Operating System Concepts

Introduction to operating System

- ❖ Definition of OS
- ❖ Structure of OS
- ❖ Objective of OS
- ❖ Operating System Functions
- ❖ Characteristics of operating System

Operating System

- An *operating system* acts as an intermediary between the user of a computer and the computer hardware. The purpose of an operating system is to provide an environment in which a user can execute programs in a *convenient* and *efficient* manner.
- An OS is a program that manages the computer hardware. It also provides a basis for application programs and acts as an intermediary between the computer user and the computer hardware.
- "An operating system (Os) is a set of program that manages computer hardware resources and provides common services for application software".
- "An operating System (Os) is a set of programs that provide interface between user and computer, creates user friendly environment".
- E.g. MAC OS, Microsoft windows 95,98 windows NT/ME/2000, Windows Xp, Windows vista, windows 7,8,8.1,10,unix Linux
- "An Operating system works as a mediator between hardware and user, for the efficient use of hardware and create user friendly environment".
- An operating system is a program designed to run other program on computers.
- Os is a collection of program and utilities.
- It acts as the interface between computer user and computer hardware.
- The operating system is most important type of system software in a computer system.
- A user cannot run application program on the computer without an operating system.

Structure Of Operating System

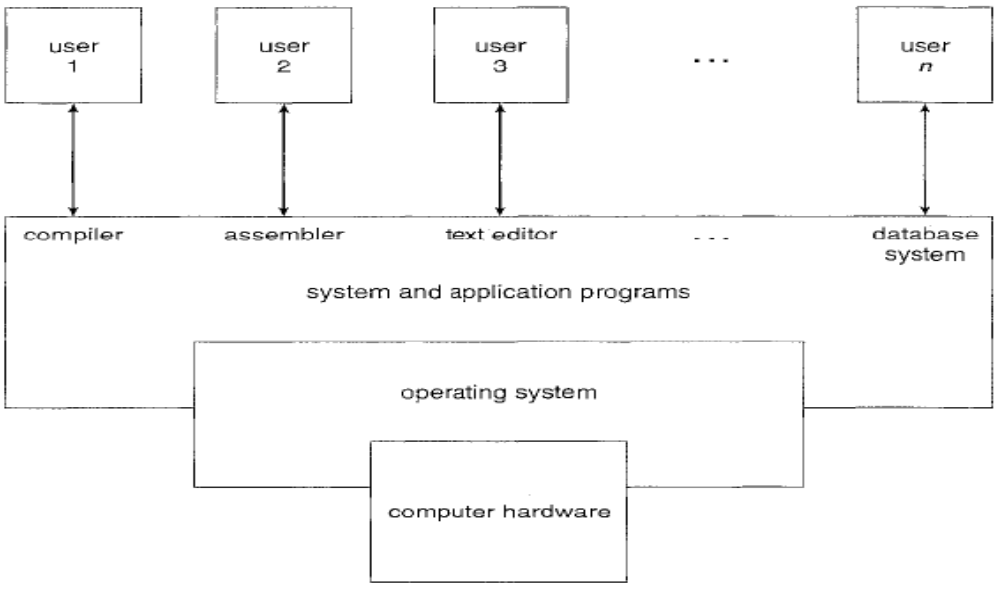


Figure 1.1 Abstract view of the components of a computer system.

- **The User**
  - Users are at the top. Users interact with the system by using application programs to perform particular task
  - Users convenience is operating systems main objective.
- **The application program**
  - It performs a particular task. They use different kinds of functionalities provided by Os to perform their task.
  - It consist compilers, assemblers ,linkers etc depend on the user
- **The Operating System**
  - The next level is for the operating system. It manages all the underlying hardware.
  - It includes process managements routines, Memory management routines,I/O control routines, file management routines etc.
  - In simple terms an interface between application program and the computer hardware
- **The Hardware**
  - It Comes at lower level, It contains various types of physical devices.
  - Eg. CPU, Main Memory, I/o Devices, Secondary Storage etc..

Objective of Operating System

- **Convenience**
  - Make computer user friendly.
  - Make a computer more convenient to use
- **Efficiency**
  - Allows computer to use resources efficiently
- **Ability to grow**
  - Constructed in a way to permit effective development, testing and introduction of new function without interfering with services.
- **To hide the details of the hardware from the user**
- **To act as an intermediary between the hardware and its user and making it easier for the user to access and use other resources.**
- **To manage the computer Resources**

Functions of Operating System

- Program Creation
- Program Execution
- Input/output operations
- Error detection
- Resource allocation
- Accounting resources
- security

- **Program Creation**
  - The operating system provides editors, debuggers, to assist the programmer in creating programs.
- **Program Execution**
  - A number of tasks required to execute a program, the task includes instruction and data must be loaded in main memory for execution, I/o operating system devices, files must be initialized, and other resources must be prepared, the os handles these tasks for the user.
  - The system must be able to load a program into memory and to run that program. The program must be able to end its execution, either normally or abnormally (indicating error).
- **Input/output operations**
  - A running program may require I/0, which may involve a file or an I/0 device.
  - For specific devices, special functions may be desired (such as recording to a CD or DVD drive or blanking a display screen).
  - For efficiency and protection, users usually cannot control I/0 devices directly.
  - The Os handles the task.
- **Error detection**
  - The operating system needs to be constantly aware of possible errors.
  - Errors may occur in the CPU and memory hardware (such as a memory error or a power failure), in I/0 devices (such as a parity error on tape, a connection failure on a network, or lack of paper in the printer), and in the user program (such as an arithmetic overflow, an attempt to access an illegal memory location, or a too-great use of CPU time).
  - For each type of error, the operating system should take the appropriate action to ensure correct and consistent computing.
- **Resource allocation**
  - When there are Multiple users or multiple jobs running at the same time, resources must be allocated to each of them.
  - Many different -types of resources are managed by the operating system
  - Some (such as CPU cycles, main memory, and file storage) may have special allocation code, whereas others (such as I/0 devices) may have much more general request and release code.
- **Accounting resources**
  - we want to keep track of which users use how much and what kinds of computer resources.
  - This record keeping may be used for accounting (so that users can be billed) or simply for accumulating usage statistics.
- **Security**
  - The owners of information stored in a multiuser or networked computer system may want to control use of that information.

- Protection involves ensuring that all access to system resources is controlled.
- Security of the system from outsiders is also important.

Characteristics of Operating System

- **Memory Management**
  - Keeps tracks of primary memory i.e. what part of it is in use by whom, what part is not in use etc. and allocates the memory when a process or program requests.

- **Process Management**
  - Allocates the processor (CPU) to a process and De-allocates processor when it is no longer required.

- **Device Management**
  - Keeps track of all devices. This is also called i/o controller that decides which process gets the device, when, and for how much time.

- **File Management**
  - Allocate and de-allocates the resources and decides who gets the resources
- **Security**
  - Prevents unauthorized access to programs and data by means of password and similar other techniques.
- **Job Accounting**
  - Keeps track of time and resources used by various jobs and/or user.
- **Control over system performance**
  - Records delays between request for a service and from the system.
- **Interaction with operators**
  - The interaction may take place via the console of the computer in the form of instructions.
  - Operating system acknowledge the same, does the corresponding action and informs the operation by a display screen
- **Error detecting aids**
  - Production of dumps, traces error messages and other debugging and error detecting methods
- **Co-ordination between other software and user**
  - Coordination and assignment of compilers, interpreters, assemblers and other software to the various user of the computer system.

Evolution of Operating System

- **hardware upgrades or new types of hardware:**

  - With hardware technologies development, the OS also needs to upgrade so as to utilize the new mechanisms introduced by new hardware. For example, Pentium IV extended instruction set of Pentium III for multimedia applications and internet transmission. An OS designed for the previous versions of Intel x86 series will have to be upgraded to be able to accommodate these new instructions.

- **New services:**
    - An OS may also expand to include more services in response to user demand.
- **Fixes:**
    - No software is perfect, and any program may contain more or less bugs or defects, thus fixes should be made from time to time. Microsoft Windows is a vivid example of this kind.
    - These situations all require OS designers to build an OS in the way that the system can be maintained and upgraded easily.

Evolution of Operating System

- Serial processing
- Batch Operating System
- Multiprogramming operating System
- Time Sharing Operating system /Multitasking OS
- Real time operating System
- Multi-user Operating System
- Multiprocessing operating system/parallel system or tightly coupled
- Distributed operating system
- Client-server operating system
- Clustered system
- Hand held operating system
- Network operating System

Serial Processing

- The earliest computer, from the late 1940s to the mid-1950s, the programmer interacted directly with the computer hardware; there was no operating system.
- These machine were run from a console, consisting of display lights, toggle switches, some form of input devices, and a printer.
- Programs in machine code were loaded via the input device(e.g. card reader)
- If error halted the program the error condition was indicated by the lights.
- If the program proceed to a normal completion, the output appeared on the printer.

- **Serial Processing presented two problems**
- **Scheduling**
  - if a user sign up for an hour and finish in 45 min this would result in wasted of computer idle time.
  - On the other hand , the user might run into problems, not finishing in the allotted time, and be forced to stop before resolving the program.
- **Setup time**
  - A single program called a job, could involve loading the compiler plus the high level language program into memory, saving the compiled program and then loading and linking together the object program and common function.
  - if an error occurred, the hapless user typically had to go to beginning of the setup sequence.
- **Batch systems**
  - Early machines were expensive, and therefore it was important to maximize machine utilization. The wasted time due to scheduling and setup time was unacceptable.
  - The common input device were card readers and tape drivers.
  - The common output devices were line printer, tape drivers and punch cards.
  - The user of such system did not directly interact with the computer systems.
  - The jobs were in form of punch card.

- The os major task was to transfer control automatically from one job to another
- The Os always resides in memory.
- To speed up processing jobs with similar needs were batched together.
- Programmer would prepare a job and submit it to the operator.
- Operator would sort them in batches with similar requirement, and as computer became available run them batch wise.
- At some later time, output appeared. The output consisted of the result of the program and error information. Programmer need to wait at this time and then collect the output from the operator.
- The output from each job would be sent back to the appropriate programmer by the operator.
- Here, memory is divided in two parts as given in figure; it is shared between the Os and the job.

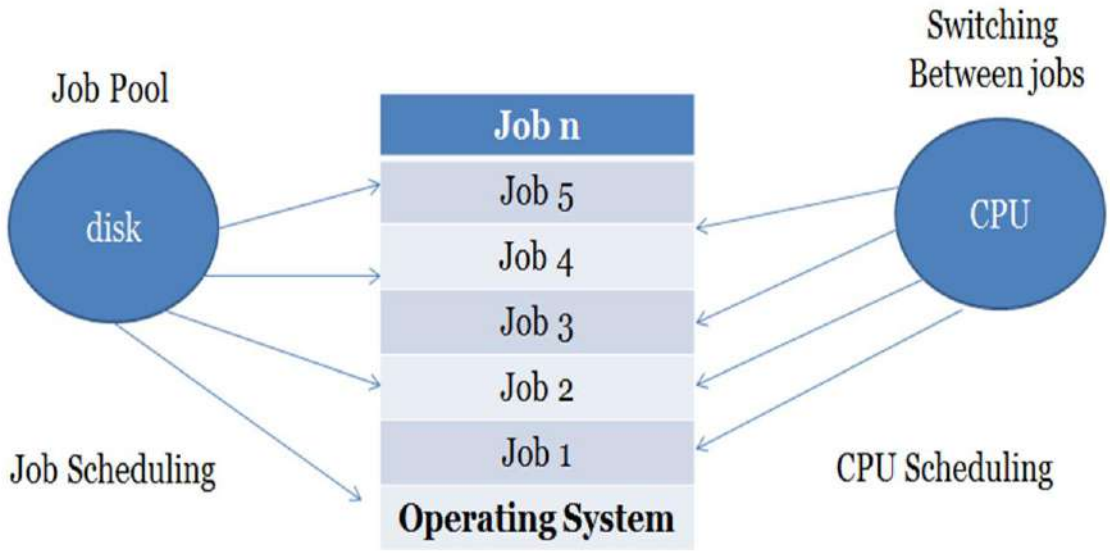
- At a time one job is selected out of the batch of jobs, and is loaded into memory for execution.
- Once its execution completes, another job will be selected and loaded into memory for execution. This process will continue untill all jobs in a batch get executed.

- **Turnaround time:-**
  - The delay between job submission and job completion called as turnaround time.
- **Disadvantage**
  - Turnaround time can be large from user standpoint
  - Difficult to debug program
  - Low throughput: because of, Cpu remains idle when I/O is going on
  - Programmers do not have direct interaction with job

- Debugging is possible only offline, after output appears
- Operation were too much time consuming
- A job could corrupt the monitors, thus affecting pending jobs
- Due to lack of protection scheme, one batch job can affect pending jobs.

- **Multiprogramming operating System**
  - Several jobs are kept in main memory at the same time, and the cpu is multiplexed among them which require memory management and protection.
  - The Os picks and begins to execute one job from memory. Once this job needs an operation the Os switches to another job(Cpu or Os is always busy).
  - A multiprogramming Os provides the ability to run more than one program concurrently by a single processor.
  - When two or more programs are in memory at the same time, sharing the processor is referred to the multiprogramming Os.
  - In multiprogramming number of process are residing in main memory at a time.

- Diagram:-



- The number of jobs in memory is less than the number of jobs in disk(job pool).

- If several jobs are ready to be brought into memory and there is not enough room for all of them, then system chooses job among them (job scheduling).
- If several jobs are ready to run at the same time, the system must choose among them (CPU Scheduling).

Implementation of Multiprogramming Os

- **Multiprogramming can be implemented as;-**
- Non Preemptive
- Preemtive
- **Non pre-emptive**
  - A program is allowed to execute until it voluntarily gives up the CPU
  - A program voluntarily gives up the CPU when it waits for some event, such as I/O operation, or when it terminates.
  - Once a CPU becomes free, it can be allocated to some other program.
- **pre-emptive**
  - A program is allowed to execute only for some maximum time duration.
  - After this time duration, A CPU is taken away from the program.
  - Hence one smaller program need not to wait for other large program to finish its execution.
- **Disadvantages**
  - User Cannot interact with their jobs, while executing.
  - A programmer cannot modify a program as it executes to study its behaviour.
- **Advantages**
  - Can give effective memory utilization
  - CPU is never idle, so the performance of CPU will increase
  - Throughput of the CPU may also increase.
  - Waiting time is limited for processes.

- **Time Sharing Operating system /Multitasking OS**
    - Logical extension of multiprogramming.
    - The CPU executes multiple jobs by switching among them, but the switches occur so frequently that the user can interact with each program while it is running.
    - An interactive computer system provides direct communication between the user and system.
    - A time shared Os allows many user to share the computer simultaneously.
    - A time shared Os uses CPU scheduling and multiprogramming to provide each user with small portion of a time-shared computer.
    - Each user has at least one separate program in memory.
    - A program loaded into memory and execution is commonly referred to as process.
    - Time sharing Os are even more complex than multi programmed Os.
    - The time slot is defined by the Os.
    - Time sharing system can run several programs at the same time, so it is also multiprogramming system. But multiprogramming is not a time sharing system.
    - The idea of time sharing was demonstrated as early as 1960 but since time shared systems are difficult and expensive to build they did not become common until the early 1970s
    - E.g. CTSS, Multics, Cal, Unix
- **Advantages**
    - Efficient CPU utilization
    - User can interact with the job when it is executing, that was not possible in batch Os.
- **Real time Operating System**
    - A System is said to be real time if it is required to complete it's work & deliver it's services on time.

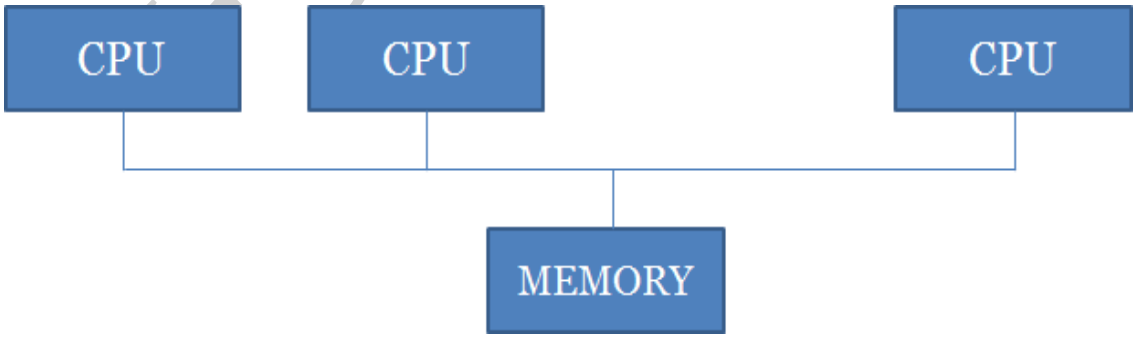
- A real time system is used when rigid time reqirements have been placed on the operation of processor or the flow of data;
- A real time system has well defined, fixed time constraints
- Process must be done with in the defined constraint, or the system will fail.
- Real time Os is one that must react to inputs and respond to them quickly. A real time system cannot afford to be late with a response to an event.
- E.g. flight control system; all tasks in that system must execute on time.
- **Hard Real time**
    - All critical task must get completed strictly within the specified time limits.
    - Tasks are guaranteed to occur in time.
    - This goal requires all delays in the system be bounded, from the retrieval of stored data to the time that it takes the operating system to finish any request made of it.
- **Soft Real time**
    - It is less restrictive type, missing an occasional dead line is acceptable.
    - A critical real time task gets priority over other task, and retains that priority until it completes.

Real Time Application	Examples
Detection	Radar System, Burglar System
Process monitoring and control	Petroleum refinery, paper mill
Communication	Telephone switching system
Flight simulation and control	Auto pilot shuttle mission simulator
Transportation	Traffic light system, Air traffic Control

- **Multi-user Operating System**
  - Single user Os allow single user to access computer system i.e. multiple user can't use single at a time.
  - Contrast to this, multi-user Os allow multiple user to access computer system at a same time.
  - Access to computer system is normally provided via network, so that user to access computer remotely using terminal or other computer.
  - A terminal contains only I/O devices such as keyboard and monitor. It is used to provide interaction between user and computer system
  - A touch screen help desk provided at railway station or an ATM machine provided at a bank is an example of such kind of terminal.
  - This can be implemented through time sharing and multiprogramming.
  - CPU is time sliced at a regular intervals among various users.
  - This kind of Os is much more complex then single user operating system.
- **Advantage**
  - Expensive hardware can be shared among several users using multi user operating system makes better utilization.
- **Disadvantage**
  - As more user access it, the performance becomes lower and slower.
  - Cost of hardware and software more expensive than single user Os.

Multiprocessing operating system/parallel system or tightly coupled

- Multiprocessing system have more than one processor in close communication, sharing the computer bus, clock, and some times memory and peripherals devices.
- All program running in parallel manner.
- Multiprogramming operating system allows more than one program to run concurrently on a uni-processes system. But a single processor can't execute more than one instruction at a time.
- **Multiprogramming is of two types**
    - Symmetric multiprocessing
    - Asymmetric multiprocessing
- **Symmetric multiprocessing**
    - Each processor runs an identical copy of operating system and they communicate with one another as needed.
    - All the CPU Shared the common memory.
    - Many processes run at once
    - SMP means that all processors are peer; no master slave Relationship exist between processors.
    - Each processor concurrently runs a copy of the operating system.
    - Virtually all modern operating systems support SMP, including Windows XP, Windows 2000, Solaris, Linux, and Mac OS X



- **Asymmetric multiprocessing**
    - Each processor has an assigned task

- Master processor controls the system schedule and allocate work for slave processors
- ASMP uses Master slave relationship.
- Master performs I/O and computations
- Only master may execute the Os. Slave can execute only user programs

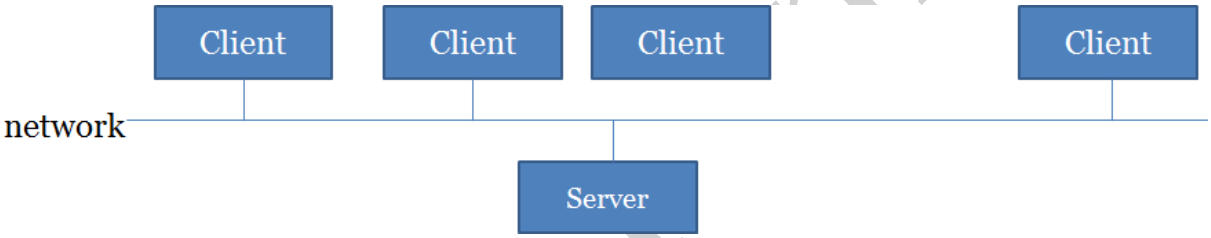
- **Features of multiprocessing Os**
  - If one processor fails, then the another processor should retrieve the interrupted process state so that execution of the process run continue.
  - The processor should support efficient context switching operation
  - Multiprocessor system support large physical address space and large virtual address space.
  - The IPC mechanism should be provided and implemented in hardware as it becomes efficient and easy.

- **Advantage of Multiprocessor**
  - Increased throughput
  - Economy of scale
  - Increased reliability

- **Distributed Os**
  - Distributed system depends on networking for their function ability.
  - By being able to communicate, distributed systems are able to share computational task.
  - Distributed Os Distribute the computation among several physical processors.
  - When there is higher load on one processor, this Os automatically distributes the computation on other machine, where the processor is idle or having low load.

- **Advantage of Multiprocessor**
  - Resource Sharing
  - Higher Reliability
  - Better price performance ration
  - Incremental growth

- **Client-server operating system**
    - Centralized system today act as server system to satisfy requests generated by client system.
    - Server system can be broadly categorized as compute server and file server.
- **Computer-server system**
    - Provides an interface to which client can send requests to perform an action, in response to which they execute the action and send back result to the client.
- **File-Server System**
    - Provide a file system interface where clients can create update, read, and delete files.



- **Peer to peer System**
    - Each computer may act as client or server depending on request & respond.
    - Service provided by several nodes distributed on network.
    - These systems are usually referred as loosely coupled system
- **Clustered system**
    - Clustered system gather together multiple CPUs to accomplish computational work.
    - Clustering allows two or more system to share storage
    - Provides high reliability
    - **Asymmetric clustering:-** one server runs the application while other servers are at standby
    - **Symmetric clustering:-** all N host are running the application or applications
    - Like multiprocessor system, clustered system gather multiple CPUs to accomplish computational work

- Clustered system differ from multiprocessor system , how ever, in that they are composed two or more individual system coupled together.

- **Asymmetric mode**
  - In asymmetric clustering, one machine is in hot-standby mode while the other is running the applications
  - The hot standby host machine does nothing but monitor the active server
  - If that server fails, the hot-standby host becomes active server.
- **Symmetric mode**
  - In symmetric mode, two or more hosts are running applications, and are monitoring each other
  - This mode is obviously more efficient, as it uses all of the available hardware
  - It does require that more than one application be available to run.
- **Handheld System**
  - **Personal Digital Assistants (PDAs)**
  - **Cellular Telephones**
  - **Issues:**
    - **Limited memory**
    - **Slow processors**
    - **Small display Screens**
- Handheld System include personal digital assistants (PDAs), Such as palm and pocket-Pcs, and cellular telephones, many of which use special-purpose embedded operating system.
- Developers of handheld system and applications face many challenges, most of which are due to the limited size of such devices.
- Because of their size, most handheld devices have a small amount of memory, show processors, and small display screens

Needs of Operating system

- Management of the process
- Management of Random Access Memory
- Management of Input/output
- Management of execution of Application
- Management of Authorization
- File management
- Information Management
- Convenient & efficient use of computer hardware
- **Management of the Processor**
    - The operating system is responsible for managing allocation of the processor between the different programs using a scheduling algorithm.
    - The type of scheduler is totally dependent on the operating system, according to desired objective.
- **Management of Random Access Memory**
    - The operating system is responsible for managing the memory space allocated to each application and where relevant, to each user.
    - If there is insufficient physical memory, the operating system can create a memory zone on the hard drive known as “Virtual memory”.
    - The virtual memory lets you run applications requiring more memory than there is available RAM on the System.
- **Management of Input/ Output**
    - The operating system allows unification and control of access of programs to material resources via drivers (also known as peripheral administrators or input/ output administrators).
- **Management of execution of application**
    - The operating system is responsible for smooth execution of application by allocating the resources required for them to operate.
    - This means an application that is not responding correctly can be “killed”.

- **Management of authorization**
    - The operating system is responsible for security relating to execution of programs by guaranteeing that the resources are used only by programs and user with the relevant authorization.
- **File management**
    - The operating system manages reading and writing in the file system and the user and application fill access authorizations.
- **Information Management**
    - The operating system provides a certain number of indicators that can be used to diagnose the correct operation of the machine.
- **Elements of an Operating System**
    - Kernel
    - Shell
    - File System
    - Memory management
- **Kernel**
    - The kernel is heart of the operating system.
    - The kernel, which represents the operating system's basic function such as management of memory, processes, files, main inputs/ outputs and communication functionalities.
    - Among its responsibilities are ensuring that each running process is given a fair amount of time to execute while a controlling the amount of resources each process can use.
- **Shell**
    - The Shell, allowing communication with the operating system Via a control language letting the user control the peripherals without knowing the characteristics of the hardware used, management of physical addresses, etc.
    - A shell is a user interface for access to an operating system's service.
    - In General, OS shell use either a command line interface(CLI) Or Graphical user interface(GUI)

- **File System**
  - The file system , allowing files to be recorded in a tree structure.
- **Memory management**
  - The name of this layer gives you a good idea what it is all about.
  - It is the responsibility of this layer to share your computers physical memory among the processes which want to use it.
  - It also has to manage such situation where there may not be enough physical memory to share out.
- **Operating System as Resource manager**
  - Resource – "Something Valuable". E.g. The Os is responsible for managing these resources. CPU, Memory Space(RAM), File Storage Space, I/O Devices.
  - A computer is a set of resources for the movement, storage and processing of data and for control of these function
  - A portion of Os is in main memory. This includes The kernel, which contains most frequently used function in Os
  - The allocation of this resource is controlled jointly by the Os and memory management hardware in the processor.
  - Os directs the processor in the use of the other system resources and in the timing of its execution of the other program.
  - A portion of Os is in main memory. This includes the kernel which contains the most frequently used function in the Os and at a given time, other portions of the Os currently in use.
- **Operation performed by resource manger**
  - Resource allocation– Os allocates resources to multiple user and multiple jobs running at same time.
  - Operation control program:- Os is a program that controls the execution of user programs and operations of I/O devices.
  - System Access:- Os ensures that all access to resource is protected, including authorization etc.
  - Accounting and usage Statistics:- Monitoring System Data and Resource usage.

- **The Os function in two ways:**
    - The Os Function in the same way as ordinary computer software , that is, it is program or suit of program executed by the processor.
    - The Os frequently relinquishes control and must depend on the processor to allow it to regain control.

History of Operating System

Generation	Year	Electronic Devices Used	Types of Os and Devices
First	1945-55	Vacuum Tubes	Plug boards
Second	1955-65	Transistors	Batch System
Third	1965-80	Integrated Circuit(IC)	Multiprogramming
Fourth	Since 1980	Large Scale Integration	PC

- **First Generation**
    - The earliest electronic digital computers had no operating systems. Machines of the time were so primitive that programs were often entered one bit at time on rows of mechanical switches.
    - Programming languages were unknown.
- **Second Generation**
    - By the early 1950's the routine had improved somewhat with the introduction of punch cards. The General motors research laboratories implemented the first operating system in early 1950's for IBM701.
    - The system of the 50's generally run one job at a time. These were known as single stream batch operating system
- **Third Generation**
    - The system of 1960's were also batch processing systems, but they were able to take better advantages of the computer's resource by running several jobs at once. So operating systems designers developed the concept of multiprogramming in which several jobs are in main memory at once
    - Operating system designers developed the concept of multiprogramming in which several jobs are in main memory at once; a processor is switched from one job to another job as needed to keep several jobs advancing while keeping the peripheral devices in use

- A major featured in third generation Os was the technique called Spooling(**simultaneous peripheral operation on line**). In spooling a high speed device like a disk interposed between a running program and a low speed device involved with the program in input/ output.
- Instead of writing directly to a printer outputs were written to a disk.
- Programs can run to completion faster, and other programs can be initiated sooner when the printer becomes available, the outputs may be printed
- Spooling technique is much like thread being spun to a spool so that it may be later be unwound as needed.

- **Fourth generation**
  - With the development of LSI (Large Scale integration) Circuits, Chips, OS entered in the system entered in the personal computer and the workstation age.
  - Microprocessor technology evolved to the point that it become possible to built desktop computers as powerful as the mainframes of the 1970s.
  - Two operating systems have dominated the personal computer scene: MS-Dos, Written by Microsoft, Inc. for the IBM PC and other machines using the Intel 8088 CPU and its successors, and UNIX, which is dominant on the large personal computers using the Motorola 6899 CPU family.

Unit 2. Introduction to File System and File Management

2.1. File Concept
2.2. Operations on File
2.3. File Access Methods (Sequential Access and Direct Access)
2.4. Directory Systems File Management Functions.
2.5. File System and Directory Structure organization.
2.6. File Protection.

File Concept

- Computers can store information on several different storage media, such as magnetic disk and optical disk.
- The operating system abstracts from the physical properties of its storage devices to define a logical storage unit(the file).
- "A file is a named collection of related information, which is recorded on a secondary storage device".
- A file is used to store information permanently. It is also suitable to store very large information.
- A file is a smallest allotment of logical secondary storage; that is, data cannot be written to secondary storage unless they are within a file.
- Data files may be numeric, alphabetic, alphanumeric or binary.
- The information in file is defined by its creater
- "A file is a sequence of bits, bytes, lines or records the meaning of which is defined by the files creator and user.
- A file is used to store various kinds of information, such as text, image, audio, video, database tables, and machine codes and so on.
- A file is an independent entity from processes, user and machines. Information is stored in files remain as it is even when-
    - A process, which creates file, terminates
    - User, owner of file, logs off from the system; or
    - Machine gets switch of.
- A file has certain defined structure according to its type
    - A text file is a sequence of characters organized into lines.
    - A source file is a sequence of subordinates and functions, each of which is further organized as declaration followed as executable statements.
    - An object file is a sequence of bytes organized into blocks understandable by the system linker
    - An executable file is a series of code section that the loader can bring into memory and execute

- A file is stored on non volatile secondary storage devices as hard-disk, CD-ROMs, Magnetic taps.
- A file has longer life span. Once program is created and stored on disk it remains there till it is not deleted.

File attribute

- ✓ Name
- ✓ identifier
- ✓ Type
- ✓ Location
- ✓ Size
- ✓ Protection
- ✓ Time
- ✓ Date
- ✓ User Identification

- Name
  - The symbolic file name is the only information kept in human readable form.
  - A string of alpha-numeric characters and some special characters like underscore(‘_’)
  - It is used by user to refer files in a convenient way
- Identifier
  - This unique tag, usually a number, identifies the file within the file system; it is non human readable name for the file.
- Type
  - This information is needed for those systems that support different types
  - It is used to identify the type of a file.
  - It is generally expressed in the form of a file extension. E.g.
    - .exe: executable files
    - .obj: object files

- .src: Source files
- Txt: textual files

- Location:-
  - This information is a pointer to a device and to the location of the file on that device.
- Size:-
  - The current size of the file(in bytes, words, or Blocks), and possibly the maximum allowed size are included in this attribute.
- Protection:-
  - Access-Control information determines who can do reading, writing, executing and so on.
  - It specifies the access control information.
- Time:-
  - It specifies the time of file creation or modification
- Date:-
  - It specifies the date of file creation and modification
- User Identification:-
  - This information may be kept for protection, security and usage monitoring

File Types

File type	Extension	Purpose/Function
Executable	.exe .com .bin	Ready to run machine language program
Source code	.c .cpp .pas .asm	Source code in various languages
Batch	.bat .sh	Commands to the command interpreter
Text	.txt .doc	Textual data , documents
Library	.lib .a	Libraries of routines for programmers
Point (or) View	.ps .dvi .sif	ASCII or binary file in a format for printing or viewing
Archieve	.arc .zip .tor	Grouped files, compressed files or archiving or storage
Object	.obj .o	Compiled, machine language, not linked
Word processor	Wp, tex, rrf, doc	Various word processor formats
Multimedia	Mpeg, mov, rm	Binary file containing audio or A/V information

Operation on file

- A file is an abstract data type.
- The operating system can provide system calls to create, write, read, reposition, delete, and truncate.
- Files are used to store information. This information need to be retrieved later. It is the responsibility of OS to provide ways to perform various operations on files.
- Operating system generally provide a set of system calls to perform various operations on files. This system calls can be called using library function provided by programming language.

Operations on file

- ✓ Creating a File
- ✓ Deleting a file
- ✓ Opening a file
- ✓ Closing a file
- ✓ Reading a file
- ✓ Writing a file
- ✓ Renaming a file
- ✓ Repositioning with in a file
- ✓ Truncating a file
- ✓ Creating a File:-
    - Two steps are necessary to create a file.
        - First, space in the file must be found for the file
        - Second an entry for the new file must be made in the directory;
- ✓ Deleting a file :-
    - To delete a file, we search the directory for the named file.
    - When a file is no longer needed, It has to decide to free up the disk space
    - First, all the file space on the disk is released
    - Second, directory entry is erased

- ✓ Opening a file
  - To use a file user must open the file
  - File attributes and data contents are fetched in main memory for rapid access on later calls
- ✓ Closing a file:
  - Once use of file is completed or finished, it should be closed to free main memory space.
  - File attributes and data contents are stored back on disk. This may contain modified information if file is updated.
- ✓ Reading a file
  - To read from file, we use a system call that specifies the name of the file and where the next block of the file should be put.
  - The system maintains the 'read' pointer. It specifies the location in a file from where to read data contents
  - The read pointer is updated automatically.
  - Data are read from file.
- ✓ Writing a file:-
  - To write a file, we make a system call specifying both the name of the file and the information to be written to the file
  - Data are written to file
  - The system maintain the 'write' pointer. It specifies the location in a file from where to read data contents
  - The system must keep a write pointer to the location in the file at where next write is to take place.
- ✓ Renaming a file
  - This operation is used to change the name of an existing file.
- ✓ Repositioning with in a file
  - The directory is searched for the appropriate entry, and the current file position is set to given value

✓ Truncating a file:

- The user may want to erase the content of a file but keep its attributes.
- Rather than forcing the delete the file and then recreate it, the function allows all attributes remain unchanged.

File Access Methods

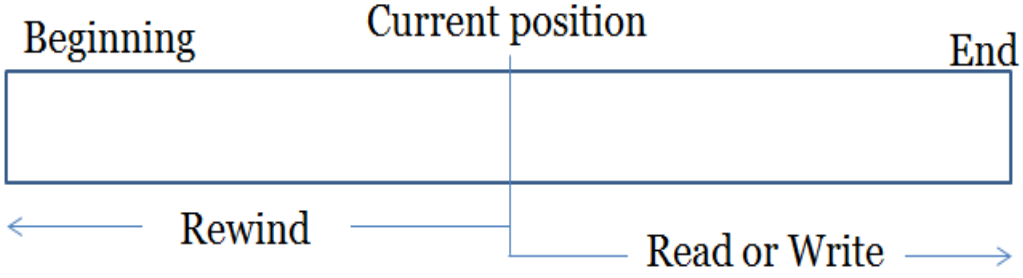
- Files store information, this information must be accessed and read into computer memory.
- There are so many ways that the information in the file can be accessed.

Basic types of File Access Methods

- **Sequence Access Method**
- **Direct Access Method**
- **Other Access Method**

✓ Sequence Access Method

- This method is simplest method.
- Information in the file is processed in order, one record after the other.
- E.g. Editors and compilers usually access file in this method
- The bulk operation on file is read and write
- A read operation reads the next portion of the file and automatically advances a pointer which tracks next I/O location.
- A write appends to the end of the file and advances to the end of the newly written material
- One file pointer is maintained here. It specifies the location of next read write operation in a file.
- Diagram for Sequential file access method.



- Sequential Access method is based on a tape model of a file, and works as well on sequential devices as it does on random access ones.

→ **Advantage:-**

- Simplicity:- one of the simplest access methods

→ **Dis-advantage:-**

- Inefficient :- an average access time of record in a file is equal to the time to access half of the file.

Direct Access Method

- Direct Access method is also known as Relative Access Method.
- A file is made up of fixed length logical records that allow programmers to read and write records rapidly in no particular order
- The direct access method is based on disk model of a file, since disks allow random access to any file block.
- For the direct access method, the file operation must be modified to include the block number as a parameter.
- We have read n, where n is the block number, rather than read next, and write n rather than write next.
- File is viewed as numbered sequence of blocks or records of fixed length.
- It is necessary to provide a block number with read, write and other operations. This is called relative block number.
- It starts from '0' or '1'. It is relative to the beginning of a file.
- This method is used for many applications such as database application. E.g. Air line reservations, Railway reservation

✓ **Advantages:-**

- Efficient: Any record can be accessed directly. It is not necessary to read entire file to access some particular record as in sequential access

✓ **Disadvantage:**

- User may not know the record number to be accessed from a file.

- **Other Access Method**
  - Other access method can be built on top of a direct access method.

- These method generally involved the construction of an index for the file.
- The index like an index in the back of a book, contains pointers to the various blocks. To find a record in the file, we first search the index and then use the pointer to access the file directly to find the desired row.

Directory Systems File Management Functions.

- Today, computer system comes with the disks having storage capacities in gigabytes and terabytes. They are used to store very large amount of information in various files.
- To manage all these files, they should be organized in a well manner.
- This organization is generally done in two parts:-
- First disks are split into one or more partitions, known as drives or local devices.(Like 'C:'.'D','E' in windows) these drivers are used to store files and directories.
- Second Each partition contains various directories
- A directory is a container for other files and sub directories.
- In GUI based system directories are generally represented as folder.
- Directories are used to group and organize files & other directories, called sub directories
- Directories provide hierarchical file structure
- A directory is a file whose data is a sequence of entries. Each entry contains a file name and some other information about a file such as file attributes
- Directories are used to store information about other files and subdirectories.
- This information need to be retrieved later when any file accessed.

✓ Directory operations

- Create
- Delete
- Read
- Write
- Open
- Close

- Rename
- Link

✓ **Directory operations**

- Create:- A new directory is created
  - Initially it contains two special entries: dot '.' and dotdot '..' to refer to current and parent directory respectively.
- Delete:- An existing directory is deleted
  - Generally a directory is deleted only if it is empty. But the rule can be violated
- Read:- directories can be read
  - For example, to list out all the files in a directory, directory entries are read to get file names.
  - A dir command in ms-dos and 'ls' command in Unix performs such operation.
- Write:- Data are written to directories also
  - When a new file is created, a new directory entry is created in a directory.
  - Information regarding a new file, such as file name and file identifier are written to that directory entry.
  - When any existing file is deleted entry related to that file is erased.
  - When any file is modified, a directory entry is updated to store that file.
- Open:- Before using a directory, a process must open it.
  - Data contents are fetched in main memory for rapid access on later calls.
  - Memory space in main memory is allocated to store fetched information
- Close:- when use of directory is finished, it should be closed to free up main memory space
  - Data contents are stored back on disk. This may contain modified information if directory is updated.
- Rename:- This operation is used to change the name of an existing file.

- Link:- Linking is a technique that allows a file to appear in more than one directory.
- A same file may appear in multiple directory

✓ Directory structure:-

- Directory structure refer to the way how directories and files are organized.
- File structure is a structure, which is according to a required format that operating system can understand.
- A file has a certain defined structure according to its type.
- A text file is a sequence of characters organized into lines.
- A source file is a sequence of procedures and functions.
- An object file is a sequence of bytes organized into blocks that are understandable by the machine.
- When operating system defines different file structures, it also contains the code to support these file structure. Unix, MS-DOS support minimum number of file structure.

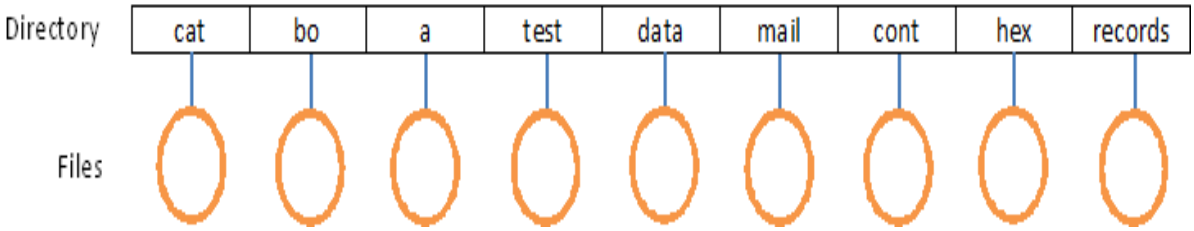
✓ Five different types of directory structure

- Single – level directory
- Two-level directory
- Tree structured directories
- Acyclic graph directory
- General graph directories

✓ Single – level directory

- This is simplest directory structure.
- All files are contained in same directory. Some times known as root directory
- Directories can only contain file(No sub directories).
- This type of structure was used in early personal computers.
- Even first super computer the CDC 6600, used this type of directory structure for simplicity.

- The following figure represents this directory structure. In this figure, directory is represented using rectangle while files are represented using circle. Here single directory contains nine different files(names written in rectangle are file names



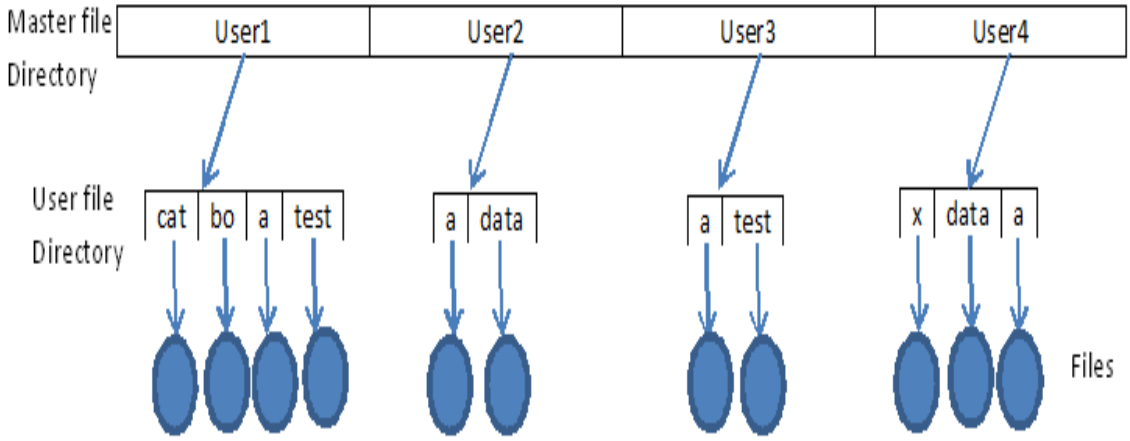
- Advantage
  - This type of structure is very simple
  - As there is only one directory, searching a file is very quick.
- Disadvantage
  - It is not suitable for multi-user system. Different user may provide same file name for different files, possibly overwriting other's files. This is called a file name collision
  - It is even not suitable for sing user system when number of files becomes too large. User cannot remember the names of files.
  - Different files cannot be grouped

✓ **Two Level directory**

- It gives each user a private directory.
- This is to avoid name collision among different users.

✓ **Two levels of directories are used here:**

- First one is a root directory or master file directory(MFD).
- Second are user file directory(UFD) or simply, user Directory.
- User files can contain only files. They cannot have subdirectories.
- Figure below represents the directory structure. In this figure, directories are represented by rectangles and files are represented using circle.
- Here root directory contains four sub-directories related to four different user named user1, user2, user3 and user4.
- User1 directories contain four different files. In similar way directories user2, user3 and user four contain 2 to 3 files respectively.

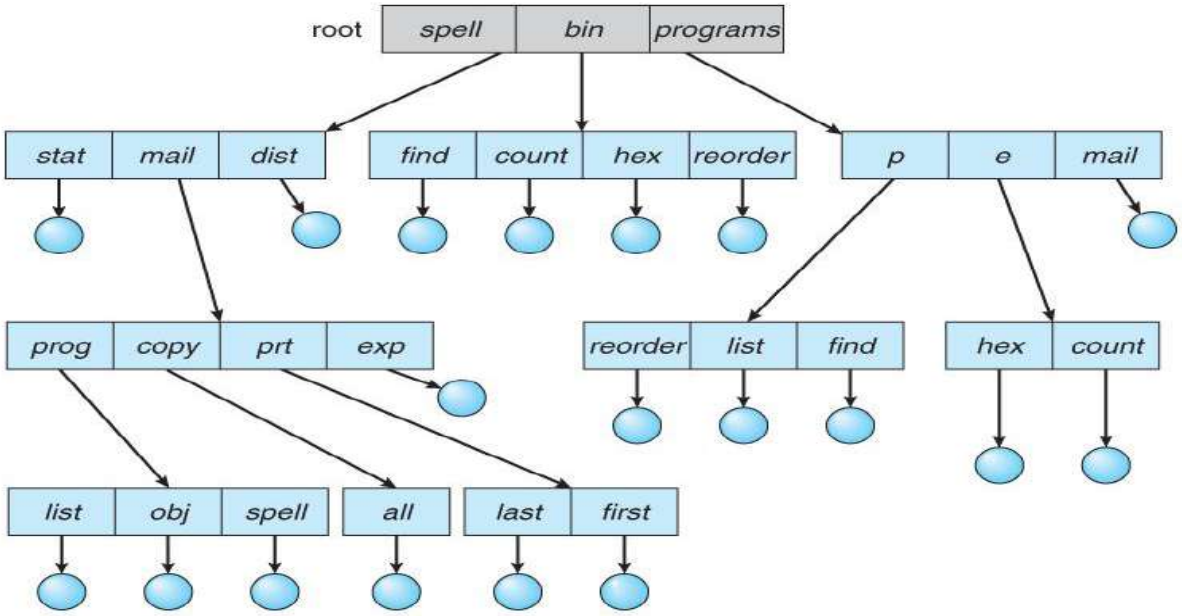


- **Advantage:**
  - It avoid file name collisions
  - It can be used in multi-user system
  - Searching is efficient here; as user has to search in its own single directory.
- **Dis-Advantage:**
  - It is not suitable for user having large number of files.
  - Different files cannot be grouped
  - One extra system directory is required to store system files

Tree Structured Directories:

- It permits user to create own sub directories
- This is also called a hierarchical file system. More than two levels of directories are possible
- Directories can contain file as well as sub directories also
- Each file contains a unique file path
- An obvious extension to the two-tiered directory structure, and the one with which we are all most familiar.
- Each user / process has the concept of a ***current directory*** from which all (relative) searches take place.
- Files may be accessed using either absolute pathnames (relative to the root of the tree) or relative pathnames (relative to the current directory.)

- Directories are stored the same as any other file in the system, except there is a bit that identifies them as directories, and they have some special structure that the OS understands.
- One question for consideration is whether or not to allow the removal of directories that are not empty - Windows requires that directories be emptied first, and UNIX provides an option for deleting entire sub-trees.



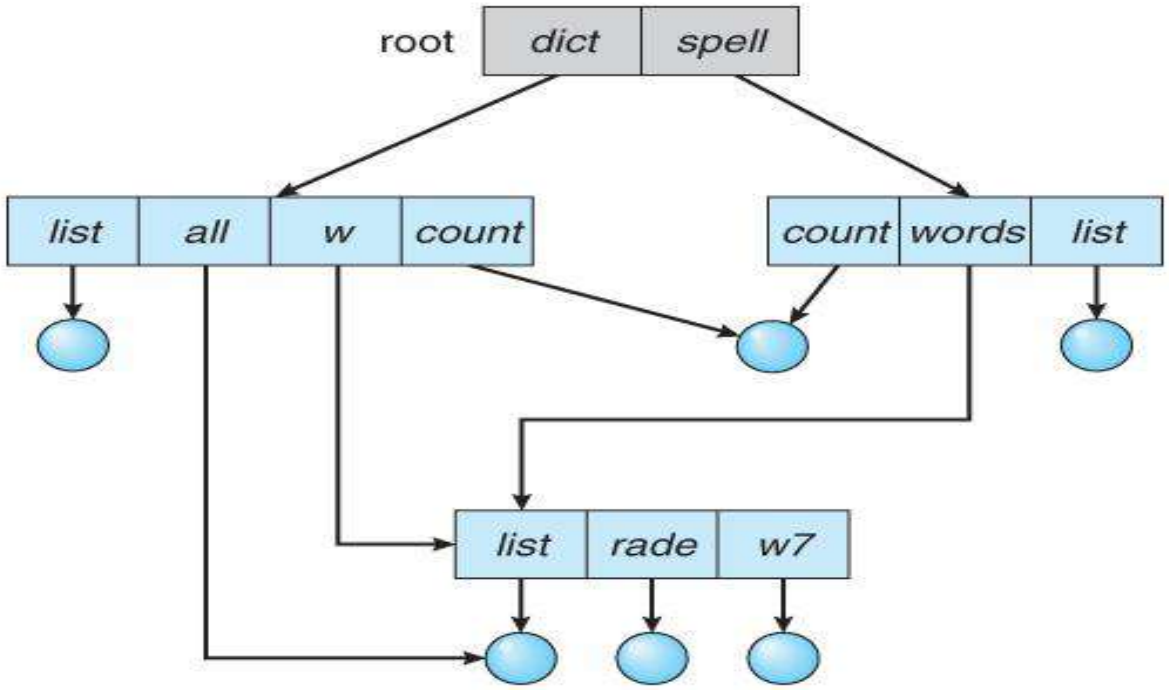
  - In this figure directories are represented using rectangles and files are represented using circles.
  - The file path for a file named "spell" is also given in figure.

- **Advantage:-**
  - It avoids file name collisions.
  - It can be used in multi-user System
  - Different files can be grouped here
- **Disadvantage:-**
  - Sharing of files and directories is not possible here.

✓ **Acyclic graph Directory**

- It permits user to create shared files and directories.
- Shared files and directories can be created using 'link' operation.
- This operation allows a file or directory to appear in more then one directory

- Directories can contain files as well as sub directories also.
- A file may contain more than one file path
- This directory structure does not contain cycles.
- When the same files need to be accessed in more than one place in the directory structure (e.g. because they are being shared by more than one user / process), it can be useful to provide an acyclic-graph structure. (Note the ***directed*** arcs from parent to child.)



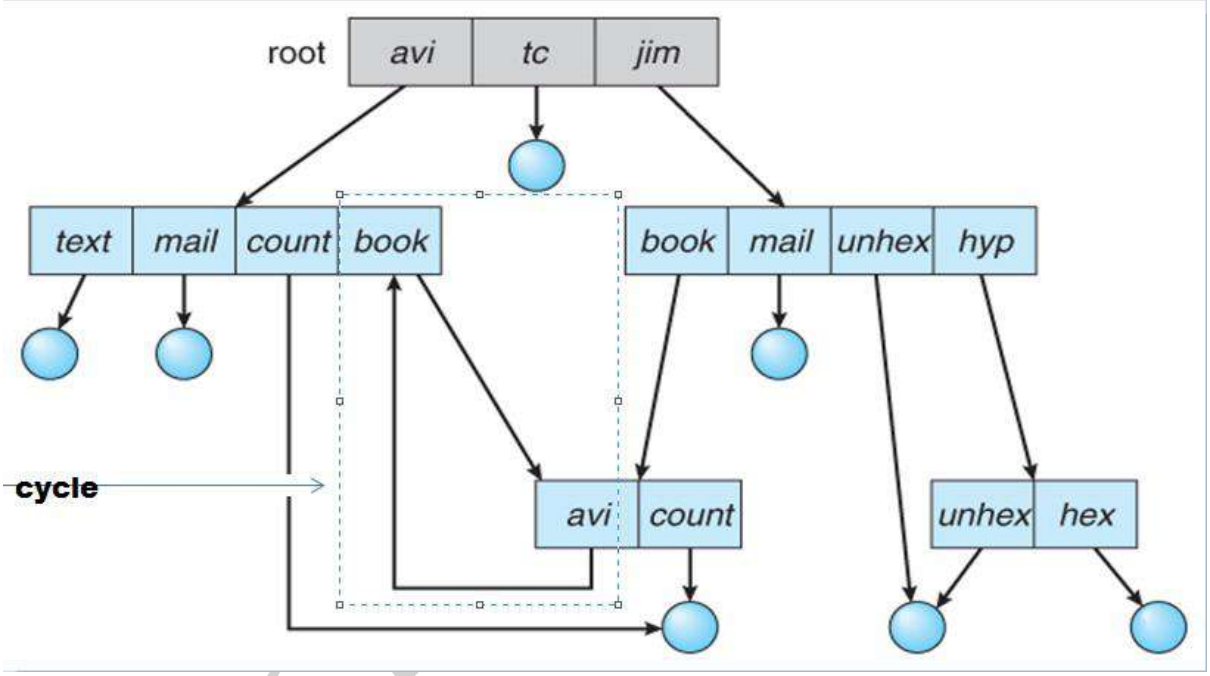
- **Advantages**
  - It allows sharing of files and directories.
- **Disadvantages**
  - Deletion of shared file or directory is complex. When such file directory is deleted, directory entry from all related directories should be deleted
  - A file may have multiple file paths. This may create problem in some operation.
  - For example if user copies all files to backup storage, then such files will be copied multiple times
  - It is difficult to ensure that there are no any cycles in a directory structure.

✓ **General graph directories**

- It is same as acyclic graph directory, but it also allows cycles in a directory structure.

- **Advantages**
  - It allows sharing of files and directories.

- **Disadvantage**
  - Searching should be done carefully. Else it may get into the infinite loops
  - The following diagram represents the directory structure.



- **Absolute path:-**
  - It starts with the root directory (or drive name) of file system.
  - Here file path are given with respect to the root directory.
  - Examples are:
    I. C:\progs\c\test.c
    II. /user/home/hello.txt

- **Relative path**
  - It starts with the current directory (or current working directory) designate by the user.

- Here file paths are given with respect to such current directories.
- Dot '.' and dot dot '..' can be used as a current directory.
- Example:-
- If C:\windows is current directory then path for file hello.txt can be given as windows\hello.txt.

Allocation methods:-

- Till now file system is described from the user point of view.
- User are concerned with how files are named, what operation are allowed on them, what the directory structure looks like and so on.
- User need convenience in using the file system.
- In this section file system is described from system's point of view. Here efficiency and reliability are major concerns
- As described earlier files are stored on disks.
- File contain a data as well as system specific attributes.
- Data means actual data contents stored in files. Attributes contain information about file
- When file is created, storage space is allocated to it. Also when new data is added in an existing file, file size grows and it needs extra storage space.
- Storage space is released when file is deleted or data from file is deleted.
- There are two main goals of allocating space on disk to files
  - Disk space should be utilized effectively
  - Files should be accessed quickly.
- When allocating space to files system must keep track of which disk block go with which file.
  - Disk is considered as a collection of fixed size logical blocks. size varies from system to system
  - These blocks are numbered to identify them uniquely on a disk, number starts from '0' or '1' to some maximum.
  - When ever a file is deleted, allocated block is freed, such blocks can be reused for other files

Three different allocation methods

- Contiguous Allocation
- Linked Allocation
- Indexed Allocation

Contiguous Allocation

- Each file occupies a set of contiguous block on disk.
- On a disk of 1KB, a file size of 50kb would contain 50 **Consecutive blocks**. With 2kb blocks it would have 25 consecutive blocks:
- When a file is created, a disk is searched to find out a chunk of free memory having enough size to store a file. Once such chunk is found memory is allocated.
- Directory entry contains file name, starting block number and length of file.

File name	Starting block number	length

- This method is widely used on CD-ROMs.
- All file sizes are known in advance.

Advantages

- Simple to implement.
  - Information here required is only two things: one starting block number of a file
  - Second length or size of file as a total number of blocks
- File access is quick.
  - All the data blocks will be on the same or neighbour tracks of disk, requiring less seek time

Disadvantages:-

- Finding free space for a new file is time consuming. This needs searching an entire disk until required free memory is found
- If size of an existing file increases. It may not be possible to accommodate such extension.
- External fragmentation is possible. When file is deleted its block are freed leaving a hole on disks.

- With time disk will consist of files and holes. Such holes may be too small to accommodate new files and will waste space on th disk.
- This problem is known as **external fragmentation**

Linked Allocation:-

- Here each file is linked list of disk blocks. Each linked block contains pointer to the next block in list
- These disk blocks may be scattered any where on the disk.
- Directory entry contains the start and last block numbers in a linked list, It looks like given below

File name	Start block number	Last block number	length

- When new file is created, a new directory entry is created, initially it contains null as both the block numbers
- A write to file causes free data blocks to be added to the file, such blocks are added to the end of a linked list. Directory entry is updated on each such occasion.
- To read a file all blocks are read by following the pointers block to block.
- An important variation on the linked allocation method is called filled allocation table (FAT).

- **Advantages:**
  - It does not suffer from external fragmentation
  - Any free disk block can be allocated to a file. Such block does not need to be a consecutive block as in previous method
  - File access is time consuming. Here it is required to access all the data blocks in a linked list to reach some particular block.
  - Random access is not possible directly.
  - Extra space is required for pointers.
- **File protection:**
  - Files are used to store information. Depending upon the users and applications, this information can be very important.
  - No one want to loose information stored in files in any ways
  - Every one want information stored in files to be safe.

- Safety comes in two Ways.
  - **Reliability**
  - **Protection**

- **Reliability:-**
  - Reliability means safety from physical damage
  - File system can be damaged by hardware problems, fire, dirt, power failure, head crashes and so on. File may be deleted accidently
  - Reliability is generally provided by keeping more than one copies of files means duplicating file,
  - Files are duplicated mostly on some different location or in different devices. Magnetic tapes are most widely used for backup of large file
  - If original file are damaged, then they can be retrieved back from backups.

- **Protection:-**
  - Protection means safety from improper access
  - Protection is required where files can be shared among various users
  - Some files need to be shared among all users while some need to be shared among limited users.
  - Protection is done by limiting the type of file access. Here, accessing a file means to perform some operation on that file
  - Several operation of files can be controlled, some of these are
  - Read:- read a file
  - Write:-write a file
  - Execute:- Load and execute a file
  - Append:- Add information at the end of file
  - Delete:- Free the space allocation to a file
  - There are two most common ways to limit the type of file access:
    - Password
    - Access Control

- **Password:-**
  - Password is associated with each file. User can access a file, only if he knows the password.
  - This scheme looks good, but it suffers from several disadvantages
  - There may be large number of files in a computer system, so user needs to remember lots of passwords.
  - To avoid this problem, if single password is used for all files, then once it is discovered, all files are accessed.
- **Access control:-**
  - A list called access control list(ACL) is associated with each file.
  - This list specifies the user name and type of access allowed for that user.
  - This information is stored for all the possible user of the system.
  - This list is generally kept in directory entry along with file name and other information
  - When a user requests to access to a file, the OS checks the ACL, associated with that file. If the user is listed for the requested access, only then access is allowed else denied
  - This method also suffers from some problem
  - First problem with this method: is Constructing an ACL is tedious task. Also user system are not known in advance
  - Second problem with this method is : ACL is kept in directory entry. So now directory should be of varying length instead of fix length. This results in more complicated space management.
  - Solution to this problem is to divide all users in various categories, and then to allow access on such categories instead of individual user.
  - It divides the user in three categories: owner group and others, Access is given on such category. All of the user of that category will be allowed that access.

3.1 Introduction to Linux Operating System

- Linux is one of popular version of UNIX operating System. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX.
- Linux is a UNIX-base operating system. Its original creator was a Finnish student name Linus Torvalds, although being 'open source' it has change a great deal since its original conception.
- It belongs to nobody, and is free to download and use. Any changes to it are open for all to adopt, and as a result it has developed into a very powerful OS that is rapidly gaining in popularity worldwide, particularly among those seeking an alternative to Windows.

- In 1991, hardware was expanding rapidly, and DOS was the king of operating systems. Software development was slower, and Macs, while better, were also much pricier than PCs. UNIX was growing, but at that time in its history the source code was jealously guarded and expensive to use.
- Linus Torvalds was a Helsinki university student who liked playing around with software and computers, and in 1991 he announced the creation of a new core operating system that he had named Linux.
- It is now one of the most used systems for the PC, and is particularly suitable for businesses with small IT budgets.
- Linux is free to use and install, and is more reliable than almost all other systems, running for many months and even years without a reboot being necessary.

- Version of linux:
  - Android
  - Arch Linux
  - Debian GNU/Linux
  - Gentoo Linux
  - Kubuntu
  - Mandriva Linux
  - PC LinuxOS
  - Linux for playstation 2
  - Red Hat Linux
  - Sabayon Linux
  - Slackware
  - SUSE Linux
  - Ubuntu

Distribution	Description
Red Hat Linux	Split into Fedora Core and Red Hat Enterprise Linux. The last official release of the unsplit distribution was Red Hat Linux 9 in March 2003.
CentOS	Community-supported Linux distribution designed as an OpenSource version of RHEL and well suited for servers.
Fedora	Community-supported Linux distribution sponsored by Red Hat. It usually features cutting-edge Linux technologies.
openSUSE	A community-developed Linux distribution, sponsored by SUSE. It maintains a strict policy of ensuring all code in the standard installs will be from FOSS solutions, including Linux kernel Modules. SUSE's enterprise Linux products are all based on the codebase that comes out of the openSUSE project.
Mandrake Linux	The first release was based on Red Hat Linux (version 5.1) and KDE 1 in July 1998. It had since moved away from Red Hat's distribution and became a completely separate distribution. The name was changed to Mandriva, which included a number of original tools, mostly to ease system configuration. Mandriva Linux was the brainchild of Gaël Duval, who wanted to focus on ease of use for new users.

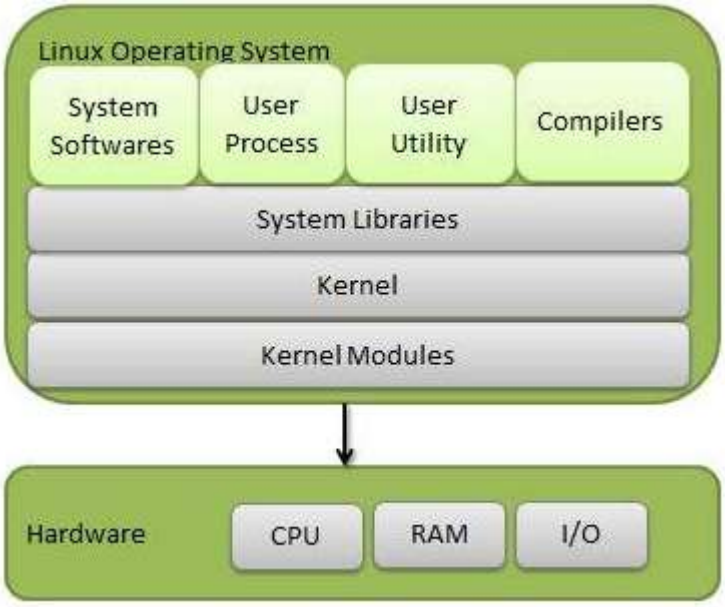
URL	Site Description
redhat.com	Red Hat Linux
fedoraproject.org	Fedora Linux
centos.org	Centos Linux
opensuse.com	openSUSE Linux
debian.org	Debian Linux
ubuntu.com	Ubuntu Linux
mepis.org	Mepis Linux
gentoo.org	Gentoo Linux
turbolinux.com	Turbo Linux
knoppix.org	Knoppix Linux
linuxiso.com	CD-ROM ISO images of Linux distributions
distrowatch.com	Detailed information about Linux distributions
kernel.org	Linux kernel

TABLE 1-1 Linux Distribution and Kernel Sites

3.2 Components of Linux System

Linux Operating System has primarily three components

- **Kernel** – Kernel is the core part of Linux. It is responsible for all major activities of this operating system. It consists of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.
- **System Library** – System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries implement most of the functionalities of the operating system and do not requires kernel module's code access rights.
- **System Utility** – System Utility programs are responsible to do specialized, individual level tasks.



Kernel Mode vs User Mode

Kernel component code executes in a special privileged mode called kernel mode with full access to all resources of the computer. This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast. Kernel runs each processes and provides system services to processes, provides protected access to hardware to processes.

Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in User Mode which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

Basic Features

Following are some of the important features of Linux Operating System.

- **Portable** − Portability means software can works on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.

- **Open Source** – Linux source code is freely available and it is community based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
- **Multi-User** – Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
- **Multiprogramming** – Linux is a multiprogramming system means multiple applications can run at same time.
- **Hierarchical File System** – Linux provides a standard file structure in which system files/ user files are arranged.
- **Shell** – Linux provides a special interpreter program which can be used to execute commands of the operating system. It can be used to do various types of operations, call application programs. etc.
- **Security** – Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.

3.3 Comparison of Windows and Linux.

Windows	Linux
Windows uses different data drives like C: D: E to stored files and folders.	Unix/Linux uses a tree like a hierarchical file system.
NT needs 12 MB RAM	Linux needs 2MB RAM
NT needs 70 MB at least.	Linux needs at least 15 MB disk space
Windows has different drives like C: D: E	There are no drives in Linux
There are 4 types of user account types 1) Administrator, 2) Standard, 3) Child, 4) Guest	There are 3 types of user account types 1) Regular, 2) Root and 3) Service Account
Administrator user has all	Root user is the super user and

administrative privileges of computers.	has all administrative privileges.
In Windows, you cannot have 2 files with the same name in the same folder	Linux file naming convention is case sensitive. Thus, sample and SAMPLE are 2 different files in Linux/Unix operating system.
In windows, My Documents is default home directory.	For every user /home/username directory is created which is called his home directory.
Window is not command line interface	Linux is command line interface.
No access This is not possible with windows as	**Full access** Linux belongs to the GNU Public License ensures that users can access (and alter) the code to the very kernel that serves as the foundation of the Linux operating system.
Licensing restriction You are bound to the number of licenses you purchase, so if you purchase 10 licenses, you can legally install that operating system (or application) on only 10 machines.	**Licensing freedom** you can download a single copy of a Linux distribution (or application) and install it on as many machines

- **4.1. Device Management Function**
- **4.2. Device Characteristics**
- **4.3. Disk space Management**
- **4.4. Allocation and Disk Scheduling Methods**

Device Management

- A process may need several resources to execute-main memory, disk drives, access to files, and so on.
- If the resources are available, they can be granted, and control can be returned to the user process.
- The various resources controlled by the operating system can be thought of as devices.
- Some of these devices are physical devices (for example, disk drives), while others can be thought of as abstract or virtual devices (for example, files).
- Devices are all pieces of equipment for a computer that perform input/output operations but that the computer does not necessary need to work.
- Devices are some times referred to as peripheral devices to emphasize the point that they are “additional” to the core hardware system. In the section the main effort is put on storage devices, that are devices that store data permanently.
- Device management is needed for offering a uniform and consistent approach to all i/o operations. There are considerable differences between all the system’s devices; their speed, how data are transferred and represented, how to prevent and detect errors and how they are handled.
- Tracking the status of each device such as tape drives, disk drives, printers, plotters and terminals.
- Using present policies to determine which process will get a device and for how long.
- Allocating the devices
- DE allocating devices at two levels at the process level when an i/o command has been executed and the devices is temprorily released, and at the job level when the job is finished and the device is permanently released.
- Deciding on policy to determine who gets a device , for how long and when.
- There are 3 basic techniques for implementing the policies of device management:
    - dedicated
    - Shared
    - Virtual

Device characteristics

- Human readable:
- Machine readable:
- Communication
- Data Rate
- Application
- Complexity of control
- Unit of transfer
- Data representation
- Error condition
- Hardware consideration
- Storage devices
- Serial access devices
- Completely direct access devices
- Direct access storage devices(DSAD)
- Human readable:
    - Suitable for communication with the computer user. Examples printers and terminals, keyboard, and perhaps other devices such as a mouse.
- Machine readable:
    - Suitable for communicating with electronic equipment.
- Communication
    - Suitable for communicating with remote devices.
- Data Rate
    - There may be difference of several orders of magnitude between the data transfer rates.
- Application:-
    - The user to which a device is put has an influence on the software and polices of file operating system and supporting utilities.

- Complexity of control:-
    - A printer requires a relatively simple control interface.
    - A disk much more complex.
- Unit of transfer:
    - Data may be transferred as a stream of bytes or characters or in large blocks
- Data representation:
    - Different data encoding schemes are used by different devices, including differences in character code and parity conventions
- Error condition:-
    - The nature of errors, the way in which they are reported, their consequences the available range of responses
- Hardware consideration:-
    - Peripheral devices can be generally categorized into 2 major groups
    - Input or output
        - Input device is one by which the computer “senses” or “feels” the outside world.
        - An output device is one by which the computer “affects” or “controls” the outside world.
    - Storage devices
        - A storage device is a mechanism by which the computer may store information in such a way that it this information may be retrieved at a later time.
- Serial access devices
    - A serial access storage device can be characterized as one that relies on strictly physical sequential positioning and accessing of information
- Completely direct access devices
    - A completely direct access device is one in which the access time is constant.
- Direct access storage devices(DSAD)
    - A direct access device is one that is characterized by small variances in the access time.

Free/Disk Space management

- Since the disk space is limited, we need to reuse the space from deleted files for new files.
- To keep track of free disk space the system maintains a free space list.
- A free space list records all free disk blocks.
- **BIT VECTOR**
    - The free space list is implemented as a bit map or bit vector. Each block is represented by bit.
    - If the block is free, the bit is 1.
    - If the block is allocated the bit is .
- **Linked list**
    - In this approach all the free disk blocks are linked together
    - Keeping a pointer to the first free block in a special location on the disk and catching it in memory
    - The first block contains a pointer to the next free disk block
- **Grouping:-**
    - A modification of the free list approach is to store the address of n free blocks in the first free block.
    - The first n-1 of these blocks are actually free.
    - The last block contains the address of another n free blocks
- **Counting**
    - In this approach we will take the advantage of the fact that several contiguous blocks may be allocateed or freed simultaneously, when the contiguous allocation algorithm
- **Techniques for device management**
    - Dedicated devices:-
        - A dedicated device is allocated to a job for the job's entire duration
    - Shared devices
        - Some devices such as disks, drums and most other direct access storage may be shared concurrently by several process

- Virtual devices
    - Some device that would normally have to be dedicated may be converted into shared devices through technique such as spooling.

Disk management

- In the disk management topic we discuss disk initialization booking from disk & bad block recovery.
- **Disk Formatting**
- **Boot Block**
- **Bad Block**
- Disk Formatting
    - **Disk formatting** is the process of preparing a data storage device such as a hard **disk** drive, solid-state drive, floppy **disk** or USB flash drive for initial use. In some cases, the**formatting** operation may also create one or more new file systems.
    - A new magnetic disk is a blank slate.
    - It is just a platter of magnetic recording material
    - The disk formatting has two types
        - Physical formatting or low level formatting.
        - Logical formatting.
- Physical formatting:
    - Low-level formatting is the process of marking out cylinders and tracks for a blank hard disk, and then dividing tracks into multiple sectors.
    - Disk must be formatted before storing data
    - Disk must be divided into sectors that the disk controller can read and write.
    - Low level formatting files the disk with a special data structure for each sector
    - The data structure consistence of three field:
    - Header
    - Data area
    - Trailer

- Header and trailer contain information used by the disk controller
- Sector number & error correcting cords (ECC) contain in the header & trailer
- When the controller writes a sector of data during normal I/O, the ECC is updated with a value calculated from all the bytes in the data area.
- When the controller writes a sector of data during normal I/O, the ECC is updated with a value calculated from all the bytes is the data area.

Logical Formatting

- After a low-level formatting has been completed, users need to make high-level formatting which makes it possible to save data and should be done on a partition.
- It is the process of writing a file system, cluster size, partition label, and so on for a newly created partition or volume.
- And we can also say high-level formatting just clears data on hard disk, generates boot information, initializes FAT, and labels logical bad sectors when the partition has existed.
- This process does no harm to hard disk in general situations, so we suggest taking such a format to fix a logically damaged partition or device, for example, Windows asks to format a SD card.
- Operating system stores the initial file system data structure to the disk.

Boot Block

- A **boot** sector or **boot block** is a region of a hard disk, floppy disk, optical disc, or other data storage device that contains machine code to be loaded into random-access memory (RAM) by a computer system's built-in firmware.
- When the computer is powered up or rebooted- it needs to have & initial program to run.
- The boot strap is stored in read only memory(ROM). This location is convenient because rom needs no initialization is at a fixed location that the processor can start executing when powered up or reset
- The problem is that changing this boot stop code required changing the ROM hardware chips.
- For this reason most system stored a tiny boot strap loader program in the boot ROM whose only job is to bring in a full boot strap program from disk
- The full boot strap program is stored in a partition called the boot block at a fixed location on the disk

- A disk that has a boot partition is called a boot disk or system disk.
- The full boot strap program is more sophisticated the boot strap loader in the boot ROM.

Bad Block

- In magnetic storage media, bad blocks can happen when a location on a hard disk is defective or when the cyclic redundancy check (CRC) for a particular storage blockdoes not match the data read by the disk. The best way to fix a file that has been affected by a bad block is to write over the original file. This will cause the hard disk to spare the bad block, or fix the CRC and/or data.
- In the disk one or more sectors become defective. Most disk even come from the factory with bad block.
- More sophisticated disk is smarter about bad block recovery.
- The controller maintains a list of bad block on the disk.
- The list is initialized during the low level format at the factory is updated over the file of the disk.
- Low level formatting also sets a side spare sectors not visible to the operating system.
- The controller can be told to replace each bad sector logically with one of the spare stores
- This scheme is known as sectors sparing or sector forwarding
- The alternative of sector sparing, some controllers can be instructed to replace a bad block by sector sleeping.

Swap space management

- Swap space use
- Swap space location
- Swap space management

Swap space management

- Swap-space management is another low-level task of the operating system.
- Virtual memory uses disk space as an extension of main memory. Since disk access is much slower than memory access, using swap space significantly decreases system performance.

The main goal for the design and implementation of swap space is to provide the best throughput for the virtual memory system.

Swap space Use

- Swap space is used in various ways by different operating systems, depending on the memory management algorithms in use.
- For instance, systems that implement swapping may use swap space to hold an entire process image, including the code and data segments.
- Paging systems may simply store pages that have been pushed out of main memory.
- The amount of swap space needed on a system can therefore vary depending on the amount of physical memory, the amount of virtual memory it is backing, and the way in which the virtual memory is used.
- It can range from a few megabytes of disk space to gigabytes.

Swap space Location

- A swap space can reside in one of two places:
- It can be carved out of the normal file system, or it can be in a separate disk partition.
- If the swap space is simply a large file within the file system, normal file system routines can be used to create it, name it, and allocate its space.
- Alternatively, swap space can be created in a separate raw partition, as no file system or directory structure is placed in this space

Disk scheduling algorithm

- **FCFS (First come First Serve)**
- **SSTF(shortest seek time first)**
- **Elevator(Scan)**
- **C-Scan(Circular scan)**
- **Look**
- **C-Look(Circular Look)**
- **Transfer rate** is rate at which data flow between drive and computer
- **Positioning time** (random-access time) = seek time + rotational latency
- **Seek time:** the time to move disk arm to desired cylinder
- **Rotational latency** : the time for desired sector to rotate under the disk head .
- **Head crash** results from disk head making contact with the disk surface

INTRODUCTION

- In operating systems, seek time is very important. Since all device requests are linked in queues, the seek time is increased causing the system to slow down. Disk Scheduling Algorithms are used to reduce the total seek time of any request.

PURPOSE

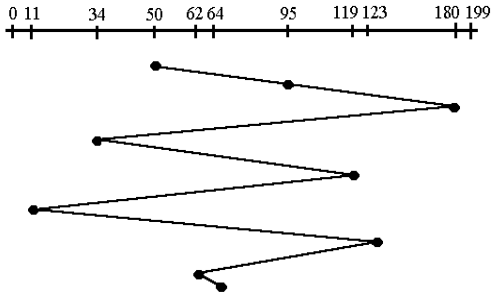
- The purpose of this material is to provide one with help on disk scheduling algorithms. Hopefully with this, one will be able to get a stronger grasp of what disk scheduling algorithms do.

TYPES OF DISK SCHEDULING ALGORITHMS

- Although there are other algorithms that reduce the seek time of all requests, I will only concentrate on the following disk scheduling algorithms:
  First Come-First Serve (FCFS)
  Shortest Seek Time First (SSTF)
  Elevator (SCAN)
  Circular SCAN (C-SCAN)
  LOOK
  C-LOOK
  These algorithms are not hard to understand, but they can confuse someone because they are so similar. What we are striving for by using these algorithms is keeping Head Movements (# tracks) to the least amount as possible. The less the head has to move the faster the seek time will be. I will show you and explain to you why C-LOOK is the best algorithm to use in trying to establish less seek time.

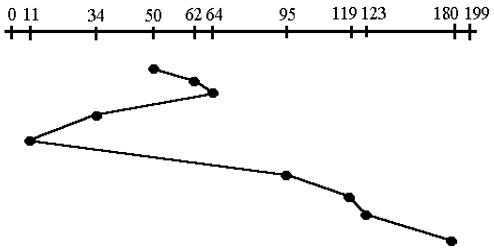
Given the following queue -- 95, 180, 34, 119, 11, 123, 62, 64 with the Read-write head initially at the track 50 and the tail track being at 199 let us now discuss the different algorithms.

- 1. *First Come -First Serve (FCFS)* All incoming requests are placed at the end of the queue. Whatever number that is next in the queue will be the next number served. Using this algorithm doesn't provide the best results.
- To determine the number of head movements you would simply find the number of tracks it took to move from one request to the next. For this case it went from 50 to 95 to 180 and so on. From 50 to 95 it moved 45 tracks.
- If you tally up the total number of tracks you will find how many tracks it had to go through before finishing the entire request. In this example, it had a total head movement of 640 tracks.
- The disadvantage of this algorithm is noted by the oscillation from track 50 to track 180 and then back to track 11 to 123 then to 64. As you will soon see, this is the worse algorithm that one can use.

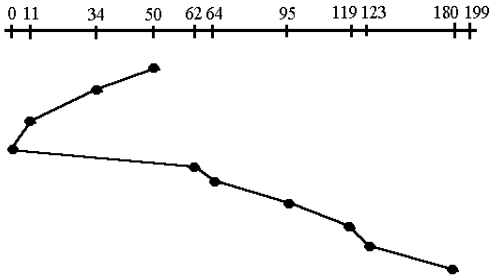


2. *Shortest Seek Time First (SSTF)*
In this case request is serviced according to next shortest distance.

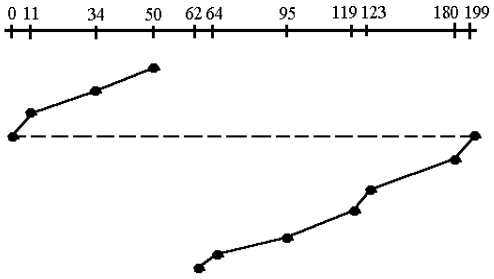
- *Starting at 50, the next shortest distance would be 62 instead of 34 since it is only 12 tracks away from 62 and 16 tracks away from 34.*
- *The process would continue until all the process are taken care of. For example the next case would be to move from 62 to 64 instead of 34 since there are only 2 tracks between them and not 18 if it were to go the other way.*
- *Although this seems to be a better service being that it moved a total of 236 tracks, this is not an optimal one. There is a great chance that starvation would take place. The reason for this is if there were a lot of requests close to eachother the other requests will never be handled since the distance will always be greater.*



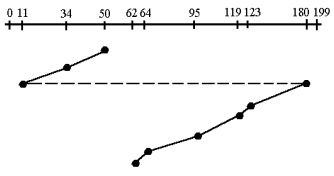
- *3. Elevator (SCAN)*
  This approach works like an elevator does. It scans down towards the nearest end and then when it hits the bottom it scans up servicing the requests that it didn't get going down. If a request comes in after it has been scanned it will not be serviced until the process comes back down or moves back up. This process moved a total of 230 tracks. Once again this is more optimal than the previous algorithm, but it is not the best.



- *4. Circular Scan (C-SCAN)*
  Circular scanning works just like the elevator to some extent. It begins its scan toward the nearest end and works it way all the way to the end of the system. Once it hits the bottom or top it jumps to the other end and moves in the same direction. Keep in mind that the huge jump doesn't count as a head movement. The total head movement for this algorithm is only 187 track, but still this isn't the mose sufficient.



- *5. C-LOOK*
  This is just an enhanced version of C-SCAN. In this the scanning doesn't go past the last request in the direction that it is moving. It too jumps to the other end but not all the way to the end. Just to the furthest request. C-SCAN had a total movement of 187 but this scan (C-LOOK) reduced it down to 157 tracks.
- *From this you were able to see a scan change from 644 total head movements to just 157. You should now have an understanding as to why your operating system truly relies on the type of algorithm it needs when it is dealing with multiple processes.*



NOTE:- Don't forget to count Total number of Head Counts And Look Algorithm. As Already discussed in lecture

DMA(Direct Memory Access)

Direct memory access (DMA) is a method that allows an input/output (I/O) device to send or receive data directly to or from the main memory, bypassing the CPU to speed up memory operations. The process is managed by a chip known as a DMA controller (DMAC).

A computer's system resource tools are used for communication between hardware and software. The four types of system resources are:

- I/O addresses
- Memory addresses
- Interrupt request numbers (IRQ)
- Direct memory access (DMA) channels

DMA channels are used to communicate data between the peripheral device and the system memory. All four system resources rely on certain lines on a bus. Some lines on the bus are used for IRQs, some for addresses (the I/O addresses and the memory address) and some for DMA channels.

A DMA channel enables a device to transfer data without exposing the CPU to a work overload. Without the DMA channels, the CPU copies every piece of data using a peripheral bus from the I/O device. Using a peripheral bus occupies the CPU during the read/write process and does not allow other work to be performed until the operation is completed.

With DMA, the CPU can process other tasks while data transfer is being performed. The transfer of data is first initiated by the CPU. During the transfer of data between the DMA channel and I/O device, the CPU performs other tasks. When the data transfer is complete, the CPU receives an interrupt request from the DMA controller.

A device applying DMA technology uses only a single channel. To avoid a conflict, sometimes the BIOS must assign a different channel to a device. A conflict can happen when more than one device tries to use the same channel.

DMA channels are slower than later data transfer methods, and therefore are not as common. One later interface is the Ultra DMA, which has a data transfer rate up to 33 MB per second. Each DMA transfers approximately 2 MB data per second.

Device Driver

A device driver is a program that controls a particular type of device that is attached to your computer

Purpose of a Device Driver

The purpose of a device driver is to handle requests made by the kernel with regard to a particular type of device. There is a well-defined and consistent interface for the kernel to

make these requests. By isolating device-specific code in device drivers and by having a consistent interface to the kernel, adding a new device is easier.

Types of Device Drivers

A device driver is a software module that resides within the Digital UNIX kernel and is the software interface to a hardware device or devices. A hardware device is a peripheral, such as a disk controller, tape controller, or network controller device. In general, there is one device driver for each type of hardware device. Device drivers can be classified as:

- Block device drivers
- Character device drivers (including terminal drivers)
- Network device drivers
- Pseudodevice drivers

Block Device Driver

A block device driver is a driver that performs I/O by using file system block-sized buffers from a buffer cache supplied by the kernel. The kernel also provides for the device driver support interfaces that copy data between the buffer cache and the address space of a process.

Block device drivers are particularly well-suited for disk drives, the most common block devices. For block devices, all I/O occurs through the buffer cache.

Character Device Driver

A character device driver does not handle I/O through the buffer cache, so it is not tied to a single approach for handling I/O. You can use a character device driver for a device such as a line printer that handles one character at a time. However, character drivers are not limited to performing I/O one character at a time (despite the name ``character'' driver). For example, tape drivers frequently perform I/O in 10K chunks. You can also use a character device driver when it is necessary to copy data directly to or from a user process.

Because of their flexibility in handling I/O, many drivers are character drivers. Line printers, interactive terminals, and graphics displays are examples of devices that require character device drivers.

A terminal device driver is actually a character device driver that handles I/O character processing for a variety of terminal devices. Like any character device, a terminal device can accept or supply a stream of data based on a request from a user process. It cannot be mounted as a file system and, therefore, does not use data caching.

Network Device Driver

A network device driver attaches a network subsystem to a network interface, prepares the network interface for operation, and governs the transmission and reception of network frames over the network interface. This book does not discuss network device drivers.

Pseudodevice Driver

Not all device drivers control physical hardware. Such device drivers are called ``pseudodevice'' drivers. Like block and character device drivers, pseudodevice drivers make use of the device driver interfaces. Unlike block and character device drivers, pseudodevice drivers do not operate on a bus. One example of a pseudodevice driver is the pseudoterminal or `pty` terminal driver, which simulates a terminal device. The `pty` terminal driver is a character device driver typically used for remote logins.

Character Device Driver

A character device driver does not handle I/O through the buffer cache, so it is not tied to a single approach for handling I/O. You can use a character device driver for a device such as a line printer that handles one character at a time. However, character drivers are not limited to performing I/O one character at a time (despite the name ``character'' driver). For example, tape drivers frequently perform I/O in 10K chunks. You can also use a character device driver when it is necessary to copy data directly to or from a user process.

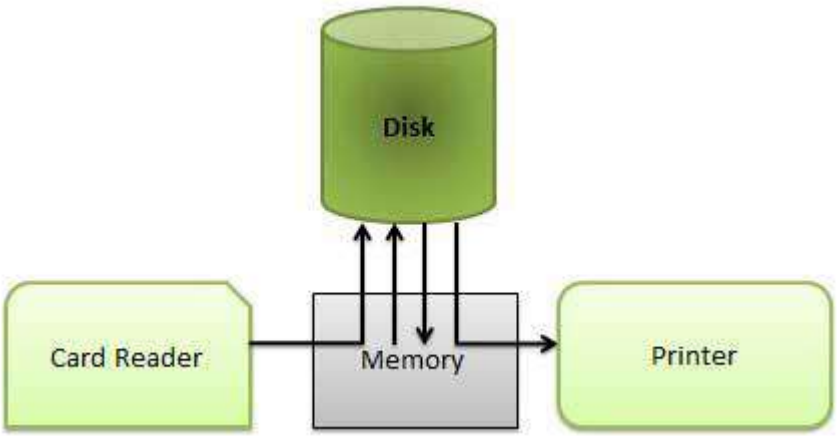
Because of their flexibility in handling I/O, many drivers are character drivers. Line printers, interactive terminals, and graphics displays are examples of devices that require character device drivers.

A terminal device driver is actually a character device driver that handles I/O character processing for a variety of terminal devices. Like any character device, a terminal device can accept or supply a stream of data based on a request from a user process. It cannot be mounted as a file system and, therefore, does not use data caching.

Spooling

Spooling is an acronym for simultaneous peripheral operations on line. Spooling refers to putting data of various I/O jobs in a buffer. This buffer is a special area in memory or hard disk which is accessible to I/O devices. Operating system does the following activites related to distributed environment.

- OS handles I/O device data spooling as devices have different data access rates.
- OS maintains the spooling buffer which provides a waiting station where data can rest while the slower device catches up.
- OS maintains parallel computation because of spooling process as a computer can perform I/O in parallel fashion. It becomes possible to have the computer read data from a tape, write data to disk and to write out to a tape printer while it is doing its computing task.



Advantages

- The spooling operation uses a disk as a very large buffer.
- Spooling is capable of overlapping I/O operation for one job with processor operations for another job.