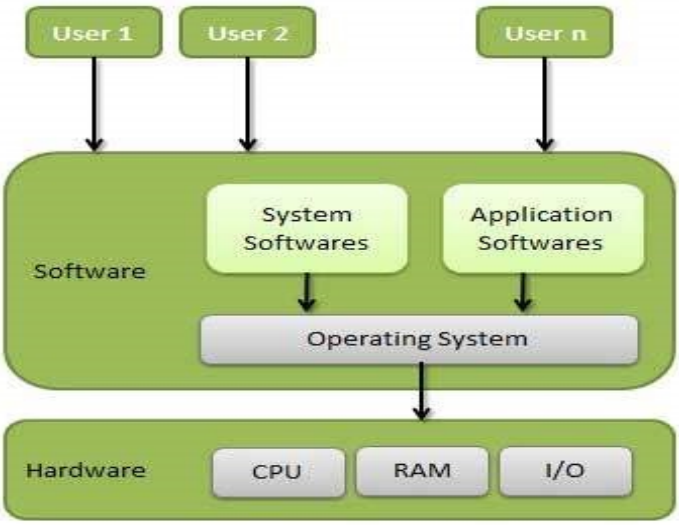
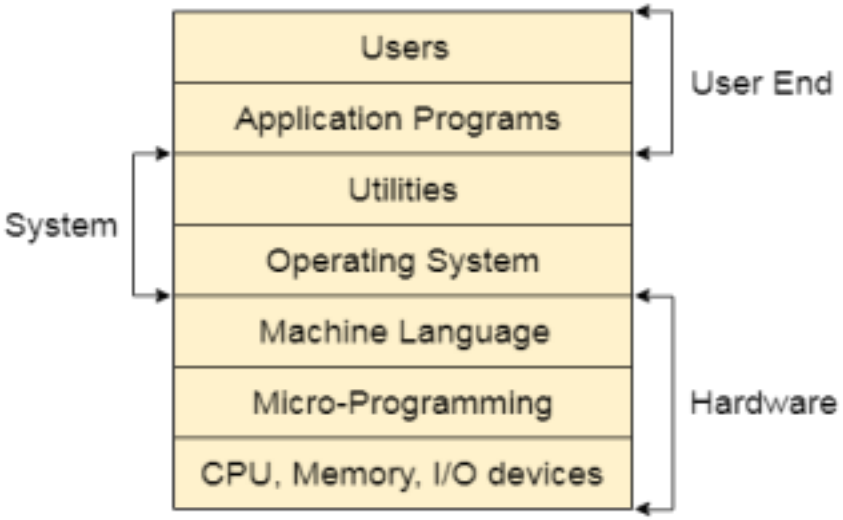
An Operating System (OS) is a software that acts as an interface between computer hardware components and the user.



Structure of a Computer System :

A Computer System consists of:

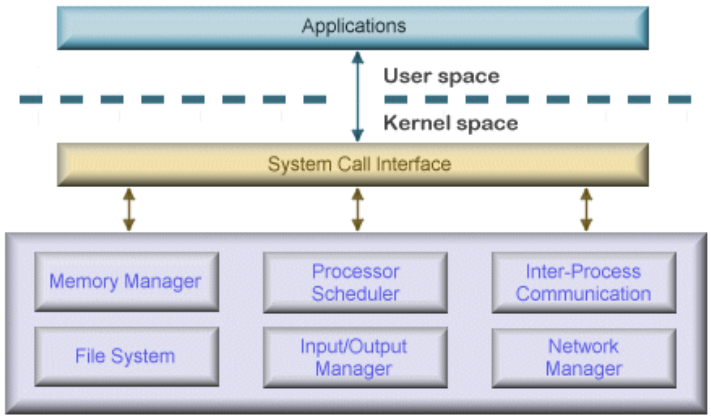
- o Users (people who are using the computer)
- o Application Programs (Compilers, Databases, Games, Video player, Browsers, etc.)
- o System Programs (Shells, Editors, Compilers, etc.)
- o Operating System (A special program which acts as an interface between user and hardware)
- o Hardware (CPU, Disks, Memory, etc)



Functions of Operating System :

- User Interface
- File and Disk Management
- CPU Scheduling
- Input/Output Management
- Memory Management
- Process Synchronisation
- Process management
- Security

Structure of Os :



Gaining the role of Operating systems :

Security – The operating system uses password protection to protect user data and similar other techniques. It also prevents unauthorised access to programs and user data.

Control over system performance : Monitors overall system health to help improve performance.

Error detecting aids – The operating system constantly monitors the system to detect errors and avoid the malfunctioning of a computer system.

Coordination between other software and users – Operating systems also coordinate and assign interpreters, compilers, assemblers, and other software to the various users of the computer systems.

Memory Management – The operating system manages the Primary Memory or Main Memory.

Processor Management – In a multiprogramming environment, the OS decides the order in which processes have access to the processor, and how much processing time each process has.

Device Management – Keeps track of all devices connected to the system. Deallocates devices when they are no longer required.

File Management – A file system is organised into directories for efficient or easy navigation and usage.Os keeps track of where information is stored, user access settings and status of every file, and more.

Operating System Types

• **Batch os :** Os does not interact directly with the computer. There is an operator which takes similar jobs having the same requirement and groups them into batches. It is the responsibility of the operator to sort jobs with similar needs.
Example: Payroll System,Bank Statements

• **Multitasking os/Time-Sharing os:** Os implements CPU scheduling and multi programming systems which deliver to every user a small piece of operating time.
Example: UNIX, Multics, Linux, Windows 2000 server,etc.

• **Multiprogramming os:** More than one program/process running at a time, it increases CPU utilisation by organising jobs so that the CPU always has one to execute.The motive is to keep multiple jobs in main memory. There is 1 cpu.

• **Clustered os (for local area network)**
• **Distributed os(for relatively large area network):** Os allows distribution of entire systems on the couples of cpu having there own RAM/main memory, and it serves on the multiple real time products as well as multiple users.
Example: Windows Server 2003, Windows Server 2008, Windows Server 2012, Ubuntu, Linux(Apache Server),etc.

• **Embedded os:** Specialised os designed to perform a specific task for embedded hardwares in computer system.They are designed to work on dedicated devices like ATM, aeroplane systems, digital home assistants, and IoT devices.

• **Real Time os :** A real-time operating system is an operating system for real-time applications that processes data and events that have defined time constraints.

There are 3 types of real time OS :

Hard Real Time os : These types of Os are used for tasks which require completing critical tasks within the defined time limit. If the response time is more, the system fails or the system does not accept the response and utility of operation/task becomes zero.

Eg: Air Traffic Control, Medical System

Soft Real Time os : A soft real-time system is a less restrictive system that can accept response delays and the system does not fail by delays and the utility of operation is degraded but does not become zero.

Eg:Multimedia Transmission and Reception, Computer Games

Context Switching:-It is a method to store/restore the state of a CPU in a PCB. So that process execution can be resumed from the same point at a later time. The context switching method is important for multitasking OS.

Types of operating system on basis of no of processors:-

i) UniProcessor(one cpu)

ii) Multiprocessor(more than 1 cpu) : Can run more than 1 process at same time parallely.

Types of Multiprocessor:-

Symmetric multiprocessor :- One os controls all cpu , each cpu has equal rights. All the cpu are in a pair to pair relationship.

Asymmetric multiprocessor:- There is a master multiprocessor that runs tasks of os and gives instructions to other processors(slaves).

Difference between multicore system and multiprocessor system :

Multicore system	Multiprocessor system
A single CPU or processor with more than one processing unit called cores that are capable of reading and executing program instructions.	A system with more than one CPU's that allows simultaneous processing of programs.
It executes a single program faster.	It executes multiple programs faster.
Not as reliable as a multiprocessor.	More reliable since failure in one CPU will not affect another.
It is very cheap and requires no configuration.	It is expensive as many CPUs are needed and requires a complex configuration.

Kernel : A Kernel is the central/core component of an operating system. Kernel acts as a bridge/interface between applications and hardware using inter-process communication and system calls while os acts as an interface between user and hardware .
It basically manages operations of memory and CPU time.

Types of Kernel :

1. Monolithic Kernel : It is one type of kernel where all operating system services operate in kernel space. It has dependencies between system components.

example: Unix, Linux, Open VMS, XTS-400 etc.

2. Microkernel: It has virtual memory and thread scheduling. It is more stable with less services in kernel space. It puts rest things in user space.A microkernel is much smaller in size than a conventional kernel and supports only the near-minimum amount of software that can provide the mechanisms needed to implement an operating system (OS). It does not provide os services/system services.
Example: Mach, L4, AmigaOS, Minix, K42 etc.

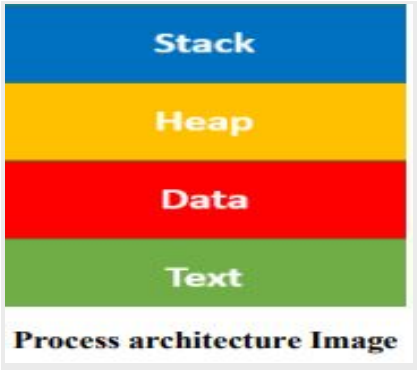
3. Hybrid Kernel: This Kernel is what we see most. Windows, Apple's macOS. It is the combination of both monolithic kernel and microkernel. It has speed and design of monolithic kernel and modularity and stability of microkernel.
Example: Windows NT, Netware, BeOS etc

4. Nano Kernel: IIt is the type of kernel that offers hardware abstraction but without system services. Micro Kernel also does not have system services therefore the Microkernel and Nano Kernel have become almost the same.
Example: EROS etc.

5. Exo Kernel: It is the type of kernel which follows end-to-end principle. It has fewest hardware abstractions as possible. It gives physical resources to applications and there is more work for application developers.
Example: Nemesis, ExOS etc.

Process : A process is a program(set of instructions to perform a specific task) in execution state.
Process requires high resources and process switching needs interaction with the OS.

Process Architecture :



Process architecture Image

- **Stack :** The Stack stores temporary data like function parameters, returns addresses, and local variables.
- **Heap :** Allocates memory, which may be processed during its run time.
- **Data :** It contains the variable.
- **Text :** Text Section includes the current activity, which is represented by the value of the Program Counter.

Process management : Process management involves various tasks like/Activities performed by os via process management :
Process creation : Creation of new process.
Process scheduling : Handles the removal of the running process from the CPU and the selection of another process.
Process synchronisation : It also protects the resources of each process from other methods and allows synchronisation among processes.
Inter Process communication : OS allocate resources that enable processes to share and exchange information
Termination of processes : termination of process
Deadlock Prevention.

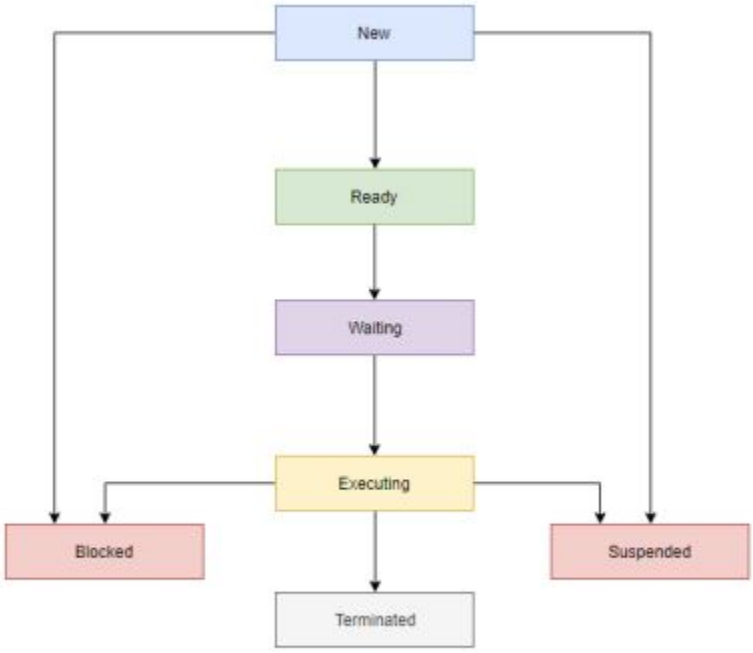
Process Control Block (PCB)/ Task Control Block : A process control block (PCB) is a data structure used by the OS to store the information related to each process .
Each process will be given a PCB which is a kind of identification card for a process by which Os manages operations like process scheduling.

Process-Id
Process state
Process Priority
Accounting Information
Program Counter
CPU Register
PCB Pointers
.....

Process Control Block

The most important components of PCB Process Control Block (PCB) is :

• **Process state:** State of the process at a certain time. It also defines the current position of the process.



Process States Diagram

There are mainly seven states/stages of a process which are:

• **New:** The new process is created when a specific program calls from secondary memory/ hard disk to primary memory/ RAM a

• **Ready:** In a ready state, the process should be loaded into the primary memory, which is ready for execution.

• **Waiting:** The process is waiting for the allocation of CPU time and other resources for execution.

• **Executing:** The process is in execution state.

• **Blocked:** It is a time interval when a process is waiting for an event like I/O operations to complete.

• **Suspended:** Suspended state defines the time when a process is ready for execution but has not been placed in the ready queue by OS.

• **Terminated:** Terminated state specifies the time when a process is terminated After completing every step, all the resources are used by a process, and memory becomes free.

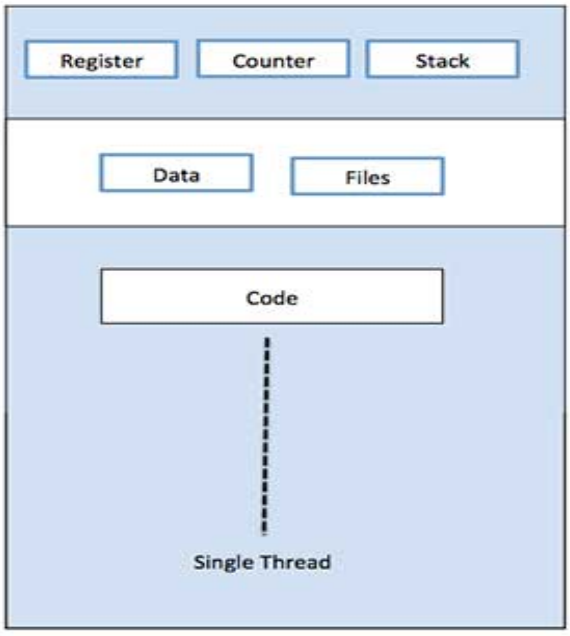
Threads/Lightweight processes :

Thread is a unit of process.which consists of its own 1) program counter, 2) stack and 3) registers , i,e, one process can have multiple threads.

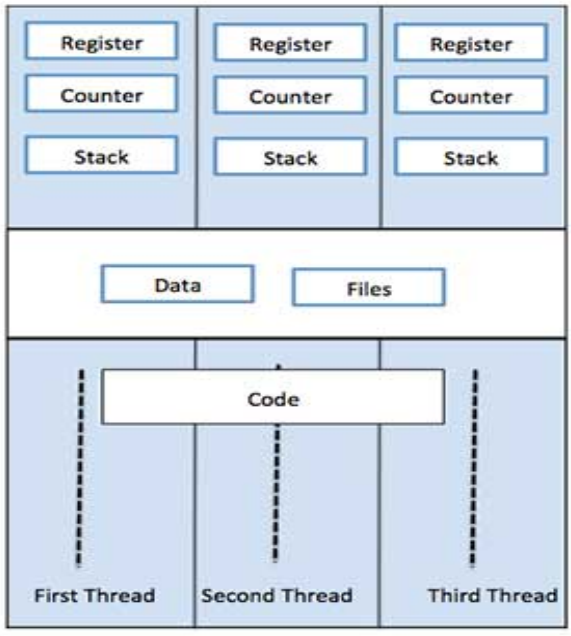
Thread requires less resources and thread switching does not need to interact with the OS.
Multiple processes without using threads use more resources as each process has its own resources.

Types of Thread :
- **User threads :** Implemented by user. It is not recognized by the OS. These are the threads that application programmers use in their programs.
- **Kernel threads** : Implemented by OS. It is recognized by the OS.

Multi threading : Use of multi thread provides execution of multiple items at same time within a process. Executing multiple threads independently but sharing process resources at the same time within a process.

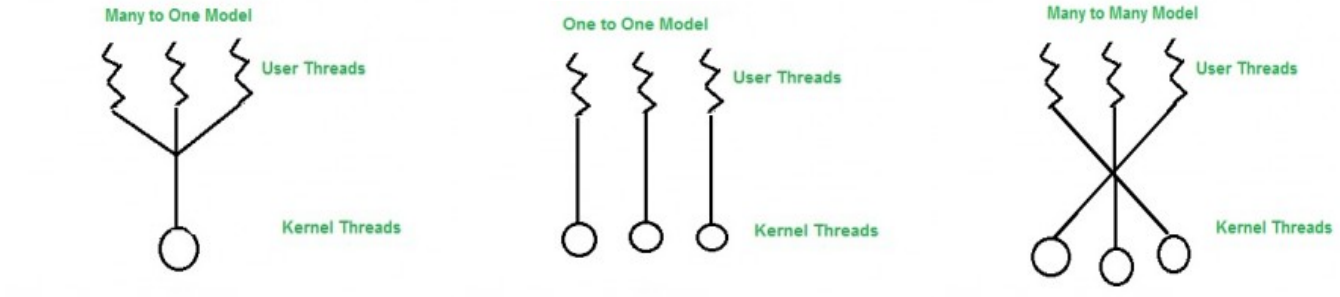


Single Process P with single thread



Single Process P with three threads

Multi threading Models : The user threads must be mapped to kernel threads, by one of the following strategies :

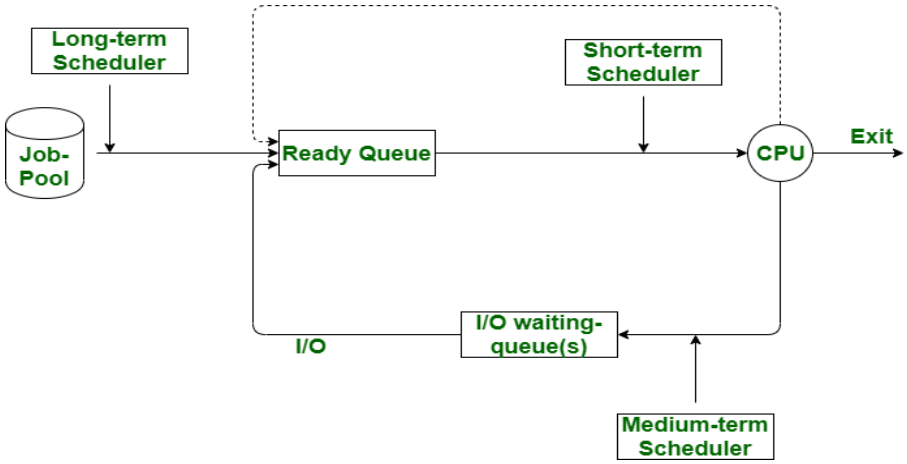


Many to One Model : Multiple user threads mapped to one kernel thread. Multiple threads are not able to access multiprocessor at the same time. The thread management is done on the user level so it is more efficient.

One to One Model : Each user thread mapped with one kernel thread.In this model multiple threads can run on multiple processors. If any user thread makes a blocking system call, the other user threads won't be blocked.

Many to Many Model : User threads are mapped with the same or less number of kernel threads.System doesn't block if a particular thread is blocked.

Process Scheduling :- Activity of the process manager that handles the removal of the running process from the CPU and the selection of another process.



Types of Schedulers :-

1) Long term Schedulers:-It brings the new process to the 'Ready State' by selecting them from the queue.
2) Short term Schedulers :- It selects one process from the ready state for scheduling it on the running state.Dispatcher is responsible for loading the process selected by Short-term scheduler on the CPU (Ready to Running State)
3) Medium term Schedulers :- It is responsible for suspending and resuming the process. It mainly does swapping (moving processes from main memory to disk and vice versa).

Operations on Process :

1)Process Creation :

Processes need to be created in the system for different operations.This can be done by the following events −

- User request for process creation
- System initialization
- Execution of a process creation system call by a running process
- Batch job initialization

A process may be created by another process using fork(). The creating process is called the parent process and the created process is the child process. A child process can have only one parent but a parent process may have many children.

2)Process Preemption :

An interrupt mechanism is used in preemption that suspends the process executing currently and the next process to execute is determined by the short-term scheduler. Preemption makes sure that all processes get some CPU time for execution.

3)Process Blocking :

The process is blocked if it is waiting for some event to occur. This event may beI/O as the I/O events are executed in the main memory and don't require the processor. After the event is complete, the process again goes to the ready state.

4)Process Termination :

After the process has completed the execution of its last instruction, it is terminated. The resources held by a process are released after it is terminated. When a parent process is terminated, its child processes are terminated as well.

System calls in operating System :-

A system call is a way for a user program to interact with the operating system.A system call is a request from computer software to an operating system's kernel. 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.

Types of System Calls:

There are commonly five types of system calls.

Process Control:

Process control is the system call that is used to direct the processes. Some process control examples include

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.

wait() : 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.

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.

File Management :-
File management is a system call that is used to handle the files. Some file Management examples include creating files, delete files.

Device Management :
Device management is a system call that is used to deal with devices. Some examples of device management include read, device,

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.

Communication:
Communication is a system call that is used for communication. There are some examples of communication, including create, delete

Interprocess Communication :-

Interprocess communication is the mechanism provided by the operating system that allows processes to communicate with each other.
These are a few different approaches for Inter- Process Communication:
1. Pipes
2. Shared Memory
3. Message Queue
4. Direct Communication
5. Indirect communication
6. Message Passing
7. FIFO

Pipe:-
The pipe is a type of data channel that is unidirectional in nature. It means that the data in this type of data channel can be moved in only a single direction at a time. Still, one can use two-channels of this type, so that he is able to send and receive data in two processes. Typically, it uses the standard methods for input and output. These pipes are used in all types of POSIX systems and in different versions of window operating systems as well.

Shared Memory:-
It can be referred to as a type of memory that can be used or accessed by multiple processes simultaneously. It is primarily used so that the processes can communicate with each other. Therefore the shared memory is used by almost all POSIX and Windows operating systems as well.

Message Queue:-
In general, several different messages are allowed to read and write the data to the message queue. In the message queue, the messages are stored or stay in the queue unless their recipients retrieve them. In short, we can also say that the message queue is very helpful in inter-process communication and used by all operating systems.

Message Passing:-
It is a type of mechanism that allows processes to synchronise and communicate with each other. However, by using the message passing, the processes can communicate with each other without restoring the

shared variables.
Usually, the inter-process communication mechanism provides two operations that are as follows:
- o send (message)
- o received (message)

Process Synchronisation :

Process Synchronisation means coordinating the execution of processes such that no two processes access the same shared resources and data. It is required in a multi-process system where multiple processes run together.

For Example, process A changing the data in a memory location while another process B is trying to read the data from the same memory location. There is a high probability that data read by the second process will be erroneous.

Sections of a Program :

• Entry Section: It is part of the process which decides the entry of a particular process.
• Critical Section: This part allows one process to enter and modify the shared variable.
• Exit Section: Exit section allows the other process that are waiting in the Entry Section, to enter into the Critical Sections. It also checks that a process that finished its execution should be removed through this Section.
• Remainder Section: All other parts of the Code, which is not in Critical, Entry, and Exit Section, are known as the Remainder Section.

Critical Section :

Critical Section is the part of a program which tries to access shared resources. The critical section cannot be executed by more than one process at the same time; operating system faces the difficulties in allowing and disallowing the processes from entering the critical section.

Rules for Critical Section:

The critical section need to must enforce all three rules:
• Mutual Exclusion:Not more than one process can execute in its critical section at one time.
• Progress: This solution is used when no one is in the critical section, and someone wants in. Then those processes not in their reminder section should decide who should go in, in a finite time.
• Bound Waiting: When a process makes a request for getting into critical section, there is a specific limit about the number of processes that can get into their critical section. So, when the limit is reached, the system must allow requests to the process to get into its critical section.