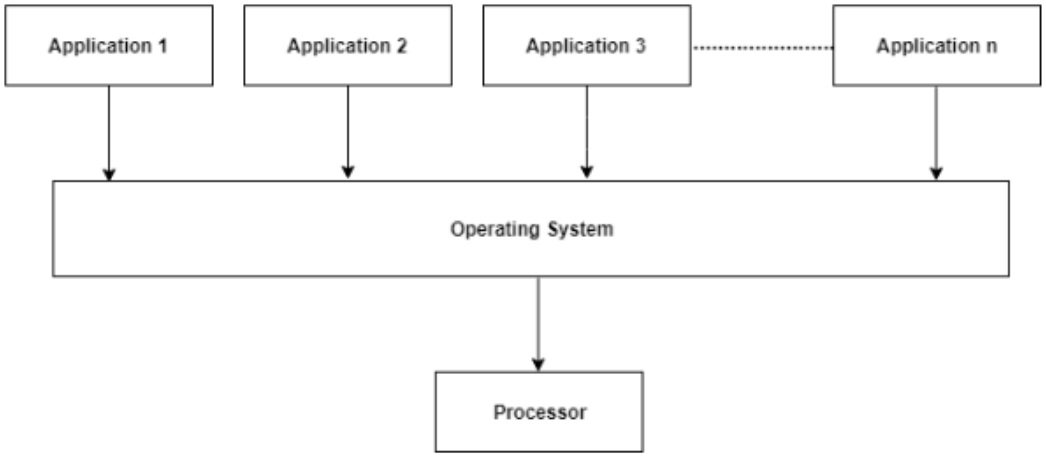
Computer-System Architecture

A computer system may be organized in a number of different ways, which we can categorize roughly according to the number of general-purpose processors used.

1. **Single-Processor Systems**

   Most systems vise a single processor. a single-processor system, there is one main CPU capable of executing a general-purpose instruction set, including instructions from user processes. Almost all systems have other special-purpose processors as well. They may come in the form of device-specific processors, such as disk, keyboard, and graphics controllers; or, on mainframes, they may come in the form of more general-purpose processors, such as I/O processors that move data rapidly among the components of the system.
   All of these special-purpose processors run a limited instruction set and do not run user processes. Sometimes they are managed by the operating system, in that the operating system sends them information about their next task and monitors their status.

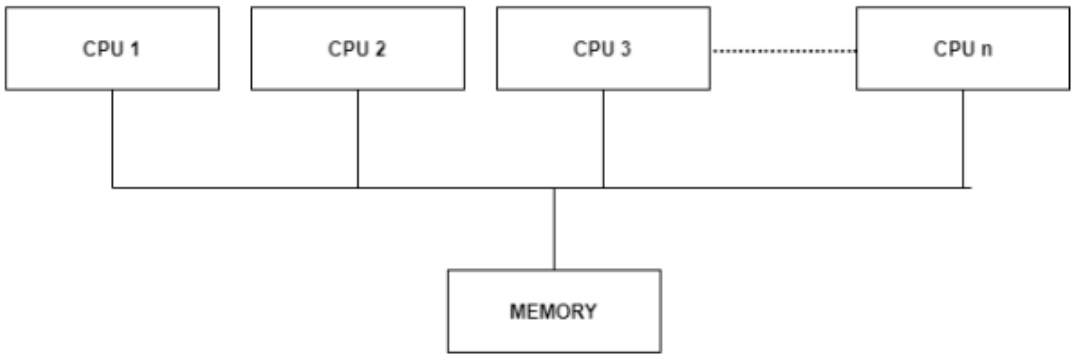


Single Processor System

2. **Multiprocessor systems** (also known as parallel systems or tightly coupled systems) are growing in importance. Such systems have two or more

processors in close communication, sharing the computer bus and sometimes the clock, memory, and peripheral devices.

Multi processor or parallel systems are increasing in importance nowadays. These systems have multiple processors working in parallel that share the computer clock, memory, bus, peripheral devices etc.



Multiprocessing Architecture

Advantages of Multiprocessor Systems

There are multiple advantages to multiprocessor systems. Some of these are −

More reliable Systems

In a multiprocessor system, even if one processor fails, the system will not halt. This ability to continue working despite hardware failure is known as graceful degradation. For example: If there are 5 processors in a multiprocessor system and one of them fails, then also 4 processors are still working. So the system only becomes slower and does not ground to a halt.

Enhanced Throughput

If multiple processors are working in tandem, then the throughput of the system increases i.e. number of processes getting executed per unit of time increase. If there are N processors then the throughput increases by an amount just under N.

More Economic Systems

Multiprocessor systems are cheaper than single processor systems in the long run because they share the data storage, peripheral devices, power supplies etc. If there are multiple processes that share data, it is better to schedule them on multiprocessor systems with shared data than have different computer systems with multiple copies of the data.

Disadvantages of Multiprocessor Systems

There are some disadvantages as well to multiprocessor systems. Some of these are:

Increased Expense

Even though multiprocessor systems are cheaper in the long run than using multiple computer systems, still they are quite expensive. It is much cheaper to buy a simple single processor system than a multiprocessor system.

Complicated Operating System Required

There are multiple processors in a multiprocessor system that share peripherals, memory etc. So, it is much more complicated to schedule processes and impart resources to processes.than in single processor systems. Hence, a more complex and complicated operating system is required in multiprocessor systems.

Large Main Memory Required

All the processors in the multiprocessor system share the memory. So a much larger pool of memory is required as compared to single processor systems.

Types of Multiprocessors

There are mainly two types of multiprocessors i.e. symmetric and asymmetric multiprocessors. Details about them are as follows −

Symmetric Multiprocessors

In these types of systems, each processor contains a similar copy of the operating system and they all communicate with each other. All the processors are in a peer to peer relationship i.e. no master - slave relationship exists between them.

An example of the symmetric multiprocessing system is the Encore version of Unix for the Multimax Computer.

Asymmetric Multiprocessors

In asymmetric systems, each processor is given a predefined task. There is a master processor that gives instruction to all the other processors. Asymmetric multiprocessor system contains a master slave relationship.

Asymmetric multiprocessor was the only type of multiprocessor available before symmetric multiprocessors were created. Now also, this is the cheaper option.

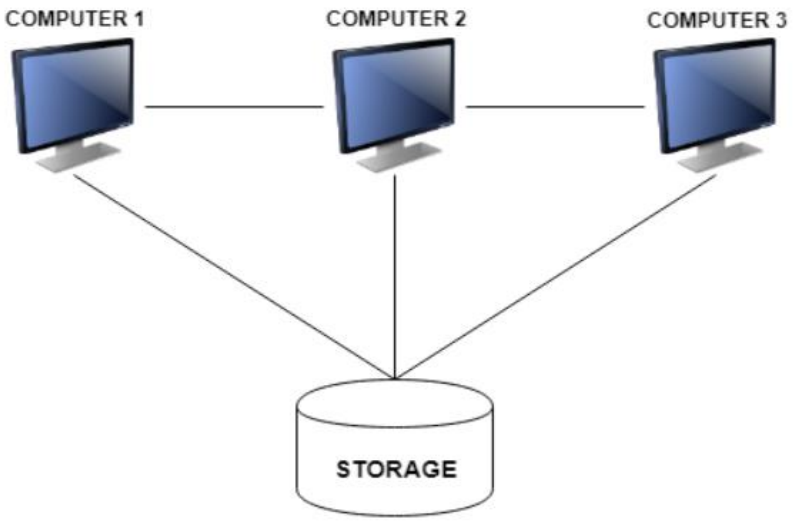
Differences Between Single Processor and Multiprocessor Systems

- Single processor system contains only one processor while multiprocessor systems may contain two or more processors.
- It is easier to design a single processor system as compared to a multiprocessor system.
- Throughput of a multiprocessor system is more than a single processor system.
- Single processor systems are less reliable than multiprocessor systems because if the processor fails for some reason then system cannot work. In multiprocessor systems, even if one processor fails than the rest of the processors can pick up the slack.
- Single processor systems can be more expensive than multiprocessor systems. If n processor multiprocessor system is available, it is cheaper than n different single processor systems because the memory, peripherals etc. are shared.

3. Clustered System

Clustered systems are similar to parallel systems 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. Basically, they have independent computer systems with a common storage and the systems work together.

A diagram to better illustrate this is −



The clustered systems are a combination of hardware clusters and software clusters. The hardware clusters help in sharing of high performance disks between the systems. The software clusters makes all the systems work together .

Each node in the clustered systems contains the cluster software. This software monitors the cluster system and makes sure it is working as required. If any one of

the nodes in the clustered system fail, then the rest of the nodes take control of its storage and resources and try to restart.

Types of Clustered Systems

There are primarily two types of clustered systems i.e. asymmetric clustering system and symmetric clustering system. Details about these are given as follows −

Asymmetric Clustering System

In this system, one of the nodes in the clustered system is in hot standby mode and all the others run the required applications. The hot standby mode is a failsafe in which a hot standby node is part of the system . The hot standby node continuously monitors the server and if it fails, the hot standby node takes its place.

Symmetric Clustering System

In symmetric clustering system two or more nodes all run applications as well as monitor each other. This is more efficient than asymmetric system as it uses all the hardware and doesn't keep a node merely as a hot standby.

Attributes of Clustered Systems

There are many different purposes that a clustered system can be used for. Some of these can be scientific calculations, web support etc. The clustering systems that embody some major attributes are –

Load Balancing Clusters

In this type of clusters, the nodes in the system share the workload to provide a better performance. For example: A web based cluster may assign different web queries to different nodes so that the system performance is optimized. Some clustered systems use a round robin mechanism to assign requests to different nodes in the system.

- **High Availability Clusters**

  These clusters improve the availability of the clustered system. They have extra nodes which are only used if some of the system components fail. So, high availability clusters remove single points of failure i.e. nodes whose failure leads to the failure of the system. These types of clusters are also known as failover clusters or HA clusters.

Benefits of Clustered Systems

The difference benefits of clustered systems are as follows −

- **Performance**

Clustered systems result in high performance as they contain two or more individual computer systems merged together. These work as a parallel unit and result in much better performance for the system.

- **Fault Tolerance**

  Clustered systems are quite fault tolerant and the loss of one node does not result in the loss of the system. They may even contain one or more nodes in hot standby mode which allows them to take the place of failed nodes.

- **Scalability**

  Clustered systems are quite scalable as it is easy to add a new node to the system. There is no need to take the entire cluster down to add a new node.

Types of Operating System

Operating systems are there from the very first computer generation and they keep evolving with time, some of the important types of operating systems which are most commonly used are:

1.Batch Processing System

In this approach similar jobs were submitted to the CPU for processing and were run together.

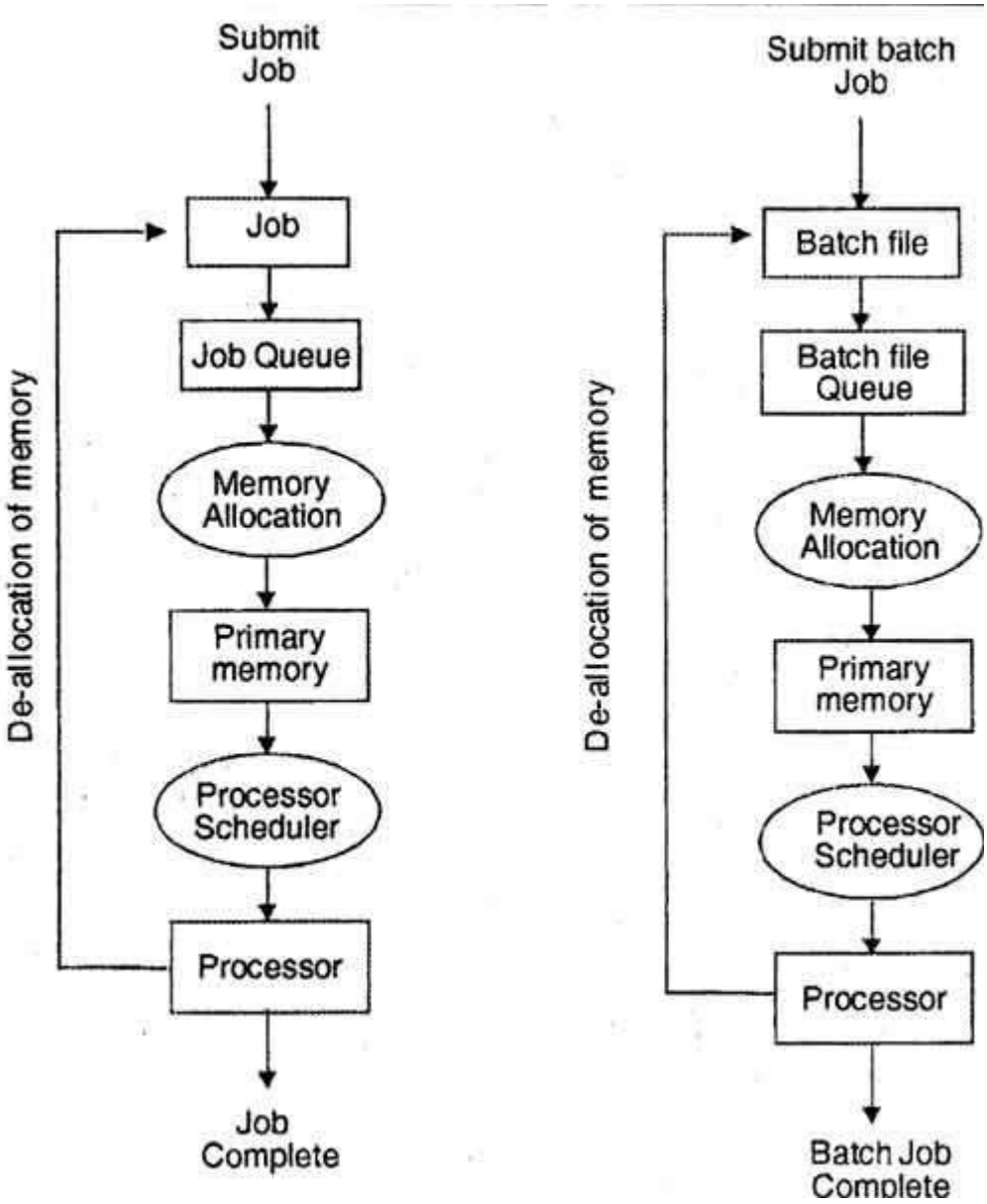
The main function of a batch processing system is to automatically keep executing the jobs in a batch. This is the important task of a batch processing system i.e. performed by the 'Batch Monitor' resided in the low end of main memory.

This technique was possible due to the invention of hard-disk drives and card readers. Now the jobs could be stored on the disk to create the pool of jobs for its execution as a batch. First the pooled jobs are read and executed by the batch monitor, and then these jobs are grouped; placing the identical jobs (jobs with the similar needs) in the same batch, So, in the batch processing system, the batched jobs were executed automatically one after another saving its time by performing the activities (like loading of compiler) only for once. It resulted in improved system utilization due to reduced turn around time.

In the early job processing systems, the jobs were placed in a job queue and the memory allocate or managed the primary memory space, when space was available in the main memory, a job was selected from the job queue and was loaded into memory.

Once the job loaded into primary memory, it competes for the processor. When the processor became available, the processor scheduler selects job that was loaded in the memory and execute it.

In batch strategy is implemented to provide a batch file processing. So in this approach files of the similar batch are processed to speed up the task.



The problems with Batch Systems are as follows −

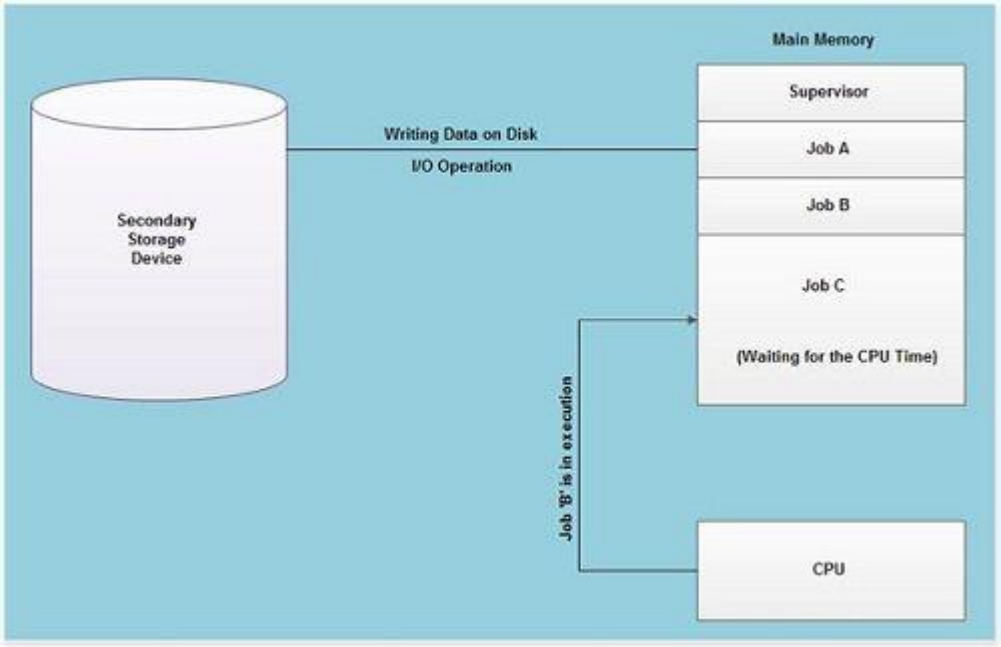
- Lack of interaction between the user and the job.
- CPU is often idle, because the speed of the mechanical I/O devices is slower than the CPU.

- Difficult to provide the desired priority.

2.Multiprogramming Operating System

To overcome the problem of under utilization of CPU and main memory, the multi-programming was introduced. The multi-programming is interleaved execution of multiple jobs by the same computer.

In multi-programming system, when one program is waiting for I/O transfer; there is another program ready to utilize the CPU. So it is possible for several jobs to share the time of the CPU. But it is important to note that multi-programming is not defined to be the execution of jobs at the same instance of time. Rather it does mean that there are a number of jobs available to the CPU (placed in main memory) and a portion of one is executed then a segment of another and so on. A simple process of multi-programming is shown in figure



As shown in fig, at the particular situation, job' A' is not utilizing the CPU time because it is busy in I/ 0 operations. Hence the CPU becomes busy to execute the job 'B'. Another job C is waiting for the CPU for getting its execution time. So in this state the CPU will never be idle and utilizes maximumofitstime.

A program in execution is called a "Process", "Job" or a "Task". The concurrent execution of programs improves the utilization of system resources and enhances the system throughput as compared to batch and serial processing. In this system, when a process requests some I/O to allocate; meanwhile the CPU time is

assigned to another ready process. So, here when a process is switched to an I/O operation, the CPU is not set idle.

Advantages

- High and efficient CPU utilization.
- User feels that many programs are allotted CPU almost simultaneously.

Disadvantages

- CPU scheduling is required.
- To accommodate many jobs in memory, memory management is required.

3.Distributed operating System

Distributed systems use multiple central processors to serve multiple real-time applications and multiple users. Data processing jobs are distributed among the processors accordingly.

The processors communicate with one another through various communication lines (such as high-speed buses or telephone lines). These are referred as loosely coupled systems or distributed systems. Processors in a distributed system may vary in size and function. These processors are referred as sites, nodes, computers, and so on.

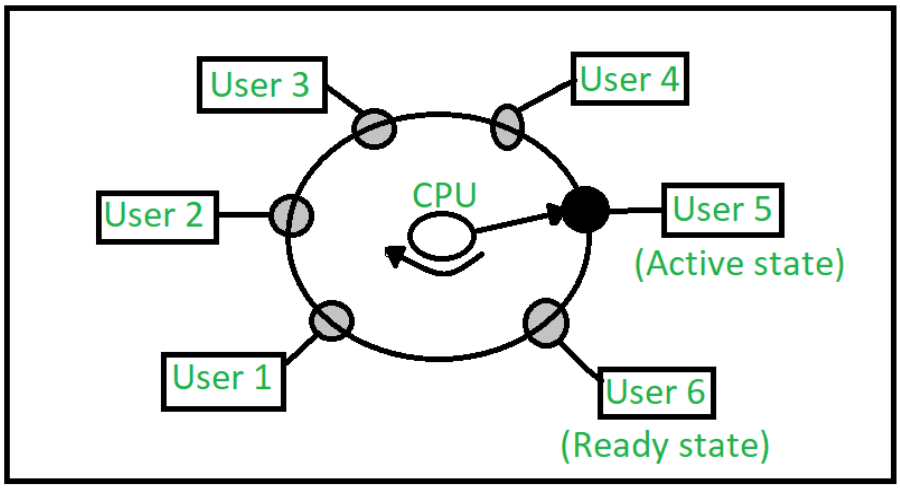
The advantages of distributed systems are as follows −

- With resource sharing facility, a user at one site may be able to use the resources available at another.
- Speedup the exchange of data with one another via electronic mail.
- If one site fails in a distributed system, the remaining sites can potentially continue operating.
- Better service to the customers.
- Reduction of the load on the host computer.
- Reduction of delays in data processing.

4.Time shared operating system

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

milliseconds. Time shared operating systems are more complex than multiprogrammed operating systems. In both, multiple jobs must be kept in memory simultaneously, so the system must have memory management and security. To achieve a good response time, jobs may have to swap in and out of disk from main memory which now serves as a backing store for main memory. A common method to achieve this goal is virtual memory, a technique that allows the execution of a job that may not be completely in memory.



In above figure the user 5 is active state but user 1, user 2, user 3, and user 4 are in waiting state whereas user 6 is in ready state.

- Active State –The user's program is under the control of CPU. Only one program is available in this state.
- Ready State –The user program is ready to execute but it is waiting for for it's turn to get the CPU. More than one user can be in ready state at a time.
- Waiting State –The user's program is waiting for some input/output operation. More than one user can be in a waiting state at a time.

Requirements of Time Sharing Operating System :
An alarm clock mechanism to send an interrupt signal to the CPU after every time slice. Memory Protection mechanism to prevent one job's instructions and data from interfering with other jobs.

Advantages :
Each task gets an equal opportunity.
Less chances of duplication of software.
CPU idle time can be reduced.

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

5.Real Time operating System

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, robots, air traffic control systems, etc.

There 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

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 directories. 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.

Components of Operating Systems

An operating system is a large and complex system that can only be created by partitioning into small pieces. These pieces should be a well-defined portion of the system, which carefully defined inputs, outputs, and functions.

Although Mac, Unix, Linux, Windows, and other OS do not have the same structure, most of the operating systems share similar OS system components like File, Process, Memory, I/O device management.

File Management

A file is a set of related information which is should define by its creator. It commonly represents programs, both source and object forms, and data. Data files can be numeric, alphabetic, or alphanumeric.

Function of file management in OS:

The operating system has the following important given activities in connections with file management:

- File and directory creation and deletion.
- For manipulating files and directories.
- Mapping files onto secondary storage.
- Backup files on stable storage media.

Process Management

The process management component is a procedure for managing the many processes that are running simultaneously on the operating system. Every software application program has one or more processes associated with them when they are running.

For example, when you use a browser like Google Chrome, there is a process running for that browser program. The OS also has many processes running, which performing various functions.

All these processes should be managed by process management, which keeps processes for running efficiently. It also uses memory allocated to them and shutting them down when needed.

The execution of a process must be sequential so, at least one instruction should be executed on behalf of the process.

Functions of process management in OS:

The following are functions of process management.

- Process creation and deletion.
- Suspension and resumption.
- Synchronization process
- Communication process

I/O Device Management

One of the important use of an operating system that helps you to hide the variations of specific hardware devices from the user.

Functions of I/O management in OS:

- It offers buffer caching system
- It provides general device driver code
- It provides drivers for particular hardware devices.
- I/O helps you to knows the individualities of a specific device.

Network Management

Network management is the process of administering and managing computer networks. It includes performance management, fault analysis, provisioning of networks, and maintaining the quality of service.

A distributed system is a collection of computers/processors that never share their own memory or a clock. In this type of system, all the processors have their local Memory, and the processors communicate with each other using different communication lines, like fiber optics or telephone lines.

The computers in the network are connected through a communication network, which can be configured in a number of different ways. With the help of network management, the network can be fully or partially connected, which helps users to design routing and connection strategies that overcome connection and security issues.

Functions of Network management:

- Distributed systems help you to various computing resources in size and function. They may involve microprocessors, minicomputers, and many general-purpose computer systems.
- A distributed system also offers the user access to the various resources the network shares.
- It helps to access shared resources that help computation to speed-up or offers data availability and reliability.

Main Memory management

Main Memory is a large array of storage or bytes, which has an address. The memory management process is conducted by using a sequence of reads or writes of specific memory addresses.

In order to execute a program , it should be mapped to absolute addresses and loaded inside the Memory. The selection of a memory management method depends on several factors.

However, it is mainly based on the hardware design of the system. Each algorithm requires corresponding hardware support. Main Memory offers fast storage that can be accessed directly by the CPU. It is costly and hence has a lower storage capacity. However, for a program to be executed, it must be in the main Memory.

Functions of Memory management in OS:

An Operating System performs the following functions for Memory Management:

- It helps you to keep track of primary memory.
- Determine what part of it are in use by whom, what part is not in use.
- In a multiprogramming system, the OS takes a decision about which process will get Memory and how much.
- Allocates the memory when a process requests
- It also de-allocates the Memory when a process no longer requires or has been terminated.

Secondary-Storage Management

The most important task of a computer system is to execute programs. These programs, along with the data, helps you to access, which is in the main memory during execution.

This Memory of the computer is very small to store all data and programs permanently. The computer system offers secondary storage to back up the main Memory. Today modern computers use hard drives/SSD as the primary storage of both programs and data. However, the secondary storage management also works with storage devices, like a USB flash drive, and CD/DVD drives. Programs like assemblers, compilers, stored on the disk until it is loaded into memory, and then use the disk as a source and destination for processing.

Functions of Secondary storage management in OS:

Here, are major functions of secondary storage management in OS:

- Storage allocation
- Free space management
- Disk scheduling

Security Management

The various processes in an operating system need to be secured from each other's activities. For that purpose, various mechanisms can be used to ensure that those processes which want to operate files, memory CPU, and other hardware resources should have proper authorization from the operating system.

For example, Memory addressing hardware helps you to confirm that a process can be executed within its own address space. The time ensures that no process has control of the CPU without renouncing it.

Lastly, no process is allowed to do its own I/O, to protect, which helps you to keep the integrity of the various peripheral devices.

Other Important Activities

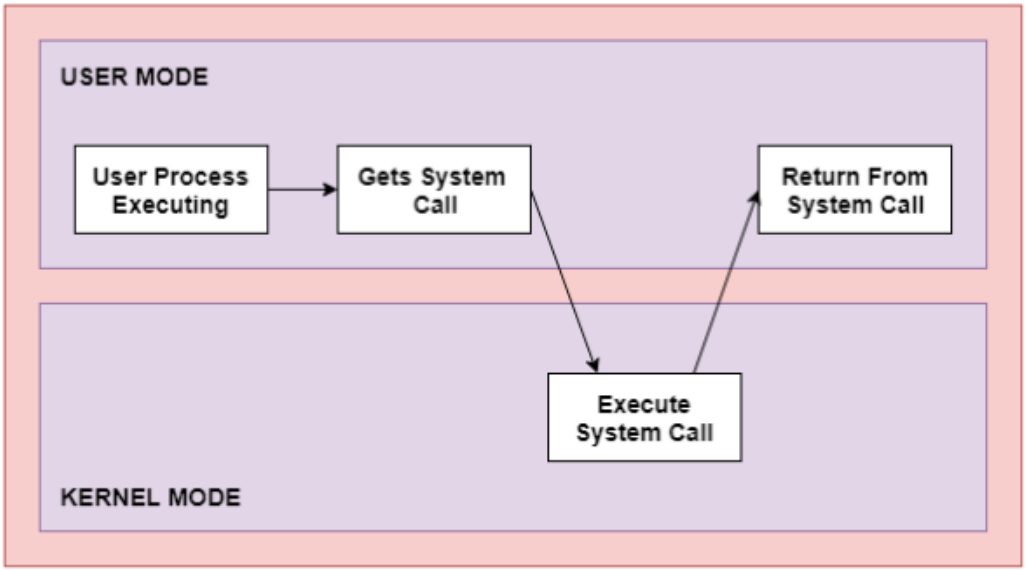
Here, are some other important activities of OS:

- The user's program can't execute I/O operations directly. The operating system should provide some medium to perform this.
- OS checks the capability of the program to read, write, create, and delete files.
- OS facilitates an exchange of information between processes executing on the same or different systems.
- OS components help you to makes sure that you get the correct computing by detecting errors in the CPU and memory hardware.

System calls in Operating System

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.

A figure representing the execution of the system call is given as follows −



the processes execute normally in the user mode until a system call interrupts this. Then the system call is executed on a priority basis in the kernel mode. After the execution of the system call, the control returns to the user mode and execution of user processes can be resumed.

In general, system calls are required in the following situations −

- If a file system requires the creation or deletion of files. Reading and writing from files also require a system call.
- Creation and management of new processes.
- Network connections also require system calls. This includes sending and receiving packets.
- Access to a hardware devices such as a printer, scanner etc. requires a system call.

Types of System Calls

There are mainly five types of system calls. These are explained in detail as follows −

Process Control

These system calls deal with processes such as process creation, process termination etc.

File Management

These system calls are responsible for file manipulation such as creating a file, reading a file, writing into a file etc.

Device Management

These system calls are responsible for device manipulation such as reading from device buffers, writing into device buffers etc.

Information Maintenance

These system calls handle information and its transfer between the operating system and the user program.

Communication

These system calls are useful for interprocess communication. They also deal with creating and deleting a communication connection.

Some of the examples of all the above types of system calls in Windows and Unix are given as follows −

Types of System Calls	Windows	Linux
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
File Management	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Device Management	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shmget() mmap()

There are many different system calls as shown above. Details of some of those system calls are as follows −

open()

The open() system call is used to provide access to a file in a file system. This system call allocates resources to the file and provides a handle that the process uses to refer to the file. A file can be opened by multiple processes at the same time or be restricted to one process. It all depends on the file organisation and file system.

read()

The read() system call is used to access data from a file that is stored in the file system. The file to read can be identified by its file descriptor and it should be opened using open() before it can be read. In general, the read() system calls takes three arguments i.e. the file descriptor, buffer which stores read data and number of bytes to be read from the file.

write()

The write() system calls writes the data from a user buffer into a device such as a file. This system call is one of the ways to output data from a program. In general, the write

system calls takes three arguments i.e. file descriptor, pointer to the buffer where data is stored and number of bytes to write from the buffer.

close()

The close() system call is used to terminate access to a file system. Using this system call means that the file is no longer required by the program and so the buffers are flushed, the file metadata is updated and the file resources are de-allocated.

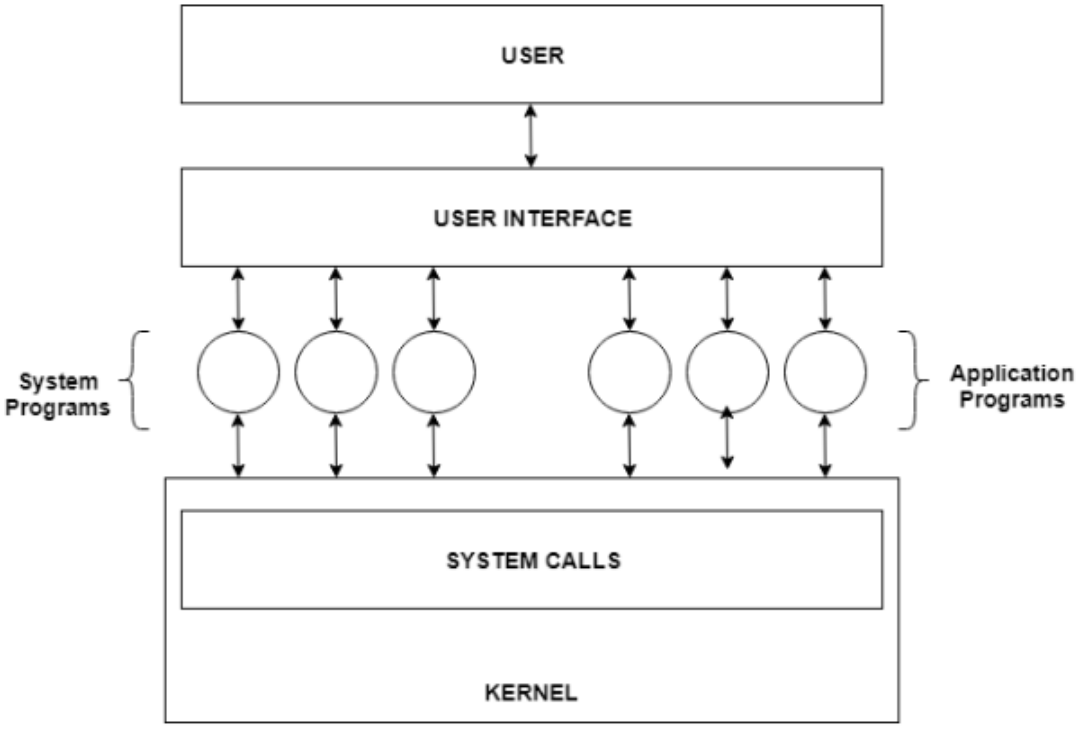
System Programs

System programs provide an 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 Purpose

The system program serves as a part of the operating system. It traditionally lies between the user interface and the system calls. The user view of the system is actually defined by system programs and not system calls because that is what they interact with and system programs are closer to the user interface.

An image that describes system programs in the operating system hierarchy is as follows −



In the above image, system programs as well as application programs form a bridge between the user interface and the system calls. So, from the user view the operating system observed is actually the system programs and not the system calls.

Types of System Programs

System programs can be divided into seven parts. These are given as follows:

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.

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.

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.

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.

Process Management

Process management involves various tasks like creation, scheduling, termination of processes, and a dead lock. Process is a program that is under execution. The OS must allocate resources that enable processes to share and exchange information. It also protects the resources of each process from other methods and allows synchronization among processes.

Process Architecture

Process architecture Image

Here, is an Architecture diagram of the Process

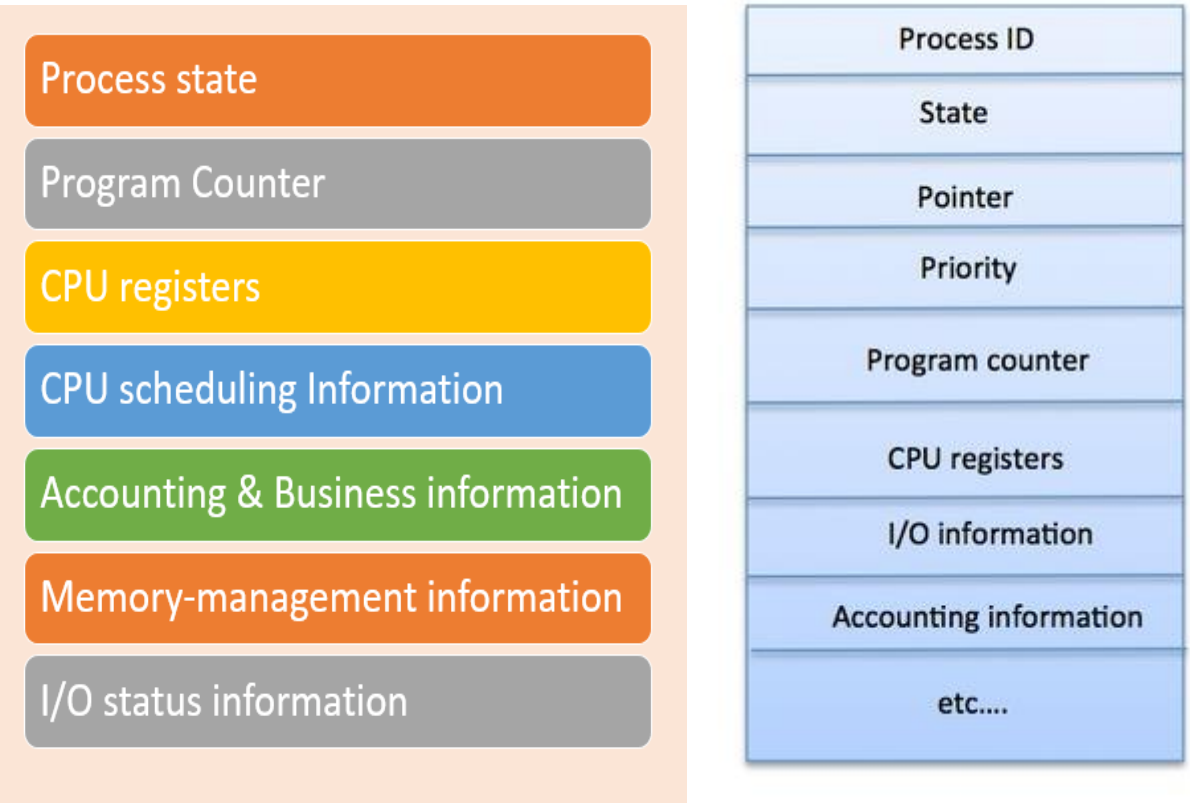
- **Stack:** The Stack stores temporary data like function parameters, returns addresses, and local variables.
- **Heap** Allocates memory, which may be processed during its run time.
- **Data:** It contains the variable.
- **Text:** Text Section includes the current activity, which is represented by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

Process Control Blocks(PCB)

The PCB is a full form of Process Control Block. It is a data structure that is maintained by the Operating System for every process. The PCB should be identified by an integer Process ID (PID). It helps you to store all the information

Every process is represented in the operating system by a process control block, which is also called a task control block.

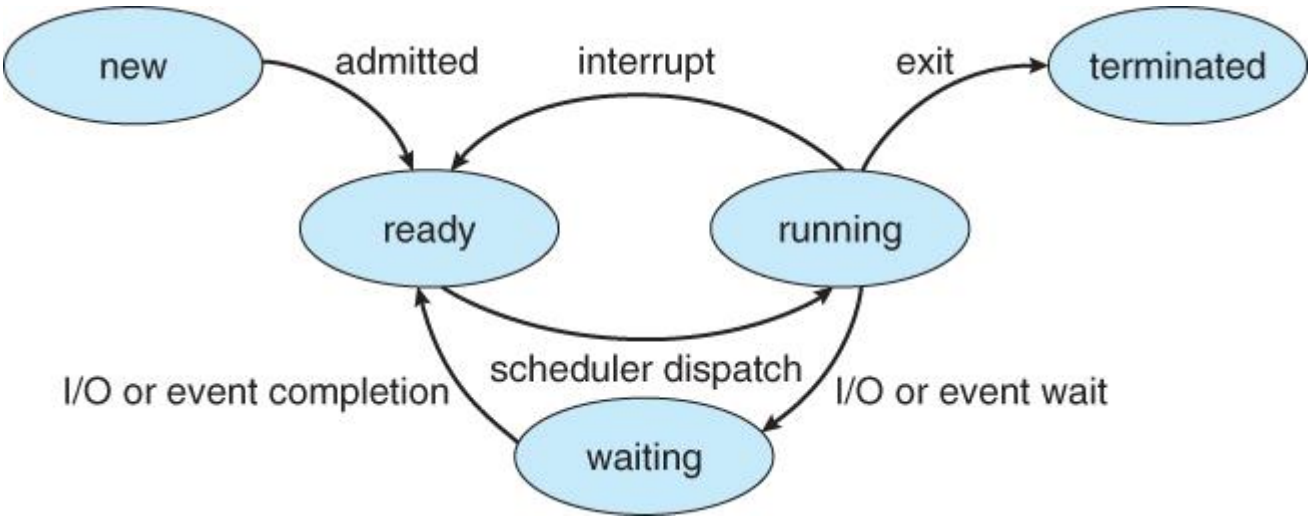
Important components of PCB are:

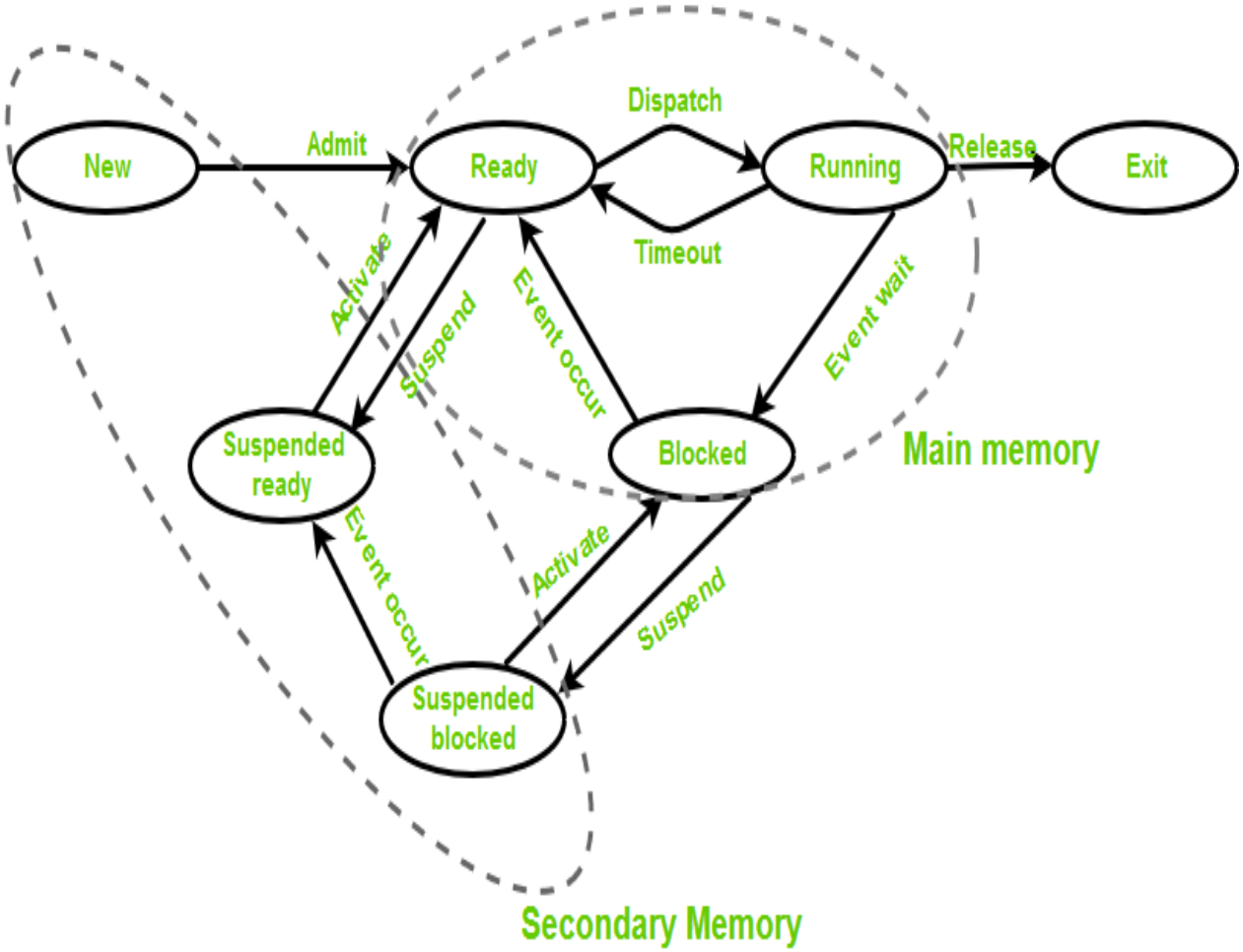


Process Control Block(PCB)

- **Process state:** A process can be new, ready, running, waiting, etc.
- **Program counter:** The program counter contains the address of the next instruction, which should be executed for that process.
- **CPU registers:** This component includes accumulators, index and general-purpose registers, and information of condition code.
- **CPU scheduling information:** This component includes a process priority, pointers for scheduling queues, and various other scheduling parameters.
- **Accounting and business information:** It includes the amount of CPU and time utilities like real time used, job or process numbers, etc.
- **Memory-management information:** This information includes the value of the base and limit registers, the page, or segment tables. This depends on the memory system, which is used by the operating system.
- **I/O status information:** This block includes a list of open files, the list of I/O devices that are allocated to the process, etc.

Process State - When a process executes, it passes through different states.





- **New (Create) –** In this step, the process is about to be created but not yet created, it is the program which is present in secondary memory that will be picked up by OS to create the process.

- **Ready –** New -> Ready to run. After the creation of a process, the process enters the ready state i.e. the process is loaded into the main memory. The process here is ready to run and is waiting to get the CPU time for its execution. Processes that are ready for execution by the CPU are maintained in a queue for ready processes.

- **Run –** The process is chosen by CPU for execution and the instructions within the process are executed by any one of the available CPU cores.

- **Blocked or wait –** Whenever the process requests access to I/O or needs input from the user or needs access to a critical region(the lock for which is already acquired) it enters the blocked or wait state. The process continues to wait in the main memory and does not require CPU. Once the I/O operation is completed the process goes to the ready state.

- **Terminated or completed –** Process is killed as well as PCB is deleted.

- **Suspend ready –** Process that was initially in the ready state but were swapped out of main memory(refer Virtual Memory topic) and placed onto external storage by scheduler are said to be in suspend ready state. The process will transition back to ready state whenever the process is again brought onto the main memory.

- **Suspend wait or suspend blocked** – Similar to suspend ready but uses the process which was performing I/O operation and lack of main memory caused them to move to secondary memory. When work is finished it may go to suspend ready.

The process scheduling

The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.

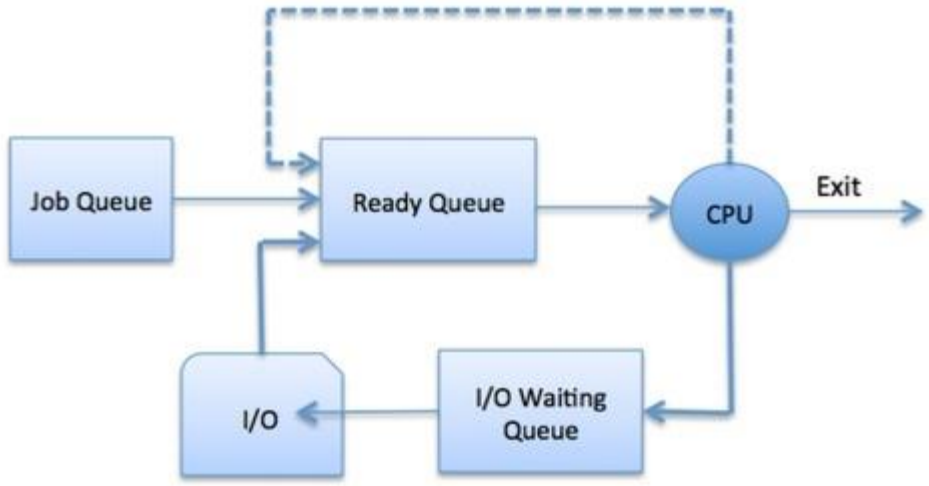
Process scheduling is an essential part of a Multiprogramming operating systems. Such operating systems allow more than one process to be loaded into the executable memory

Process Scheduling Queues

The OS maintains all PCBs in Process Scheduling Queues. The OS maintains a separate queue for each of the process states and PCBs of all processes in the same execution state are placed in the same queue. When the state of a process is changed, its PCB is unlinked from its current queue and moved to its new state queue.

The Operating System maintains the following important process scheduling queues −

- **Job queue** − This queue keeps all the processes in the system.
- **Ready queue** − This queue keeps a set of all processes residing in main memory, ready and waiting to execute. A new process is always put in this queue.
- **Device queues** − The processes which are blocked due to unavailability of an I/O device constitute this queue.



P1,p3,p5-60msec
P2-80msec-20msec
P4-130msec-70msec

30msec

P1,p2,p3,p4,p5-30msec
P1,p2,p3,p4,p5-30msec
P1,p3,p5 completed the task

P2-20msec completed.
P4-30msec not completed

P4-30msec not completed
P4-10msec completed

The OS can use different policies to manage each queue (FIFO, Round Robin, Priority, etc.). The OS scheduler determines how to move processes between the ready and run queues which can only have one entry per processor.

Schedulers

Schedulers are special system software which handle process scheduling in various ways. Their main task is to select the jobs to be submitted into the system and to decide which process to run. Schedulers are of three types −

- Long-Term Scheduler
- Short-Term Scheduler
- Medium-Term Scheduler

Long Term Scheduler

It is also called a **job scheduler**. A long-term scheduler determines which programs are admitted to the system for processing. It selects processes from the queue and loads them into memory for execution. Process loads into the memory for CPU scheduling.

The primary objective of the job scheduler is to provide a balanced mix of jobs, such as I/O bound and processor bound. It also controls the degree of multiprogramming. If the degree of multiprogramming is stable, then the average rate of process creation must be equal to the average departure rate of processes leaving the system.

On some systems, the long-term scheduler may not be available or minimal. Time-sharing operating systems have no long term scheduler. When a process changes the state from new to ready, then there is use of long-term scheduler.

Short Term Scheduler

It is also called as **CPU scheduler**. Its main objective is to increase system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects a process among the processes that are ready to execute and allocates CPU to one of them.

Short-term schedulers, also known as dispatchers, make the decision of which process to execute next. Short-term schedulers are faster than long-term schedulers.

Medium Term Scheduler

Medium-term scheduling is a part of **swapping**. It removes the processes from the memory. It reduces the degree of multiprogramming. The medium-term scheduler is in-charge of handling the swapped out-processes.

A running process may become suspended if it makes an I/O request. A suspended processes cannot make any progress towards completion. In this condition, to remove the process from memory and make space for other processes, the suspended process is moved to the secondary storage. This process is called **swapping**, and the process is said to be swapped out or rolled out. Swapping may be necessary to improve the process mix.

Comparison among Scheduler

S.N.	Long-Term Scheduler	Short-Term Scheduler	Medium-Term Scheduler
1	It is a job scheduler	It is a CPU scheduler	It is a process swapping scheduler.
2	Speed is lesser than short term scheduler	Speed is fastest among other two	Speed is in between both short and long term scheduler.
3	It controls the degree of multiprogramming	It provides lesser control over degree of multiprogramming	It reduces the degree of multiprogramming.
4	It is almost absent or minimal in time sharing system	It is also minimal in time sharing system	It is a part of Time sharing systems.
5	It selects processes from pool and loads them into memory for execution	It selects those processes which are ready to execute	It can re-introduce the process into memory and execution can be continued

Cooperating processes

In the computer system, there are many processes which may be either independent processes or **cooperating processes** that run in the operating system. A process is said to be independent when it cannot affect or be affected by any other processes that are running the system. It is clear that any process which does not share any data (temporary or persistent) with any another process then the process independent. On the other hand, a cooperating process is one which can affect or affected by any another process that is running on the computer. the cooperating process is one which shares data with another process.

There are several reasons for providing an environment that allows **process cooperation**:

- **Information sharing**
  In the information sharing at the same time, many users may want the same piece of information(for instance, a shared file) and we try to provide that environment in which the users are allowed to concurrent access to these types of resources.
- **Computation speedup**
  When we want a task that our process run faster so we break it into a subtask, and each subtask will be executing in parallel with another one. It is noticed that the speedup can be achieved only if the computer has multiple processing elements (such as CPUs or I/O channels).
- **Modularity**
  In the modularity, we are trying to construct the system in such a modular fashion, in which the system dividing its functions into separate processes.
- **Convenience**
  An individual user may have many tasks to perform at the same time and the user is able to do his work like editing, printing and compiling.

A cooperating process is one that can affect or be affected by other process executing in the system cooperating process an:

1. **Directly share a logical address data space (i.e. code & data)** - This may result in data inconsistency. It is implemented on threads.
2. **Share data only through files/ messages** - So we will deal with various to order....orderly execution of cooperating process so that data consistency is maintained.

Example- producer-consumer problem

There is a producer and a consumer, producer produces on the item and places it into buffer whereas consumer consumes that item.

Inter Process Communication (IPC)

Inter Process Communication (IPC) refers to a mechanism, where the operating systems allow various processes to communicate with each other.

A process can be of two types:

- Independent process.
- Co-operating process.

An independent process is not affected by the execution of other processes while a co-operating process can be affected by other executing processes. Though one can think that those processes, which are running independently, will execute very efficiently, in reality, there are many situations when co-operative nature can be utilized for increasing computational speed, convenience and modularity.

Inter process communication (IPC) is a mechanism which allows processes to communicate with each other and synchronize their actions. The communication between these processes can be seen as a method of co-operation between them. Processes can communicate with each other through both:

1. Shared Memory
2. Message passing

The Figure shows a basic structure of communication between processes via the shared memory method and via the message passing method.

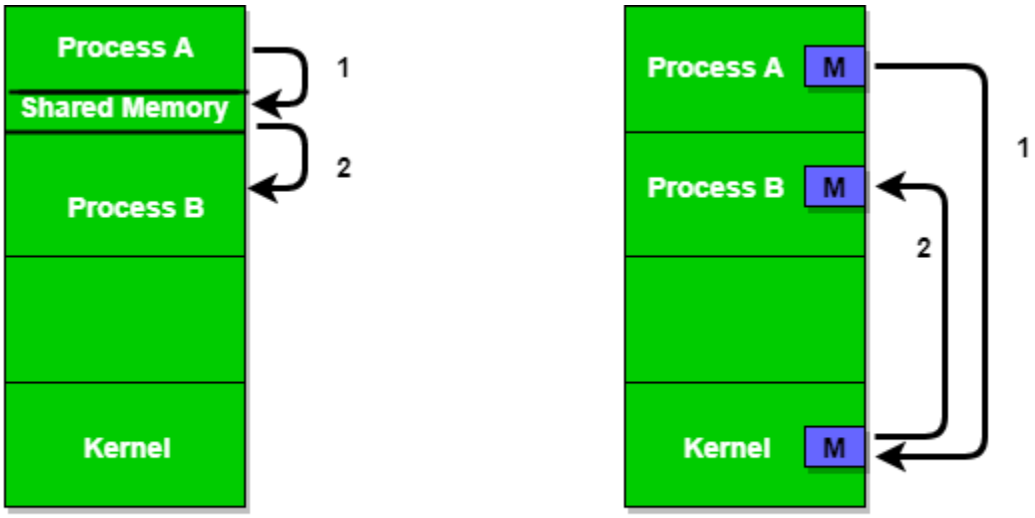


Figure 1 - Shared Memory and Message Passing

. Communication between processes using shared memory requires processes to share some variable and it completely depends on how programmer will implement it.

One way of communication using shared memory is: Suppose process1 and process2 are executing simultaneously and they share some resources or use some information from another process.

Process1 generate information about certain computations or resources being used and keeps it as a record in shared memory. When process2 needs to use the shared information, it will check in the record stored in shared memory and take note of the information generated by process1 and act accordingly. Processes can use shared memory for extracting information as a record from another process as well as for delivering any specific information to other processes.

Let’s discuss an example of communication between processes using shared memory method.

Ex: Producer-Consumer problem
There are two processes: Producer and Consumer.

Producer produces some item and Consumer consumes that item.

The two processes share a common space or memory location known as a buffer where the item produced by Producer is stored and from which the Consumer consumes the item, if needed.

There are two versions of this problem: the first one is known as unbounded buffer problem in which Producer can keep on producing items and there is no limit on the size of the buffer

the second one is known as the bounded buffer problem in which Producer can produce up to a certain number of items before it starts waiting for Consumer to consume it.

We will discuss the bounded buffer problem. First, the Producer and the Consumer will share some common memory, then producer will start producing items.

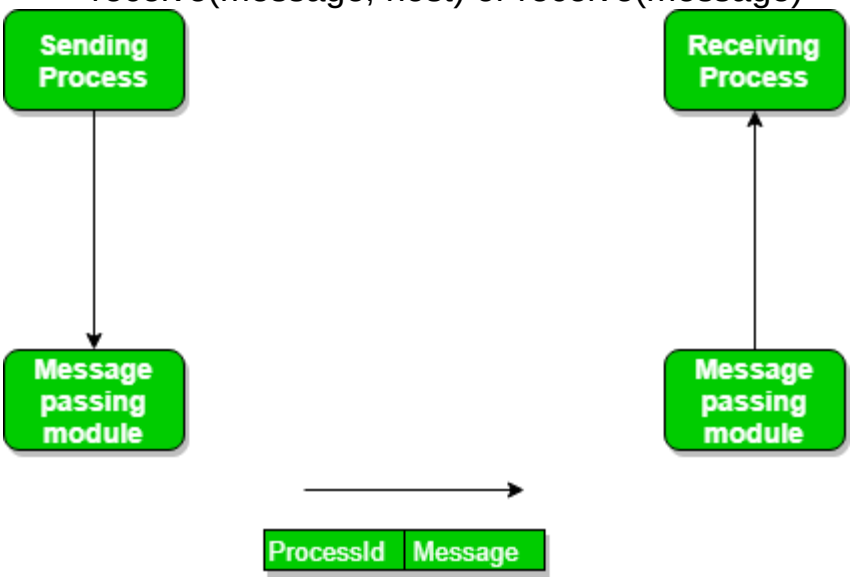
If the total produced item is equal to the size of buffer, producer will wait to get it consumed by the Consumer. Similarly, the consumer will first check for the availability of the item. If no item is available, Consumer will wait for Producer to produce it. If there are items available, Consumer will consume it.

ii) Messaging Passing Method

The communication between processes via message passing. In this method, processes communicate with each other without using any kind of shared memory.

If two processes p1 and p2 want to communicate with each other, they proceed as follows:

- Establish a communication link (if a link already exists, no need to establish it again.)
- Start exchanging messages using basic primitives.
  We need at least two primitives:
  – send(message, destinaion) or send(message)
  – receive(message, host) or receive(message)



The message size can be of fixed size or of variable size. If it is of fixed size, it is easy for an OS designer but complicated for a programmer and if it is of variable size then it is easy for a programmer but complicated for the OS designer. A standard message can have two parts: **header and body.**
The **header part** is used for storing message type, destination id, source id, message length, and control information. The control information contains information like what to do if runs out of buffer space, sequence number, priority. Generally, message is sent using FIFO style.