# UNIT-1

# Operating Systems Overview & System Structures

## Operating System:

- An Operating System (OS) can be defined as an interface between a computer user and computer hardware.
- An operating system is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.
- It is responsible for the execution of all the processes, Resource Allocation, CPU management, File Management etc and many ither tasks.
- An operating system (OS) is system software that manages computer hardware, software resources, and provides common services for computer programs.
- The purpose of an operating system is to provide an environment in which a user can execute programs in convenient and efficient manner.
- Many popular Operating systems are Windows , Linux, Mac OS by Apple etc.
- OS is a program that acts as an intermediary between a user of a computer and the computer hardware
- Operating system goals:
    - Execute user programs and make solving user problems easier
    - Make the computer system convenient to use
    - Use the computer hardware in an efficient manner

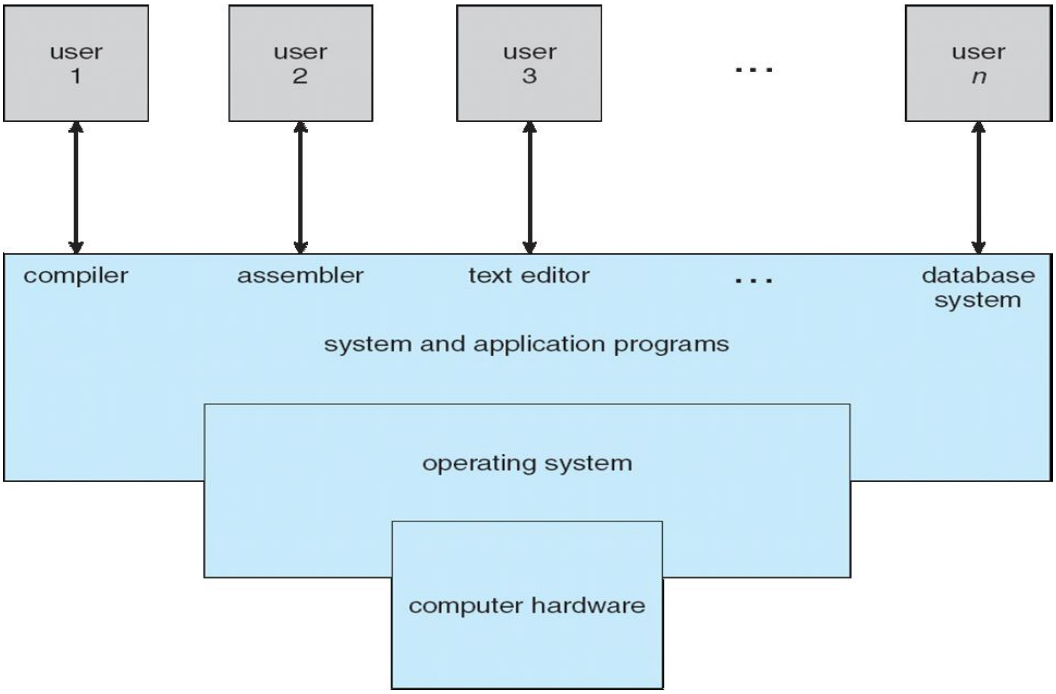
## Functions of operating system

Some of important functions of an operating System.

1. Process Management
2. Memory Management
3. File Management
4. Processor Management
5. Device Management
6. Security
7. Job accounting
8. Job priority
9. Error detecting aids
10. Coordination between other software and users
11. Control over system performance

**Computer System Structure:**

- Computer system can be divided into four components:
  - Hardware – provides basic computing resources
    - CPU, memory, I/O devices
  - Operating system
    - Controls and coordinates use of hardware among various applications and users
  - Application programs – define the ways in which the system resources are used to solve the computing problems of the users
    - Word processors, compilers, web browsers, database systems, video games
  - Users
    - People, machines, other computers




## What Operating Systems Do:

Depends on the point of view

**User view:**

- Users want convenience, **ease of use** and **good performance**
  - Don't care about **resource utilization**

- But shared computer such as **mainframe** or **minicomputer** must keep all users happy
- Users of dedicate systems such as **workstations** have dedicated resources but frequently use shared resources from **servers**
- Some computers have little or no user interface, such as embedded computers in devices and automobiles

**System view:**

- From computer view OS is the program most involved with the hardware.
- In this point of view , the OS is viewed as a Resource allocator.
- Computer resources are like CPU time, memory space, I/O devices, storage space etc.
- OS acts as a manager of these resources.
- In other view, OS needs to control the various I/O devices and user programs. So it is control program.

## Operating System Definition

- OS is a resource allocator
  - It manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- OS is a control program
  - Controls execution of programs to prevent errors and improper use of the computer
- No universally accepted definition
- Everything a vendor ships when you order an operating system" is good approximation
  - But varies wildly
- "The one program running at all times on the computer" is the **kernel**. Everything else is either a system program (ships with the operating system) or an application program

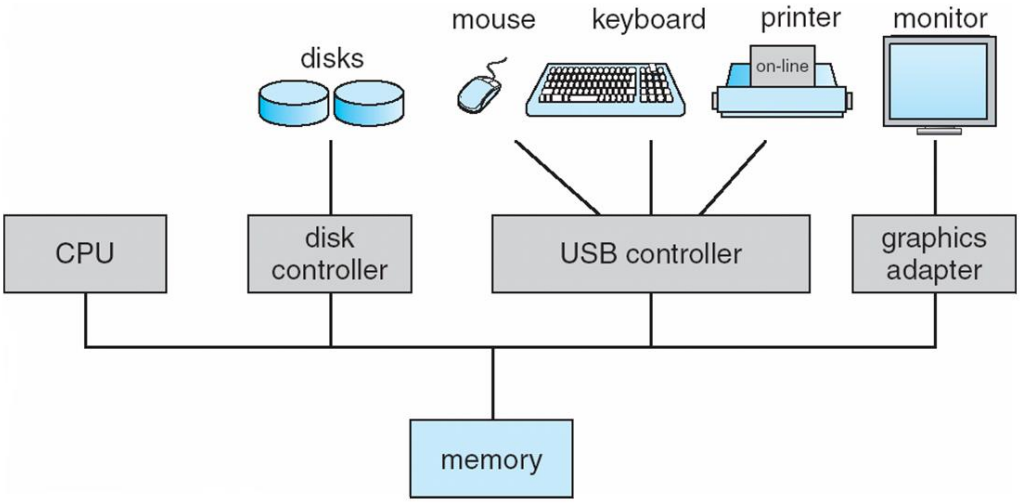
**Computer Startup:**

- For a computer to start running when it is powered on, it needs to have an initial program to run.
- There is **bootstrap program** is loaded at power-up or reboot
  - Typically stored in ROM or EPROM, generally known as **firmware**
  - Initializes all aspects of system
  - Loads operating system kernel and starts execution

## Computer System Organization:

- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory

– Concurrent execution of CPUs and devices competing for memory cycles




## Computer-System Operation:

- A general purpose computer consists of one or more CPUs and a number of device controllers connected through a common bus that provides a shared memory.(figure shown in before slide)
- I/O devices and the CPU can execute concurrently
- Each device controller is in charge of a particular device type
- Each device controller has a local buffer
- CPU moves data from/to main memory to/from local buffers
- I/O is from the device to local buffer of controller
- Device controller informs CPU that it has finished its operation by causing an interrupt

## Interrupts:

- In a computer , the operating system executes different processes. When the processes are executing , the operating system switches from one process to another when an interrupt occurs.
- An interrupt is an event that alters the sequence in which the processor executes instructions.
- The occurrence of the event is usually signaled by an interrupt, from hardware or software.
- Hardware Interrupts are generated by hardware devices in form of signal to CPU by bus
- Software programs give an interrupt by executing a special operation called as system call.

- When an interrupt occurs, the CPU stops what is doing and immediately transfers execution to fixed location where . On completion it returns back to the previous location.

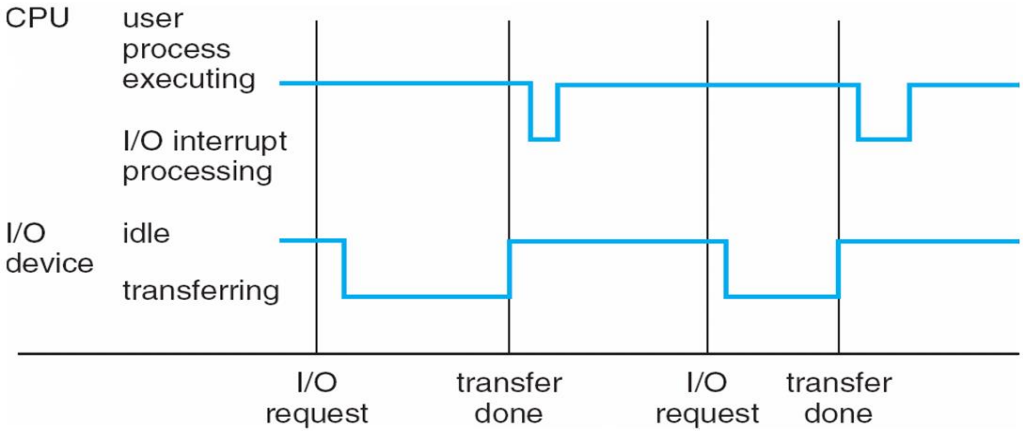
**Common Functions of Interrupts:**

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

**Interrupt Handling:**

- The operating system preserves the state of the CPU by storing registers and the program counter
- Determines which type of interrupt has occurred:
  - **polling**
  - **vectored** interrupt system
- Separate segments of code determine what action should be taken for each type of interrupt

**Interrupt Timeline:**




**I/O Structure:**

- After I/O starts, control returns to user program only upon I/O completion
  - Wait instruction idles the CPU until the next interrupt
  - Wait loop (contention for memory access)

  - At most one I/O request is outstanding at a time, no simultaneous I/O processing
- After I/O starts, control returns to user program without waiting for I/O completion
  - **System call** – request to the OS to allow user to wait for I/O completion
  - **Device-status table** contains entry for each I/O device indicating its type, address, and state
  - OS indexes into I/O device table to determine device status and to modify table entry to include interrupt

## Storage Structure:

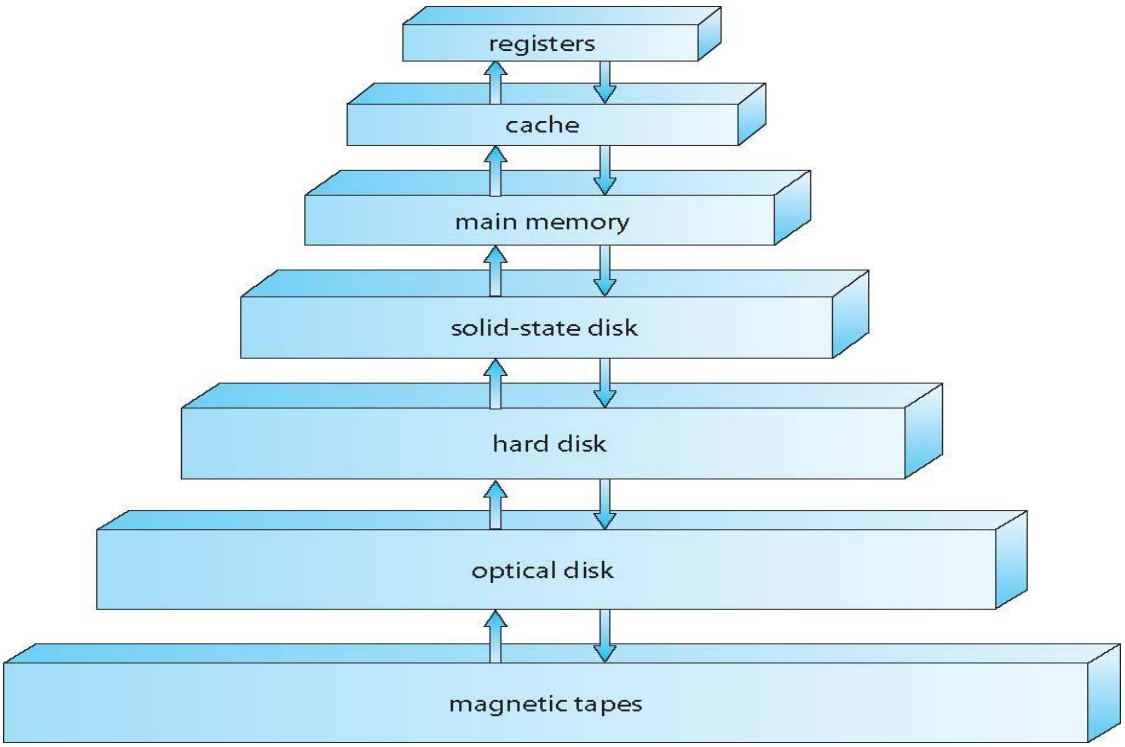
- Main memory – only large storage media that the CPU can access directly
  - **Random access**
  - Typically **volatile**
- Secondary storage – extension of main memory that provides large **nonvolatile** storage capacity
- Hard disks – rigid metal or glass platters covered with magnetic recording material
  - Disk surface is logically divided into **tracks**, which are subdivided into **sectors**
  - The **disk controller** determines the logical interaction between the device and the computer
- **Solid-state disks** – faster than hard disks, nonvolatile
  - Various technologies
  - Becoming more popular

**Storage Hierarchy:**

- Storage systems organized in hierarchy
  - Speed
  - Cost
  - Volatility
- **Caching** – copying information into faster storage system; main memory can be viewed as a cache for secondary storage

**Storage-Device Hierarchy:**

The storage devices hierarchy can be shown as in the below figure:




**Caching:**

- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast)
  - If not, data copied to cache and used there
- Cache smaller than storage being cached Cache management important design problem
- Cache size and replacement policy

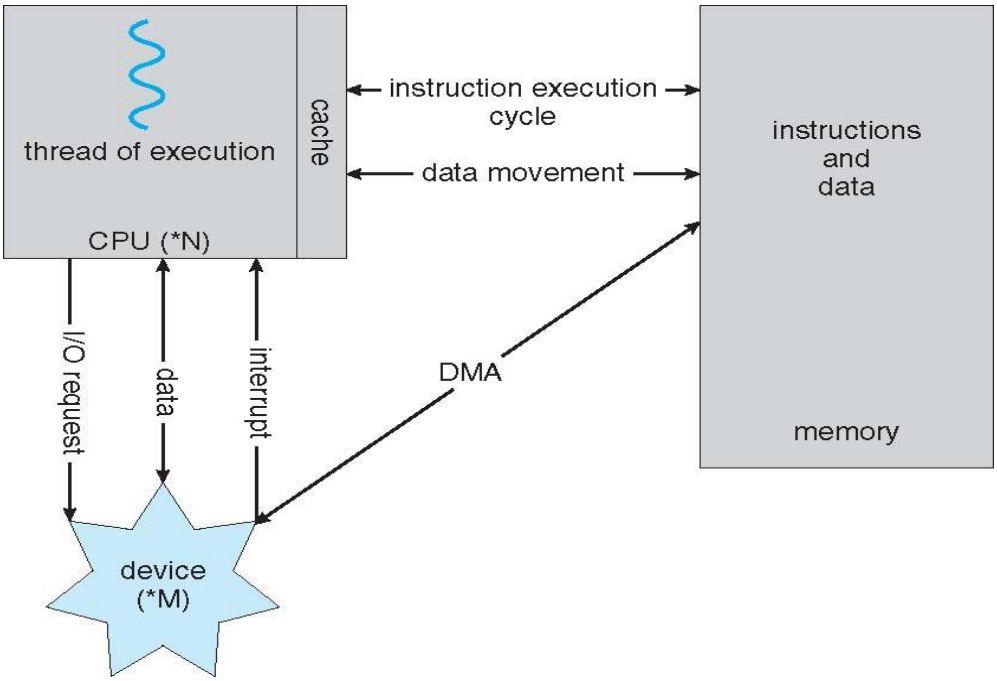
**Device Driver :**

Device Driver for each device controller to manage I/O

- Provides uniform interface between controller and kernel

**Direct Memory Access Structure:**

- Used for high-speed I/O devices able to transmit information at close to memory speeds
- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention
- Only one interrupt is generated per block, rather than the one interrupt per byte
- The below figure shows the how the computer works with DMA




## Computer-System Architecture:

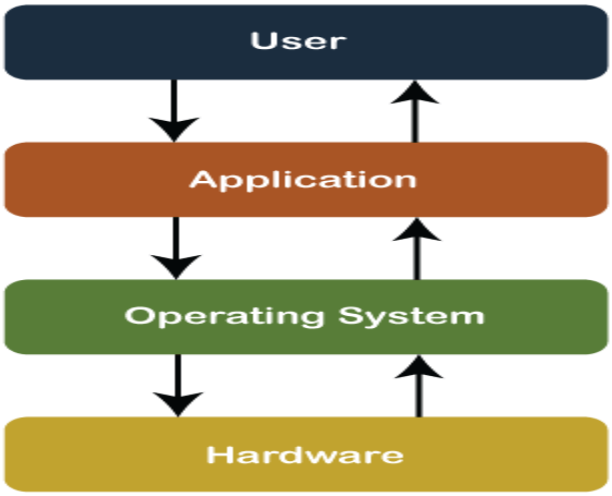
- Most systems use a single general-purpose processor (PDAs through mainframes)
- Most systems have special-purpose processors as well
- Multiprocessors systems growing in use and importance
- Also known as parallel systems, tightly-coupled systems
- Advantages include
  - 1.Increased throughput
  - 2.Economy of scale
  - 3.Increased reliability – graceful degradation or fault tolerance
- Two types
  1. Asymmetric Multiprocessing
  2. Symmetric Multiprocessing

## Clustered Systems:

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

- Applications must be written to use parallelization

## History of operating systems:




- 1940s computer invented
- 1950s batch processing done without os.
- 1964: IBM System/360
- 1967-1968: **Mouse** was invented.
- 1969: The UNIX Time-Sharing System from Bell Telephone Laboratories
- 1972: IBM comes out with VM: the Virtual Machine Operating System
- 1983 Microsoft begins work on MS-Windows
- 1984 Apple Macintosh comes out
- 1990 Microsoft Windows 3.0 comes out
- 1991 GNU/Linux
- 1993 Windows NT
- 2007: iOS
- 2008: Android OS

- Operating systems were first developed in the late 1950s to manage tape storage
- The General Motors Research Lab implemented the first OS in the early 1950s for their IBM 701
- In the mid-1960s, operating systems started to use disks
- In the late 1960s, the first version of the Unix OS was developed
- The first OS built by Microsoft was DOS. It was built in 1981 by purchasing the 86-DOS software from a Seattle company

- The present-day popular OS Windows first came to existence in 1985 when a GUI was created and paired with MS-DOS.

## Types of Operating Systems:

There are different types of operating systems. They are:

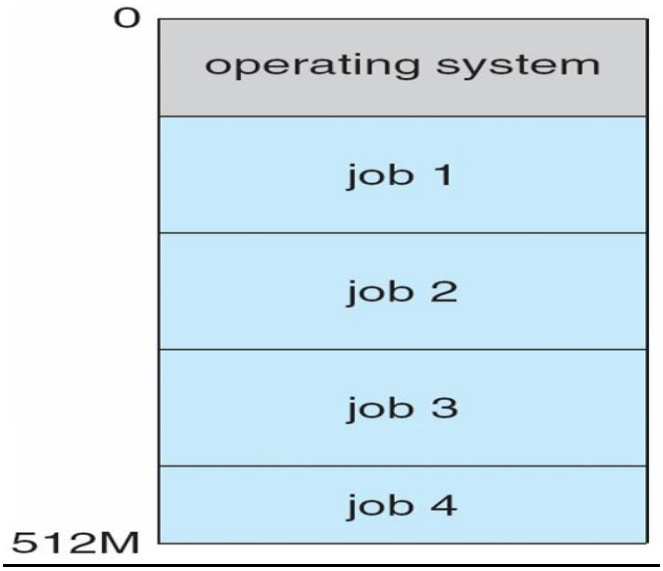
1. Batch Operating System
2. Time-sharing operating systems
3. Distributed Operating System
4. Network operating System
5. Real Time Operating System
6. Embedded operating system.
7. Multiprogramming Operating System
8. Multiprocessing Operating System

(Explanation given in the end of the unit-1 notes)

### Operating System Structure:

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

### Memory Layout for Multiprogrammed System




## Operating-System Operations: (or) Dual Mode operation of OS:

- The modern Operating systems are interrupt driven.
- Interrupt driven (hardware and software)
  - Hardware interrupt by one of the devices
  - Software interrupt (exception or trap):
    - Software error (e.g., 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
  - A bit called mode bit is added to hardware to computer to indicate mode current mode:
    Kernel- mode bit is 0
    User –mode bit is 1

**User Mode:**

The system is in user mode when the operating system is running a user application such as handling a text editor. The transition from user mode to kernel mode occurs when the application requests the help of operating system or an interrupt or a system call occurs.

The mode bit is set to 1 in the user mode. It is changed from 1 to 0 when switching from user mode to kernel mode.

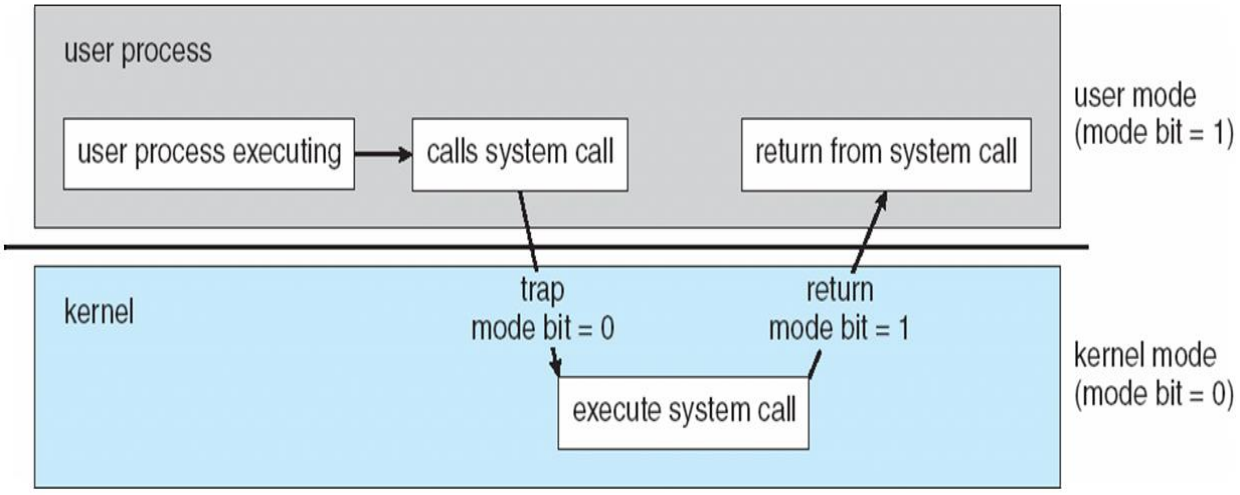
**Kernel Mode:**

The system starts in kernel mode when it boots and after the operating system is loaded, it executes applications in user mode. There are some privileged instructions that can only be executed in kernel mode.
These are interrupt instructions, input output management etc. If the privileged instructions are executed in user mode, it is illegal and a trap is generated.
The mode bit is set to 0 in the kernel mode. It is changed from 0 to 1 when switching from kernel mode to user mode.

- The mode bit Provides ability to distinguish when system is running user code or kernel code
- Some instructions designated as **privileged**, only executable in kernel mode
- System call changes mode to kernel, return from call resets it to user

**Transition from User to Kernel Mode:**

- Timer to prevent infinite loop / process hogging resources
  - Timer is set to interrupt the computer after some time period
  - Keep a counter that is decremented by the physical clock.
  - Operating system set the counter (privileged instruction)
  - When counter zero generate an interrupt
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time




In the above image, the user process executes in the user mode until it gets a system call. When a system occurs, then a system trap is generated and the mode bit is set to zero. The system call gets executed in kernel mode. After the execution is completed, again a system trap is generated and the mode bit is set to 1. The system control returns to kernel mode and the process execution continues.

## OPERATING SYSTEM FUNCTIONS :

## Process Management

- In multiprogramming environment, the OS decides which process gets the processor when and for how much time. This function is called **process scheduling**.
- The operating system is very useful for process management.
- A process is a program in execution. It is a unit of work within the system. Program is a ***passive entity***, process is an ***active entity***.
- Process needs resources to accomplish its task
  - CPU, memory, I/O, files
  - Initialization data
- Process termination requires reclaim of any reusable resources
- Single-threaded process has one **program counter** specifying location of next instruction to execute
  - Process executes instructions sequentially, one at a time, until completion
- Multi-threaded process has one program counter per thread
- Typically system has many processes, some user, some operating system running concurrently on one or more CPUs

## Process Management Activities:

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

- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling
- Keeps tracks of processor and status of process. The program responsible for this task is known as traffic controller.
- Allocates the processor (CPU) to a process.
- De-allocates processor when a process is no longer required.

## Memory Management:

- The operating system handles the responsibility of storing any data, system programs, and user programs in memory. This function of the operating system is called memory management.
- All data in memory before and after processing
- All instructions in memory in order to execute
- Memory management determines what is in memory when
  - Optimizing CPU utilization and computer response to users
- Memory management activities

- Keeping track of memory: which parts of memory are currently being used and by whom
- In multiprogramming, the OS decides which process will get memory when and how much.
- Allocating and deallocating memory space as needed
- Allocates the memory when a process requests it to do so.
- De-allocates the memory when a process no longer needs it or has been terminated.

## Storage Management:

- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - **file**
  - Each medium is controlled by device (i.e., disk drive, tape drive)
    - These devices Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
  - OS activities include
    - Creating and deleting files and directories
    - Supporting Primitives to manipulate files and directories
    - Mapping files onto secondary storage
    - Backup files onto stable (non-volatile) storage media

**File Management**:

The operating system is helpful in making changes in the stored files and in replacing them. It also plays an important role in transferring various files to a device.

An Operating System does the following activities for file management −

1. Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.
2. Decides who gets the resources.
3. Allocates the resources.
4. De-allocates the resources.

**Device Management :**

An Operating System manages device communication via their respective drivers.

An Operating System does the following activities for device management −

1. Keeps tracks of all devices. Program responsible for this task is known as the **I/O controller**.
2. Decides which process gets the device when and for how much time.
3. Allocates the device in the efficient way.

4. De-allocates devices.

## Mass-Storage Management

- Usually disks used to store data that does not fit in main memory or data that must be kept for a "long" period of time
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystem and its algorithms
- OS activities
  - Free-space management
  - Storage allocation
  - Disk scheduling
- Some storage need not be fast
- Tertiary storage includes optical storage, magnetic tape
- Still must be managed
- Varies between WORM (write-once, read-many-times) and RW (read-write)

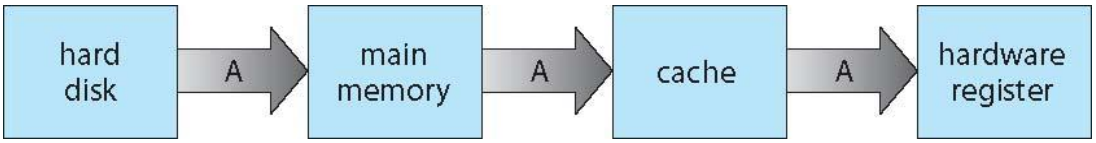
Performance of Various Levels of Storage

| Level | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Name | registers | cache | main memory | solid state disk | magnetic disk |
| Typical size | < 1 KB | < 16MB | < 64GB | < 1 TB | < 10 TB |
| Implementation technology | custom memory with multiple ports CMOS | on-chip or off-chip CMOS SRAM | CMOS SRAM | flash memory | magnetic disk |
| Access time (ns) | 0.25 - 0.5 | 0.5 - 25 | 80 - 250 | 25,000 - 50,000 | 5,000,000 |
| Bandwidth (MB/sec) | 20,000 - 100,000 | 5,000 - 10,000 | 1,000 - 5,000 | 500 | 20 - 150 |
| Managed by | compiler | hardware | operating system | operating system | operating system |
| Backed by | cache | main memory | disk | disk | disk or tape |

Movement between levels of storage hierarchy can be explicit or implicit

**Migration of data "A" from Disk to Register:**

- Multitasking environments must be careful to use most recent value, no matter where it is stored in the storage hierarchy




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

## Protection and Security:

- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control
  - Group identifier (**group ID**) allows set of users to be defined and controls managed, then also associated with each process, file
  - **Privilege escalation** allows user to change to effective ID with more rights

- Computer security is a very important aspect of any operating system. The reliability of an operating system is determined by how much better security it provides us. Modern operating systems use a firewall for security. A firewall is a security system that monitors every activity happening in the computer and blocks that activity in case of any threat.

## Advantages of Operating System:

- It is helpful to monitor and regulate resources.
- The performance of the computer system is based on the CPU.
- The response time and throughput time of any process or program are fast.
- It can share different resources like fax, printer, etc.

- We can easily install any game or application on the Operating system easily and can run them
- An operating system can be refreshed easily from time to time without having any problems.
- An operating system can be updated easily.
- The operating system mainly acts as an interface between the hardware and the software.
- Users can easily access the hardware without writing large programs.
- With the help of an Operating system, sharing data becomes easier with a large number of users.
- There are various operating systems that are accessible in open source. Example: Unix/Linux these can easily run on the personal computer with no cost that means Free.
- An operating system is mainly used to hide the complexity of the hardware.
- With the help of an operating system, multitasking becomes easier.
- The different components of a system are independent of each other, thus failure of one component does not affect the functioning of another.

**Disadvantage of the Operating System**

- It allows only a few tasks that can run at the same time.
- If any error occurred in the operating system; the stored data can be destroyed.
- It is a very difficult task or works for the OS to provide entire security from the viruses because any threat or virus can occur at any time in a system.
- An unknown user can easily use any system without the permission of the original user.
- The cost of operating system costs is very high.

# Computing Environments:

- Computer Environment is a collection of Computer Devices which are used to process and exchange information to solve various types of computing problems.
- It is arrangement of computer devices to a solve a problem.

**Personal Computing Environment**

In the personal computing environment, there is a single computer system. All the system processes are available on the computer and executed there. The different devices that

constitute a personal computing environment are laptops, mobiles, printers, computer systems, scanners etc.

**Traditional Computing**

- Stand-alone general purpose machines
- **Network computers** (**thin clients**) are like Web terminals
- Mobile computers interconnect via **wireless networks**
- Networking becoming ubiquitous – even home systems use **firewalls** to protect home computers from Internet attacks

**Computing Environments – Mobile:**

- Handheld smartphones, tablets, etc
- What is the functional difference between them and a "traditional" laptop?
- Extra feature – more OS features (GPS)
- Allows new types of apps like ***augmented reality***
- Use IEEE 802.11 wireless, or cellular data networks for connectivity
- Leaders are **Apple iOS** and **Google Android**

**Distributed computing Environment:**

- A distributed computing environment contains multiple nodes that are physically separate but linked together using the network. All the nodes in this system communicate with each other and handle processes in tandem. Each of these nodes contains a small part of the distributed operating system software
- Collection of separate, possibly heterogeneous, systems networked together
  - **Network** is a communications path, **TCP/IP** most common
    - **Local Area Network** (**LAN**)
    - **Wide Area Network** (**WAN**)
    - **Metropolitan Area Network** (**MAN**)
    - **Personal Area Network** (**PAN**)
  - **Network Operating System** provides features between systems across network
    - Communication scheme allows systems to exchange messages
    - Illusion of a single system

**Client-Server Computing.**

- This is a form of distributed computing.
- In client server computing, the client requests a resource and the server provides that resource. A server may serve multiple clients at the same time while a client is in contact with only one server. Both the client and server usually communicate via a computer network but sometimes they may reside in the same system.
- Many computers are connected together .
- Dumb terminals supplanted by smart PCs
- Many systems now **servers**, responding to requests generated by **clients**

**1)Compute-server system** provides an interface to client to request services to perform an action

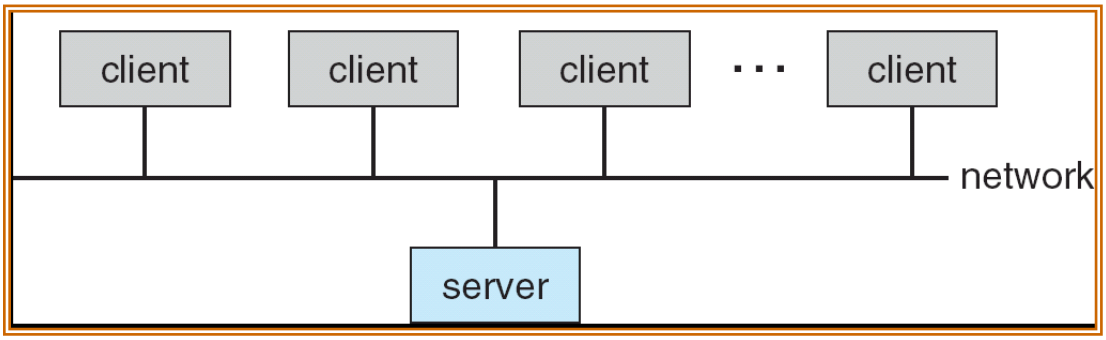
- Ex: database server to read and respond to clients request.

**2)File-server system** provides interface for clients to store and retrieve files

Ex: Web server

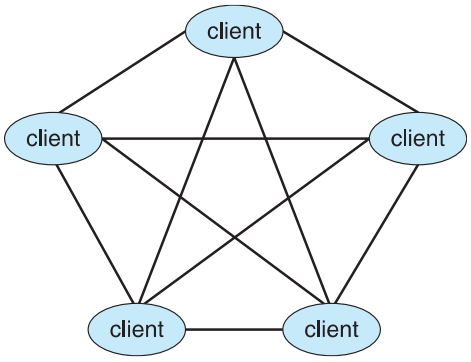
General Structure of Client-Server System:

It is shown in the below diagram




**Peer to Peer computing:**

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



**Web based computing:**

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

**Time Sharing Computing Environment**

The time sharing computing environment allows multiple users to share the system simultaneously. Each user is provided a time slice and the processor switches rapidly among the users according to it. Because of this, each user believes that they are the only ones using the system.

**Cloud Computing Environment:**

The computing is moved away from individual computer systems to a cloud of computers in cloud computing environment. The cloud users only see the service being provided and not the internal details of how the service is provided. This is done by pooling all the computer resources and then managing them using a software.

**Cluster Computing Environment**

The clustered computing environment is similar to parallel computing environment as they both have multiple CPUs. However a major difference is that clustered systems are created by two or more individual computer systems merged together which then work parallel to each other.

## Open source Operating Systems:

- Open source refers to the computer software or applications where the owners or copyright holders allow the users or third party to see, use and provide the right to modify the source code of the product.
- An **Open-source Operating System** is the Operating System in which source code is visible publicly and editable. The generally known Operating Systems like **Microsoft's Windows, Apple's iOS and Mac OS, are closed Operating system.**
- In Closed Operating Systems are, the source code is kept secret by the respective companies (owners) and inaccessible to third parties. By doing so, they ensure the safety and secure the Operating System and computer from any threats.
- In the case of an Open Source Operating system, everyone can access and edit the source code.

Examples:

- Linux

-Linux is the best-known and most-used open source operating system. As an operating system, Linux is software that sits underneath all of the other software on a computer, receiving requests from those programs and relaying these requests to the computer's hardware.

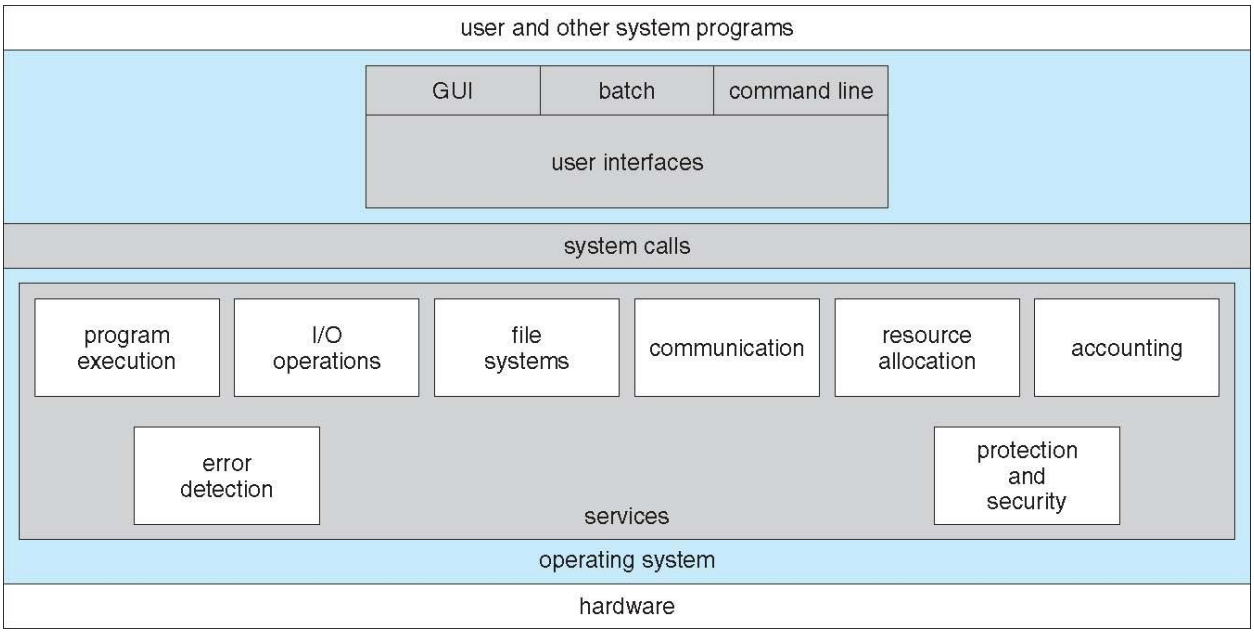
- Ubuntu.
- Linux Lite.
- Fedora.
- Linux Mint.
- Sun Solaris
- GNU/Linux, BSD UNIX

# Operating System Services:

- Following are a few common services provided by an operating system −

1. User interface
2. Program execution
3. I/O operations
4. File System manipulation
5. Communication
6. Error Detection
7. Resource Allocation
8. Protection
9. Accounting

**A View of Operating System Services**




- One set of operating-system services provides functions that are helpful to the user:

**1. User interface** -

- Operating system provides user interface as an service in types
  - 1.Command line interface(CLI) 2.Graphical user interface(GUI)
- We can access command line interface using the small commands . Also we can access graphical user interface using the clicking of applications through keyboard or mouse.

**2. Program execution** –

The operating system must have the capability to load a program into memory and to run that program, end execution, either normally or abnormally (indicating error)

Following are the major activities of an operating system with respect to program management −

- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

3. **I/O operations** -

- A running program may require I/O, which may involve a file or an I/O device
- I/O operation means read or write operation with any file or any specific I/O device.
- Operating system provides the access to the required I/O device when required
- An I/O subsystem comprises of I/O devices and their corresponding driver software. Drivers hide the peculiarities of specific hardware devices from the users.

- An Operating System manages the communication between user and device drivers.

4. **File-system manipulation** -

- The file system is of particular interest. Obviously, programs need to read and write files and directories, create and delete them, search them, list file Information, permission management.
- A file represents a collection of related information. Computers can store files on the disk (secondary storage), for long-term storage purpose. Examples of storage media include magnetic tape, magnetic disk and optical disk drives like CD, DVD. Each of these media has its own properties like speed, capacity, data transfer rate and data access methods.
- A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Following are the major activities of an operating system with respect to file management −

Program needs to read a file or write a file.

- The operating system gives the permission to the program for operation on file.
- Permission varies from read-only, read-write, denied and so on.
- Operating System provides an interface to the user to create/delete files.
- Operating System provides an interface to the user to create/delete directories.
- Operating System provides an interface to create the backup of file system.

One set of operating-system services provides functions that are helpful to the user :

**5. Communications** – Processes may exchange information, on the same computer or between computers over a network

  – Communications may be via shared memory or through message passing (packets of information moved by the OS)

Process needs to swap over information with other process. Processes executing on same computer system or on different computer systems can communicate using operating system support. Communication between two processes can be done using shared memory or via message passing.

. Following are the major activities of an operating system with respect to communication −

- Two processes often require data to be transferred between them
- Both the processes can be on one computer or on different computers, but are connected through a computer network.
- Communication may be implemented by two methods, either by Shared Memory or by Message Passing.

6. **Error detection** –

OS needs to be constantly aware of possible errors

- May occur in the CPU and memory hardware, in I/O devices, in user program
- For each type of error, OS should take the appropriate action to ensure correct and consistent computing
- Debugging facilities can greatly enhance the user's and programmer's abilities to efficiently use the system

Errors can occur anytime and anywhere. An error may occur in CPU, in I/O devices or in the memory hardware. Following are the major activities of an operating system with respect to error handling −

- The OS constantly checks for possible errors.
- The OS takes an appropriate action to ensure correct and consistent computing.

Another set of OS functions exists for ensuring the efficient operation of the system itself via resource sharing

**7. Resource allocation -** When multiple users or multiple jobs running concurrently, resources must be allocated to each of them

- Many types of resources - Some (such as CPU cycles, main memory, and file storage) may have special allocation code, others (such as I/O devices) may have general request and release code
- Following are the major activities of an operating system with respect to resource management −
- The OS manages all kinds of resources using schedulers.
- CPU scheduling algorithms are used for better utilization of CPU.

**8. Accounting -**

This service of the operating system keeps track of which users are using how much and what kinds of computer resources have been used for accounting or simply to accumulate usage statistics.

**9. Protection and security -** The owners of information stored in a multiuser or networked computer system may want to control use of that information. When separate processes are running concurrently, processes should not interfere with each other

- **Protection** involves ensuring that all access to system resources is controlled

Considering a computer system having multiple users and concurrent execution of multiple processes, the various processes must be protected from each other's activities. Protection includes in ensuring all access to system resources in a controlled manner. For making a system secure, the user needs to authenticate him or her to the system before using (usually via login ID and password).

Following are the major activities of an operating system with respect to protection −

- The OS ensures that all access to system resources is controlled.
- The OS ensures that external I/O devices are protected from invalid access attempts.
- The OS provides authentication features for each user by means of passwords.

- **Security** of the system from outsiders requires user authentication, extends to defending external I/O devices from invalid access attempts

## User Operating System Interface (or) User Interface:

A **User interface (UI)** facilitates communication between an application and its user by acting as an intermediary between them. Each application including the operating system is provided with a specific UI for effective communication. The two basic function of a user interface of an application is to take the inputs from the user and to provide the output to the users. However, the types of inputs taken by the UI and the types of output provided by the UI may vary from one application to another.

- There are two fundamental types for users to interface with the operating system.
    1. Graphical user interface (GUI)
    2. Command line user interface (CLI)

### 1) Command Line Interface

Command line interface is a type of UI that enables the users to interact with the operating system by issuing some specific commands. In order to perform a task in this interface, the user needs to type a command at the command line. When the user enters the key, the command line interpreter received a command.

(CLI) or command interpreter allows direct command entry-

-It allows the users to directly enter commands and perform the action by the operating system.

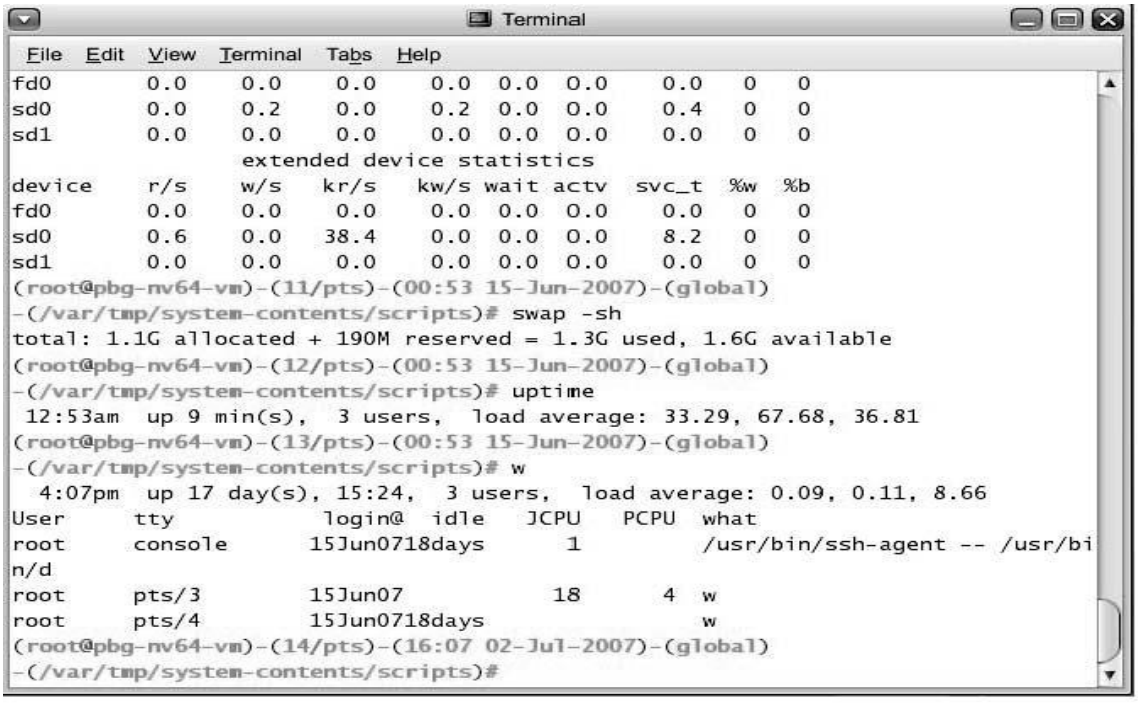
- Sometimes implemented in kernel, sometimes by systems program
- On systems with multiple command interpreters to choose from, these interpreters are known as – shells
- In Linux and Unix, there are several shells like Bourne shell, C shell, Korn shell etc
- Primarily fetches a command from user and executes it
    - Sometimes commands built-in, sometimes just names of programs

**The disadvantages of the CLI** is that the user needs to remember a lot to interact with the operating system. Therefore these types of interface are not considered very friendly from the users perspective.

## 2) Graphical User Interface(GUI):

- The graphical user interface is a type of GUI that enables the users to interact with the operating system by means of point-and-click operations. GUI contains several icons representing pictorial representation of the variables such as a file, directory, and device.
- The graphical icon provided in the UI can be manipulated by the users using a suitable pointing device such as a mouse, trackball, touch screen and light pen. The other input devices like keyboard can also be used to manipulate these graphical icons.
- GUIs are considered to be very user- friendly interface because each object is represented with a corresponding icon.
- User-friendly desktop metaphor interface
  - Usually mouse, keyboard, and monitor
  - Icons represent files, programs, actions, etc
  - Various mouse buttons over objects in the interface cause various actions (provide information, options, execute function, open directory (known as a folder)
- Many systems now include both CLI and GUI interfaces
  - Microsoft Windows is GUI with CLI "command" shell
  - Apple Mac OS X as "Aqua" GUI interface with UNIX kernel underneath and shells available
  - Solaris is CLI with optional GUI interfaces

Bourne Shell Command Interpreter:



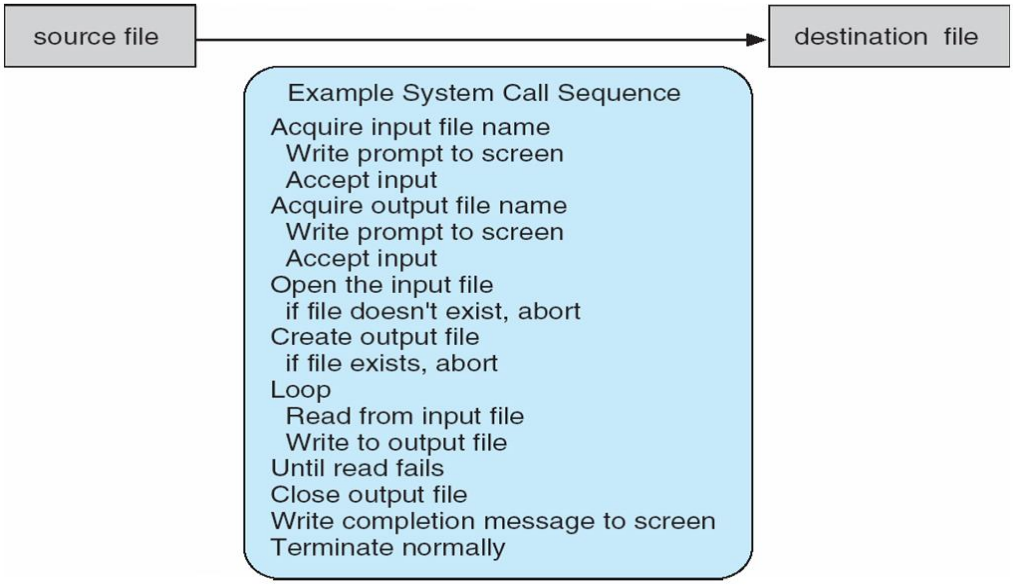
The Mac OS X GUI:


# System Calls:

- System calls provide an Programming interface to the services provided by the OS.
- The interface between a process and an operating system is provided by system calls. In general, system calls are available as assembly language instructions. They are also included in the manuals used by the assembly level programmers.
- System calls are usually made when a process in user mode requires access to a resource. Then it requests the kernel to provide the resource via a system call.
- It is a programmatic method in which a computer program requests a service from the kernel of the OS.
- These calls are generally available as routines written in a high-level language (C or C++).
- System call offers the services of the operating system to the user programs via API. Mostly accessed by programs via a high-level Application Program Interface (API) rather than direct system call use
- Three most common APIs are Win32 API for Windows, POSIX API for POSIX-based systems (including virtually all versions of UNIX, Linux, and Mac OS X), and Java API for the Java virtual machine (JVM)

Example of System Calls:

- System call sequence to copy the contents of one file to another file




## Why do you need system calls in Operating System?

There are various situations where you must require system calls in the operating system. Following of the situations are as follows:

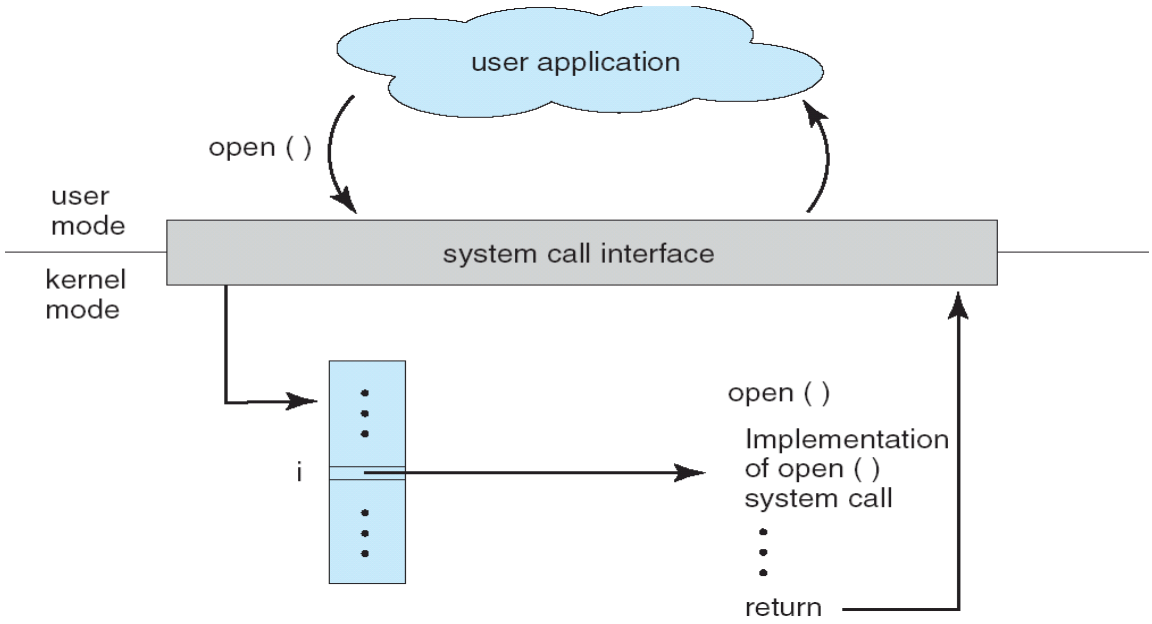
1. It is must require when a file system wants to create or delete a file.
2. Network connections require the system calls to sending and receiving data packets.
3. If you want to read or write a file, you need to system calls.
4. If you want to access hardware devices, including a printer, scanner, you need a system call.
5. System calls are used to create and manage new processes.

**System Call Implementation:**

- Typically, a number associated with each system call
  - System-call interface maintains a table indexed according to these numbers
- The system call interface invokes intended system call in OS kernel and returns status of the system call and any return values
- The caller need know nothing about how the system call is implemented

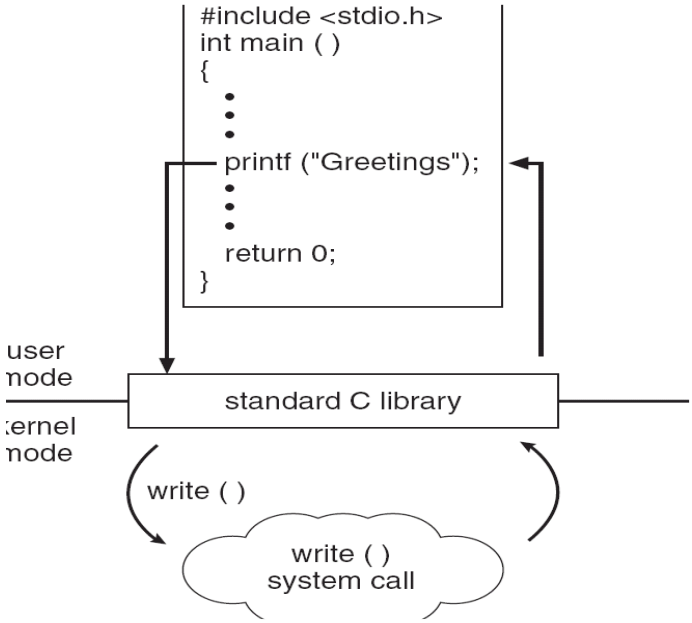
- Just needs to obey API and understand what OS will do as a result of execution of system call.
- Most details of OS interface hidden from programmer by API
  - Managed by run-time support library (set of functions built into libraries included with compiler)

**API – System Call – OS Relationship:**




**Standard C Library Example:**

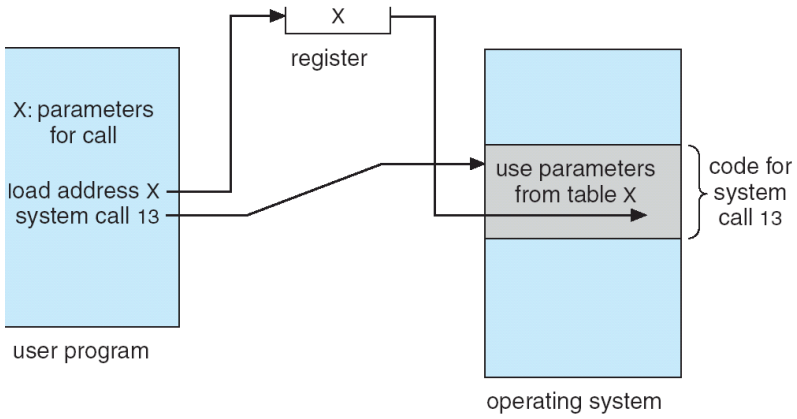
- C program invoking printf() library call, which calls write() system call



**System Call Parameter Passing:**

- System calls occur in different ways depending on computer in use. Often, more information is required than simply identity of desired system call
  - Exact type and amount of information vary according to OS and call
- Three general methods used to pass parameters to the OS
  - Simplest method: to pass the parameters in *registers*
    - In some cases, may be more parameters than registers
  - In this case, Parameters stored in a *block,* or table, in memory, and address of block passed as a parameter in a register
    - This approach taken by Linux and Solaris
  - Parameters placed, or *pushed,* onto the *stack* by the program and *popped* off the stack by the operating system
  - Block and stack methods do not limit the number or length of parameters being passed

**Parameter Passing via Table:**




# Types of System Calls:

There are five types of system calls. These are as follows:

- Process control
- File management
- Device management
- Information maintenance
- Communications
- Protection

Examples of Windows and Unix System Calls:

| | Windows | Unix |
|---|---|---|
| Process Control | CreateProcess()<br>ExitProcess()<br>WaitForSingleObject() | fork()<br>exit()<br>wait() |
| File Manipulation | CreateFile()<br>ReadFile()<br>WriteFile()<br>CloseHandle() | open()<br>read()<br>write()<br>close() |
| Device Manipulation | SetConsoleMode()<br>ReadConsole()<br>WriteConsole() | ioctl()<br>read()<br>write() |
| Information Maintenance | GetCurrentProcessID()<br>SetTimer()<br>Sleep() | getpid()<br>alarm()<br>sleep() |
| Communication | CreatePipe()<br>CreateFileMapping()<br>MapViewOfFile() | pipe()<br>shmget()<br>mmap() |
| Protection | SetFileSecurity()<br>InitlializeSecurityDescriptor()<br>SetSecurityDescriptorGroup() | chmod()<br>umask()<br>chown() |

**1) Process Control**

This system calls perform the task of process creation, process termination, etc. It is a system call that is used to direct the processes. Some examples are ending, abort, load, execute, create, process, terminate the process, etc.

**Functions:**

- End and Abort
- Load and Execute
- Create Process and Terminate Process
- Wait and Signal Event
- Allocate and free memory

**2) File Management**

File management system calls handle file manipulation jobs like creating a file, reading, and writing, etc. These types of system calls are used to handle files. Some examples are creating files, delete files, open, close, read, write, etc.

**Functions:**

- Create a file
- Delete file
- Open and close file
- Read, write, and reposition
- Get and set file attributes

**3)Device Management**

Device management system calls does the job of device manipulation like reading from device buffers, writing into device buffers, etc.

This system call is used to deal with devices.

**Functions:**

- Request and release device
- Logically attach/ detach devices
- Get and Set device attributes

**4) Information Maintenance**

It handles information and its transfer between the OS and the user program. This type of system call is used to maintain information.

**Functions:**

- Get or set time and date
- Get process , file and device attributes
- Set process , file and device attributes

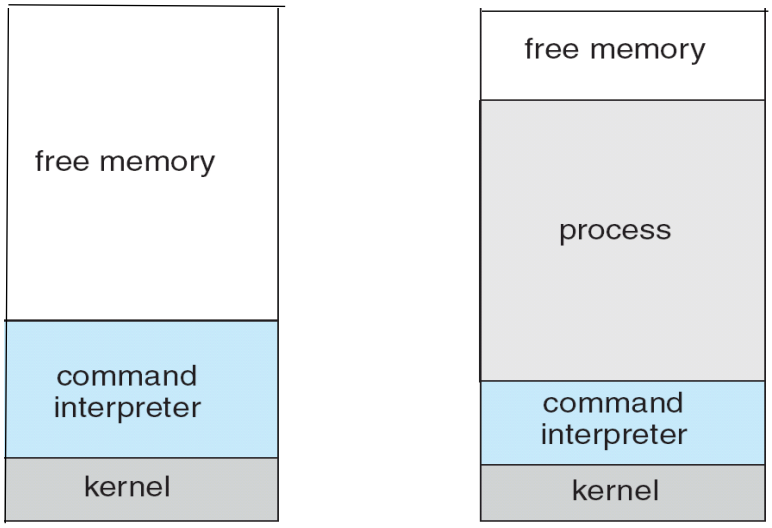
**5)Communication:**

These types of system calls are specially used for interprocess communications. These types of system calls are used for communication.

**Functions:**

- Create, delete communications connections
- Send, receive message
- Help OS to transfer status information
- Attach or detach remote devices

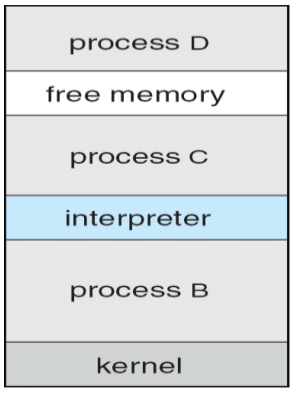
## MS-DOS execution:




a) At system startup (b) running a program

MS-DOS is a single task operating system
It executes only one process at a time.

**FreeBSD Running Multiple Programs:**




## System Programs:

System programs provide an convenient environment where programs can be developed and executed. In the simplest sense, system programs also provide a bridge between the user interface and system calls. In reality, they are much more complex. For example, a compiler is a complex system program.
System programs can be divided into different types. These are given as follows:

- File manipulation
- Status information
- File modification

- Programming language support
- Program loading and execution
- Communications
- Application programs

- **Status Information**

The status information system programs provide required data on the current or past status of the system. This may include the system date, system time, available memory in system, disk space, logged in users etc.

- **Communications**

These system programs are needed for system communications such as web browsers. Web browsers allow systems to communicate and access information from the network as required.

Communications system calls Provide the mechanism for creating virtual connections among processes, users, and computer systems. Allow users to send messages to one another's screens, browse web pages, send electronic-mail messages, log in remotely, transfer files from one machine to another

- **File Manipulation**

These system programs are used to manipulate system files. This can be done using various commands like create, delete, copy, rename, print etc. These commands can create files, delete files, copy the contents of one file into another, rename files, print them etc.

- **File management** - Create, delete, copy, rename, print, dump, list, and generally manipulate files and directories

- **Program Loading and Execution**

The system programs that deal with program loading and execution make sure that programs can be loaded into memory and executed correctly. Loaders and Linkers are a prime example of this type of system programs. Some examples are: Absolute loaders, relocatable loaders, linkage editors, and overlay, loaders, debugging systems for higher-level and machine language

- **File Modification**

System programs that are used for file modification basically change the data in the file or modify it in some other way. Text editors are a big example of file modification system programs.

- **Application Programs**

Application programs can perform a wide range of services as per the needs of the users. These include programs for database systems, word processors, plotting tools, spreadsheets, games, scientific applications etc.

- **Programming Language Support**

These system programs provide additional support features for different programming languages. Some examples of these are compilers, debuggers etc. These compile a program and make sure it is error free respectively.

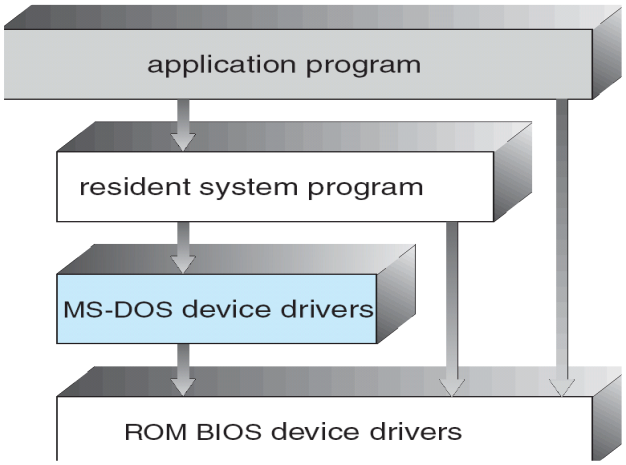
# Operating System Structure:

Operating system can be implemented with the help of various structures. The structure of the OS depends mainly on how the various common components of the operating system are interconnected and melded into the kernel. Depending on this we have following structures of the operating system:

**Simple Structure :**

- More commercial operating systems do not have well defined structure.
- They are small, simple and limited systems. The interfaces and levels of functionality are not well separated. MS-DOS is an example of such operating system. In MS-DOS application programs are able to access the basic I/O routines. These types of operating system cause the entire system to crash if one of the user programs fails.
- MS-DOS – written to provide the most functionality in the least space
  - Not divided into modules
  - Although MS-DOS has some structure, its interfaces and levels of functionality are not well separated

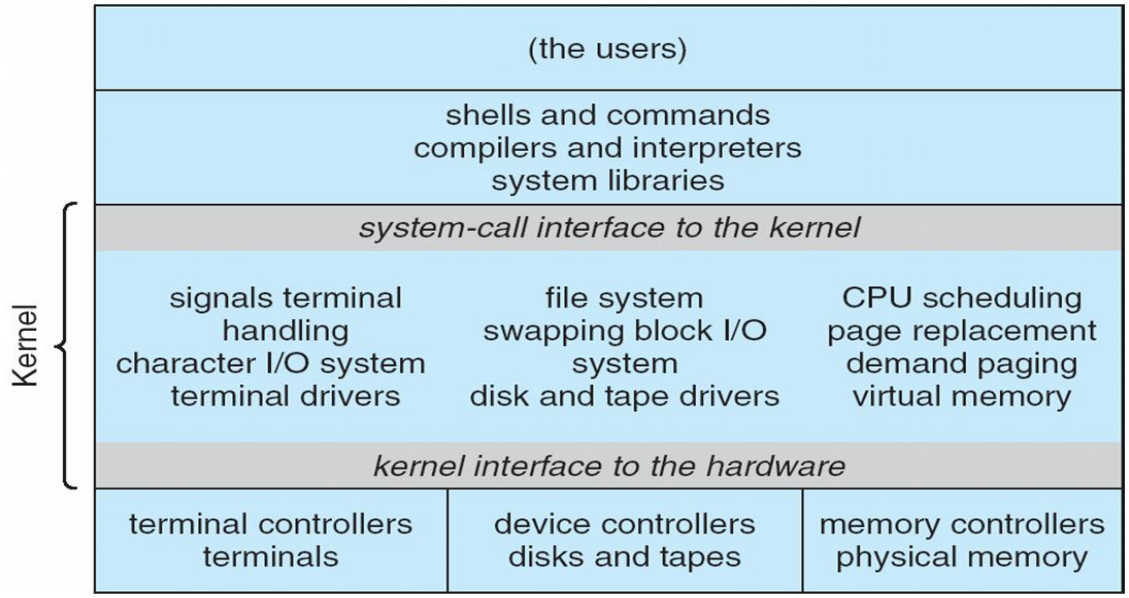
MS-DOS Layer Structure:




## UNIX structure:

- UNIX – limited by hardware functionality, the original UNIX operating system had limited structuring. The UNIX OS consists of two separable parts
  - Systems programs
  - The kernel
    - Consists of everything below the system-call interface and above the physical hardware
    - Provides the file system, CPU scheduling, memory management, and other operating-system functions; a large number of functions for one level

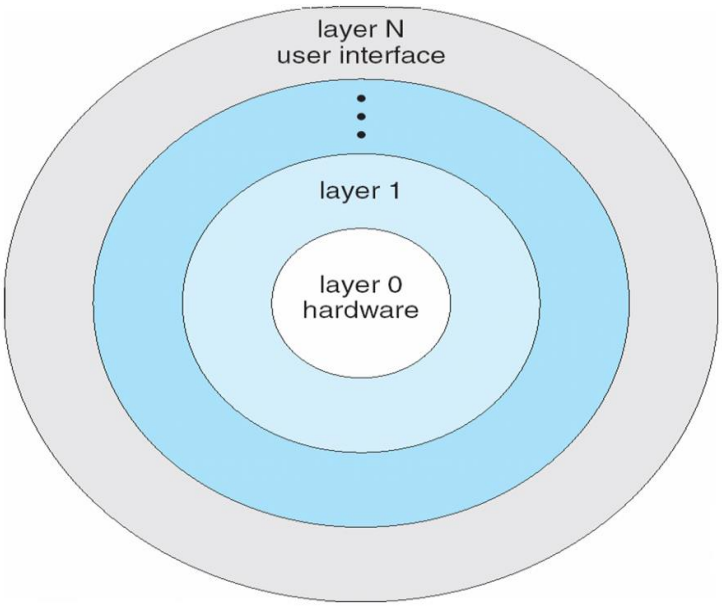
Traditional UNIX System Structure:




## Layered Approach:

- The operating system is divided into a number of layers (levels), each built on top of lower layers. The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.
- With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers
- Each upper layer is built on the bottom layer
- Every layer performs different functions like Hardware , Cpu scheduling, process management, memory management etc,
- These layers are so designed that each layer uses the functions of the lower-level layers only. It simplifies the debugging process as if lower-level layers are debugged, and an error occurs during debugging. The error must be on that layer only as the lower-level layers have already been debugged.
- Advantage is It is very easy to perform debugging and system verification.

Layered Operating System:




## System Boot

- Booting the system is process of loading the kernel into main memory, and starting its execution.
- A small program known as bootstrap program is used to load the kernel into main memory. It locates the kernel, loads it into memory, and starts it
- The initial bootstrap program is found in the BIOS read-only memory.
- This program can run diagnostics, initialize all components of the system, loads and starts the Operating System loader. (Called boot strapping)
- The loader program loads and starts the operating system.
- When the Operating system starts, it sets up needed data structures in memory, sets several registers in the CPU, and then creates and starts the first user level program.

## Types of Operating Systems:

There are different types of operating systems. They are:

1. Batch Operating System
2. Multiprogramming Operating System
3. Multiprocessing Operating System
4. Time-sharing operating systems
5. Distributed Operating System
6. Network operating System
7. Real Time Operating System
8. Embedded operating system.

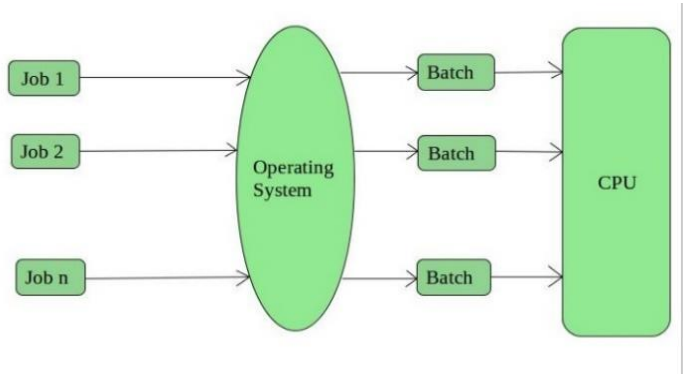
**1. Batch Operating System :**

In the era of 1970s, the Batch processing was very popular. The Jobs were executed in batches. People were used to have a single computer which was called mainframe.

This type of operating system does not interact with the computer directly. There is an operator which takes similar jobs having the same requirement and group them into batches. It is the responsibility of the operator to sort jobs with similar nee

In Batch operating system, access is given to more than one person; they submit their respective jobs to the system for the execution.

The system put all of the jobs in a queue on the basis of first come first serve and then executes the jobs one by one. The users collect their respective output when all the jobs get executed.




**Examples of Batch based Operating System:** Payroll System, Bank Statements, etc.

**Advantages of Batch OS**

- It is very difficult to guess or know the time required for any job to complete. ...
- Multiple users can share the batch systems.
- It is easy to manage large work repeatedly in batch systems.

**Disadvantages of Batch OS:**

The problems with Batch Systems are as follows −

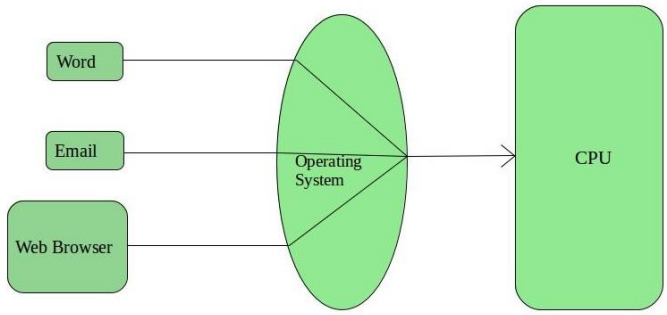
- Lack of interaction between the user and the job.
- CPU is often idle
- Difficult to provide the desired priority.

- **Starvation**: Batch processing suffers from starvation. If there are five jobs J1, J2, J3, J4 and J5 present in the batch. If the execution time of J1 is very high then other four jobs will never be going to get executed or they will have to wait for a very high time. Hence the other processes get starved.
- **Not Interactive**: Batch Processing is not suitable for the jobs which are dependent on the user's input. If a job requires the input of two numbers from the console then it will never be going to get it in the batch processing scenario since the user is not present at the time of execution.

## Time-sharing operating systems:

Each task is given some time to execute so that all the tasks work smoothly. Each user gets the time of CPU as they use a single system. These systems are also known as Multitasking Systems. The task can be from a single user or different users also. The time that each task gets to execute is called quantum. After this time interval is over OS switches over to the next task.

**Examples of Time-Sharing OSs are:** Multics, Unix, etc.




**Advantages of Timesharing operating systems** are as follows −

1. Provides the advantage of quick response.
2. Avoids duplication of software.
3. Reduces CPU idle time.

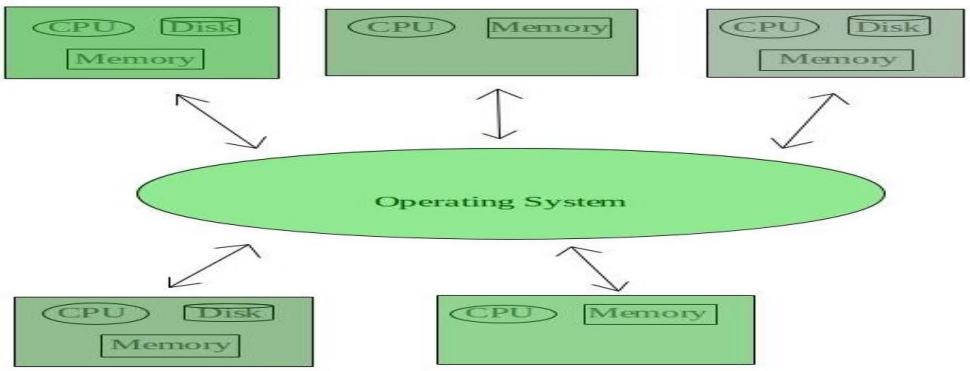
**Disadvantages of Time-sharing operating systems** are as follows −

1. Problem of reliability.
2. Question of security and integrity of user programs and data.
3. Problem of data communication.

## Distributed Operating System

These advancements in technology have made it possible to design and develop distributed systems comprising of many computers that are inter connected by communication networks. The main benefit of distributed systems is its low price/performance ratio. Independent systems possess their own memory unit and CPU. These are referred to as **loosely coupled systems** or distributed systems. These system's processors differ in size and function. The major benefit of working with these types of the operating system is that it is always possible that one user can access the files or software which are not actually present on his system but some other system connected within this network i.e., remote access is enabled within the devices connected in that network.

**Examples of Distributed Operating System are-** sun multiprocessor workstations, solaris operating systems etc.




## Advantages Distributed Operating System

The advantages of distributed systems are as follows –

1. Resource sharing facility is available .
2. If one site fails in a distributed system, the remaining sites can potentially continue operating.
3. Better service to the customers.
4. Reduction of the load on the host computer.

## DisAdvantages Distributed Operating System

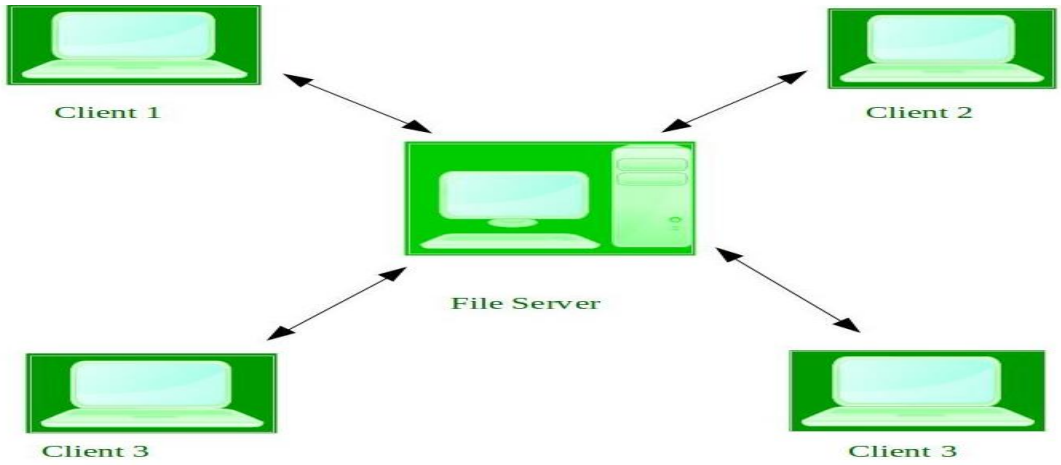
1. it is difficult to provide adequate security in distributed systems because the nodes as well as the connections need to be secured.
2. Some messages and data can be lost in the network while moving from one node to another.

## Network operating System:

A Network Operating System runs on a server and provides the server the capability to manage data, users, groups, security, applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), a private network or to other networks.

One more important aspect of Network Operating Systems is that all the users are well aware of the underlying configuration, of all other users within the network, their individual connections, etc. and that's why these computers are popularly known as **tightly coupled systems**.

Examples of network operating systems include Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD.




The **advantages of network operating systems** are as follows −

1. Centralized servers are highly stable.
2. Security is server managed.
3. Upgrades to new technologies and hardware can be easily integrated into the system.
4. Remote access to servers is possible from different locations and types of systems.

The **disadvantages of network operating systems** are as follows −

1. High cost of buying and running a server.
2. Dependency on a central location for most operations.
3. Regular maintenance and updates are required.

## Real Time Operating System:

In Real Time systems, each job carries a certain deadline within which the Job is supposed to be completed, otherwise the huge loss will be there or even if the result is produced then it will be completely useless.

The Application of a Real Time system exists in the case of military applications, if you want to drop a missile then the missile is supposed to be dropped with certain precision.

A real-time system is defined as a data processing system in which the time interval required to process and respond to inputs is so small that it controls the environment. The time taken by the system to respond to an input and display of required updated information is termed as the response time. So in this method, the response time is very less as compared to online processing.

Real-time systems are used when there are rigid time requirements on the operation of a processor or the flow of data and real-time systems can be used as a control device in a dedicated application. A real-time operating system must have well-defined, fixed time constraints, otherwise the system will fail. For example, Scientific experiments, medical imaging systems, industrial control systems, weapon systems, robots, air traffic control systems, etc.

There are two types of real-time operating systems.

here are two types of real-time operating systems.

**Hard real-time systems :** Hard real-time systems guarantee that critical tasks complete on time. In hard real-time systems, secondary storage is limited or missing and the data is stored in ROM. In these systems, virtual memory is almost never found.

**Soft real-time systems:** Soft real-time systems are less restrictive. A critical real-time task gets priority over other tasks and retains the priority until it completes. Soft real-time systems have limited utility than hard real-time systems. For example, multimedia, virtual reality, Advanced Scientific Projects like undersea exploration and planetary rovers, etc.

**Examples of Real-Time Operating Systems are:** Scientific experiments, medical imaging systems, industrial control systems, weapon systems, robots, air traffic control systems, etc.

## Advantages of real time operating system

1. Easier Testing.
2. Improved Efficiency.
3. Priority Based Scheduling.

**Disadvantages of real time operating system**

1. Limited Tasks.
2. Use Heavy System resources.
3. Complex Algorithms.

## Embedded Operating System

An Embedded Operating System is designed to perform a specific task for a particular device which is not a computer. For example, the software used in elevators is dedicated to the working of elevators only and nothing else. So, this can be an example of Embedded Operating System. The Embedded Operating System allows the access of device hardware to the software that is running on the top of the Operating System.

### Advantages:

1. Since it is dedicated to a particular job, so it is fast.
2. Low cost.
3. These consume less memory and other resources.

### Disadvantages:

1. Only one job can be performed.
2. It is difficult to upgrade or is nearly scalable.

## Multiprogramming Operating System:

Multiprogramming is an extension to the batch processing where the CPU is kept always busy. Each process needs two types of system time: CPU time and IO time.

In multiprogramming environment, for the time a process does its I/O, The CPU can start the execution of other processes. Therefore, multiprogramming improves the efficiency of the system.

**Advantages of multiprogramming systems**

- CPU is used most of time and never become idle
- The system looks fast as all the tasks runs in parallel
- Short time jobs are completed faster than long time jobs
- Multiprogramming systems support multiply users

- Response time is shorter

**Disadvantages of multiprogramming systems**

- Tracking all tasks/processes is sometimes difficult to handle
- Due to high load of tasks, long time jobs have to wait long

## Multiprocessing Operating System

In Multiprocessing, Parallel computing is achieved. There are more than one processors present in the system which can execute more than one process at the same time. This will increase the throughput of the system.

**Examples for Symmetric Multiprocessor** – Windows NT, Solaris, Digital UNIX, OS/2 & Linux.

## Advantages of Multiprocessor Systems

1. Enhanced performance
2. Execution of several tasks by different processors concurrently, increases the system's throughput without speeding up the execution of a single task.
3. **High Throughput:** Throughput is the number of processes executed by the CPU at a given time so this type of system has higher throughput.
4. **High Reliability:** As multiple processors share their work between one and another so work is completed with collaboration. That means these systems are reliable.

**Disadvantages of multiprocessor systems:-**

1. **More memory required**
2. **High Chances of Deadlock**
3. **Expensive:** These type of systems are expensive to buy.