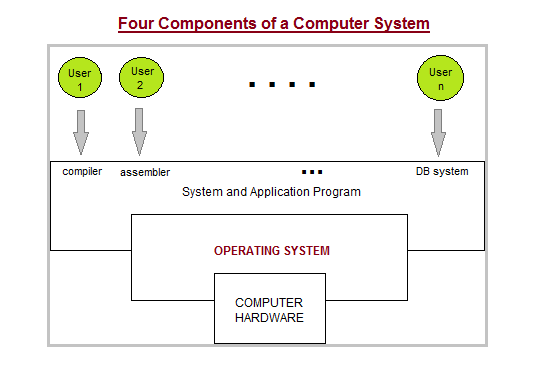
**Introduction of Operating Systems**

A computer system has many resources (hardware and software), which may be require to complete a task. The commonly required resources are input/output devices, memory, file storage space, CPU etc.

The operating system acts as a manager of the above resources and allocates them to specific programs and users as necessary for their task. Therefore operating system is the resource manager i.e. it can manage the resource of a computer system internally. The resources are processor, memory, files, and I/O devices.



**Two Views of Operating System**

1. User's View
2. System View

**User View :**

The user view of the computer refers to the interface being used. Such systems are designed for one user to monopolize its resources, to maximize the work that the user is performing. In these cases, the operating system is designed mostly for ease of use, with some attention paid to performance, and none paid to resource utilization.

**System View :**

Operating system can be viewed as a resource allocator also. A computer system consists of many resources like - hardware and software - that must be managed efficiently. The operating system acts as the manager of the resources, decides between conflicting requests, controls execution of programs etc.

**Operating System Management Tasks**

1. **Processor management** which involves putting the tasks into order and pairing them into manageable size before they go to the CPU.
2. **Memory management** which coordinates data to and from RAM (random-access memory) and determines the necessity for virtual memory.
3. **Device management** which provides interface between connected devices.
4. **Storage management** which directs permanent data storage.
5. **Application** which allows standard communication between software and your computer.
6. **User interface** which allows you to communicate with your computer.

**Functions of Operating System**

1. It boots the computer
2. It performs basic computer tasks e.g. managing the various peripheral devices e.g. mouse, keyboard
3. It provides a user interface, e.g. command line, graphical user interface (GUI)
4. It handles system resources such as computer's memory and sharing of the central processing unit(CPU) time by various applications or peripheral devices.
5. It provides file management which refers to the way that the operating system manipulates, stores, retrieves and saves data.
6. Error Handling is done by the operating system. It takes preventive measures whenever required to avoid errors.

**Types of Operating Systems**

Following are some of the most widely used types of Operating system.

1. Simple Batch System
2. Multiprogramming Batch System
3. Multiprocessor System
4. Distributed Operating System
5. Realtime Operating System

**SIMPLE BATCH SYSTEMS**

* In this type of system, there is no direct interaction between user and the computer.
* The user has to submit a job (written on cards or tape) to a computer operator.
* Then computer operator places a batch of several jobs on an input device.
* Jobs are batched together by type of languages and requirement.
* Then a special program, the monitor, manages the execution of each program in the batch.
* The monitor is always in the main memory and available for execution.

**Following are some disadvantages of this type of system :**

1. Zero interaction between user and computer.
2. No mechanism to prioritize processes.

**MULTIPROGRAMMING BATCH SYSTEMS**

* In this the operating system, picks and begins to execute one job from memory.
* Once this job needs an I/O operation operating system switches to another job (CPU and OS always busy).
* Jobs in the memory are always less than the number of jobs on disk(Job Pool).
* If several jobs are ready to run at the same time, then system chooses which one to run (CPU Scheduling).
* In Non-multiprogrammed system, there are moments when CPU sits idle and does not do any work.
* In Multiprogramming system, CPU will never be idle and keeps on processing

**Time-Sharing Systems**

It is very similar to Multiprogramming batch systems. In fact time sharing systems are an extension of multiprogramming systems.

In time sharing systems the prime focus is on minimizing the response time, while in multiprogramming the prime focus is to maximize the CPU usage.

**MULTIPROCESSOR SYSTEMS**

A multiprocessor system consists of several processors that share a common physical memory. Multiprocessor system provides higher computing power and speed. In multiprocessor system all processors operate under single operating system. Multiplicity of the processors and how they do act together are transparent to the others.

**Following are some advantages of this type of system.**

1. Enhanced performance
2. Execution of several tasks by different processors concurrently, increases the system's throughput without speeding up the execution of a single task.
3. If possible, system divides task into many subtasks and then these subtasks can be executed in 0parallel in different processors. Thereby speeding up the execution of single tasks.

**DISTRIBUTED OPERATING SYSTEMS**

The motivation behind developing distributed operating systems is the availability of powerful and inexpensive microprocessors and advances in communication technology.

These advancements in technology have made it possible to design and develop distributed systems comprising of many computers that are inter connected by communication networks. The main benefit of distributed systems is its low price/performance ratio.

Following are some advantages of this type of system.

1. As there are multiple systems involved, user at one site can utilize the resources of systems at other sites for resource-intensive tasks.
2. Fast processing.
3. Less load on the Host Machine.

**REAL-TIME OPERATING SYSTEM**

It is defined as an operating system known to give maximum time for each of the critical operations that it performs, like OS calls and interrupt handling.

The Real-Time Operating system which guarantees the maximum time for critical operations and complete them on time are referred to as Hard Real-Time Operating Systems.

While the real-time operating systems that can only guarantee a maximum of the time, i.e. the critical task will get priority over other tasks, but no assurity of completeing it in a defined time. These systems are referred to as Soft Real-Time Operating Systems.

**Process**

**What is a Process?**

A program in the execution is called a Process. Process is not the same as program. A process is more than a program code. A process is an 'active' entity as opposed to program which is considered to be a 'passive' entity. Attributes held by process include hardware state, memory, CPU etc.

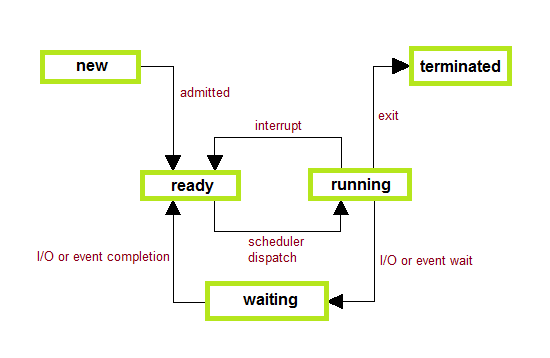
Process memory is divided into four sections for efficient working :

* The text section is made up of the compiled program code, read in from non-volatile storage when the program is launched.
* The data section is made up the global and static variables, allocated and initialized prior to executing the main.
* The heap is used for the dynamic memory allocation, and is managed via calls to new, delete, malloc, free, etc.
* The stack is used for local variables. Space on the stack is reserved for local variables when they are declared.

**PROCESS STATE**

Processes can be any of the following states :

* **New** - The process is in the stage of being created.
* **Ready** - The process has all the resources available that it needs to run, but the CPU is not currently working on this process's instructions.
* **Running** - The CPU is working on this process's instructions.
* **Waiting** - The process cannot run at the moment, because it is waiting for some resource to become available or for some event to occur.
* **Terminated** - The process has completed.



**PROCESS CONTROL BLOCK**

There is a Process Control Block for each process, enclosing all the information about the process. It is a data structure, which contains the following :

* Process State - It can be running, waiting etc.
* Process ID and parent process ID.
* CPU registers and Program Counter. ProgramCounter holds the address of the next instruction to be executed for that process.
* CPU Scheduling information - Such as priority information and pointers to scheduling queues.
* Memory Management information - Eg. page tables or segment tables.
* Accounting information - user and kernel CPU time consumed, account numbers, limits, etc.
* I/O Status information - Devices allocated, open file tables, etc.

