at a time).
- Operating System handles the following responsibilities

Process Management

Operating System handles the following responsibilities

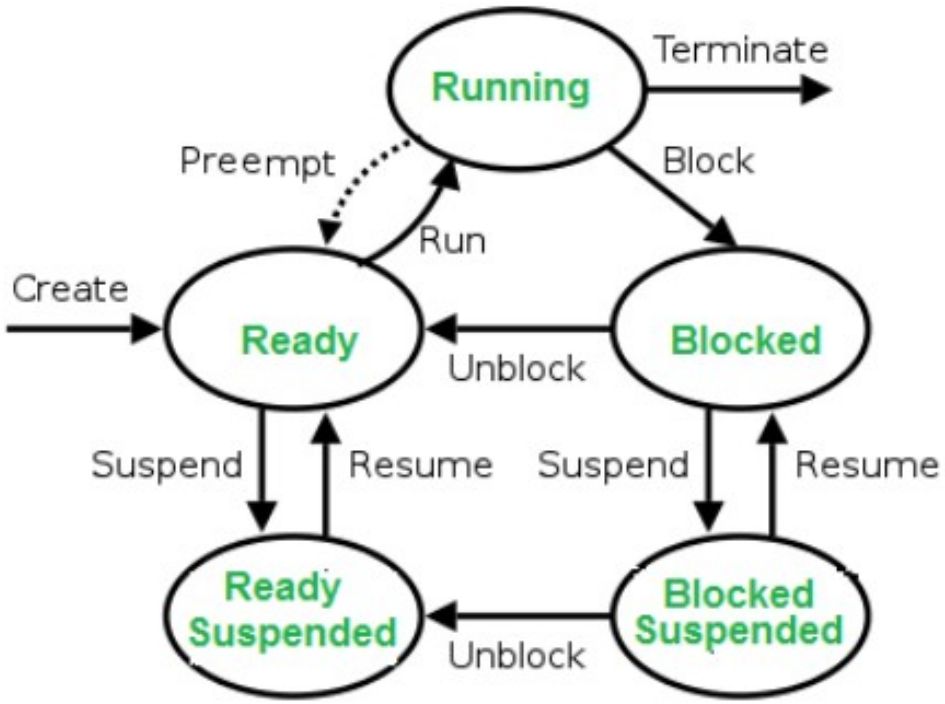


Fig 18. Process Management in OS

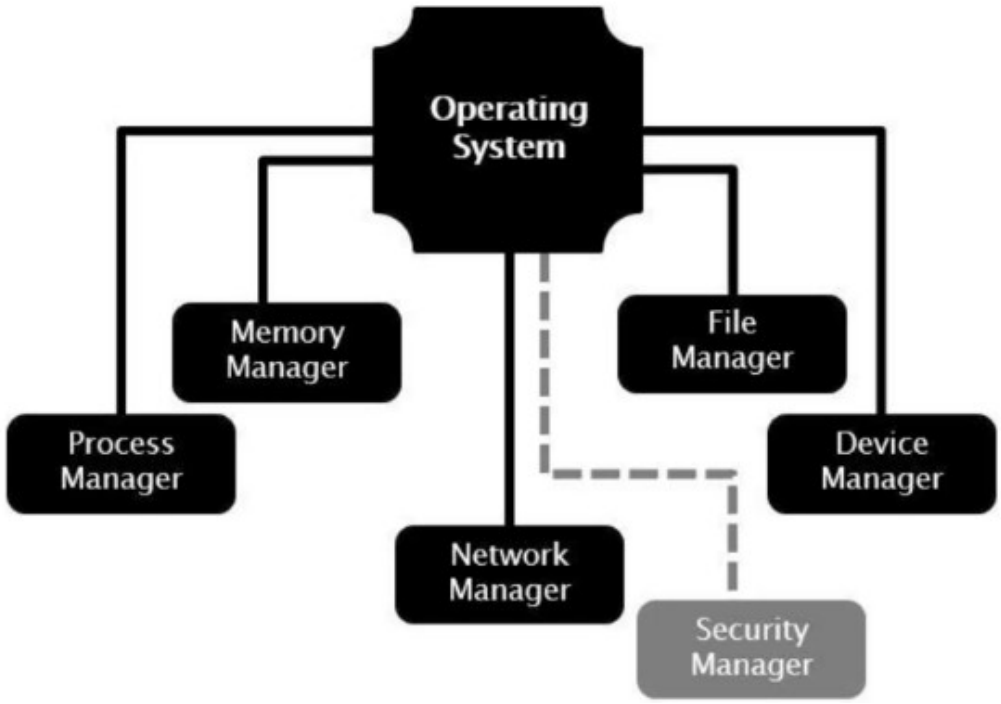
- In a multi-programming environment, the OS decides the order in which processes have access to the processor, and how much processing time each process has.
- This function of OS is called Process Scheduling.

- An Operating System performs the following activities for Processor Management.
- An operating system manages the processor's work by allocating various jobs to it and ensuring that each process receives enough time from the processor to function properly.
- Keeps track of the status of processes. The program which performs this task is known as a traffic controller. Allocates the CPU that is a processor to a process. De-allocates processor when a process is no longer required.

Device Management

- OS Keeps track of all devices connected to the system.
- It designates a program responsible for every device known as the Input/Output controller.
- It decides which process gets access to a certain device and for how long.
- It allocates devices effectively and efficiently. Deallocates devices when they are no longer required.
- There are various input and output devices. An OS controls the working of these input-output devices.
- It receives the requests from these devices, performs a specific task, and communicates back to the requesting process.

Fig 19. Device Management in OS



Operating System handles the following responsibilities

File Management

- A file system is organized into directories for efficient or easy navigation and usage.
- These directories may contain other directories and other files.
- An Operating System carries out the following file management activities.
- It keeps track of where information is stored, user access settings, the status of every file, and more.
- These facilities are collectively known as the file system.
- An OS keeps track of information regarding the creation, deletion, transfer, copy, and storage of files in an organized way.
- It also maintains the integrity of the data stored in these files, including the file directory structure, by protecting against unauthorized access.

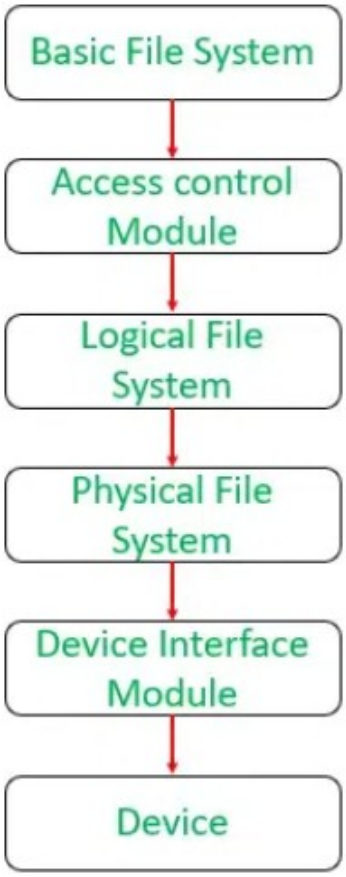


Fig 20 . File Management in OS

Operating System handles the following responsibilities

User Interface or Command Interpreter

- The user interacts with the computer system through the operating system.
- Hence OS acts as an interface between the user and the computer hardware.
- This user interface is offered through a set of commands or a graphical user interface (GUI). Through this interface, the user makes interacts with the applications and the machine hardware.

Booting the Computer

- The process of starting or restarting the computer is known as booting.
- If the computer is switched off completely and if turned on, then it is called cold booting.
- Warm booting is a process of using the operating system to restart the computer.

Fig 21 .Command Interpreter in OS

Operating System: Functions

Operating System handles the following responsibilities

Security

- Protection against unauthorized access through login.
- Protection against intrusion by keeping the firewall active.
- Protecting the system memory against malicious access.
- Displaying messages related to system vulnerabilities.

Error-Detecting Aids

- The operating system constantly monitors the system to detect errors and avoid malfunctioning computer systems.
- From time to time, the operating system checks the system for any external threat or malicious software activity.

Network Management

- OS manage how data is packaged and sent over the network, making sure it arrives safely and in the right order.
- OS also enable to set up your network connections, like Wi-Fi or Ethernet, and keep an eye on how your network is doing.
- OS keep make sure your computer is using the network efficiently and securely.

Services of Operating System

Operating systems (OS) offer a wide range of services to facilitate efficient and secure computing for users and applications.

User Interface Service

The user interface is the service that practically enables users to interact with an operating system. So, it means interacting with the entire computer system itself.

The user interface of operating systems has two common forms: the **Command Line Interface (CLI)** and the **Graphical User Interface (GUI).**

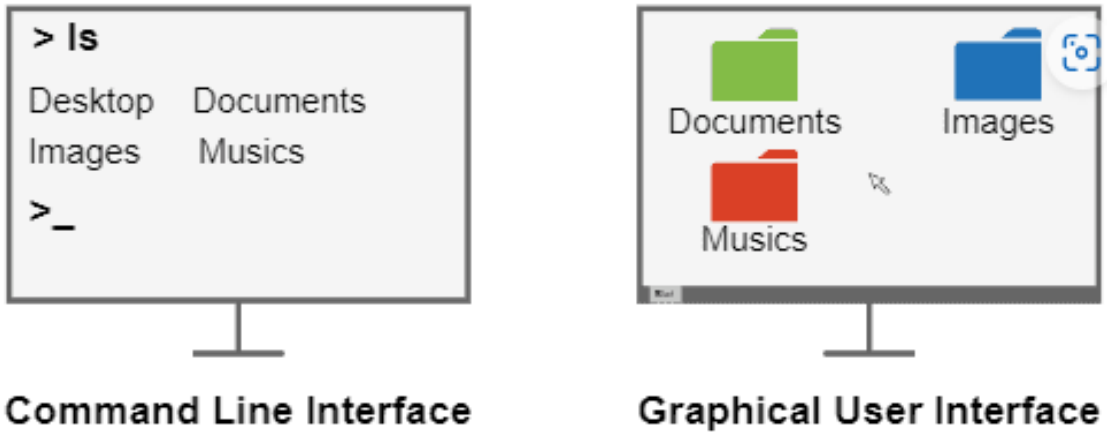
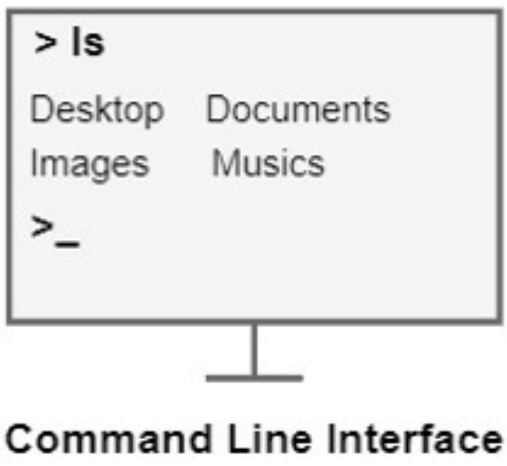


Fig 1. User Interface service in OS

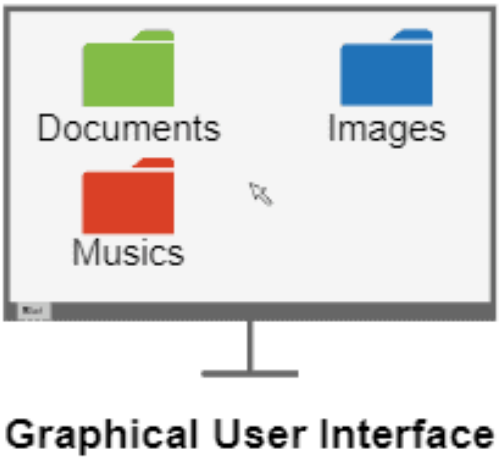
A CLI establishes the interaction between users and the operating system through a text-based set of commands. Thus, the users type commands and their respective arguments and submit them to be processed by the operating system.

Examples of CLI interfaces are the command prompt of Windows and the terminal of Linux-based systems.



- GUI, consists of a windowed graphical design with which the users can interact by clicking with the mouse and typing with the keyboard.
- Graphical interfaces are considered more user-friendly than the commando line ones. So, they are widely adopted in general-purpose operating systems.

Examples of GUI designs are the Microsoft Metro and the Linux GNOME.



Program Execution Service

- Another vital service of the operating system is program execution.
- To execute a program, operating systems tackle several operations, such as program loading, management of execution stacks, and process scheduling.
- The following image sketches the program execution service of operating systems
- This service allows different users to use the same operating system to deal with distinct tasks through running **heterogeneous programs.**

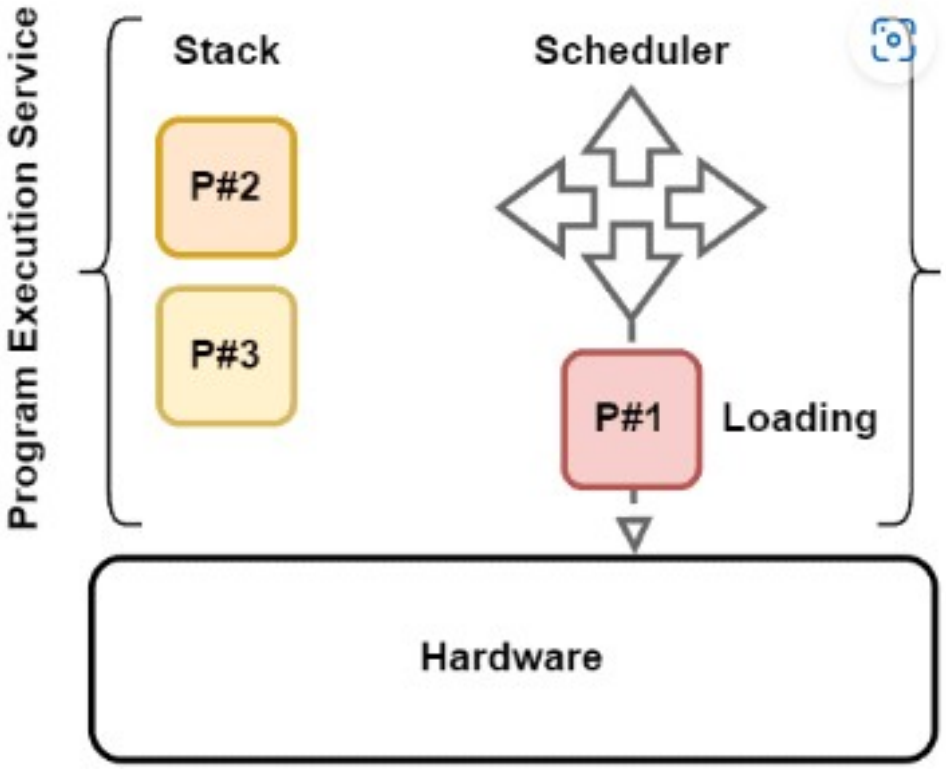


Fig 2. Program execution service in OS

Input/Output Operations Service

- A user can not directly control Input/Output (I/O) devices, such as monitors, speakers, keyboards, and mouses.
- So, operating systems mediate the communication between the user, I/O devices, and the computer system.
- The operating system provides a I/O service with several system calls and interrupts that can both send and receive operation requests to the available I/O devices.
- It is important that controlling the I/O operations of a specific device may require the installation of drivers. Drivers, in turn, supply a kind of guideline to the operating system on how to communicate with particular I/O devices.

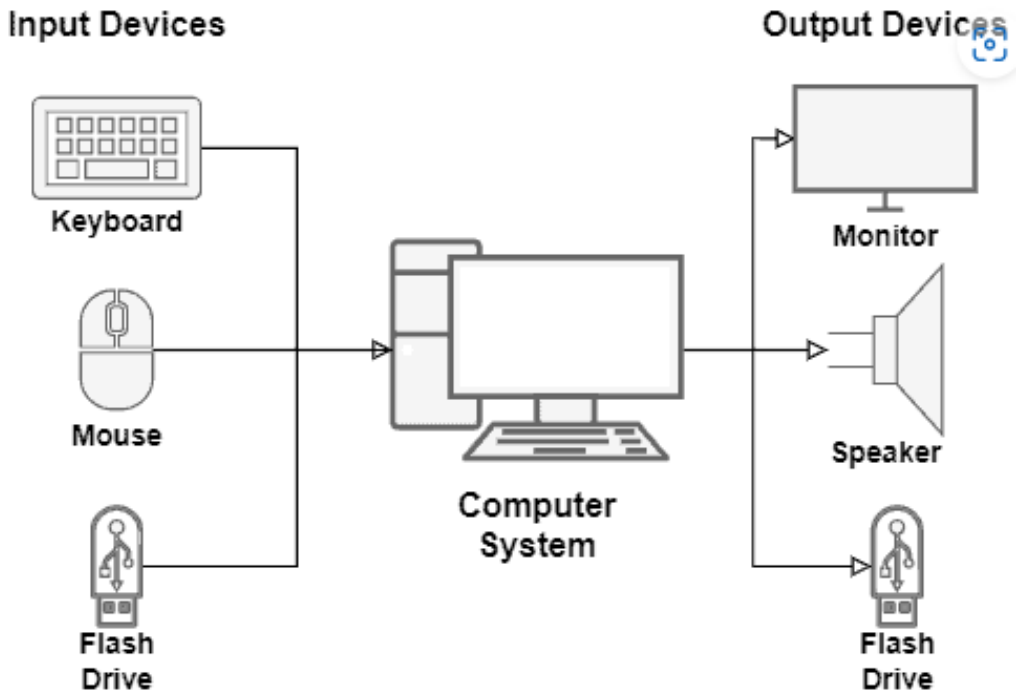


Fig 3. I/O Operation service in OS

File System Management Service

- The file system keeps and organizes all the files of a computer in persistent memory, usually a Hard Disk (HD) or Solid-State Drive (SSD).
- However, it is normal for the user to create, modify, delete, and search data files, Directory in the file system. Thus, the operating system provides a service for the users to manipulate the file system, enabling them to execute the cited management operations.
- Furthermore, the operation systems control the permissions given to users and programs to access files. These permissions (or restrictions) are applied to avoid unauthorized modifications and reads of crucial files, which can damage the computer system.

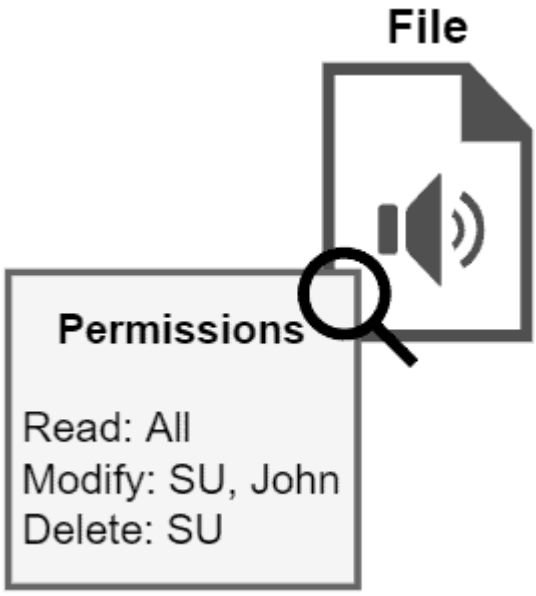


Fig 4. Permission on a File in OS

Communications Service

- In OS, communication represents the data exchange between processes.
- These processes, in turn, may execute on the same computer or different computers. So, in different computers case, different computers communicate with each other through a network.
- The operating system has specific mechanisms to enable processes running in the same computer to communicate. The most common examples are **pipes and shared memory**.

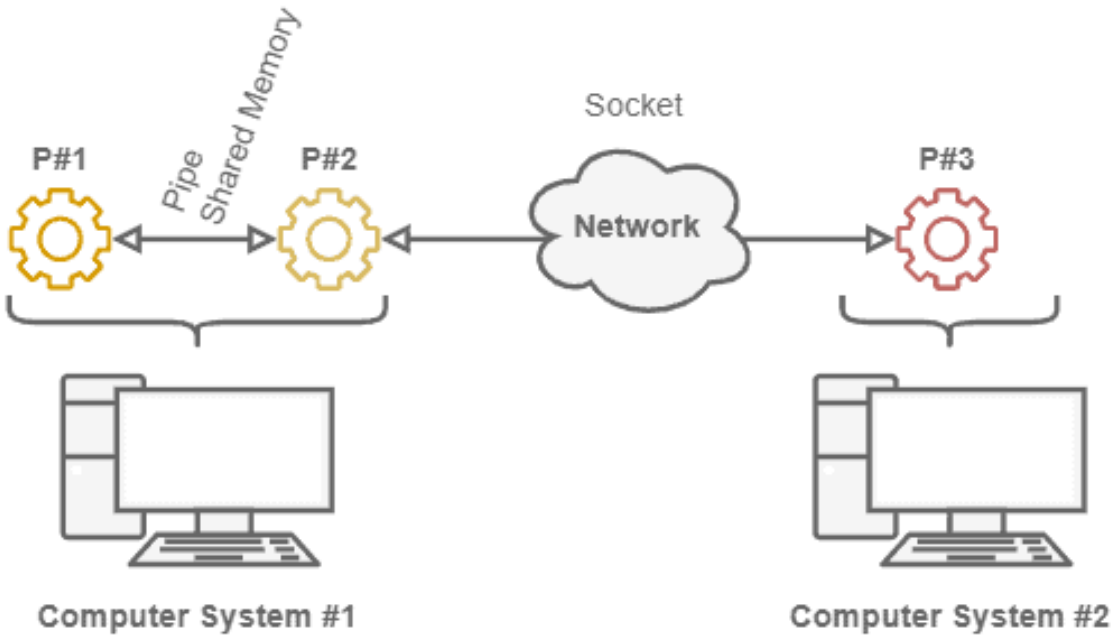


Fig 5. Communication between Different systems

Pipes

It consist of data buffers that connect two processes. In technical terms, the operating system associates file descriptors to each pipe: one for reading and another for writing it.

Shared Memory Regions

The other option is using memory regions shared between two or more processes. In this case, the operating system defines the shared memory regions. The addresses of these regions are included in the addressing spaces of the processes, which can exchange data through them.

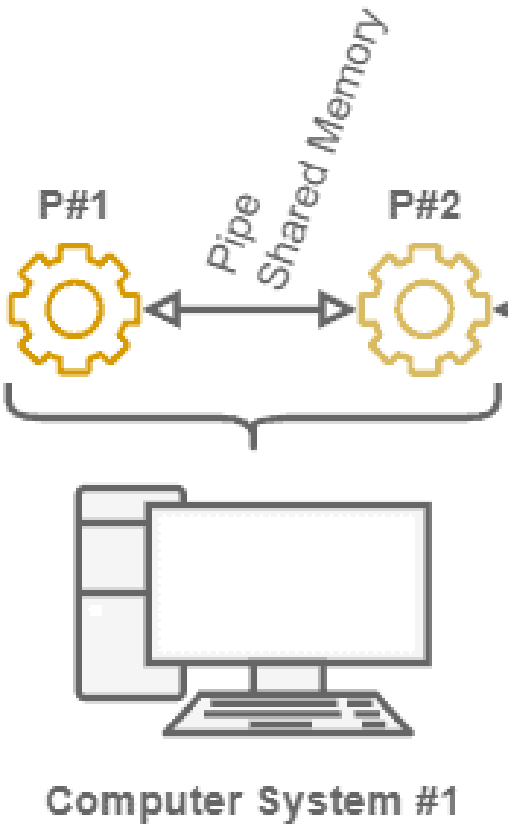
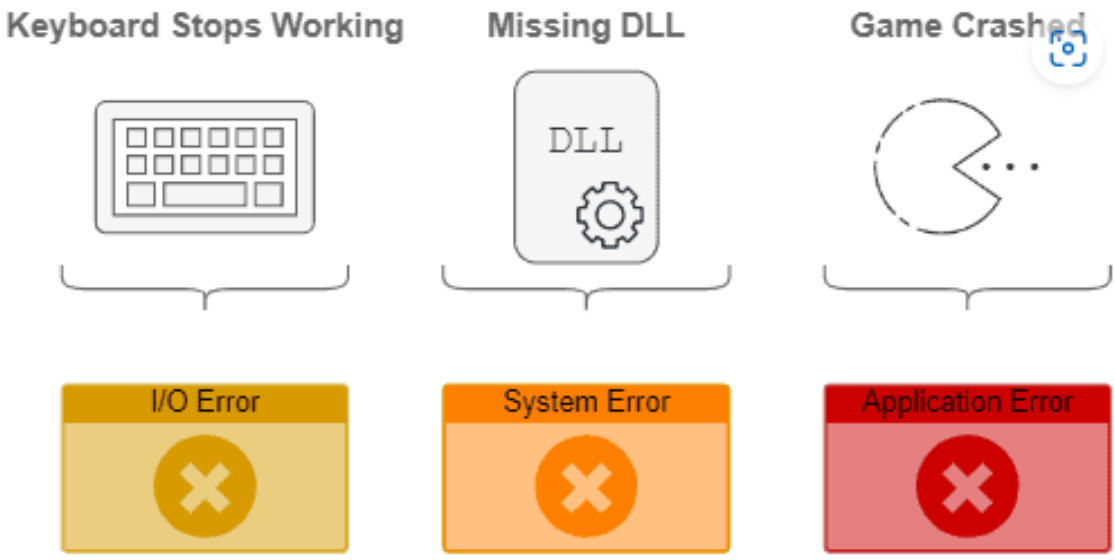


Fig 6. Pipe/ Shared Memory between Processes

Error Detection Service

During a computer system lifecycle, several errors can occur. For example, we can have CPU errors, memory bad allocations or accesses, a component failure causing a hardware error, or even an error caused by an I/O device**.**

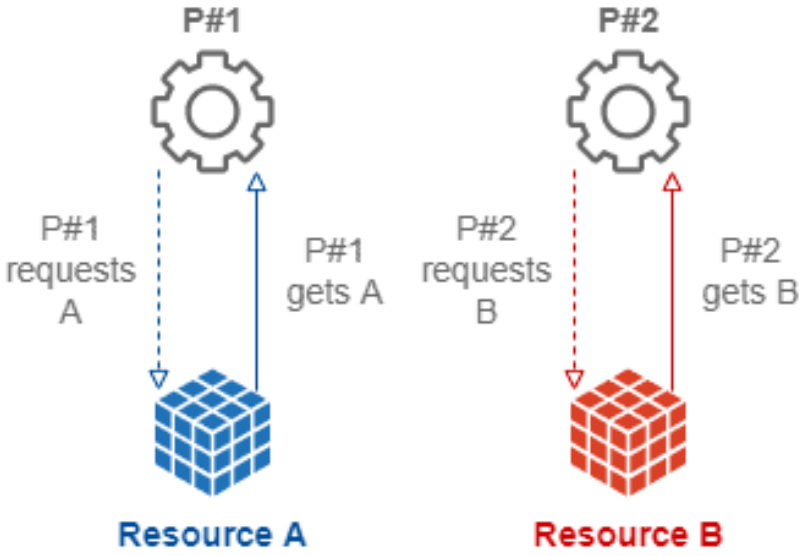
- **The error detection service of operating systems must avoid a computer system completely breaking down when an error happens**.
- So, this service must catch errors and manage them, keeping the entire system as functional as possible**.**
- Furthermore, the error detection service must be able to inform the user about the errors, showing their codes, descriptions, and, if known, manners to fix them.
- The image next presents error situations that may occur and be handled by the operating system



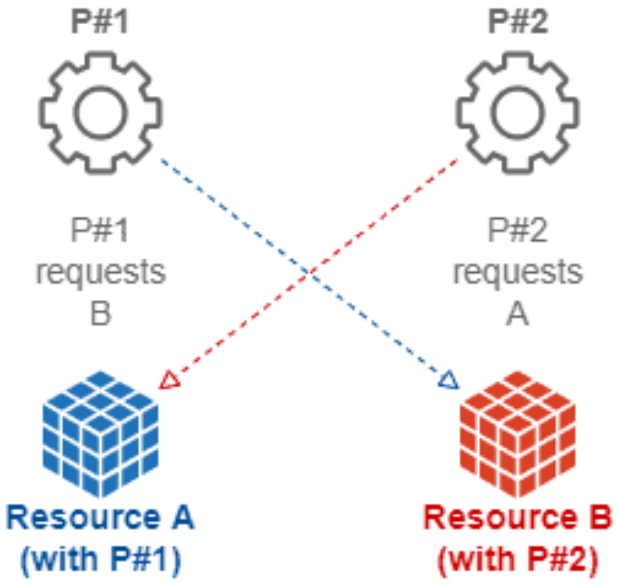
Resource Allocation Service

Resource allocation means dedicating computing resources to processes and users. We have many resource types, such as CPU (actually, CPU time), memory, networking, and I/O devices. So, as the operating system controls these resources, it naturally decides which processes use them.

- During the lifecycle, a process will typically require multiple computing resources.
- There are two main resource management challenges:
    - A process demanding a particular resource never gets it;
    - A process with a resource never releases it.



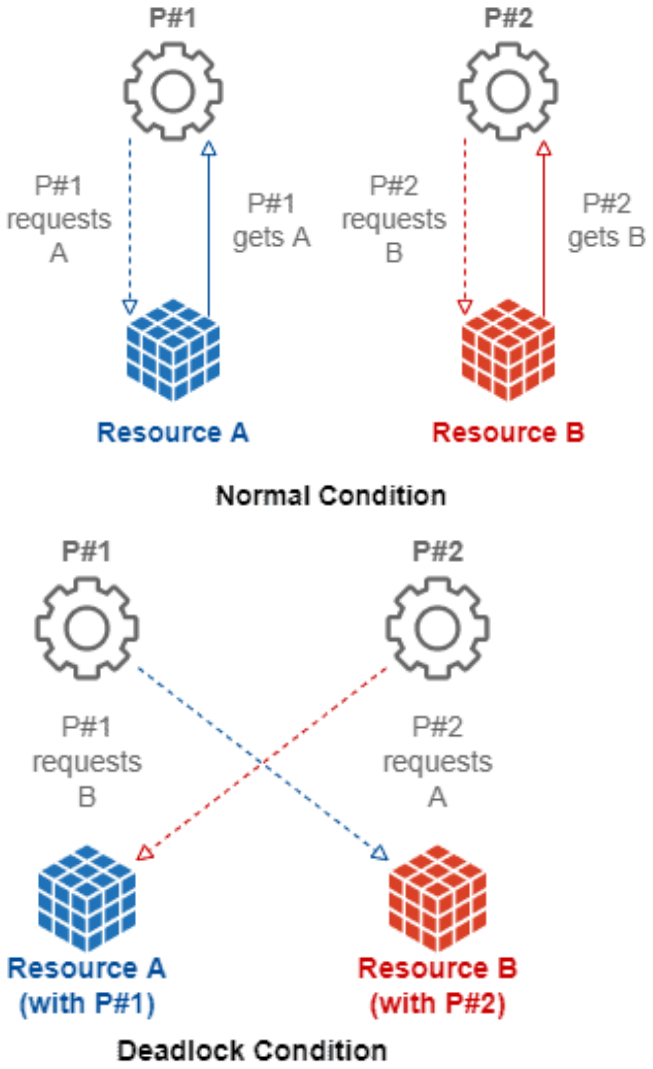
Normal Condition



Deadlock Condition

In normal condition, processes request or their required resources and get it. Once computation is completed, they release the resources. Thus, resource becomes available for next process.

Two processes hold particular resources required by both. So, they indefinitely wait for each other to release these resources. We call this condition a deadlock.



Accounting Service

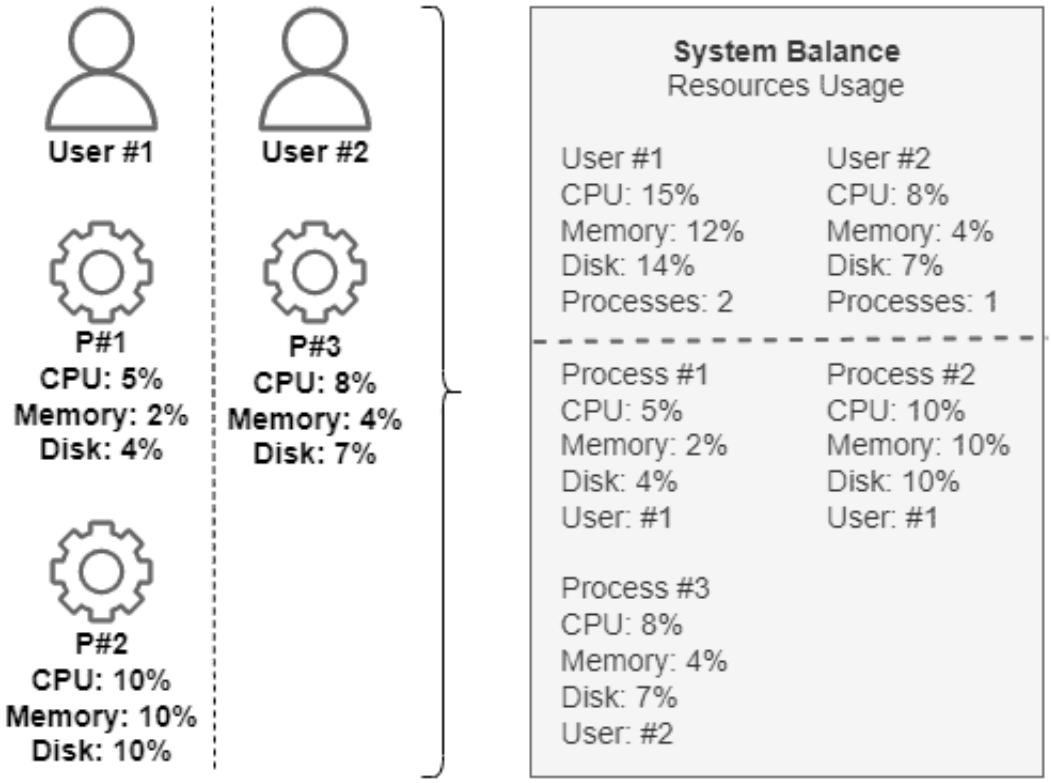


Fig 10. Accounting service in OS

- Accounting consists of keeping track of the behavior of both users and processes in the computing system.
- For instance, the operating system analyzes which users require the execution of which processes. So, it can also investigate how many computing resources are requested by the processes executing on the computer.
- Finally, the operating system can correlate the obtained information and generate statistics about the users of the computer system and the processes running on it.

Security Service

Regarding the operating system, we can understand security in different aspects.

- The first aspect consists of internal security for the processes. The operating system must be able to guarantee the correct execution of processes.
- Another relevant aspect of security is guaranteeing that only authorized users trigger the creation, modification, and remotion of resources and processes in the computer system.
- So, the operating system must provide authentication methods to the users to prove they are who they claim to be. The most usual authentication method employed by operating systems **is single-factor password-based authentication.**

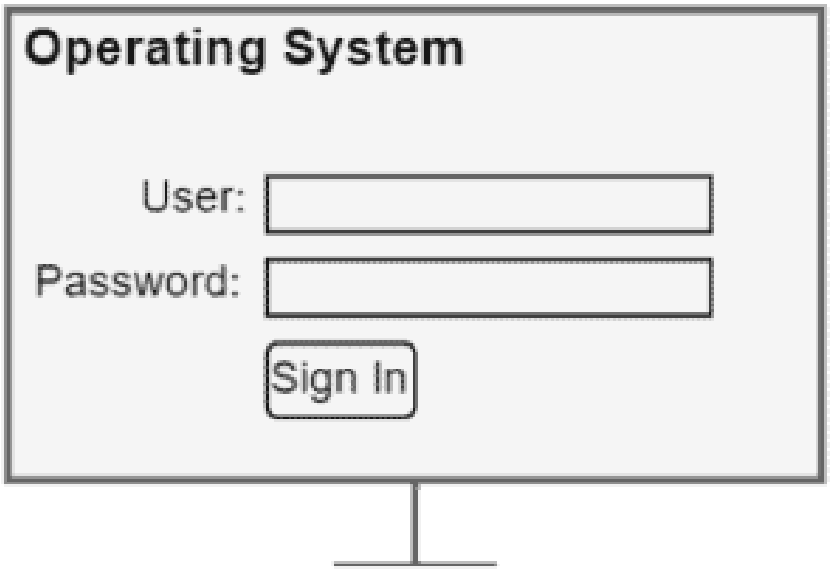


Fig 11. Security service in OS

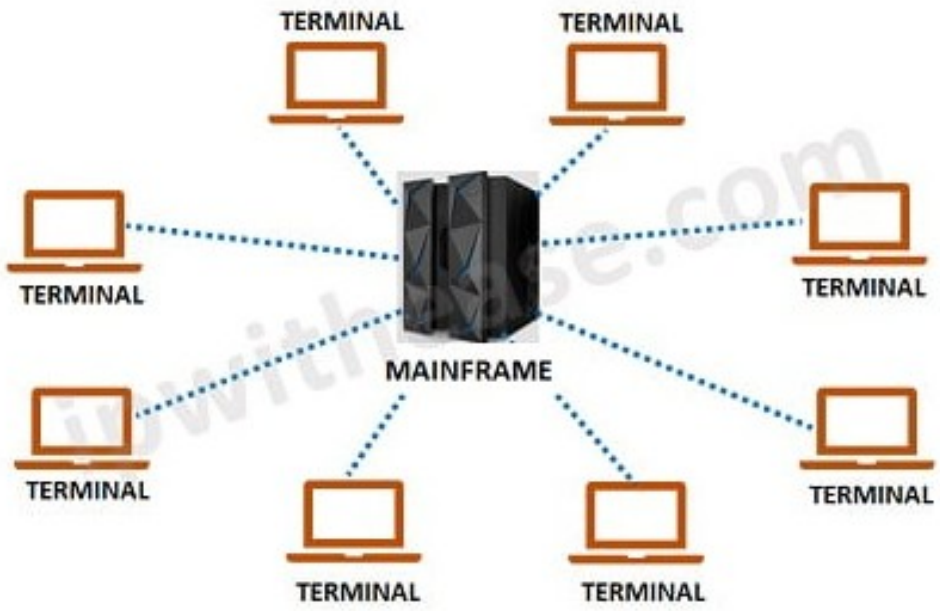
Operating System: Computing Environments

- **Mainframe Computing Environment**: A large and powerful computer system used for critical applications and large-scale data processing.

Example: IBM Z Series mainframes used by large financial institutions

Components (Hardware)

- **Mainframe Computer:** Centralized, high-performance computers designed for large-scale processing.
- **Storage Systems:** High-capacity storage units (e.g., disk arrays, tape drives).
- **Input/Output Devices:** High-speed printers, terminals.



-
- **Networking Components:** High-bandwidth network interfaces to connect with various systems and devices.

Components (Software)

- **Operating System:** Specialized mainframe OS such as z/OS, z/VM, or Linux on Z.
- **Middleware:** Software to manage communication and data exchange between applications (e.g., CICS for transaction processing, DB2 for database management).
- **Applications:** Business-critical applications for finance, insurance, retail, and other industries.

Utility Areas: Enterprise Resource Planning (ERP), Data Analysis and Reporting, ATM and Online Banking, Historical Data Storage

Example: A company's internal email system

Components

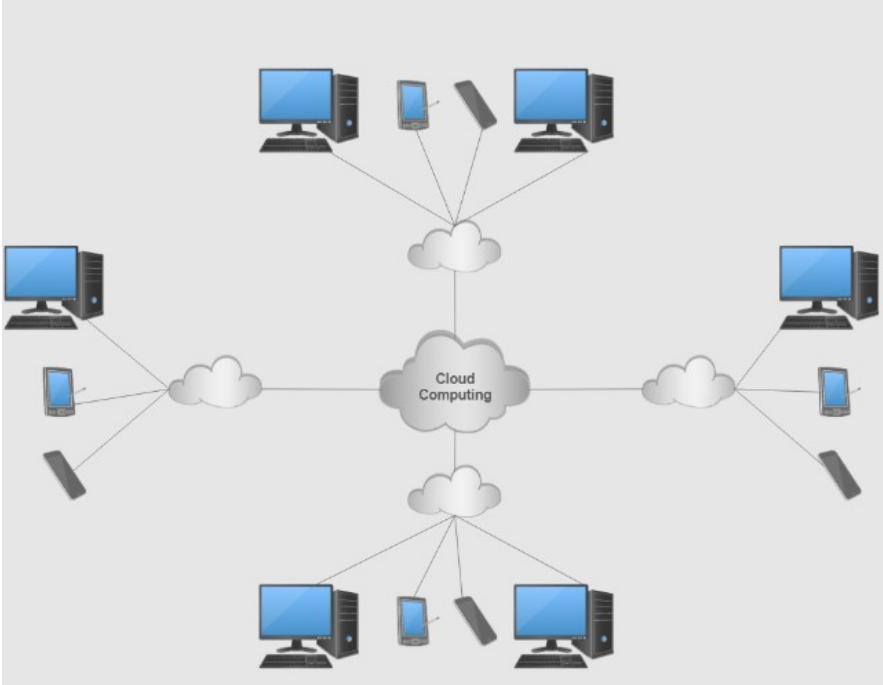
- **Hardware:** Servers (email servers, database servers), client machines (desktops, laptops)
- **Software:** Server OS (Windows Server, Linux), client applications (Outlook, web browsers)

Utility Areas

- **Enterprise Applications:** Email services, database management, internal websites.
- **Resource Sharing:** Centralized storage, shared printers and devices.
- **Security and Management:** Centralized authentication, access controls, and monitoring.

Cloud Computing Environment:

- **Cloud Computing Environment** : A computing environment in which resources and services are provided over the Internet and accessed through a web browser or client software.



Example: Google Cloud Platform (GCP), Amazon Web Services

(AWS), Microsoft Azure

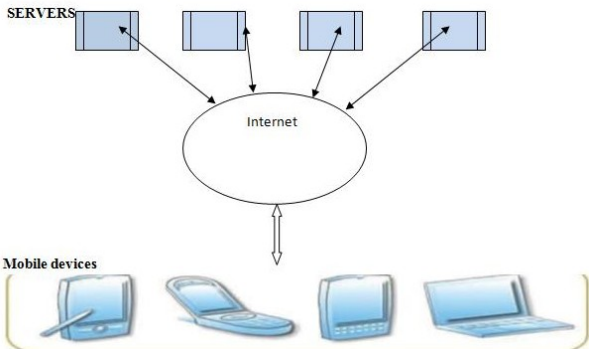
Components

- **Hardware**: Data centers with thousands of servers
- **Software**: Virtualization software, cloud management platforms, APIs.

Utility Areas

- **Scalability**: Dynamic allocation of resources for web applications, databases.
- **Cost Efficiency**: Pay-as-you-go model for resources.
- **Disaster Recovery:** Backup and restore capabilities across multiple data centers

Mobile Computing Environment :



- **Mobile Computing Environment** : A computing environment in which users access information and applications using handheld devices such as smartphones and tablets.

Example: Smartphones, Tablets

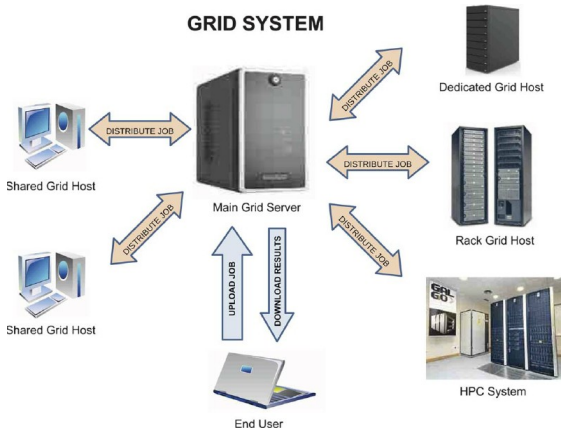
Components

- **Hardware:** Mobile processors, touchscreens, sensors (GPS, accelerometer).
- **Software:** Mobile OS (iOS, Android), mobile applications (apps)

Utility Areas

- **Communication:** Messaging, video calls.
- **Navigation**: GPS, maps.
- **Entertainment**: Games, media consumption

Grid Computing Environment :



- **Grid Computing Environment** : A computing environment in which resources and services are shared across multiple computers to perform large-scale computations.

Example: The Large Hadron Collider (LHC) computing grid used

by CERN for particle physics research.

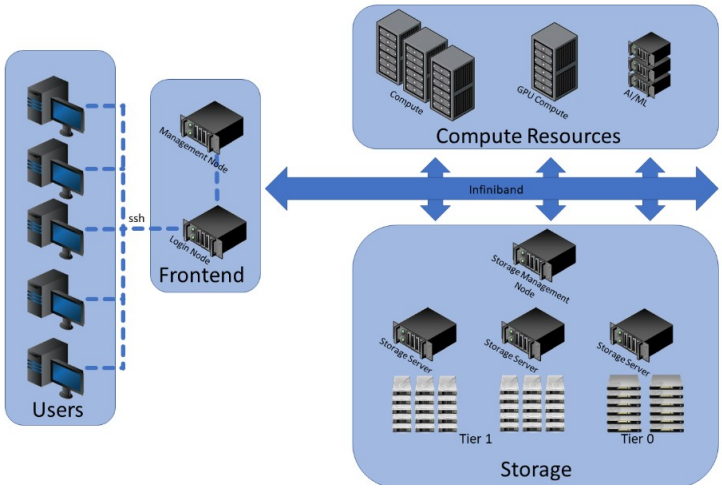
Components

- **Hardware:** Multiple interconnected computers, Distributed storage infrastructure, High-speed networks and interconnects
- **Software**: OS (Linux), Application Software (task specific), security tools

Utility Areas

- **Scientific Research:** Physics, Biology.
- **Data-Intensive Applications**: Climate Modeling, earthquack simulation
- **Healthcare**: Medical Research, Personalized medicine

High-Performance Computing (HPC) Environment:



- **High-Performance Computing (HPC) Environment**: A computing environment in which resources and services are shared across multiple computers to perform large-scale computations.

Example: Supercomputers like Summit, used by the Oak

Ridge National Laboratory

Components

- **Hardware:** Thousands of CPUs/GPUs, high-speed interconnects.
- **Software:** Specialized OS (Linux variants), parallel computing frameworks (MPI, CUDA)

Utility Areas

- **Scientific Simulations**: Climate modeling, astrophysics
- **Complex Calculations**: Cryptography, genomic research.
- **Data Analysis**: Real-time data processing, financial modeling.

Virtualization and Containerization

Virtualization is the process of creating a virtual version of a physical machine or resource, such as a computer, server, storage device, or network. It allows multiple virtual instances to run on a single physical system, enabling better resource utilization, flexibility, and isolation

In other words:

Virtualization in operating systems (OS) is a technology that creates a virtual version of a computing resource, such as hardware platforms, storage devices, or network resources. It enables the creation of multiple simulated environments or dedicated resources from a single physical hardware system.

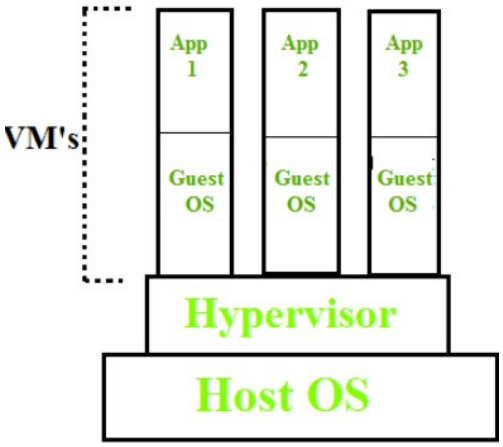
Virtualization: Key Concepts

1. Virtual Machines (VMs)

A virtual machine is an emulation of a computer system. Each VM runs its own operating system and applications, functioning as if it were a separate physical machine.

2. Hypervisor

The hypervisor, also known as a virtual machine monitor (VMM), is the software layer that enables virtualization. It manages the hardware resources and ensures that VMs are isolated from each other while sharing the same physical hardware.



There are two types of hypervisors:

Type 1 (Bare-metal): Runs directly on the physical hardware (e.g., **VMware ESXi**, **Microsoft Hyper-V**).

Type 2 (Hosted): Runs on top of a host operating system (e.g., **VMware Workstation**, **Oracle VM VirtualBox**).

Isolation

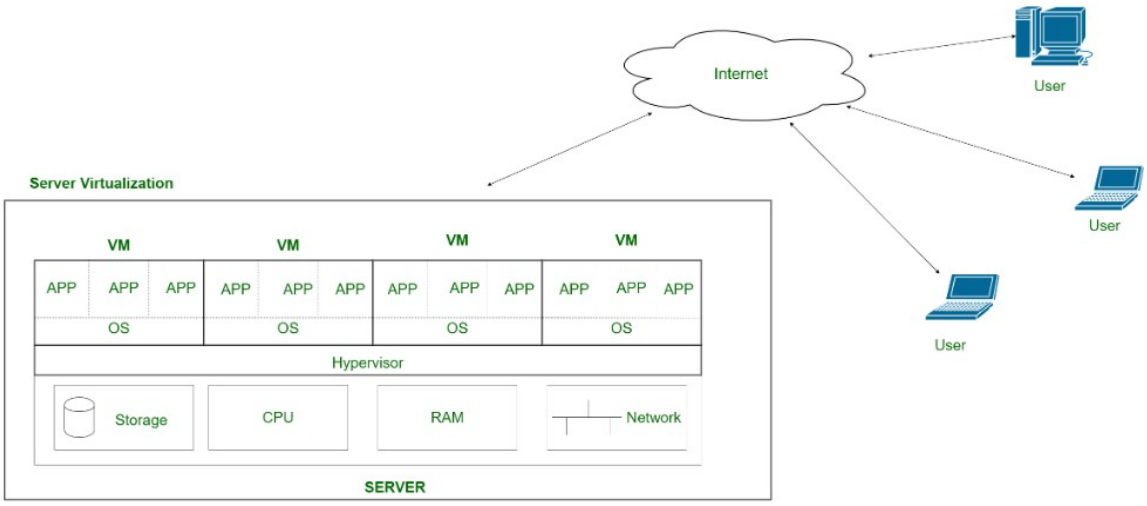
Virtualization provides strong isolation between VMs. Each VM operates independently and securely, ensuring that issues in one VM do not affect others.

Resource Management

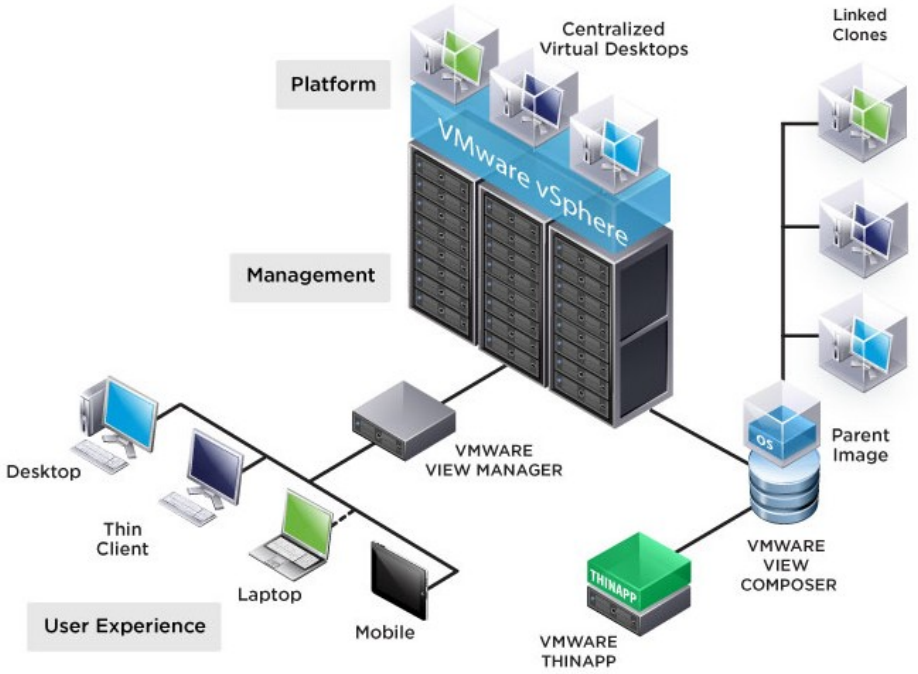
Virtualization allows for efficient resource allocation and management. Resources such as CPU, memory, and storage can be dynamically allocated to different VMs based on demand.

Virtualization Types

Server Virtualization: It Divides a physical server into multiple virtual servers, each running its own operating system and applications.



Desktop Virtualization: It Provides virtual desktops to end-users, which they can access from various devices. Examples include Virtual Desktop Infrastructure (VDI) solutions.



Storage Virtualization: It Abstracts physical storage resources to create a single storage pool that can be managed and allocated efficiently.

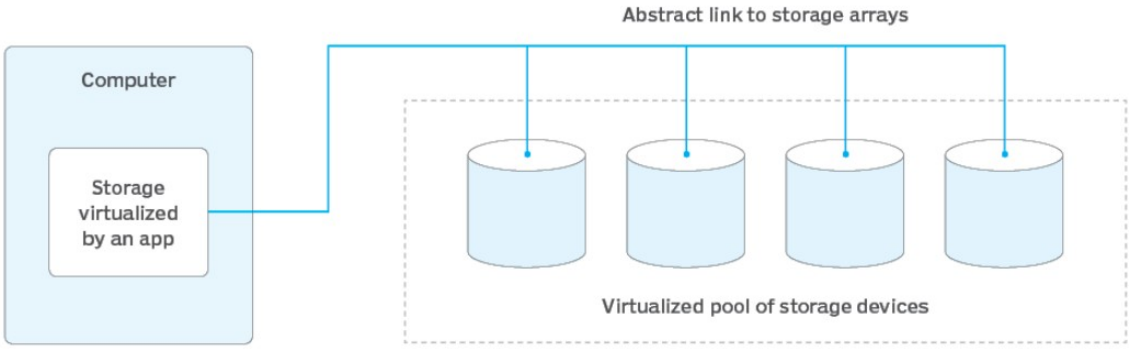
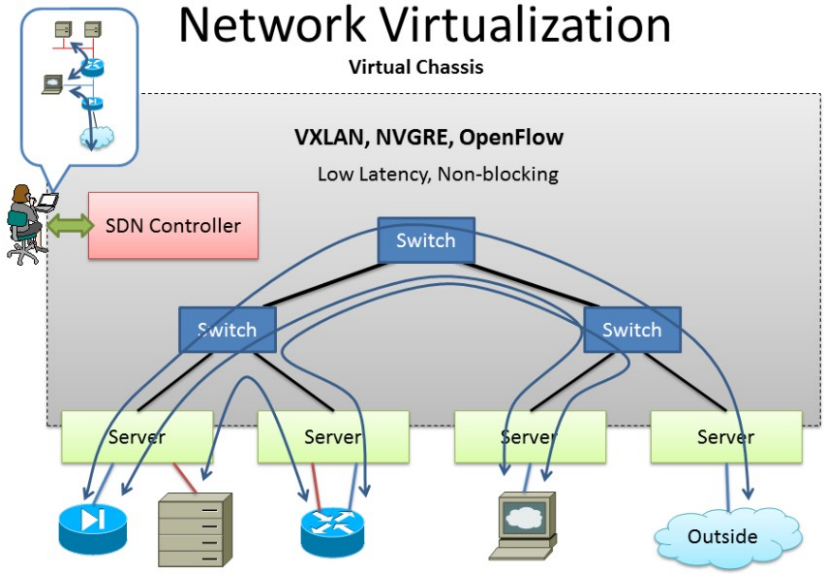


Fig 4. Storage Virtualization

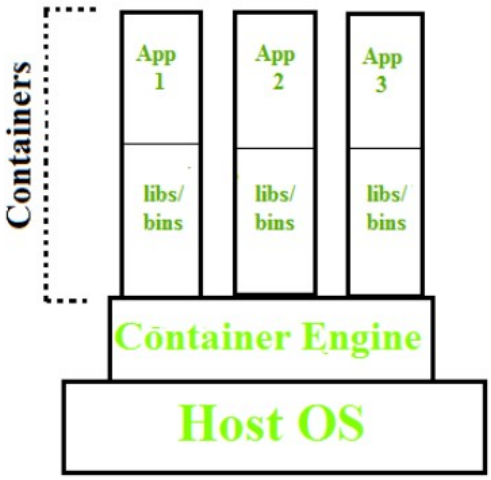
Network Virtualization: It Combines hardware and software network resources into a single, software-based administrative entity.



Containerization

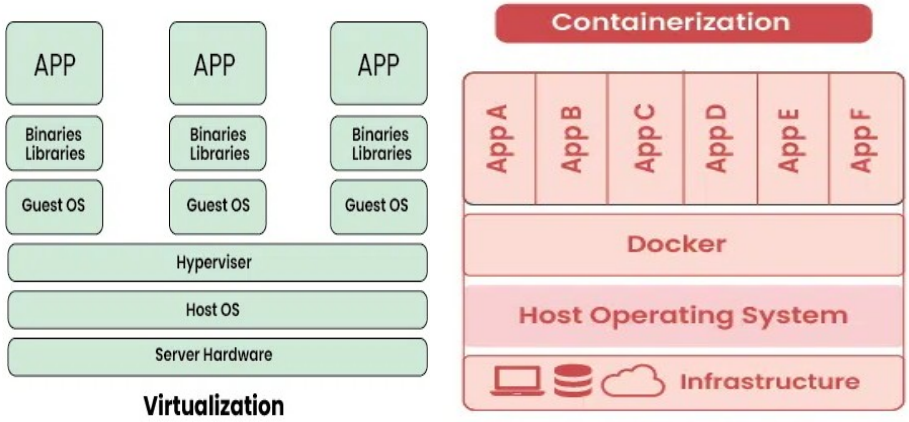
Containerization in operating systems (OS) involves encapsulating an application and its dependencies into a container, ensuring it runs consistently across different environments.

Containerization was introduced to address certain limitations and inefficiencies associated with traditional virtualization. While virtualization provided significant benefits over physical hardware by allowing multiple operating systems to run on a single physical machine, containerization introduced a more efficient and streamlined approach to application deployment and management



Containerization

Virtualization vs Containerization



Virtualization vs Containerization

Feature	Virtualization	Containerization
Isolation	Each VM runs its own OS, providing strong isolation from the host and other VMs.	Containers share the host OS kernel, providing lightweight isolation, but not as strong as VMs.
Resource Usage	VMs require a separate OS instance and consume more resources (CPU, memory, storage).	Containers share the host OS kernel, resulting in more efficient resource utilization.
Performance	Slightly slower performance due to overhead of hypervisor and separate OS instances.	Generally faster performance due to less overhead, as containers leverage the host OS directly.
Portability	VMs can be migrated between different hypervisors and platforms, but with some effort.	Containers are highly portable and can run consistently across different environments (development, testing, production).
Scalability	Scalability is limited by the available resources on the host machine.	Highly scalable as containers are lightweight and can be spun up or down quickly to match demand.
Overhead	Higher overhead due to running multiple OS instances.	Lower overhead as containers share the host OS kernel.
Examples	VMware, VirtualBox, Hyper-V.	Docker, Podman.

Best Scenarios for Virtualization and Containerization

Scenario	Containerization	Virtualization
Running legacy applications	✗	✓
Isolating applications from one another	✗	✓
Building microservices architecture	✓	✗
Rapid development and deployment (CI/CD)	✓	✗
Maximizing resource utilization	✓	✗
Lifting and shifting existing applications to the cloud	✗	✓

Fig 8. Scenarios for Containerization and Virtualization

Operating System Structures

- The operating system can be implemented with the help of various structures.
- The structure of the OS depends mainly on how the various standard components of the operating system are interconnected and melded into the kernel.
- A design known as an operating system enables user application programs to communicate with the machine's hardware.

Types of structures in Operating systems

- Simple/Monolithic Structure
- Micro-Kernel Structure
- Hybrid-Kernel Structure
- Exo-Kernel Structure
- Layered Structure
- Client Server Model
- Virtual Machines

Operating System Structures: Types

Client-Server Model

An operating system structure where services are divided into client and server components, with servers providing specific functionalities to client applications upon request.

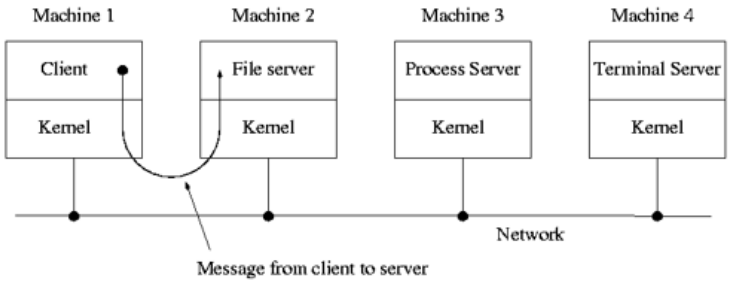


Fig 14. Client-Server Structure

Operating System Structures: Types

Layered Structure

An operating system structure organized into layers, with each layer providing a set of services to the layer above, facilitating modular design and ease of maintenance.

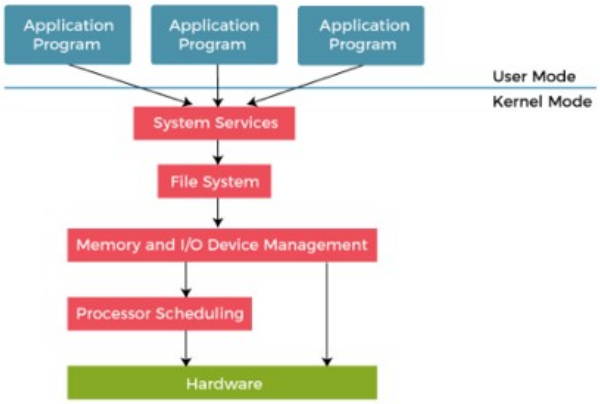


Fig 13. Layered Structure

Operating System Structures: Types

Exo-Kernel Structure

An operating system architecture that exposes hardware resources directly to applications, allowing greater flexibility but requiring applications to manage resources explicitly.

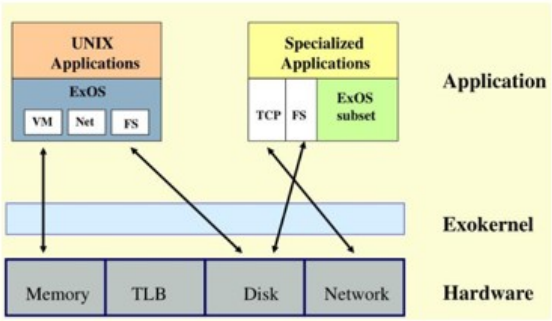


Fig 12. Exo-Kernal Structure

Operating System Structures: Types

Hybrid-Kernel Structure

An operating system architecture combining aspects of both monolithic and microkernel designs, offering a balance between performance and flexibility.

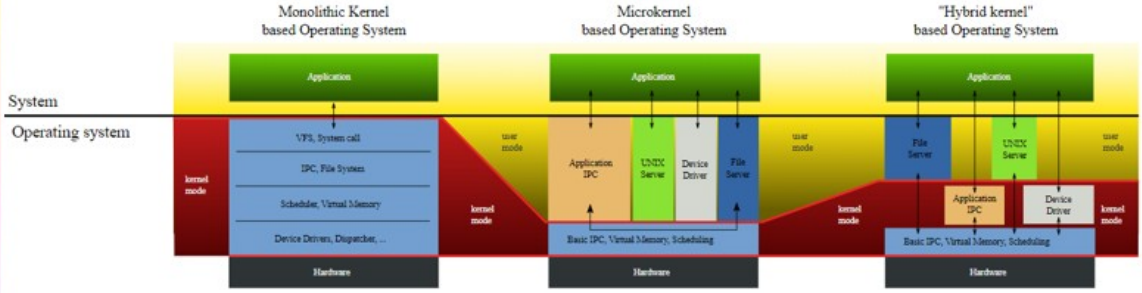


Fig 11. Hybrid-Kernal Structure

Operating System Structures: Types

Micro-Kernel Structure

A minimalistic operating system architecture where the kernel provides only essential services, and additional functionalities are implemented as user-space processes.

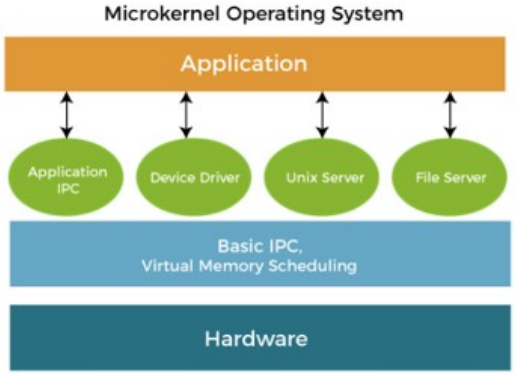


Fig 10. Micro-Kernal Structure

Operating System Structures: Types

Simple/Monolithic Structure

A unified operating system architecture where the kernel provides all essential services directly to user applications

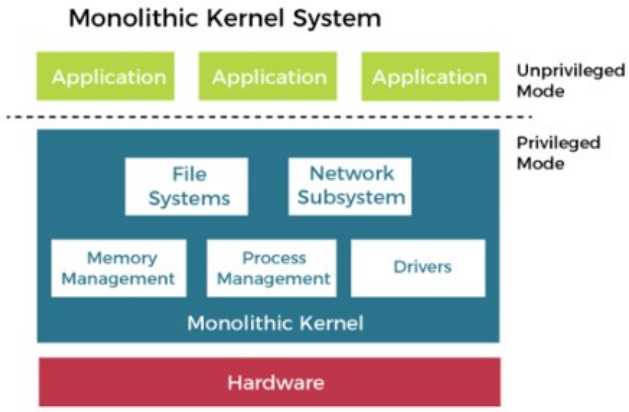


Fig 9. Simple/Monolithic OS Structure

Operating System Structures: Types

Virtual Machines

A layer of software that enables multiple operating systems to run concurrently on the same hardware by virtualizing resources and providing isolation between guest operating systems.

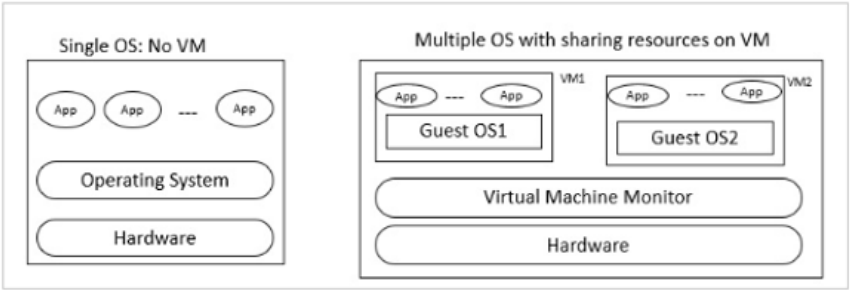


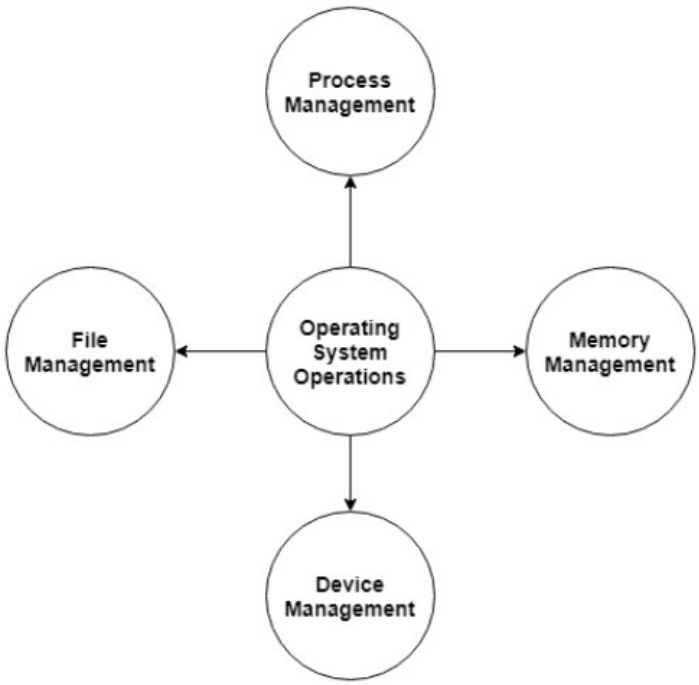
Fig 15. Virtual Machine Structure

Operating System Operations

An operating system allows the user application programs to interact with the system hardware. Operating system by itself does not provide any function but it provides an atmosphere in which different applications and programs can do useful work.

The major operations of the operating system are:

- Process management
- Memory management
- Device management
- File management.



Process Management

The operating system is responsible for managing the processes i.e assigning the processor to a process at a time. This is also known as process scheduling.

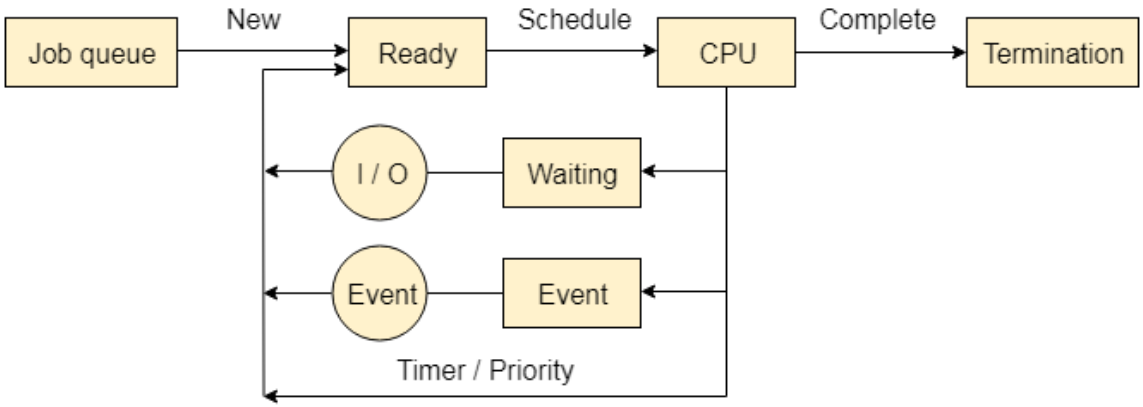
The different algorithms used for process

scheduling are:

- FCFS (first come first served),
- SJF (shortest job first),
- priority scheduling,

- round robin scheduling etc.

- There are many scheduling queues that are used to handle processes in process management.
- When the processes enter the system, they are put into the job queue.
- The processes that are ready to execute in the main memory are kept in the ready queue. The processes that are waiting for the I/O device are kept in the device queue.



Memory Management

Memory management plays an important part in operating system. It deals with memory and the moving of processes from disk to primary memory for execution and back again.

The activities performed by the operating system for memory management are

- The operating system assigns memory to the processes as required. This can be done using best fit, first fit and worst fit algorithms.
- All the memory is tracked by the operating system i.e. it nodes what memory parts are in use by the processes, and which are empty.
- The operating system deallocated memory from processes as required. This may happen when a process has been terminated or if it no longer needs the memory.

Device Management

There are many I/O devices handled by the operating system such as

- mouse,
- keyboard,

- disk drive etc.

- There are different device drivers that can be connected to the operating system to handle a specific device.
- The device controller is an interface between the device and the device driver.
- The user applications can access all the I/O devices using the device drivers, which are device specific codes.

File Management

Files are used to provide a uniform view of data storage by the operating system. All the files are mapped onto physical devices that are usually non-volatile so data is safe in the case of system failure.

The files can be accessed by the system in two ways

- Sequential access
- Direct access
- **Sequential Access** The information in a file is processed in order using sequential access. The files records are accessed one after another. Most of the file systems such as editors, compilers etc. use sequential access.
- **Direct Access** In direct access or relative access, the files can be accessed in random for read and write operations. The direct access model is based on the disk model of a file, since it allows random accesses.

System Booting

Booting is the process of starting a computer. It can be initiated by hardware such as a button press or by a software command.

After it is switched on, a **CPU has no software in its main memory, so some processes must load software into memory before execution**. This may be done by hardware or firmware in the CPU or by a separate processor in the computer system.

Booting Operation Sequence:

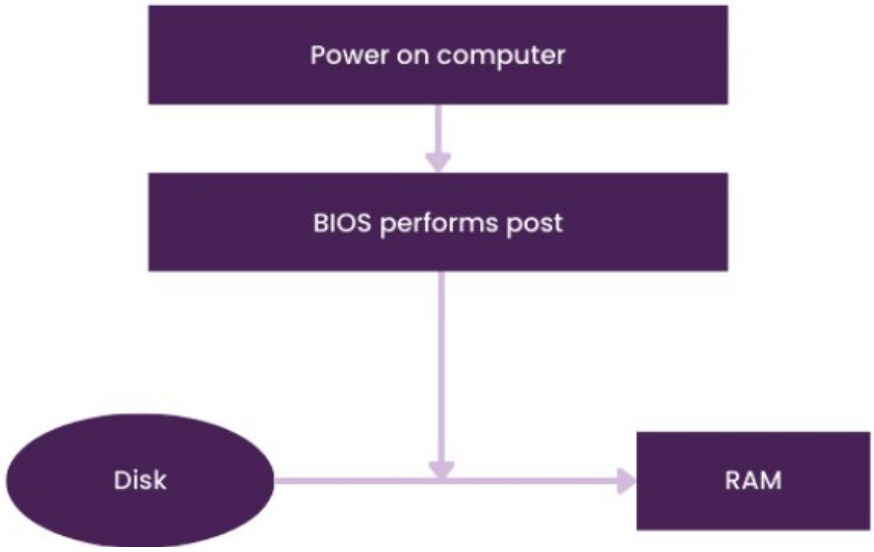
- Booting is a start-up sequence that starts the operating system of a computer when it is turned on.
- A boot sequence is the initial set of operations that the computer performs when it is switched on. Every computer has a boot sequence.

Fig 1. Booting Operation Sequence

System Booting

When a computer or any other computing device is in a powerless state, its operating system remains stored in secondary storage like a hard disk or SSD. But, when the computer is started, the operating system must be present in the main memory or RAM of the system.

When a computer system is started, there is a mechanism in the system that loads the operating system from the secondary storage into the main memory, or RAM, of the system. This is called the booting process of the system.



System Booting: Process

The following components are involved during booting process

1. Power Supply
2. CPU
3. BIOS (ROM)
4. CMOS
5. Master Boot Record
6. RAM
7. Boot Loader
8. Storage Drives

The booting process is as follows:

1. Power Supply

- The first step to switch on the power supply.
- This power supply provides electricity to all motherboard components like CPU, Hard disk etc.

2. CPU

- CPU loads the BIOS (Basic Input output System) to execute instructions stored in it.

3. BIOS

- When we turn on the computer, so the CPU looks for another program, called the BIOS (Basic Input/Output System), and runs it. The BIOS is a firmware that is located on the motherboard and is run by the CPU to start the booting sequence

3. BIOS Initialization, RUN Test and Initialization Hardware

- BIOS settings are stored in a non-volatile memory called, CMOS (Complementary metal oxide semiconductor).
- BIOS loads the setting from CMOS

- CMOS is backed by CMOS battery to provide it continuous

power supply even after system is shutdown or restart.

- Now BIOS program is loaded with setting received from CMOS
- BIOS program initialized POST test, called Power on Self-Test.
- POST contains some test cases for each hardware to ensure whether each

Connected device working properly or not.

    - Printer, Keyboard, Mouse, Speaker, RAM etc.

4. BIOS Sends Control to Boot Device

- Once the BIOS finished all initialization and hardware test. It will find the Boot Device to load operating system
- Now it looks for the actual program that will search and run the operating system
- Boot device contains the instruction for this actual program called boot loader.
- The boot device is the device from which the operating system is loaded. A modern **PC BIOS** (Basic Input/Output System) supports booting from various devices.
- These devices includes
    - Local hard disk drive,
    - Optical drive,
    - Floppy drive,
    - A network interface card,
    - A USB device.
- The BIOS will allow the user to configure a boot order. If the boot order is set to:
    - CD Drive
    - Hard Disk Drive
    - Network

Boot Loader

- Boot Loader is a software program that is responsible for “actually loading” the operating system. The Boot Loader is typically a part of the Operating System itself.
- Till the point Boot Loader starts loading the OS, there is nothing in the Main Memory of the machine.
- After POST test, BIOS search for a place where bootloader is located.
- Boot loader is located at MBR (Master Boot Record) which further exists at 0^{th} index of boot disk
- BIOS instructs CPU to run bootloader in main memory (RAM) to start running OS

5. Boot loader initialize the OS

- Control is transferred from BIOS to CPU and CPU to boot loader
- Boot loader responsible to initialize the OS into main memory
- Boot loader is not a single program for all kind of OS
- Each operating system has its own instruction for initialization.
- Each OS manufacturer write its own program for bootloader
    - Microsoft has written bootmgr.exe bootloader to initialize Windows
    - Apple has written boot.efi bootloader to initialize Mac:
    - Linux has written Grub bootloader to initialize its OS
- The whole control is managed by bootloader
- Bootloader performs all the checks to initialize OS

Boot Types

Cold Booting

- When the computer starts for the first time or is in a shut-down state and switch on the power button to start the system

- This type of process to start the computer is called cold booting.
- During cold booting, the system will read all the instructions from the ROM (BIOS) and the Operating System will be automatically get loaded into the system.
- This booting takes more time than Hot or Warm Booting.

Warm Booting

- When computer systems come to no response or hang state, and then the system is allowed to restart during on condition.
- It is referred to as rebooting (Warm Booting).
- There are many reasons for this state, and the only solution is to reboot the computer. Rebooting may be required when we install new software or hardware or to set software or hardware configuration changes
- sometimes systems may behave abnormally or may not respond properly.
- In such a case, the system has to be a force restart.
- Most commonly Ctrl+Alt+Del button is used to reboot the system.

System Booting: Dual Booting

Dual booting is a configuration where two operating systems are installed on a single computer, allowing the user to select which OS to boot into at startup. This setup is useful for users who need to use features or applications specific to different operating systems.

Key Concepts

Boot Loader

- A boot loader is a small program that manages the boot process of a computer.
- It allows the user to choose between multiple operating systems at startup.
- Common boot loaders include GRUB (GNU GRand Unified Bootloader), LILO (Linux Loader), and Windows Boot Manager.

Partitioning

- Hard disk partitioning is crucial for dual booting. Each OS needs its own partition, and sometimes additional partitions are required for data or swap space.
- A typical dual-boot setup might include partitions for each OS and possibly a shared data partition accessible by both.

Step-by-Step Process

Backup Data

- Before making any changes to the disk, back up important data to prevent data loss.

Partition the Disk

- Use a partitioning tool (e.g., GParted, Disk Management in Windows) to create separate partitions for each operating system.
- Ensure there is enough space for each OS and its applications.

Install the First Operating System

- Install one of the operating systems first, if it's not already installed.
- During installation, allocate the appropriate partition for this OS.

Install the Second Operating System

- Boot from the installation media of the second OS.
- During installation, select the partition designated for this OS. Be careful not to overwrite the existing OS.
- The installer of the second OS usually detects the presence of the first OS and configures the boot loader to allow choosing between the two.

Configure the Boot Loader

- If using GRUB, it will usually detect both operating systems and present a menu at startup.
- For Windows Boot Manager, you might need to add the second OS manually using tools like EasyBCD.

Post-Installation

- Update the OSes and drivers.
- Set up shared partitions or directories if needed.
- Customize the boot loader menu (e.g., setting default OS, timeout duration).

  Example: Dual Booting Windows and Linux

1. **Install Windows**

    - Install Windows first as it tends to overwrite existing boot loaders without asking.
    - During installation, create a partition for Windows, leaving space for Linux.

    2. Partitioning for Linux

    - After installing Windows, boot from a Linux live CD/USB.
    - Use a partitioning tool to create partitions for Linux (e.g., root /, home /home, swap).

    3. Install Linux

    - Install Linux on the newly created partitions.
    - The Linux installer will usually detect the Windows installation and add it to the GRUB boot menu.

    4. Boot Loader Configuration

    - GRUB will be installed and set as the default boot loader.
    - After installation, you can edit the GRUB configuration to set the default OS and timeout.

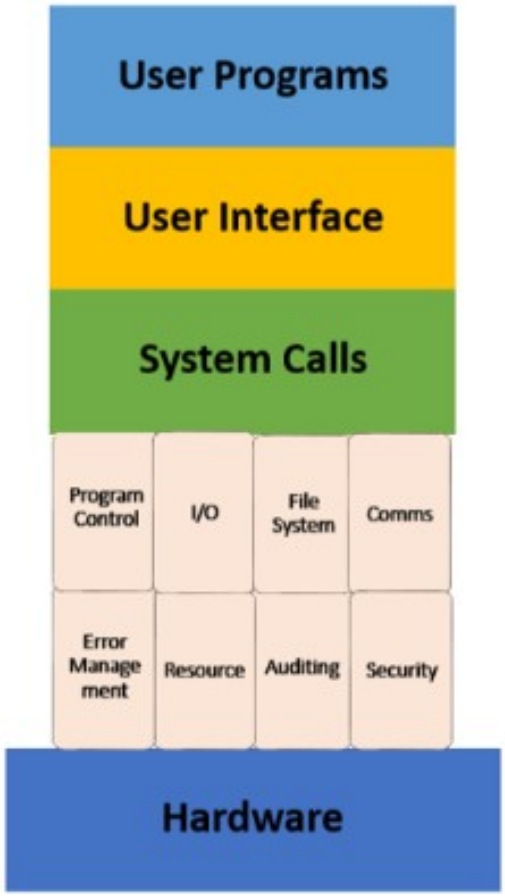
system call

A system call is a mechanism that provides the interface between a process and the operating system. It is a programmatic method in which a computer program requests a service from the kernel of the OS.

In other words

A system call is a procedure, written in c, c++ or assembly level languages for performing various operations. A system call is a method of interacting with the operating system via programs. A system call is a request from computer software to an operating system's kernel.

System call offers the services of the operating system to the user programs via API (Application Programming Interface). System calls are the only entry points for the kernel system.



- A system call is a programmatic way in which a computer program requests a service from the kernel of the operating system it is executed on.
- It is a Programming interface to the services provided by the OS.
- It is typically written in a high-level language (C or C++)
- System call provides the services of the operating system to the user programs via Application Program Interface(API).
- It provides an interface between a process and an operating system to allow user-level processes to request services of the operating system.
- System calls are the only entry points into the kernel system. All programs needing resources must use system calls.

- 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 mode to Kernal 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

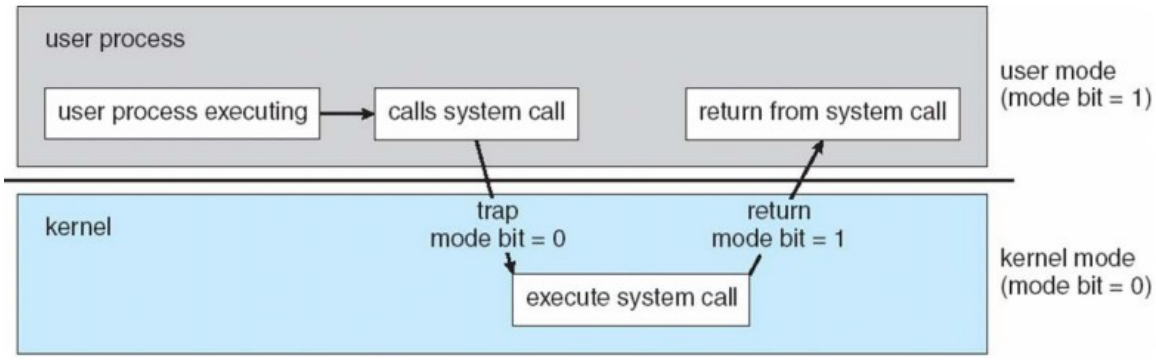


Fig 7. Transition from User mode to Kernal mode