An operating system (OS) is system software that manages computer hardware and software resources and provides common services for computer programs. An operating system is a collection of system programs that control computer and any other peripherals connected to it. The program that hides the truth about the hardware from the programmer and present and a nice simple view a named file that can be read & written as "operating system". Operating system shields the programmer from the interface, the abstraction offers by the operating system is slower & easier to use than the underlying hardware. An operating System is a layer of software on a bare hardware machine that performs two basic functions: Resource management, Virtual machine management. Resource Management : A computer consists of a set of resources such as processors, memories, timers, disks, printers and many others. The Operating System manages these resources and allocates them to specific programs. As a resource manager, the Operating system provides the controlled allocation of the processors, memories, I/O devices among various programs. Moreover, multiple user programs are running at the same time, the processor itself is a resource and the Operating System decides how much processor time should be given for the execution of a particular user program. An operating system is a control program, a control program manages the execution of user program to prevent errors and improve use of computer. It is especially concerned with the operation and control of I/O devices. When a computer has multiple users, the operating system manages and protects the memory I/O devices. The operating system also keeps in trace that who is using which resource and to whom to the grant resource. **Virtual Machine Management** : The architecture (instruction set, memory, I/O, and bus structure) of most computers at the machine level language is primitive and awkward to program, especially for input/output operations. Users do not want to be involved in the programming of storage devices. Moreover, Operating System provides a simple, high-level abstraction such that these devices contain a collection of named files. Such files consist of the useful piece of information like a digital photo, email messages, or web page. Operating System provides a set of basic commands or instructions to perform various operations such as read, write, modify, save or close. Also, dealing with them is easier than directly dealing with hardware. Thus, Operating System hides the complexity of hardware and presents a beautiful interface to the users. Just as the operating system shields (protect from an unpleasant experience) the programmer from the disk hardware and presents a simple file-oriented interface, it also conceals a lot of unpleasant business concerning interrupts, timers, memory management, and other low-level features. In each case, the abstraction offered by the operating system is simpler and easier to use than that offered by the underlying hardware. Moreover, in this view, the function of the operating system is to present the user with the equivalent of an **extended machine** or **virtual machine** that is easier to work with than the underlying hardware. The operating system provides a variety of services that programs can obtain using special instructions called system calls.**Types of Operating System** :**Batch Processing**: In the batch system requires the grouping of similar jobs which consists of programs data and system commands. The instructions, data and some controlled information are submitted to the computer operator in the form of job. The users are not allowed to interact with the computer system. Thus, the programs that do not require interaction are well served by batch OS. Since the jobs are executed in the FIFO manner, the batch OS require very simple CPU scheduling techniques. In addition, batch system allow only one user program to reside in the memory at a time, thus memory management is also very simple affairs in batch OS. Since only one program is in execution at a time any time critical device management is not required, which simplifies the I/O management. Since the files are accessed in a serial

manner, a concurrency control mechanism is not required, making the file management also very simple matter in batch OS. 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.**Multi-Processing:** A computer's capability to process more than one task simultaneously is called *multiprocessing*. A multiprocessing operating system is capable of running many programs simultaneously, and most modern network operating systems (NOSs) support multiprocessing. These operating systems include Windows NT, 2000, XP, and UNIX. A multiprocessing system uses more than one processor to process any given workload, increasing the performance of a system's application environment beyond that of a single processor's capability. Collections of processors arranged in a loosely coupled configuration and interacting with each other over a communication channel have been the most common multiprocessor architecture. **Advantage:** Increase throughput ,Economy of scale ,Increased reliability .**Disadvantage:** If one processor fails then it will affect in the speed , Multiprocessor systems are expensive ,Complex OS is required, Large main memory required. **Time Sharing Operating System** : Time-sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing. The main difference between Multiprogrammed Batch Systems and Time-Sharing Systems is that in case of multiprogrammed batch systems, the objective is to maximize processor use, whereas in Time-Sharing Systems, the objective is to minimize response time. Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, the processor executes each user program in a short burst or quantum of computation. That is, if n users are present, then each user can get a time quantum. When the user submits the command, the response time is in few seconds at most. Advantages :Provides the advantage of quick response ,Avoids duplication of software, Reduces CPU idle time .Disadvantages :Problem of reliability ,Question of security and integrity of user programs and data, Problem of data communication .**Real Time System**: The time taken by the system to respond to an input and display of required updated information is termed as the response time. 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. It is a multitasking operating system that aims at executing real-time applications. Real-time operating systems often use specialized scheduling algorithms so that they can achieve a deterministic nature of behavior. The main object of real-time operating systems is their quick and predictable response to events. They either have an event-driven design or a time-sharing one. An event-driven system switches between tasks based of their priorities while time-sharing operating systems switch tasks based on clock interrupts : 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 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. Network Operating System : A Network Operating System runs on a server and provides the server the capability to manage data, users, groups, security, applications, and other networking functions. The

primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), and a private network or to other networks. Certain standalone operating systems, such as Microsoft Windows NT and Digital's OpenVMS, come with multipurpose capabilities and can also act as network operating systems. Some of the most well-known network operating systems include Microsoft Windows Server 2003, Microsoft Windows Server 2008, Linux and Mac OS X.Mainly there are two types of network operating. **Peer-to-peer** network operating systems allow users to share resources and files located on their computers and to access shared resources found on other computers. In a peer-to-peer network, all computers are considered equal; they all have the same privileges to use the resources available on the network. Peer-to-peer networks are designed primarily for small to medium local area networks. Windows for Workgroups is an example of the program that can function as peer-to-peer network operating systems. **Client/server** network operating systems allow the network to centralize functions and applications in one or more dedicated file servers. The file servers become the heart of the system, providing access to resources and providing security. The workstations (clients) have access to the resources available on the file servers. The network operating system allows multiple users to simultaneously share the same resources irrespective of physical location. Novell Netware and Windows 2000Server are examples of client/ server network operating systems . Distributed Operating System: A distributed operating system is an operating system that runs on several machines. Its purpose is to provide a useful set of services, generally to make the collection of machines behave more like a single machine. Distributed operating systems typically run cooperatively on all machines whose resources they control. These machines might be capable of independent operation, or they might be usable merely as resources in the distributed system. In some architectures, each machine is an equally powerful peer as all the others. In other architectures, some machines are permanently designated as master or are given control of particular resources. In yet others, elections or other selection mechanisms are used to designate some machines as having special roles, often controlling roles. Advantages :Sharing of resources, Reliability, Communication, Computation speedup **Operating system components** : An operating system provides the environment within which programs are executed. To construct such an environment, the system is partitioned into small modules with a well-defined interface. The design of a new operating system is a major task. It is very important that the goals of the system be will defined before the design begins. The type of system desired is the foundation for choices between various algorithms and strategies that will be necessary. **Process Management System** : The operating system is responsible for the following activities in connection with processes managed. The creation and deletion of both user and system process,the suspension is resumption of process, the provision of mechanism for process synchronization , the provision of mechanism for deadlock handling. **Memory Management System :** Memory is central to the operation of a modern computer system. The operating system is responsible for the following activities in connection with memory management. Keep track of which parts of memory are currently being used and by whom. Decide which processes are to be loaded into memory when memory space becomes available.Allocate and deallocate memory space as needed. **Secondary Storage Management System :** The main purpose of a computer system is to execute programs. The operating system is responsible for the following activities in connection with disk management Free space management, Storage allocation, Disk scheduling. **I/O System :** One of the important jobs of OS is to manage various I/O devices including mouse, keyboard, touch pad etc. The I/O system

consists of: A buffer caching system, A general device driver code, Drivers for specific hardware devices. Only the device driver knows the peculiarities of a specific device. **File Management system** : The operating system is responsible for the following activities in connection with file management: The creation and deletion of files, The creation and deletion of directory, The support of primitives for manipulating files and directories, The mapping of files onto disk storage, Backup of files on stable (nonvolatile) storage. **Protection System** : Protection refers to a mechanism for controlling the access of programs, processes, or users to the resources defined by a computer controls to be imposed, together with some means of enforcement. Protection can improve reliability by detecting latent errors at the interfaces between component subsystems. Early detection of interface errors can often prevent contamination of a healthy subsystem by a subsystem that is malfunctioning. An unprotected resource cannot defend against use (or misuse) by an unauthorized or incompetent user. **Networking :** A networking system of a computer OS that is designed of networking use. Networking system is an OS with file, task and job management. However in some earlier OS, it was a separate component that enhance a basic, non- networking OS by adding networking capabilities. NOS is designed primarily to support workstation, PC and in some instances, older terminal that are connected to LAN. Networking system allows to share file and printer access among multiple computers in a Network, to enable the sharing of data . **Command Interpreter System** : The command interpreter is the primary interface between the user and the rest of the system. The command statement themselves deal with process management, I/O handling, secondary storage management, main memory management, file system access, protection, and networking. **Operating System Services :System Calls** : A system call is a mechanism that is used by the application program to request a service from the operating system. They use a machine-code instruction that causes the processor to change mode. An example would be from supervisor mode to protected mode. This is where the operating system performs actions like accessing hardware devices or the memory management unit. Generally, the operating system provides a library that sits between the operating system and normal programs. **Services Provided by System Calls:** Process creation and management, Main memory management ,File Access, Directory and File system management , Device handling(I/O) ,Protection,Networking, etc. **A process** is defined as an entity which represents the basic unit of work to be implemented in the system i.e. a process is a program in execution. The execution of a process must progress in a sequential fashion. In general, a process will need certain resources such as the CPU time, memory, files, I/O devices and etc. to accomplish its task. Operation on Process: Process Creation , Destroy of a process , Run a process, Change a process priority , Get process information, Set process information. **Process Creation:** There are four ways achieving it: For a batch environment a process is created in response to submission of a job. In Interactive environment, a process is created when a new user attempt to log on. The OS can create process to perform functions on the behalf a user program. A number of process can be generated from the main process. For the purpose of modularity or to exploit parallelism a user can create numbers of process. **Process Termination :** A process terminates when it finishes executing its last statement. Its resources are returned to the system, it is purged from any system lists or tables, and its process control block (PCB) is erased. The new process terminates the existing process, usually due to following reasons: **Normal Exit (Voluntary):** Most processes terminate because they have done their job. **Error Exist(Voluntary):** When process discovers a fatal error. For example, a user tries to compile a program that does not exist. **Fatal Error (Involuntary):** An error caused by process due to a bug in program for example, executing an illegal instruction, referring

non-existing memory or dividing by zero. **Killed by another Process (Industrial)**: A process executes a system call telling the Operating Systems to terminate some other process. In UNIX, this call is kill. In some systems when a process kills all processes it created are killed as well (UNIX does not work this way).

Process State (Two state model) **: In this model, we consider two main states of the process. These two states are State 1: Process is Running on CPU ,State 2: Process is Not Running on CPU .** New: First of all, when a new process is created, then it is in Not Running State. Suppose a new process P2 is created then P2 is in NOT Running State. CPU: When CPU becomes free, Dispatcher gives control of the CPU to P2 that is in NOT Running state and waiting in a queue. Dispatcher: Dispatcher is a program that gives control of the CPU to the process selected by the CPU scheduler. Suppose dispatcher allow P2 to execute on CPU. Running: When dispatcher allows P2 to execute on CPU then P2 starts its execution. Here we can say that P2 is in running state. Now, if any process with high priority wants to execute on CPU, Suppose P3 with high priority, then P2 should be the pause or we can say that P2 will be in waiting state and P3 will be in running state. Now, when P3 terminates then P2 again allows the dispatcher to execute on CPU. **Process State (Five state model): New:** A process that has just been created but has not yet been admitted to the pool of executable processes by the OS. **Ready:** Process that is prepared to execute when given the opportunity. That is, they are not waiting on anything except the CPU availability. **Running:** the process that is currently being executed. **Blocked:** A process that cannot execute until some event occurs, such as the completion of an I/O operation. **Exit:** A process that has been released from the pool of executable processes by the OS, either because it is halted or because it is aborted for some reason A process that has been released by OS either after normal termination or after abnormal termination (error). **Process Control Block (PCB) :** A process control block or PCB is a data structure (a table) that holds information about a process. Every process or program that runs needs a PCB. When a user requests to run a particular program, the operating system constructs a process control block for that program. It is also known as Task Control Block (TCB). It contains many pieces of information associated with a specific process i.e. it simply serves as the respiratory for any information that may vary from process to process.The information stored in the Process Control Block in given below **Process State:** The state may be new, ready, running, and waiting, halted, and so on. **Program Counter:** the counter indicates the address of the next instruction to be executed for this process. **CPU register**: The registers vary in number and type, depending on the computer architecture. They include accumulator, index registers, stack pointers, and general-purpose registers. Along with the program counter, this state information must be saved when an interrupt occurs, to allow the process to be continued correctly afterward. **CPU Scheduling information:** This information includes a process priority, pointers to scheduling queues, and other scheduling parameters. **Memory management information:** this information includes the value of the base and limit registers, the page table, or the segment tables, depending on the memory system used by the OS. **Accounting information:** This information includes the amount of CPU and real time used, time limits, account numbers, job or process numbers and so on. **I/O status information:** This information includes the list of I/O devices allocated to the process, a list of open files and so on. A thread is a single sequence stream within in a process. Because threads have some of the properties of processes, they are sometimes called lightweight processes. In many respect, threads are popular way to improve application through parallelism. The CPU switches rapidly back and forth among the threads giving

illusion that the threads are running in parallel. Like a traditional process i.e., process with one thread, a thread can be in any of several states (Running, Blocked, Ready or terminated). A thread consists of a program counter (PC), a register set, and a stack space. Threads are not independent of one other like processes. Threads shares address space, program code, global variables, OS resources with other thread. **Why Threads?** Process with multiple threads makes a great server (e.g. print server) . Increase responsiveness, i.e. with multiple threads in a process, if one threads blocks then other can still continue executing . Sharing of common data reduce requirement of inter-process communication. Proper utilization of multiprocessor by increasing concurrency . Threads are cheap to create and use very little resources. Context switching is fast (only have to save/reload PC, Stack, SP, Registers). In an operating system (OS), a process scheduler performs the important activity of scheduling a process between the ready queues and waiting queue and allocating them to the CPU. The OS assigns priority to each process and maintains these queues. The scheduler selects the process from the queue and loads it into memory for execution. There are two types of process scheduling: Preemptive scheduling, Non-preemptive scheduling. Preemptive Scheduling : The scheduling in which a running process can be interrupted if a high priority process enters the queue and is allocated to the CPU is called preemptive scheduling. In this case, the current process switches from the running queue to ready queue and the high priority process utilizes the CPU cycle.Non-preemptive Scheduling : The scheduling in which a running process cannot be interrupted by any other process is called non-preemptive scheduling. Any other process which enters the queue has to wait until the current process finishes its CPU cycle.