Types and Structure of Operating Systems (4 hours)

1. **Introduction and History of Operating system**
2. **Operating System Concepts and Functionalities**
   - **2.1. Processes**
   - **2.2. Files**
   - **2.3. System Calls**
   - **2.4. The Shell**
3. **Operating System Structure**
   - **3.1. Monolithic Systems**
   - **3.2. Layered**
   - **3.3. Virtual Machines**
   - **3.4. Client-Server**
4. **Evolution of Operating System**
5. **Types of operating system**
6. **Features of Operating system**

1. Introduction:

An operating system is the program that manages all the application programs in a computer system. Operating system acts as an intermediary between a user of a computer and the computer hardware.

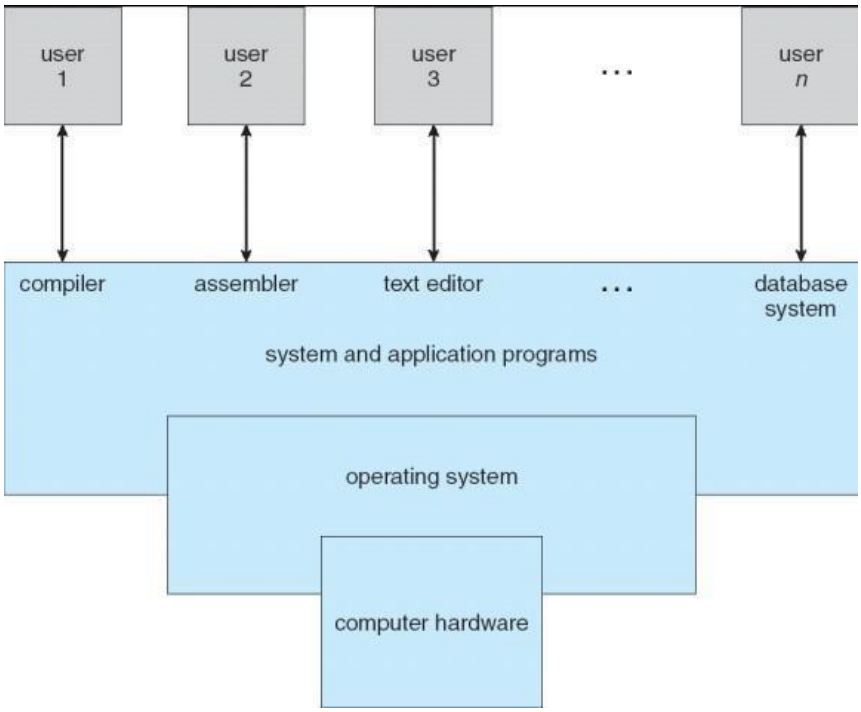


Fig: Abstract view of the components of a Computer System.

An **operating system** (**OS**) is a collection of software that manages computer hardware resources and provides common services for computer programs. The operating system is an essential component of the system software in a computer system.

Operating Systems mainly plays two roles;

- Operating systems plays a role as a resource manager.
- They provide a convenient environment for the development and execution of the programs, user machine.

OS: A Resource Manager:

A computer system has many resources that may be required to solve a problem such as CPU time, memory space, files storage space, I/O devices, and so on. The OS acts as the manager of these resources. Many resources can be conflicted while requested by various users; the OS is responsible how to allocate them to specific programs and users so that it can operate the computer system efficiently and fairly.

OS: Virtual Machine:

A computer system consists of one or more processors, main memory and many types of I/O devices such as disk, tapes, terminals, network interfaces, etc. Writing programs using these hardware resources correctly and efficiently is an extremely difficult job, requires depth knowledge of functions of such resources. Hence to make computer systems usable by a large number of users, OS provides a mechanism to shield programmers and other users from the complexity of hardware resources. This problem is solved by putting a layer of software on top of the bare hardware. This layer of hardware manages all hardware resources of the system, and presents the user with an interface or virtual machine that is easier, safer and efficient to program and use i.e. an OS hides details of hardware resources from programmers and other users. It provides a high level interface to low-level hardware resources, making it easier for programmers and other users to use a computer system.

2. Operating system concepts and functionalities

2.1. Processes

A process is an instance of a program in execution. A program by itself is not a process; a program is a ***passive entity***, such as a file containing a list of instructions stored on disks, whereas a process is an active entity, with a program counter specifying the next instruction to execute and a set of associated resources. A program becomes a process when an executable file is loaded into memory.

2.2. Files

A file is the smallest allotment of logical secondary storage. A computer file is a block of arbitrary information, or resource for storing information. System calls are obviously needed to create files, remove files, read files, and write files. Before a file can be read, it must be opened, and after it has been read it should be closed, so calls are provided to do these things. A file

system organizes data in an efficient manner and is tuned to the specific characteristics of the device.

2.3. System Calls

A **system call** is the programmatic way in which a computer program requests a service from the kernel of the operating system in which it is executed on. This may include services such as accessing the hard disk, creating and executing new processes, etc. System calls provide an essential interface between a process and the operating system.

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. System calls provide the interface between a process and the operating system. Most operations interacting with the system require permissions not available to a user level process, e.g. I/O performed with a device present on the system, or any form of communication with other processes requires the use of system calls.
System calls can be roughly grouped into five major categories:

1. Process Control
   - load
   - execute
   - create process
   - terminate process
   - get/set process attributes
   - wait for time, wait event, signal event
   - allocate, free memory

2. File management
   - create file, delete file
   - open, close
   - read, write, reposition
   - get/set file attributes

3. Device Management
   - request device, release device
   - read, write, reposition
   - get/set device attributes
   - logically attach or detach devices

4. Information Maintenance
   - get/set time or date
   - get/set system data
   - get/set process, file, or device attributes

5. Communication
   - create, delete communication connection
   - send, receive messages

- transfer status information
- attach or detach remote devices

2.4. The Shell and kernel

Shell and the kernel are the parts of this Operating system. When a user gives his command for performing any operation, then the request will go to the shell parts, the shell parts is also called as the interpreter which translate the human program into the machine language and then the request will be transferred to the kernel that means shell is just as the interpreter of the commands which converts the request of the user into the machine language.

Kernel is also known as heart of operating system and the every **operation is performed by using the kernel,** when the kernel receives the request from the shell then this will process the request and display the results on the screen. The various types of operations those are performed by the kernel are as followings:-

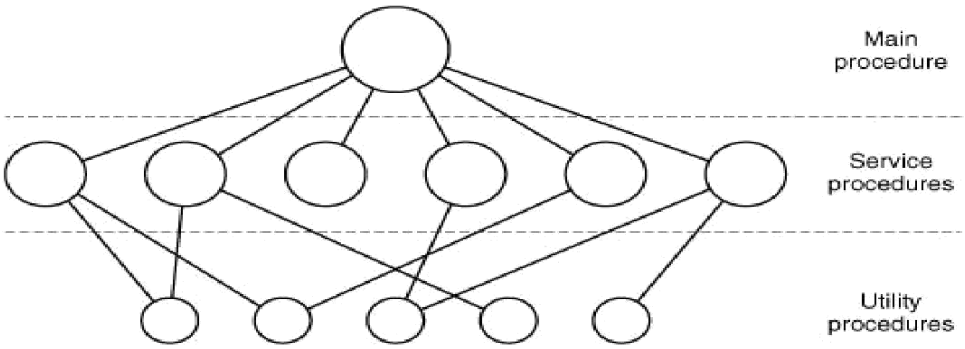
- It controls the state of the process means it checks whether the process is running or process is waiting for the request of the user.
- It provides the memory for the processes those are running on the system means kernel runs the allocation and de-allocation process , first when we request for the service then the kernel will provides the memory to the process and after that it also release the memory which is given to a process.
- The kernel also maintains a time table for all the processes those are running means the kernel also prepare the schedule time means this will provide the time to various process of the CPU and the kernel also puts the waiting and suspended jobs into the different memory area.
- When a kernel determines that the logical memory doesn't fit to store the programs. Then he uses the concept of the physical memory which will store the programs into temporary manner i.e. virtual memory.
- Kernel also maintains all the files those are stored into the computer system and the kernel also stores all the files into the system as no one can read or write the files without any permission. So that the kernel system also provides us the facility to use the passwords and also all the files are stored into the particular manner.

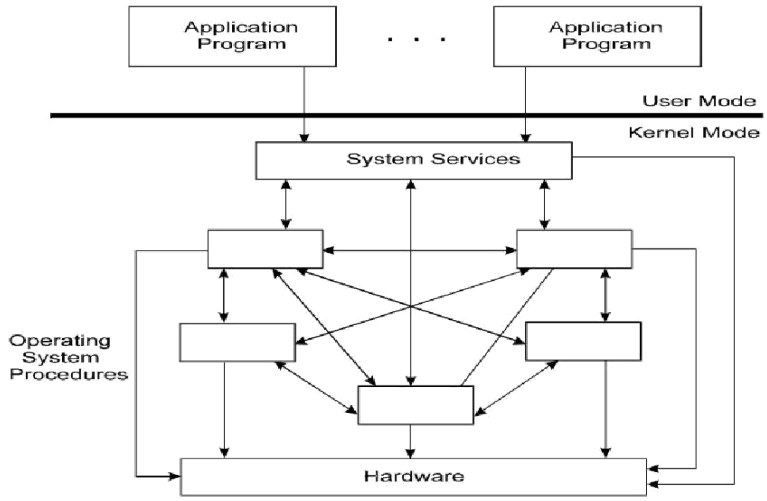
3. Operating system structure

An operating system might have many structures. Some of the main structures used in operating systems are:

3.1. Monolithic architecture of operating system

It is the oldest architecture used for developing operating system. The components of monolithic operating system are organized haphazardly and any module can call any other module without any arrangement. Applications in monolithic OS are also separated from the operating system itself. That is, the operating system code runs in a privileged processor mode (referred to as kernel mode), with access to system data and to the hardware; applications run in a non-privileged processor mode (called the user mode), with a limited set of interfaces available and with limited access to system data. The monolithic operating system structure with separate user and kernel processor mode is shown in Figure.





The structure is that there is no structure. The operating system is written as a collection of procedures, each of which can call any of the other ones whenever it needs to. **Example Systems: CP/M and MS-DOS**

3.2. Layered Architecture of operating system

In this approach, the operating system is broken up into number of layers. The bottom layer (layer 0) is the hardware layer and the highest layer (layer n) is the user interface layer as shown in the figure.

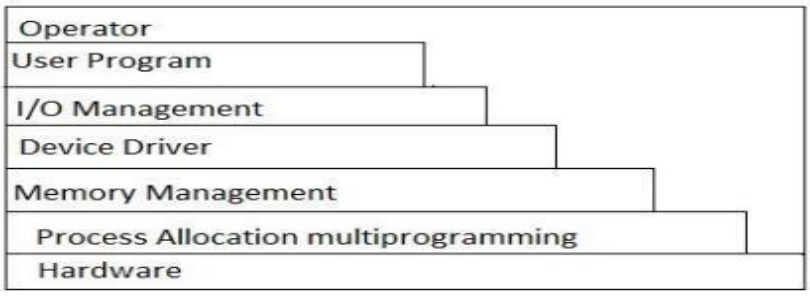


fig:- layered Architecture

The main advantage of the layered approach is modularity. The layers are selected such that each uses functions (operations) and services of only lower-level layers. This approach simplifies debugging and system verifications. That is in this approach, the Nth layer can access services provided by the (N-1)th layer and provide services to the (N+1)th layer. This structure also allows the operating system to be debugged starting at the lowest layer, adding one layer at a time until the whole system works correctly. Layering also makes it easier to enhance the operating system; one entire layer can be replaced without affecting other parts of the system.

Example Systems: VAX/VMS, Multics, UNIX, Os/2, earlier version of Windows NT.

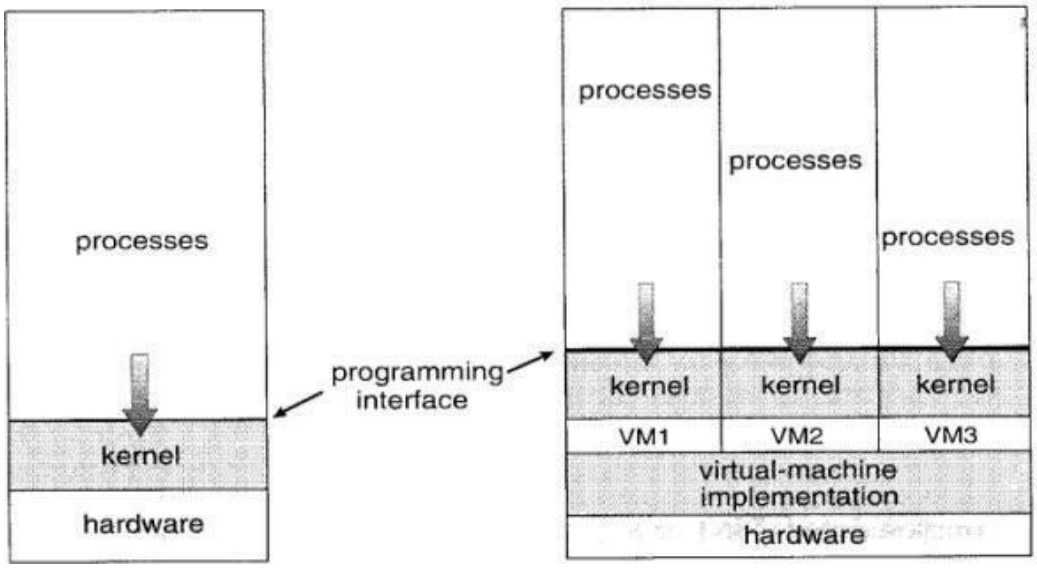
The main disadvantage of this architecture is that it requires an appropriate definition of the various layers and a careful planning of the proper placement of the layer.

The system had 6 layers apart from hardware and their functions are as follows:

- Layer 0 deals with allocation of the processor, switching between processes when interrupts occurred or timers expired.
- Layer 1 deal with the memory management. It allocates space for process in main memory.
- Layer 2 handles communication between each process and the operator console.
- Layer 3 takes care of managing I/O devices and buffering the information streams to and from them.
- Layer 4 is the one where the user programs are found. This layer does not have to worry about process, memory, console or I/O management.
- The system operator process is located in layer 5.

3.3. Virtual memory architecture of operating system

Virtual machine is an illusion of a real machine. Each process is provided with a (virtual) copy of the underlying computer. The resources of the physical computer are shared to create the virtual machine. CPU scheduling can be used to share the CPU and to create the appearance that users have their own processors.



The best example of virtual machine architecture is IBM 370 computer. In this system each user can choose a different operating system. Actually, virtual machine can run several operating systems at once, each of them on its virtual machine. Its multiprogramming shares the resource of a single machine in different manner.

3.4. Client/server architecture of operating system

A trend in modern operating system is to move maximum code into the higher level and remove as much as possible from operating system, minimizing the work of the kernel. The basic approach is to implement most of the operating system functions in user processes to request a service, such as request to read a particular file, user send a request to the server process, server checks the parameter and finds whether it is valid or not, after that server does the work and send back the answer to client server model. It works on request- response basis. The figure below shows client server architecture.

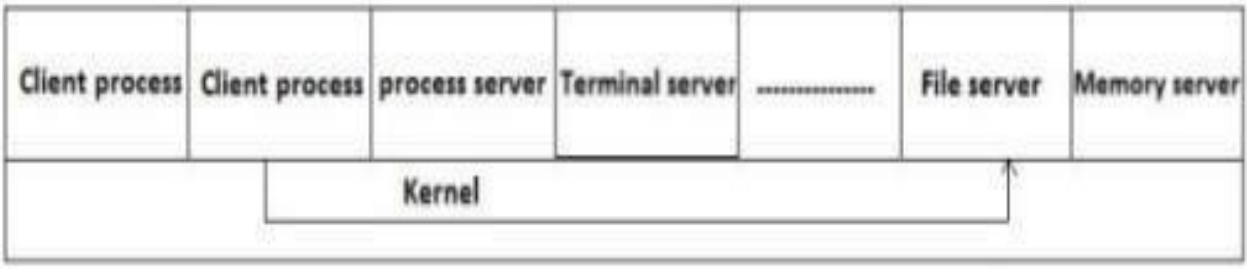


fig:- The client- server model

In this model, the main task of the kernel is to handle all the communication between the client and the server by splitting the operating system into number of ports, each of which only handle some specific task i.e. file server, process server, terminal server and memory service. Another advantage of the client-server model is it's adaptability to user in distributed system.

4. Evolution of operating system

Operating systems have been evolving through the years. The history of OS is closely tied to the architecture of the computers on which they run.

4.1. Serial Processing (First Generation, 1945-55)

In earlier computer system, from the late 1940s to the mid-1950s, the programmer interacted directly with the computer hardware, because there was no operating system. A single group of people designed, built, programmed, operated and maintained each machine. These machines were run from a console consisting of display lights, toggle switches, some form of input device and a printer. Programs in machine code were loaded with the input device (e.g. card reader). If an error halted the program, the error condition was indicated by the lights. The programmer could proceed to examine register and main memory to determine the cause of the error. If the program proceeded to a normal completion, the output appeared on the printer.

This system presented two major problems.

- Scheduling
- Set up time

I. Scheduling:

A user may sign up for an hour but finishes his job in 45 minutes. This would result in wasted computer idle time, also the user might run into the problem if not finish his job in allotted time.

II. Set up time:

A single program involves:

- Loading compiler and source program in memory
- Saving the compiled program (object code)
- Loading and linking together object program and common function

Each of these steps involves the mounting or dismounting tapes on setting up punch cards. If an error occur user had to go to the beginning of the set up sequence. Thus, a considerable amount of time is spent in setting up the program to run. This mode of operation is turned as serial processing, reflecting the fact that users access the computer in series.

4.2. Simple batch system (Second Generation, 1955-65)

To overcome the problems of serial processing batch operating system was developed. The first batch operating system was developed in the mid-1950s by General Motors for use on an IBM701.

In such system, a user doesn't have direct access to the machine. The user submits the job on cards or tape to a computer operator, who batches the jobs together sequentially and places the entire batch on an input device for the use by the monitor. Each program is constructed to branch back to the monitor when it completes processing, at which point the monitor automatically begins loading the next program. In such system the processor is often idle. The problem is that I/O devices are slow compared to the processor.

4.3. Multiprogramming (Third Generation, 1965 – 1980)

Multiprogramming allows the processor to handle multiple batch jobs at a time, multiprogramming can be used to handle multiple interactive jobs. Multiprogramming increases CPU utilization by organizing jobs in such a manner that CPU has always one job to execute. If computer is required to run several programs at the same time, the processor could be kept busy for the most of the time by switching its attention from one program to the next. Additionally I/O transfer could overlap the processor activity i.e, while one program is waiting for an I/O transfer, another program can use the processor. So CPU never sits idle or if comes in idle state then after a very small time it is again busy.

4.4. Fourth generation operating system

The operating system such as time sharing, real-time operating system, distributed operating system, etc. were developed in 4th generation of operating system.

5. Types of operating system

The description of serial processing operating system, batch processing operating system, and multiprogramming operating system are stated above. Apart from them rest of the types of operating system are described below;

5.1. Time Sharing:

The basic technique for a time sharing system is to have multiple users simultaneously using the system through terminals, with the OS interleaving the execution of each user program in a short burst, or quantum of computation.

The first time sharing operating system to be developed was Compatible Time sharing System (CTSS), developed at MIT by a MAC (Machine Aided Cognition, Multiple Access Computers) project. It was developed for the IBM 70 in 1961 and later transferred to an IBM 7094.

5.2. Distributed System:

In distributed system, each processor has its own local memory. The processors communicate with one another through various communication lines such as computer network. A distributed operating system controls and manages the hardware and software resources of a distributed system. When a program is executed on a distributed system, user is not aware of where the program is executed or the location of the resources accessed.

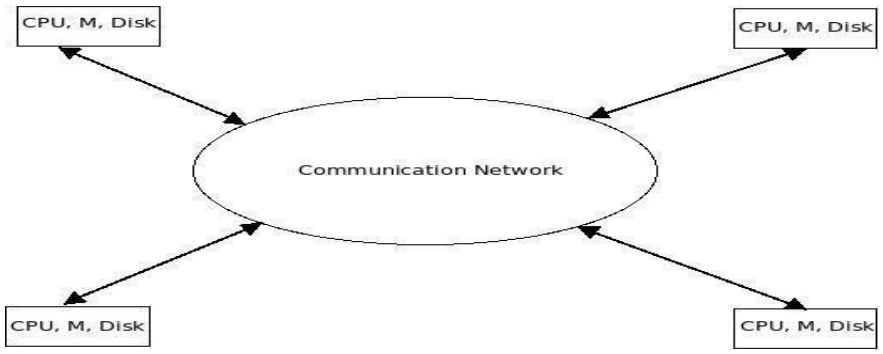


Fig: Architecture of a Distributed system.

Example of Distributed OS: Amoeba, Chorus, Alpha Kernel.

5.3. Multiprocessor operating system:

Multiprocessor operating system aims to support high performance through the

use of multiple CPUs. It consists of a set of processors that share a set of physical memory blocks over an interconnected network. An important goal is to make the number of CPUs transparent to the application. Achieving such transparency is relatively easy because the communication between different applications uses the same primitives as those in uni-processor OS.

The idea is that all communication is done by manipulating data at the shared memory locations and that we only have to protect that data segment against simultaneous access. Protection is done through synchronization primitives like semaphores and monitors.

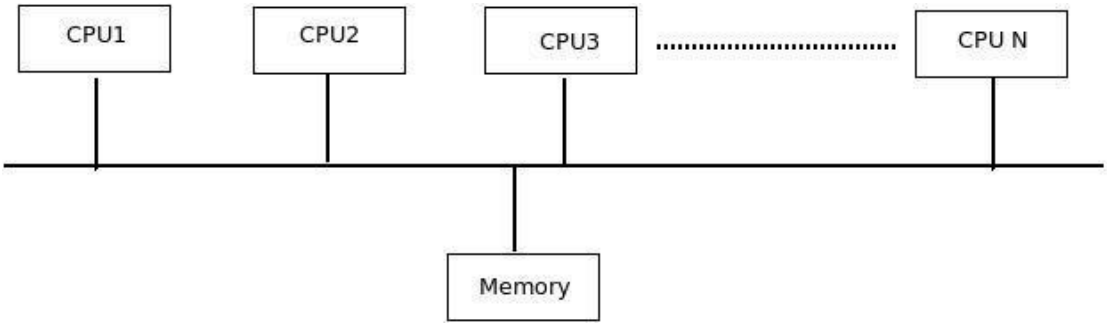


Fig: Multiprocessor System.

5.4. Real Time Operating System:

Primary objective of Real Time Operating System is to provide quick response time and thus to meet a scheduling deadline. User convenience and resource utilization are secondary concern to these systems. A real time system has many events that must be accepted and processed in a short time or within certain deadline. Such applications include: Rocket launching, flight control, robotics, real time simulation, telephone switching equipment etc.

Real time systems are classified into two categories:

- **Soft Real time System:** If certain deadlines are missed then system continues its working with no failure but its performance degrades.
- **Hard Real time System:** If any deadline is missed then system will fail to work or does not work properly. This system guarantees that critical task is completed on time.

6. Features of Operating system

Features of operating system are as follows;

I. Process Management

In multiprocessing environment, operating system allows more than one application (or process) to run simultaneously. Process management is a part of an operating system which manages the processes in such a way that system performance can be enhanced.

A process is an activity that needs certain resources to complete its task. Various computer resources are CPU time, main memory, and I/O devices. These resources are allocated to the processes and based on decision that which process should be assigned for the allocation of resource is taken by process management, implementing the process scheduling algorithm.

It is important to note that a process is not a program. A process is only ONE instant of a program in execution. There are many processes running the same program.

The five major activities of an operating system in regard to process management are:

- Creation and deletion of user and system processes.
- Suspension and re-activation of processes.
- A mechanism for process synchronization.
- A mechanism for process communication.
- A mechanism for deadlock handling.

II. Main-Memory Management

Memory management is the most important part of an operating system that deals directly with both the primary (known as main memory) memory and secondary memory. Main memory provides the storage for a program that can be accessed directly by the CPU for its exertion. So for a program to be executed, the primary task of memory management is to load the program into main memory.

Memory management performs mainly two functions, these are:

- Each process must have enough memory in which it has to execute.
- The different locations of memory in the system must be used properly so that each and every process can run most effectively.

The major activities of an operating system in regard to memory-management are:

- Keep track of which part of memory are currently being used and by whom.
- Decide which processes should be loaded into memory when the memory space is free.
- Allocate and de-allocate memory spaces as and when required.

III. File Management

A file is a collection of related information defined by its creator. Computer can store files on the disk (secondary storage), which provide long term storage. Some examples of storage media are magnetic tape, magnetic disk and optical disk. Each of these media has its own properties like speed, capacity, and data transfer rate and access methods.

A file system is normally organized into directories to make ease of their use. These directories may contain files and other directories. Every file system is made up of similar directories and subdirectories.

The main activities of an operating system in regard to file management are:

- creation and deletion of files/ folders
- support of manipulating files/ folders
- mapping of files onto secondary storage and

- Taking back up of files.

IV. I/O device Management

Input/ Output device management is a part of an operating system that provides an environment for the better interaction between system and the I/O devices (such as printers, scanners tape drives etc.). To interact with I/O devices in an effective manner, the operating system uses some special programs known as device driver. The device drivers take the data that operating system has defined as a file and then translate them into streams of bits or a series of laser pulses (in regard with laser printer).

A device driver is a specific type of computer software that is developed to allow interaction with hardware devices. Typically this constitutes an interface for, communicating with the I/O device, through the specific computer bus or communication subsystem that the hardware is connected with. The device driver is a specialized hardware dependent computer program that enables another program, typically an operating system to interact transparently with a hardware device, and usually provides the required interrupt handling necessary for the time dependent hardware interfacing.

V. Secondary-Storage Management

Secondary storage is considered as permanent memory. Secondary storage consists of tapes drives, disk drives, and other media.

The secondary storage management provides an easy access to the file and folders placed on secondary storage using several disk scheduling algorithms.

The four major activities of an operating system in regard to secondary storage management are:

- Managing the free space available on the secondary-storage device.
- Allocation of storage space when new files have to be written.
- Scheduling the requests for memory access.
- Creation and deletion of files.

VI. Network Management

An operating system works as a network resource manager when multiple computers are in a network or in a distributed architecture. A distributed system is a collection of processors that do not share memory, peripheral devices, or a clock. The processors communicate with one another through communication lines called network, the communication-network design must consider routing and network strategies, and the problems with network and security.

Most of today's networks are based on client-server configuration. A client is a program running on the local machine requesting to a server for the service, whereas a server is a program running on the remote machine providing service to the clients by responding their request.

VII. Protection (User Authentication)

Protection (or security) is the most demanding feature of an operating system. Protection is an ability to authenticate the users for an illegal access of data as well as system. Operating system provides various services for data and system security by the means of passwords, file permissions and data encryption. Generally computers are connected through a network or

Internet link, allowing the users for sharing their files accessing web sites and transferring their files over the network. For these situations a high level security is expected.

If a computer system has multiple users and allows the concurrent execution of multiple processes, then the various processes must be protected from one another's activities. Protection refers to mechanism for controlling the access of programs, processes, or users to the resources defined by a computer system.

VIII. Command Interpreter System

A command interpreter is an interface of the operating system with the user. The user gives commands which are executed by operating system (usually by turning them into system calls). The main function of a command interpreter is to get and execute the user specified command.

Command-Interpreter is usually not a part of the kernel, since multiple command interpreters may be supported by an operating system, and they do not really need to run in kernel mode. There are two main advantages of separating the command interpreter from the kernel.

If we want to change the way the command interpreter looks, i.e., we want to change the interface of command interpreter, then we can do that if the command interpreter is separate from the kernel. But if it is not then we cannot change the code of the kernel and will not be able to modify the interface.

If the command interpreter is a part of the kernel; it is possible for an unauthenticated process to gain access to certain part of the kernel. So it is advantageous to have the command interpreter separate from kernel