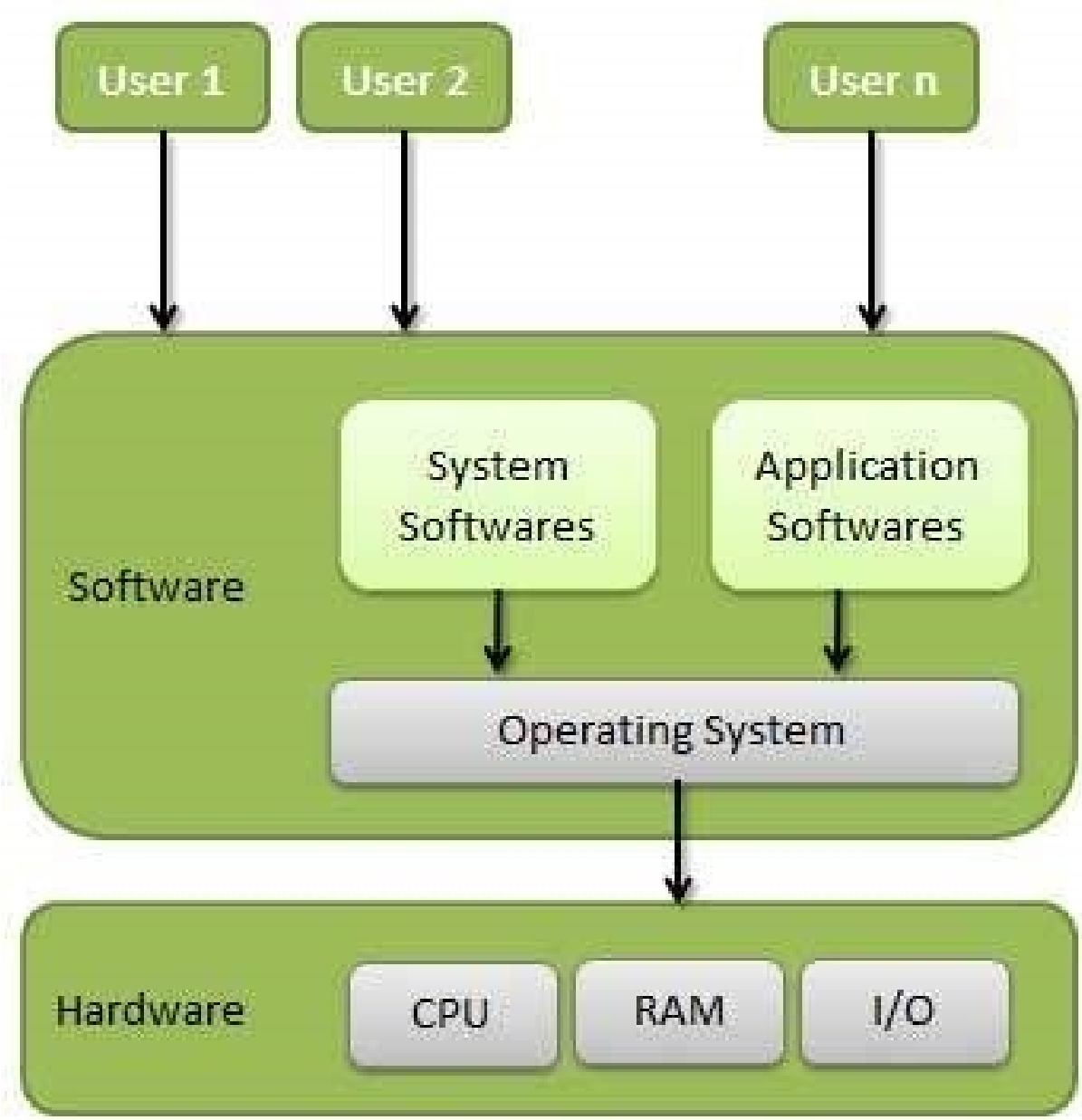
Operating System

Unit – 1

operating system

- An operating system is a software programme required to manage and operate a computing device like smartphones, tablets, computers, etc.,
- An Operating System (OS) is 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.



Some operating system

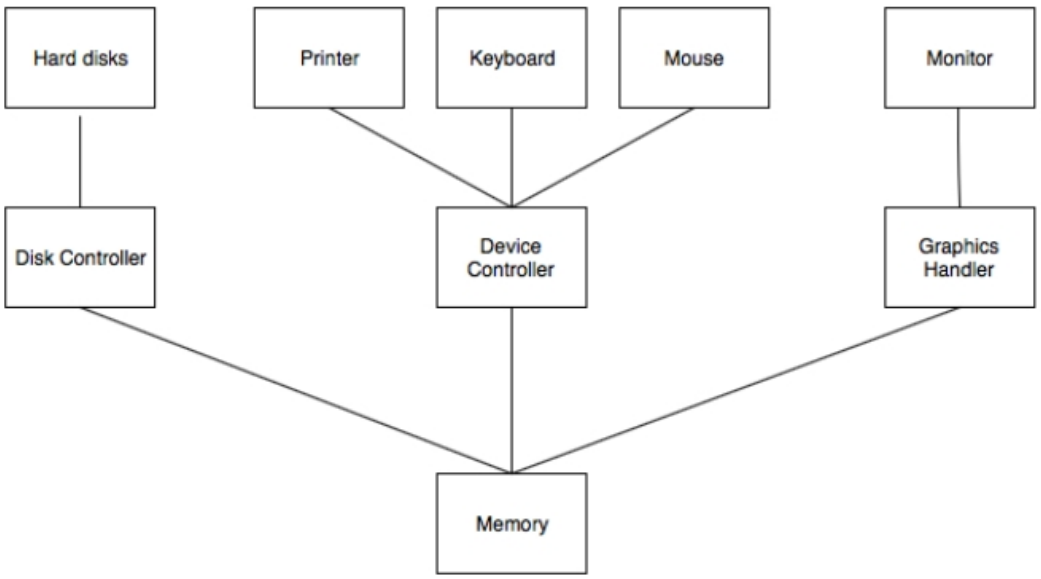
- Some popular Operating Systems include Linux Operating System, Windows Operating System,

Some common operating systems:



Computer System Organisation

The computer system is a combination of many parts such as peripheral devices, secondary memory, CPU etc. This can be explained more clearly using a diagram.



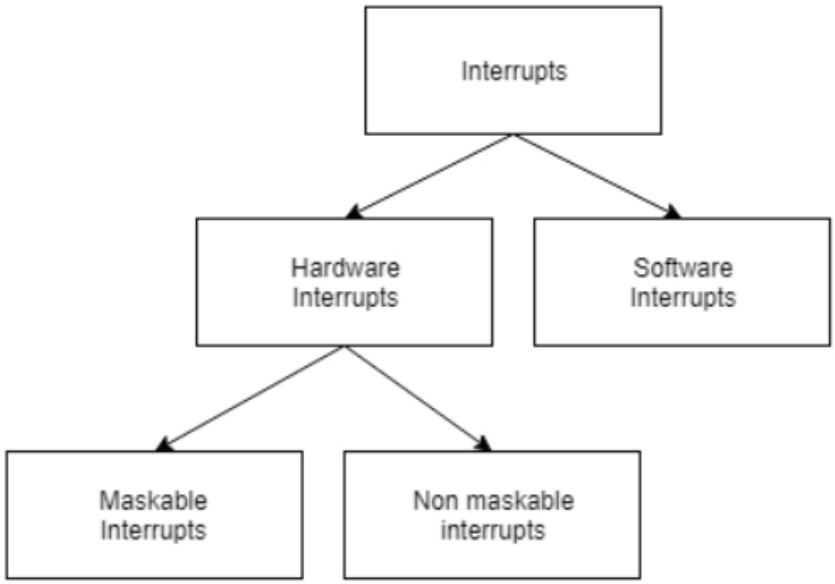
The salient points about the above figure displaying Computer System Organisation is −

- The I/O devices and the CPU both execute concurrently. Some of the processes are scheduled for the CPU and at the same time, some are undergoing input/output operations.
- There are multiple device controllers, each in charge of a particular device such as keyboard, mouse, printer etc.
- There is buffer available for each of the devices. The input and output data can be stored in these buffers.
- The data is moved from memory to the respective device buffers by the CPU for I/O operations and then this data is moved back from the buffers to memory.
- The device controllers use an interrupt to inform the CPU that I/O operation is completed.

Interrupt Handling

An interrupt is a necessary part of Computer System Organisation as it is triggered by hardware and software parts when they need immediate attention.

An interrupt can be generated by a device or a program to inform the operating system to halt its current activities and focus on something else. The types of interrupts are better explained using the following diagram −



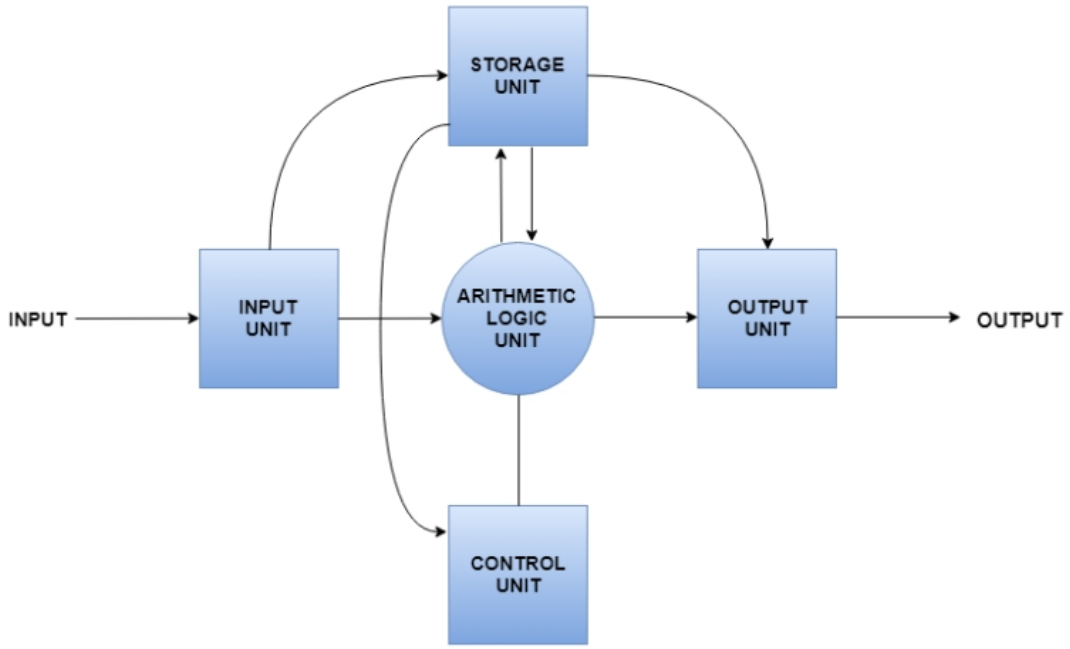
Hardware and software interrupts are two types of interrupts. Hardware interrupts are triggered by hardware peripherals while software interrupts are triggered by software function calls.

Hardware interrupts are of further two types. Maskable interrupts can be ignored or disabled by the CPU while this is not possible for non maskable interrupts.

Computer System Architecture

A computer system is basically a machine that simplifies complicated tasks. It should maximize performance and reduce costs as well as power consumption.The different components in the Computer System Architecture are Input Unit, Output Unit, Storage Unit, Arithmetic Logic Unit, Control Unit etc.

A diagram that shows the flow of data between these units is as follows –



The input data travels from input unit to ALU. Similarly, the computed data travels from ALU to output unit. The data constantly moves from storage unit to ALU and back again. This is because stored data is computed on before being stored again. The control unit controls all the other units as well as their data.

Details about all the computer units are –

- **Input Unit**
  The input unit provides data to the computer system from the outside. So, basically it links the external environment with the computer. It takes data from the input devices, converts it into

machine language and then loads it into the computer system. Keyboard, mouse etc. are the most commonly used input devices.

- **Output Unit**
  The output unit provides the results of computer process to the users i.e it links the computer with the external environment. Most of the output data is the form of audio or video. The different output devices are monitors, printers, speakers, headphones etc.
- **Storage Unit**
  Storage unit contains many computer components that are used to store data. It is traditionally divided into primary storage and secondary storage.Primary storage is also known as the main memory and is the memory directly accessible by the CPU. Secondary or external storage is not directly accessible by the CPU. The data from secondary storage needs to be brought into the primary storage before the CPU can use it. Secondary storage contains a large amount of data permanently.
- **Arithmetic Logic Unit**
  All the calculations related to the computer system are performed by the arithmetic logic unit. It can perform operations like addition, subtraction, multiplication, division etc. The control unit transfers data from storage unit to arithmetic logic unit when calculations need to be performed. The arithmetic logic unit and the control unit together form the central processing unit.
- **Control Unit**
  This unit controls all the other units of the computer system and so is known as its central nervous system. It transfers data throughout the computer as required including from storage unit to central processing unit and vice versa. The control unit also dictates how the memory, input output devices, arithmetic logic unit etc. should behave.

Operating System Structure

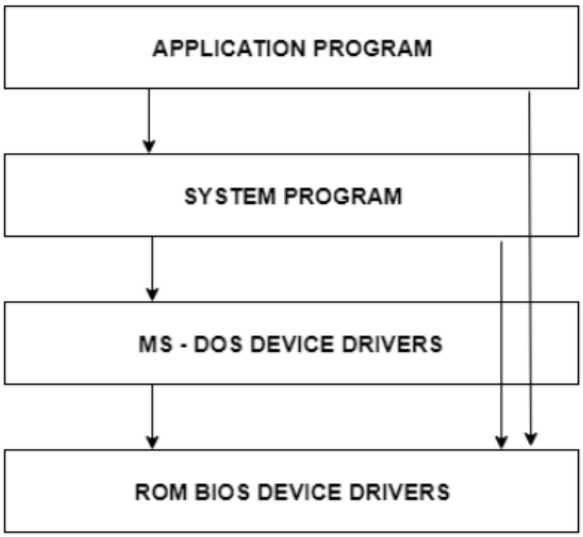
An operating system is a construct that allows the user application programs to interact with the system hardware. Since the operating system is such a complex structure, it should be created with utmost care so it can be used and modified easily. An easy way to do this is to create the operating system in parts. Each of these parts should be well defined with clear inputs, outputs and functions.

Simple Structure

There are many operating systems that have a rather simple structure. These started as small systems and rapidly expanded much further than their scope.

A common example of this is MS-DOS. It was designed simply for a niche amount for people. There was no indication that it would become so popular.

An image to illustrate the structure of MS-DOS is as follows −



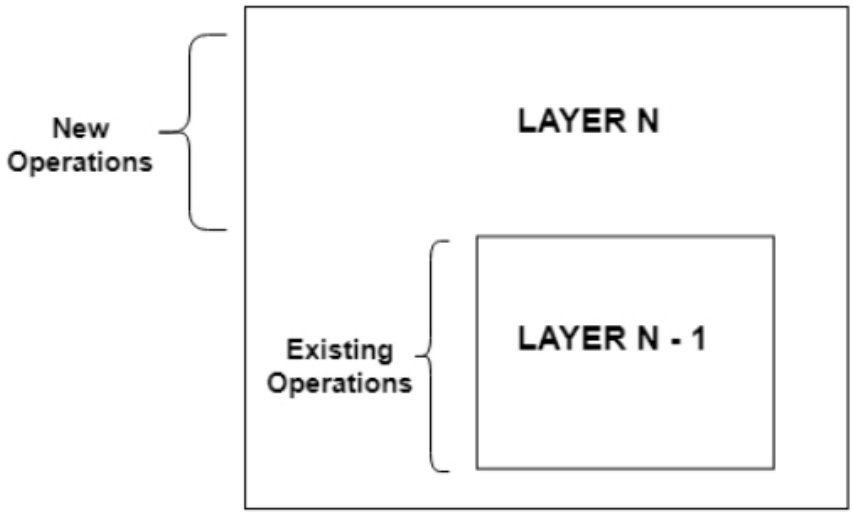
MS-DOS STRUCTURE

It is better that operating systems have a modular structure, unlike MS-DOS. That would lead to greater control over the computer system and its various applications. The modular structure would also allow the programmers to hide information as required and implement internal routines as they see fit without changing the outer specifications.

Layered Structure

One way to achieve modularity in the operating system is the layered approach. In this, the bottom layer is the hardware and the topmost layer is the user interface.

An image demonstrating the layered approach is as follows −



Layered Structure of Operating System

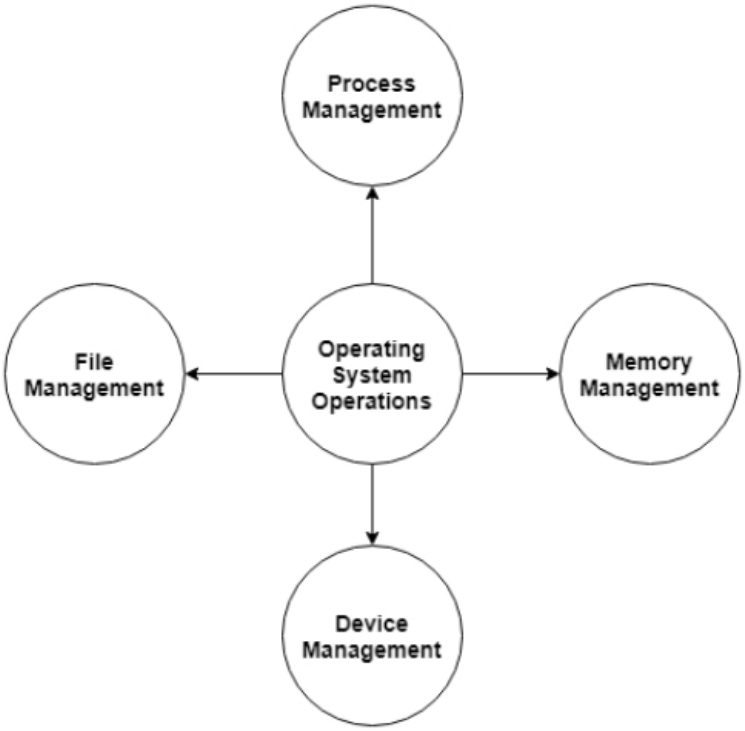
As seen from the image, each upper layer is built on the bottom layer. All the layers hide some structures, operations etc from their upper layers.

One problem with the layered structure is that each layer needs to be carefully defined. This is necessary because the upper layers can only use the functionalities of the layers below them.

Operating System Operations

An operating system is a construct that 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 and file management. These are given in detail as follows:



Process Management

The operating system is responsible for managing the processes i.e assigning the processor to a process at a time. This is 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 i.e. sequential access and direct access −

- **Sequential Access**
  The information in a file is processed in order using sequential access. The files records are accessed on 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.

what operating system do

Following are some of important functions of an operating System.

- Memory Management
- Processor Management
- Device Management
- File Management
- Security
- Control over system performance

- Job accounting
- Error detecting aids
- Coordination between other software and users

Memory Management

Memory management refers to management of Primary Memory or Main Memory. Main memory is a large array of words or bytes where each word or byte has its own address.

Main memory provides a fast storage that can be accessed directly by the CPU. For a program to be executed, it must in the main memory. An Operating System does the following activities for memory management −

- Keeps tracks of primary memory, i.e., what part of it are in use by whom, what part are not in use.
- In multiprogramming, the OS decides which process will get memory when and how much.
- 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.

Processor Management

In multiprogramming environment, the OS decides which process gets the processor when and for how much time. This function is called **process scheduling**. An Operating System does the following activities for processor management −

- 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.

Device Management

An Operating System manages device communication via their respective drivers. It does the following activities for device management −

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

File Management

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions.

An Operating System does the following activities for file management −

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

Other Important Activities

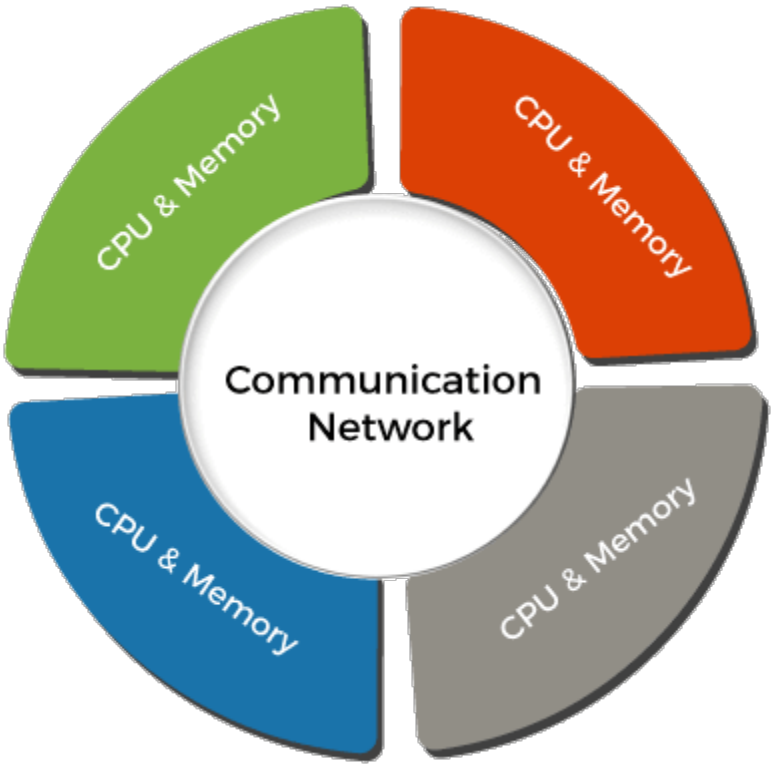
Following are some of the important activities that an Operating System performs −

- **Security** − By means of password and similar other techniques, it prevents unauthorized access to programs and data.
- **Control over system performance** − Recording delays between request for a service and response from the system.
- **Job accounting** − Keeping track of time and resources used by various jobs and users.
- **Error detecting aids** − Production of dumps, traces, error messages, and other debugging and error detecting aids.
- **Coordination between other softwares and users** − Coordination and assignment of compilers, interpreters, assemblers and other software to the various users of the computer systems.

Distributed Operating System

A distributed operating system **(DOS)** is an essential type of operating system. Distributed systems use many central processors to serve multiple real-time applications and users. As a result, data processing jobs are distributed between the processors.

It connects multiple computers via a single communication channel. Furthermore, each of these systems has its own processor and memory. Additionally, these **CPUs** communicate via high-speed buses or telephone lines. Individual systems that communicate via a single channel are regarded as a single entity. They're also known as **loosely coupled systems**.



This operating system consists of numerous computers, nodes, and sites joined together via **LAN/WAN** lines. It enables the distribution of full systems on a couple of center processors, and it supports many real-time products and different users. Distributed operating systems can share their computing resources and I/O files while providing users with virtual machine abstraction.

Types of Distributed Operating System

There are various types of Distributed Operating systems. Some of them are as follows:

1. **Client-Server Systems**
2. **Peer-to-Peer Systems**
3. **Middleware**
4. **Three-tier**
5. **N-tier**

Client-Server System

This type of system requires the client to request a resource, after which the server gives the requested resource. When a client connects to a server, the server may serve multiple clients at the same time.

Client-Server Systems are also referred to as "Tightly Coupled Operating Systems". This system is primarily intended for multiprocessors and homogenous multicomputer. Client-Server Systems function as a centralized server since they approve all requests issued by client systems.

Server systems can be divided into two parts:

1. Computer Server System

This system allows the interface, and the client then sends its own requests to be executed as an action. After completing the activity, it sends a back response and transfers the result to the client.

2. File Server System

It provides a file system interface for clients, allowing them to execute actions like file creation, updating, deletion, and more.

Peer-to-Peer System

The nodes play an important role in this system. The task is evenly distributed among the nodes. Additionally, these nodes can share data and resources as needed. Once again, they require a network to connect.

The Peer-to-Peer System is known as a "Loosely Couple System". This concept is used in computer network applications since they contain a large number of processors that do not share memory or clocks. Each processor has its own local memory, and they interact with one another via a variety of communication methods like telephone lines or high-speed buses.

Middleware

Middleware enables the interoperability of all applications running on different operating systems. Those programs are capable of transferring all data to one other by using these services.

Three-tier

The information about the client is saved in the intermediate tier rather than in the client, which simplifies development. This type of architecture is most commonly used in online applications.

N-tier

When a server or application has to transmit requests to other enterprise services on the network, n-tier systems are used.

Special Purpose Computer Systems

There are various classes of computer systems based upon their computational speed, usage and hardware. The following are some special purpose systems according to specific applications. They use:

1. Real-time embedded systems Multimedia systems
2. Multimedia systems
3. Hand held and portable systems

These are explained as following below.

1. **Real-time embedded systems Multimedia systems:**
   Embedded systems are small computers having a limited set of hardware like a small processor capable of processing a limited set of instructions (often called as an Application Specific Integrated Circuit (ASIC) a small memory (RAM or EPROM) and I/O devices. These systems usually do specific tasks.
   Examples are microwave ovens, robots in a manufacturing unit, latest automotive engines, etc.

   A variety of embedded systems exists of which some are computers with standard operating systems, some have dedicated programs embedded in their limited memories and often some don't even have any software, hardware (ASIC) to do processing. Nearly all embedded systems use real-time operating system because they are used as control devices and have rigid time requirements. Sensors are used to input data such as temperature, air pressure, etc, from the environment to embedded system where that data is analyzed and several other controls are adjusted by embedded system itself to control the situation of system.

   Few examples are home appliance controllers, weapon controllers, boiler temperature controllers, fuel injection system in automobile engines, etc. A real-time system needs that processing to be done in fixed time constraints.

2. **Multimedia systems:**
   Multimedia refers to data of multiple types that includes audio and video including conventional data like text-files, word-processing documents, spreadsheets, etc. It requires that audio and video data must be processed based upon certain time restrictions. This is called as streaming. It is usually 30 frames per second for a video file.

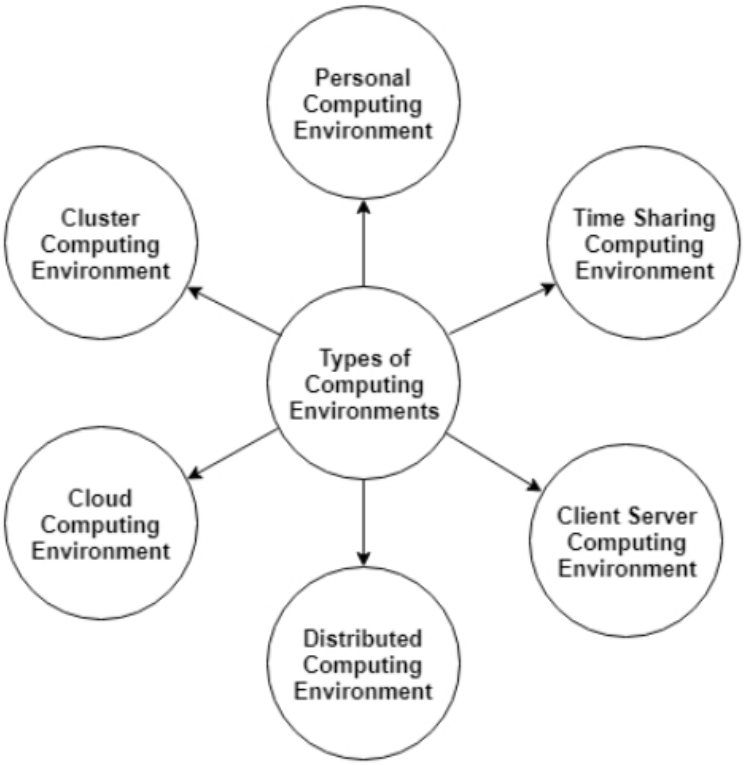
Applications such as video conferencing, movies and clips downloaded over internet, mp3, DVD, VCD playing and recording are examples of various multimedia applications. A multimedia application is usually a combination of both audio and video. The multimedia is not limited to desktop operating system or computers but it is also becoming popular in handheld.

3. **Hand held and portable systems:**
   Hand-held systems refers to small portable devices that can be carried along and are capable of performing normal op erations. They are usually battery powered. Examples include Personal Digital Assistants (PDAs), mobile phones, palm-top computers, pocket-PCs etc. As they are handheld devices, their weights and sizes have certain limitations as a result they are equipped with small memories, slow processors and small display screens, etc. The physical memory capacity is very less (512 KB to 128 MB) hence the operating systems of these devices must manage the memory efficiently. As the processors are slower due to battery problem, the operating system should not burden.

Types of Computing Environments

A computer system uses many devices, arranged in different ways to solve many problems. This constitutes a computing environment where many computers are used to process and exchange information to handle multiple issues.

The different types of Computing Environments are –



Let us begin with Personal Computing Environment −

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.

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.

Client Server Computing Environment

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.

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.

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.

Operating System - Services

An Operating System provides services to both the users and to the programs.

- It provides programs an environment to execute.
- It provides users the services to execute the programs in a convenient manner.

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

- Program execution
- I/O operations
- File System manipulation
- Communication
- Error Detection
- Resource Allocation
- Protection

Program execution

Operating systems handle many kinds of activities from user programs to system programs like printer spooler, name servers, file server, etc. Each of these activities is encapsulated as a process.

A process includes the complete execution context (code to execute, data to manipulate, registers, OS resources in use). 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.

I/O Operation

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.

- 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.

File system manipulation

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.

Communication

In case of distributed systems which are a collection of processors that do not share memory, peripheral devices, or a clock, the operating system manages communications between all the processes. Multiple processes communicate with one another through communication lines in the network.

The OS handles routing and connection strategies, and the problems of contention and security. 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.

Error handling

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.

Resource Management

In case of multi-user or multi-tasking environment, resources such as main memory, CPU cycles and files storage are to be allocated to each user or job. 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.

Protection

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 refers to a mechanism or a way to control the access of programs, processes, or users to the resources defined by a computer system. 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.

What is the user interface and operating system interface?

The user and operating system are connected with each other with the help of interface, so interface is used to connect the user and OS.

In computers there are different types of interface that can be used for connection with computers to users and their connection is responsible for data transfer.

Also, in computers there are different interfaces. These interfaces are not necessarily used but can be used in computers whenever it is needed. So, different types of tasks can be performed by the help of different interfaces.

Command line interface

The command-line interface is an interface whenever the user needs to have different commands regarding the input and output and then a task is performed so this is called the command-line argument and it is used to execute the output and create, delete, print, copy, paste, etc.

All these operations are performed with the help of the command-line interface.

The interface is always connected to the OS so that the command given by the user directly works by the OS and a number of operations can be performed with the help of the command line interface because multiple commands can be interrupted at same time and execute only one.

The command line interface is necessary because all the basic operations in the computer are performed with the help of the OS and it is responsible for memory management. By using this we can divide the memory and we can use the memory.

Command Line Interface advantages −

- Controls OS or application

- faster management
- ability to store scripts which helps in automating regular tasks.
- Troubleshoot network connection issues.

Command Line Interface disadvantages –

- The steeper learning curve is associated with memorizing commands and a complex syntax.
- Different commands are used in different shells.

Graphical user interface

The graphical user interface is used for playing games, watching videos, etc. these are done with the help of GUI because all these applications require graphics.

The GUI is one of the necessary interfaces because only by using the user can clearly see the picture, play videos.

So we need GUI for computers and this can be done only with the help of an operating system.

When a task is performed in the computer then the OS checks the task and defines the interface which is necessary for the task. So, we need GUI in the OS.

The basic components of GUIs are –

- Start menu with program groups
- Taskbar which showing running programs
- Desktop screen
- Different icons and shortcuts.

Choice of interface

The interface that is used with the help of OS for a particular task and that task can be performed with minimum possible time and the output is shown on the screen in that case we use the choice of interface.

The choice of interface means the OS checks the task and finds out which interface can be suitable for a particular task. So that type of interface is called the choice of interface and this can be done with the help of an OS.

System Calls in Operating System

A system call is a way for a user program to interface with the operating system. The program requests several services, and the OS responds by invoking a series of system calls to satisfy the request. A system call can be written in assembly language or a high-level language like **C** or **Pascal**. System calls are predefined

functions that the operating system may directly invoke if a high-level language is used.

In this article, you will learn about the system calls in the operating system and discuss their types and many other things.

What is a System Call?

A system call is a method for a computer program to request a service from the kernel of the operating system on which it is running.

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.

The **Application Program Interface (API):** connects the operating system's functions to user programs. It acts as a link between the operating system and a process, allowing user-level programs to request operating system services. The kernel system can only be accessed using system calls. System calls are required for any programs that use resources.

How are system calls made?

When a computer software needs to access the operating system's kernel, it makes a system call. The system call uses an API to expose the operating system's services to user programs. It is the only method to access the kernel system. All programs or processes that require resources for execution must use system calls, as they serve as an interface between the operating system and user programs.

Below are some examples of how a system call varies from a user function.

1. A system call function may create and use kernel processes to execute the asynchronous processing.
2. A system call has greater authority than a standard subroutine. A system call with kernel-mode privilege executes in the kernel protection domain.
3. System calls are not permitted to use shared libraries or any symbols that are not present in the kernel protection domain.
4. The code and data for system calls are stored in global kernel memory.

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.

How System Calls Work

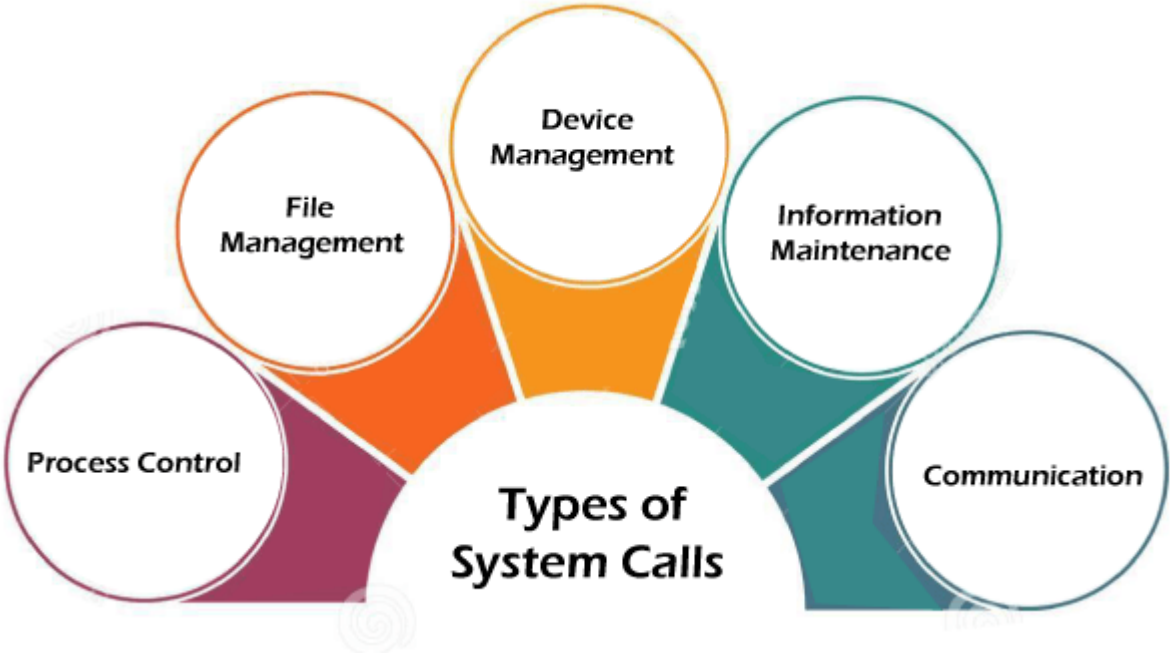
The Applications run in an area of memory known as user space. A system call connects to the operating system's kernel, which executes in kernel space. When an application creates a system call, it must first obtain permission from the kernel. It achieves this using an interrupt request, which pauses the current process and transfers control to the kernel.

If the request is permitted, the kernel performs the requested action, like creating or deleting a file. As input, the application receives the kernel's output. The application resumes the procedure after the input is received. When the operation is finished, the kernel returns the results to the application and then moves data from kernel space to user space in memory.

A simple system call may take few nanoseconds to provide the result, like retrieving the system date and time. A more complicated system call, such as connecting to a network device, may take a few seconds. Most operating systems launch a distinct kernel thread for each system call to avoid bottlenecks. Modern operating systems are multi-threaded, which means they can handle various system calls at the same time.

Types of System Calls

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



1. **Process Control**
2. **File Management**
3. **Device Management**
4. **Information Maintenance**
5. **Communication**

Now, you will learn about all the different types of system calls one-by-one.

Process Control

Process control is the system call that is used to direct the processes. Some process control examples include creating, load, abort, end, execute, process, terminate the process, etc.

File Management

File management is a system call that is used to handle the files. Some file management examples include creating files, delete files, open, close, read, write, etc.

Device Management

Device management is a system call that is used to deal with devices. Some examples of device management include read, device, write, get device attributes, release device, etc.

Information Maintenance

Information maintenance is a system call that is used to maintain information. There are some examples of information maintenance, including getting system data, set time or date, get time or date, set system data, etc.

Communication

Communication is a system call that is used for communication. There are some examples of communication, including create, delete communication connections, send, receive messages, etc.

Examples of Windows and Unix system calls

There are various examples of Windows and Unix system calls. These are as listed below in the table:

Process	Windows
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()
File Manipulation	CreateFile() ReadFile() WriteFile() CloseHandle()
Device Management	SetConsoleMode() ReadConsole() WriteConsole()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()

Protection	SetFileSecurity() InitializeSecurityDescriptor() SetSecurityDescriptorgroup()

Here, you will learn about some methods briefly:

open()

The **open()** system call allows you to access a file on a file system. It allocates resources to the file and provides a handle that the process may refer to. Many processes can open a file at once or by a single process only. It's all based on the file system and structure.

read()

It is used to obtain data from a file on the file system. It accepts three arguments in general:

- A file descriptor.
- A buffer to store read data.
- The number of bytes to read from the file.

The file descriptor of the file to be read could be used to identify it and open it using **open()** before reading.

wait()

In some systems, a process may have to wait for another process to complete its execution before proceeding. When a parent process makes a child process, the parent process execution is suspended until the child process is finished. The **wait()** system call is used to suspend the parent process. Once the child process has completed its execution, control is returned to the parent process.

write()

It is used to write data from a user buffer to a device like a file. This system call is one way for a program to generate data. It takes three arguments in general:

- A file descriptor.
- A pointer to the buffer in which data is saved.

- The number of bytes to be written from the buffer.

fork()

Processes generate clones of themselves using the **fork()** system call. It is one of the most common ways to create processes in operating systems. When a parent process spawns a child process, execution of the parent process is interrupted until the child process completes. Once the child process has completed its execution, control is returned to the parent process.

close()

It is used to end file system access. When this system call is invoked, it signifies that the program no longer requires the file, and the buffers are flushed, the file information is altered, and the file resources are de-allocated as a result.

exec()

When an executable file replaces an earlier executable file in an already executing process, this system function is invoked. As a new process is not built, the old process identification stays, but the new process replaces data, stack, data, head, etc.

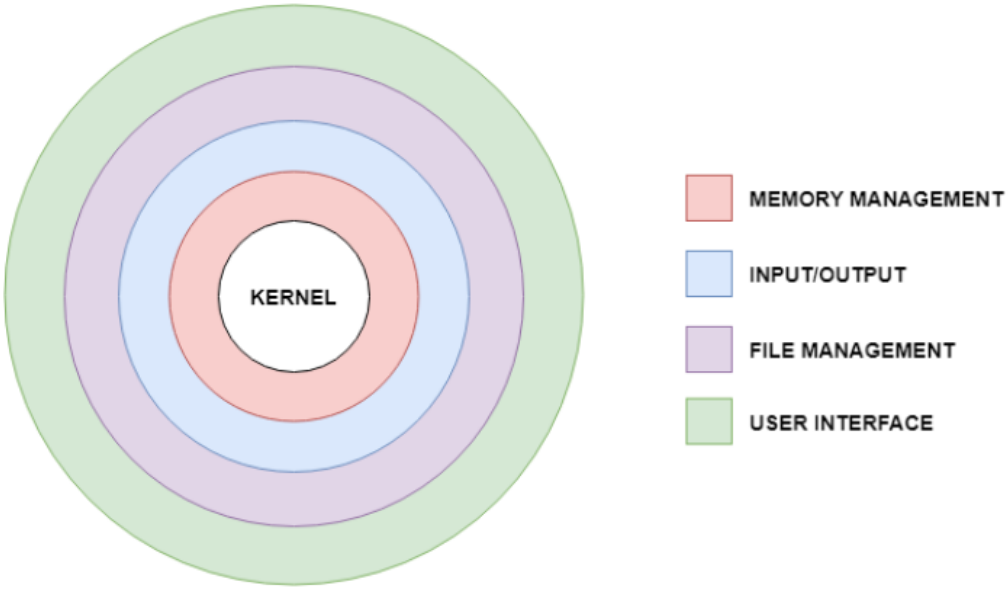
exit()

The **exit()** is a system call that is used to end program execution. This call indicates that the thread execution is complete, which is especially useful in multi-threaded environments. The operating system reclaims resources spent by the process following the use of the **exit()** system function.

Operating System Design and Implementation

An operating system is a construct that 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.

There are many problems that can occur while designing and implementing an operating system. These are covered in operating system design and implementation.



Layered Operating System Design

Operating System Design Goals

It is quite complicated to define all the goals and specifications of the operating system while designing it.The design changes depending on the type of the operating system i.e if it is batch system, time shared system, single user system, multi user system, distributed system etc.

There are basically two types of goals while designing an operating system. These are −

User Goals

The operating system should be convenient, easy to use, reliable, safe and fast according to the users. However, these specifications are not very useful as there is no set method to achieve these goals.

System Goals

The operating system should be easy to design, implement and maintain. These are specifications required by those who create, maintain and operate the operating system. But there is not specific method to achieve these goals as well.

Operating System Implementation

The operating system needs to be implemented after it is designed. Earlier they were written in assembly language but now higher level languages are used. The first system not written in assembly language was the Master Control Program (MCP) for Burroughs Computers.

Advantages of Higher Level Language

There are multiple advantages to implementing an operating system using a higher level language such as: the code is written more fast, it is compact and also easier to debug and understand. Also, the operating system can be easily moved from one hardware to another if it is written in a high level language.

Disadvantages of Higher Level Language

Using high level language for implementing an operating system leads to a loss in speed and increase in storage requirements. However in modern systems only a small amount of code is needed for high performance, such as the CPU scheduler and memory manager. Also, the bottleneck routines in the system can be replaced by assembly language equivalents if required.

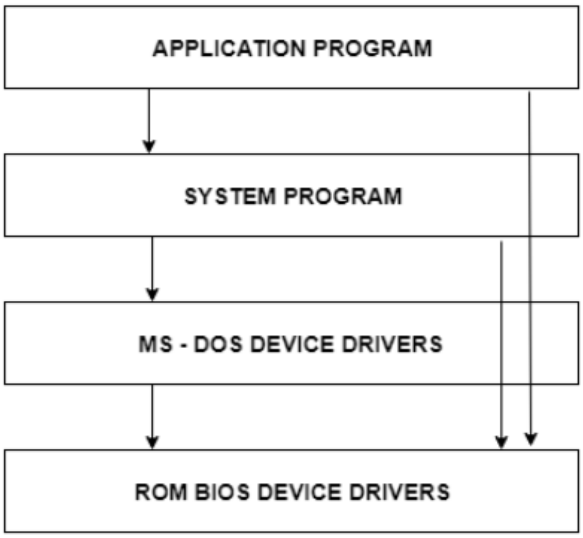
Operating System Structure

An operating system is a construct that allows the user application programs to interact with the system hardware. Since the operating system is such a complex structure, it should be created with utmost care so it can be used and modified easily. An easy way to do this is to create the operating system in parts. Each of these parts should be well defined with clear inputs, outputs and functions.

Simple Structure

There are many operating systems that have a rather simple structure. These started as small systems and rapidly expanded much further than their scope. A common example of this is MS-DOS. It was designed simply for a niche amount for people. There was no indication that it would become so popular.

An image to illustrate the structure of MS-DOS is as follows −



MS-DOS STRUCTURE

It is better that operating systems have a modular structure, unlike MS-DOS. That would lead to greater control over the computer system and its various applications. The modular structure would also allow the programmers to hide information as required and implement internal routines as they see fit without changing the outer specifications.

Virtual Machines in Operating System

Virtual Machine abstracts the hardware of our personal computer such as CPU, disk drives, memory, NIC (Network Interface Card) etc, into many different execution environments as per our requirements, hence giving us a feel that each execution environment is a single computer. For example, VirtualBox.
When we run different processes on an operating system, it creates an illusion that each process is running on a different processor having its own virtual memory, with the help of CPU scheduling and virtual-memory techniques. There are additional features of a process that cannot be provided by the hardware alone like system calls and a file system. The virtual machine approach does not provide these additional functionalities but it only provides an interface that is same as basic hardware. Each process is provided with a virtual copy of the underlying computer system.

We can create a virtual machine for several reasons, all of which are fundamentally related to the ability to share the same basic hardware yet can also support different execution environments, i.e., different operating systems simultaneously.

The main drawback with the virtual-machine approach involves disk systems. Let us suppose that the physical machine has only three disk drives but wants to support seven virtual machines. Obviously, it cannot allocate a disk

drive to each virtual machine, because virtual-machine software itself will need substantial disk space to provide virtual memory and spooling. The solution is to provide virtual disks.

Users are thus given their own virtual machines. After which they can run any of the operating systems or software packages that are available on the underlying machine. The virtual-machine software is concerned with multi-programming multiple virtual machines onto a physical machine, but it does not need to consider any user-support software. This arrangement can provide a useful way to divide the problem of designing a multi-user interactive system, into two smaller pieces.

Advantages:

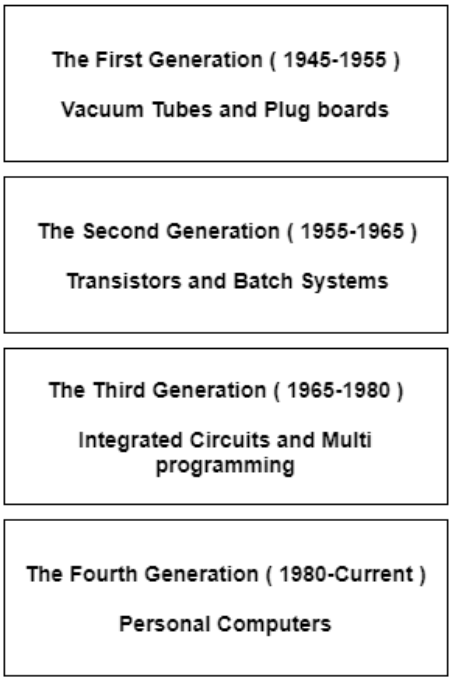
1. There are no protection problems because each virtual machine is completely isolated from all other virtual machines.
2. Virtual machine can provide an instruction set architecture that differs from real computers.
3. Easy maintenance, availability and convenient recovery.

Disadvantages:

1. When multiple virtual machines are simultaneously running on a host computer, one virtual machine can be affected by other running virtual machines, depending on the workload.
2. Virtual machines are not as efficient as a real one when accessing the hardware.

Operating System Generations

Operating Systems have evolved over the years. So, their evolution through the years can be mapped using generations of operating systems. There are four generations of operating systems. These can be described as follows –



OPERATING SYSTEM GENERATIONS

The First Generation (1945 - 1955): Vacuum Tubes and Plugboards

Digital computers were not constructed until the second world war. Calculating engines with mechanical relays were built at that time. However, the mechanical relays were very slow and were later replaced with vacuum tubes. These machines were enormous but were still very slow.

These early computers were designed, built and maintained by a single group of people. Programming languages were unknown and there were no operating systems so all the programming was done in machine language. All the problems were simple numerical calculations.

By the 1950's punch cards were introduced and this improved the computer system. Instead of using plugboards, programs were written on cards and read into the system.

The Second Generation (1955 - 1965): Transistors and Batch Systems

Transistors led to the development of the computer systems that could be manufactured and sold to paying customers. These machines were known as mainframes and were locked in air-conditioned computer rooms with staff to operate them.

The Batch System was introduced to reduce the wasted time in the computer. A tray full of jobs was collected in the input room and read into the magnetic tape. After that, the tape was rewound and mounted on a tape drive. Then the batch operating system was loaded in which read the first job from the tape and ran it. The output was written on the second tape. After the whole batch was done, the input and output tapes were removed and the output tape was printed.

The Third Generation (1965 - 1980): Integrated Circuits and Multiprogramming

Until the 1960's, there were two types of computer systems i.e the scientific and the commercial computers. These were combined by IBM in the System/360. This used integrated circuits and provided a major price and performance advantage over the second generation systems.

The third generation operating systems also introduced multiprogramming. This meant that the processor was not idle while a job was completing its I/O operation. Another job was scheduled on the processor so that its time would not be wasted.

The Fourth Generation (1980 - Present): Personal Computers

Personal Computers were easy to create with the development of large-scale integrated circuits. These were chips containing thousands of transistors on a square centimeter of silicon. Because of these, microcomputers were much cheaper than minicomputers and that made it possible for a single individual to own one of them.

The advent of personal computers also led to the growth of networks. This created network operating systems and distributed operating systems. The users were aware of a network while using a network operating system and could log in to remote machines and copy files from one machine to another.

Booting in Operating System

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.

Restarting a computer also is called rebooting, which can be "***hard***", e.g., after electrical power to the CPU is switched from off to on, or "***soft***", where the power is not cut. On some systems, a soft boot may optionally clear RAM to zero. Hard and soft booting can be initiated by hardware such as a button press or a

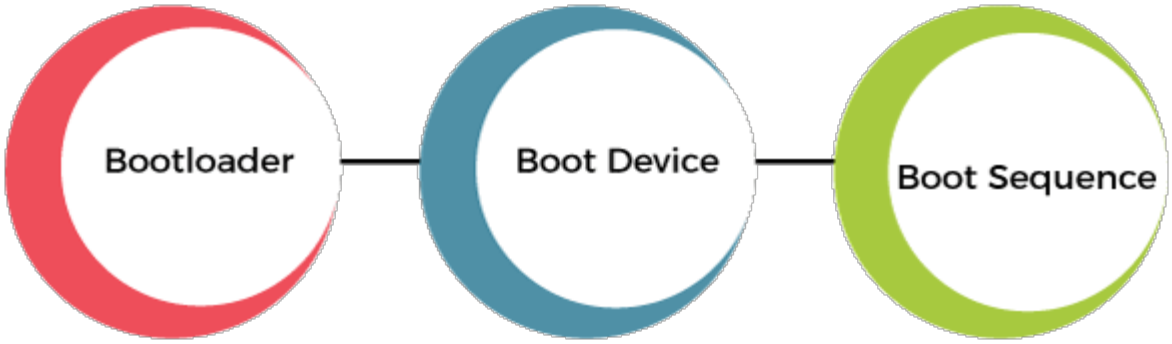
software command. Booting is complete when the operative runtime system, typically the operating system and some applications, is attained.

The process of returning a computer from a state of sleep does not involve booting; however, restoring it from a state of hibernation does. Minimally, some embedded systems do not require a noticeable boot sequence to begin functioning and, when turned on, may run operational programs that are stored in ROM. All computer systems are state machines and a reboot may be the only method to return to a designated zero-state from an unintended, locked state.

In addition to loading an operating system or stand-alone utility, the boot process can also load a storage dump program for diagnosing problems in an operating system.

Sequencing of Booting

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.



1. Boot Loader: Computers powered by the central processing unit can only execute code found in the system's memory. Modern operating systems and application program code and data are stored on nonvolatile memories. When a computer is first powered on, it must initially rely only on the code and data stored in nonvolatile portions of the system's memory. The operating system is not really loaded at boot time, and the computer's hardware cannot perform many complex systems actions.

The program that starts the chain reaction that ends with the entire operating system being loaded is the boot loader or bootstrap loader. The boot loader's only job is to load other software for the operating system to start.

2. Boot Devices: 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 include the local hard disk drive, optical drive, floppy drive, a network interface card, and 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

The BIOS will try to boot from the CD drive first, and if that fails, then it will try to boot from the hard disk drive, and if that fails, then it will try to boot from the network, and if that fails, then it won't boot at all.

3. Boot Sequence: There is a standard boot sequence that all personal computers use. First, the CPU runs an instruction in memory for the BIOS. That instruction contains a jump instruction that transfers to the BIOS start-up program. This program runs a power-on self-test (POST) to check that devices the computer will rely on are functioning properly. Then, the BIOS goes through the configured boot sequence until it finds a bootable device. Once BIOS has found a bootable device, BIOS loads the bootsector and transfers execution to the boot sector. If the boot device is a hard drive, it will be a master boot record (MBR).

The MBR code checks the partition table for an active partition. If one is found, the MBR code loads that partition's boot sector and executes it. The boot sector is often operating system specific, and however, in most operating systems, its main function is to load and execute the operating system kernel, which continues start-up. Suppose there is no active partition, or the active partition's boot sector is invalid. In that case, the MBR may load a secondary boot loader which will select a partition and load its boot sector, which usually loads the corresponding operating system kernel.

Types of Booting

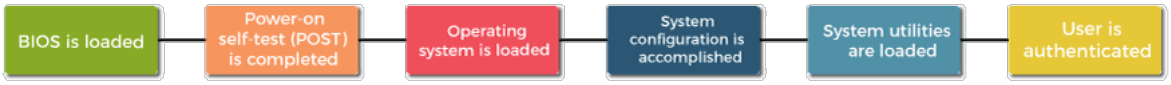
There are two types of booting in an operating system.



1. **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.
2. **Warm Booting:** Warm or Hot Booting process is when computer systems come to no response or hang state, and then the system is allowed to restart during on condition. It is also referred to as rebooting. 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. The system requires a reboot to set software or hardware configuration changes, or 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. Else, in some systems, the external reset button may be available to reboot the system.

Booting Process in Operating System

When our computer is switched on, it can be started by hardware such as a button press, or by software command, a computer's central processing unit (CPU) has no software in its main memory, there is some process which must load software into main memory before it can be executed. Below are the six steps to describe the boot process in the operating system, such as:



Step 1: Once the computer system is turned on, ***BIOS*** (Basic Input /Output System) performs a series of activities or functionality tests on programs stored

in ROM, called on ***POST*** (Power-on Self Test) that checks to see whether peripherals in the system are in perfect order or not.

Step 2: After the BIOS is done with pre-boot activities or functionality test, it read bootable sequence from ***CMOS*** (Common Metal Oxide Semiconductor) and looks for master boot record in the first physical sector of the bootable disk as per boot device sequence specified in ***CMOS***. For example, if the boot device sequence is:

- Floppy Disk
- Hard Disk
- CDROM

Step 3: After this, the master boot record will search first in a floppy disk drive. If not found, then the hard disk drive will search for the master boot record. But if the master boot record is not even present on the hard disk, then the CDROM drive will search. If the system cannot read the master boot record from any of these sources, ROM displays **"*No Boot device found*"** and halted the system. On finding the master boot record from a particular bootable disk drive, the operating system loader, also called Bootstrap loader, is loaded from the boot sector of that bootable drive· into memory. A bootstrap loader is a special program that is present in the boot sector of a bootable drive.

Step 4: The bootstrap loader first loads the ***IO.SYS*** file. After this, ***MSDOS.SYS*** file is loaded, which is the core file of the DOS operating system.

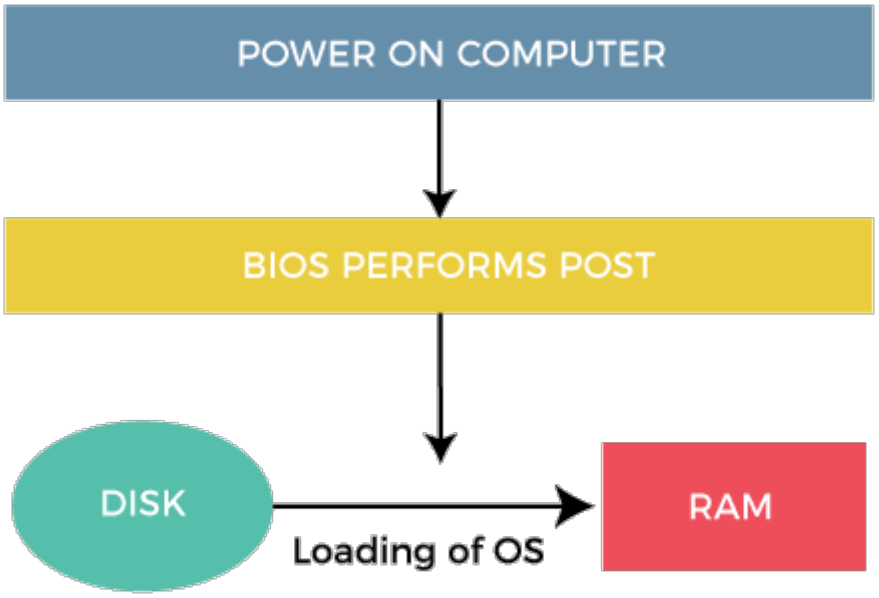
Step 5: After this, ***MSDOS.SYS*** file searches to find Command Interpreter in ***CONFIG.SYS*** file, and when it finds, it loads into memory. If no Command Interpreter is specified in the ***CONFIG.SYS*** file, the ***COMMAND.COM*** file is loaded as the default Command Interpreter of the DOS operating system.

Step 6: The last file is to be loaded and executed is the ***AUTOEXEC.BAT*** file that contains a sequence of DOS commands. After this, the prompt is displayed. We can see the drive letter of bootable drive displayed on the computer system, which indicates that the operating system has been successfully on the system from that drive.

What is Dual Booting

When two operating systems are installed on the computer system, then it is called dual booting. Multiple operating systems can be installed on such a system. But to know which operating system is to boot, a boot loader that

understands multiple file systems and multiple operating systems can occupy the boot space.



Once loaded, it can boot one of the operating systems available on the disk. The disk can have multiple partitions, each containing a different type of operating system. When a computer system turns on, a boot manager program displays a menu, allowing the user to choose the operating system to use.