Unit I - Introduction (ch-1) Chapter (1)

Ch-1-1

Operating System

An operating system is an organised collection of programs that controls and manages the computer hardware. The operating system acts as an interface between users and hardware of a computer system.

Internally an operating system acts as a manager of resources of the computer system such as processor, memory, I/O devices, files etc.

Thus, the purpose of an operating system is to provide an environment in which a user can execute programs in a convenient and efficient manner.

Components of Computer System

Entire Computer system can be divided into four components

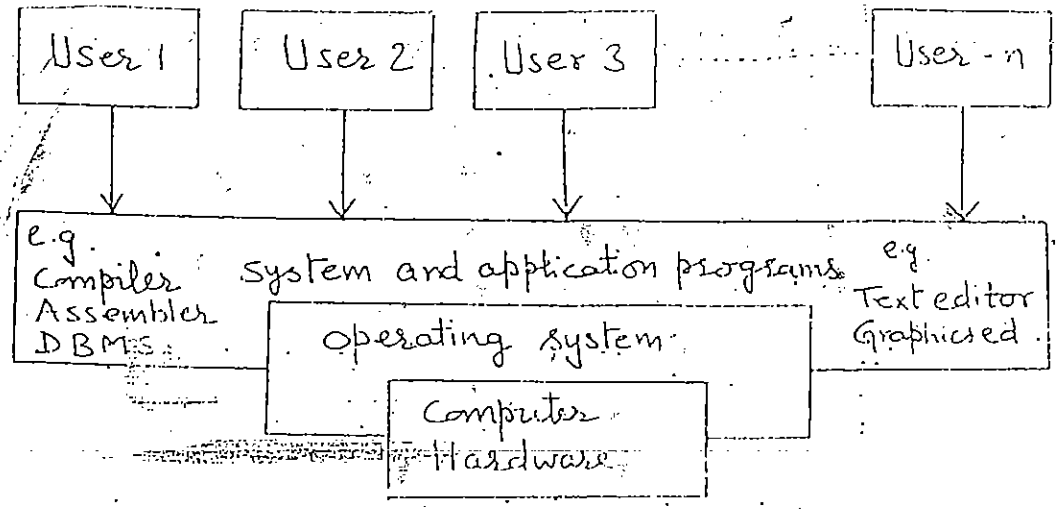
1) Hardware: Central processing unit, memory, Input-output devices which provide basic computing resources

2) Application programs: Word processors, spreadsheets, compilers and web browsers which provide the way to solve the problems.

3) User: who works on the system

4) Operating System: The operating system controls and coordinates the use of the hardware among the various application programs for the various users

Neelam Bhatnagar



Components of a Computer System

Operating System - different views

1. Operating System as a government: Like a government, it does not perform any useful function by itself. It provides an environment within which other departments can work properly and efficiently. Similarly, operating system provides an environment to the user to interact with the hardware to work with different application programs effectively and efficiently.

2. Operating System as a resource allocator: Operating System runs all the time as long as the computer operational and exit only when the computer is shut down. The operating system manages the different resources like CPU, main storage and input/output devices and file. It resolves the conflicts between the resources and makes effective use of the system resources. Thus, operating system must decide the strategy to allocate resources to different programs

(3)

and users so that it can operate the computer system effectively and fairly.

3) Operating System as a control program: (An operating system is a control program which manages the execution of user programs to prevent errors and improper use of the computer especially I/O devices.) It is responsible for controlling the process programs of processes and acting on exceptional conditions arising during the execution of a process.

[As a resource manager and controller, operating system performs the following activities:

1) It assigns processors to different tasks being performed by the computer system.

2) It allocates the main memory and other storage areas to the system programs as well as user programs and data.

3) It manages files on various storage devices and the transfers of these files from one storage device to another.

4) It carries out the input/output management and coordinates and assigns different input and output devices while one or more programs are being executed.

5) It is able to interpret commands and instructions.

6) It coordinates and assigns compilers, assemblers, utility programs and other software packages to various users working on the computer system.

7) It establishes data security and integrity. That is, it keeps different programs and data in such a manner that they do not interfere with each other.

8) It establishes and enforces the job priority. That is,

it determines and maintains the order in which jobs are to be executed on the computer system.
9. It also produces error messages, and other debugging and error detecting codes.
10. It maintains internal time clock and log off system usage for all users.

Operating System — goals

Goals of an operating system depend on the type of operating system. Two general goals are associated with the design of operating system which are as follows:
1) Efficient operation of Computer system.
2) Convenience of the user.

For small PCs, primary goal of operating system is convenience for the user i.e. to make the work easier.

The primary goal of other operating systems is efficient use of and operation of the computer system as in multiuser system. If the system hardware is expensive, it is desirable to make them as efficient as possible.

In past, when hardware was expensive, efficiency was more important than convenience. Over time, when hardware's cost came down, ease of use became primary goal. Many graphic user interfaces were added.

Different Systems and their requirements for OS

Most operating systems contain components which have similar functionalities in all OSs such as process management memory management, file management etc. The techniques used to implement these functions may vary from one OS to another (but the fundamental concepts are same) due to the different requirements of different type of systems

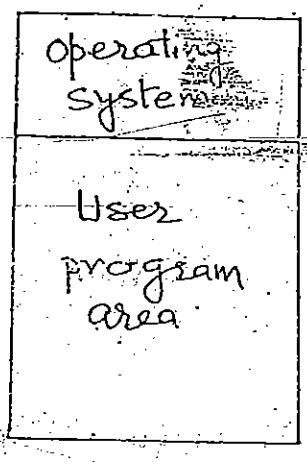
(1) Mainframe Systems: These are the first computers for commercial and scientific applications. Different types of systems are as follows:

Batch Systems: Batch processing requires program, data and appropriate system commands to be submitted together in the form of a job. Batch operating system does not allow direct interaction between users and executing programs. The common input devices were card reader and tape drives. The jobs were usually in form of punch cards. The output consisted of the result of the program, dump of the final memory and register contents for debugging.

These were the systems of early days, and operating system in these computers was fairly simple. The main task of the operating system was to transfer control automatically from one job to another job. The operating system was always resident in memory.

To speed up processing, operators batched together jobs with similar needs and submitted to

the computer as a group. The operator would sort programs into batches with similar requirements and, as the computer became available, would run each batch. The output from each job would be sent back to the appropriate programmer.



Memory layout for a simple batch system

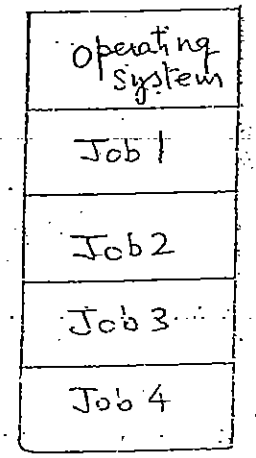
Speed of mechanical I/O devices are very less than CPU, so the CPU is often idle in batch processing. CPU can execute thousands of instructions per second. A card reader can read only 20 cards/second. The introduction of disk technology allowed the operating system to keep all the jobs on a disk, rather than a serial card reader. The OS could perform job scheduling to use resources and perform tasks efficiently due to direct access to several jobs.

2) Multiprogrammed Systems

Multiprogramming increases CPU utilization by organizing jobs so that CPU always has one job to execute. Normally, Operating System keeps several jobs in memory simultaneously. The operating system picks and begins to execute one of the jobs in the memory. The job may have to wait for some task, such as an I/O operation,

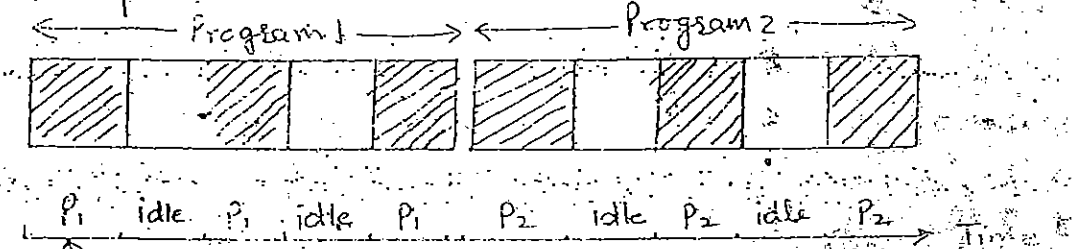
to complete. The operating system switches from one program to another program and executes. Then, CPU is idle for very less time or no time.

In a non-multiprogrammed system, the CPU would sit idle.

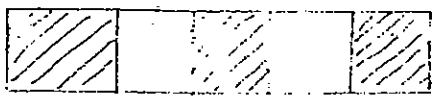


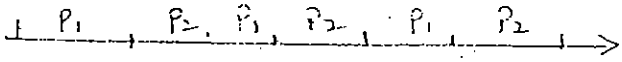
Memory layout for a multiprogramming system.

Example:



(a) Sequential execution





(b) Multiprogrammed execution

If two programs are run sequentially, processor will be idle whenever I/O interaction is required.

In multiprogramming, when one program is busy in I/O interaction, processor can execute program-2 and save the time. Similarly when program-2 is busy to do I/O job, processor can come back to program-1 and execute.

Thus, for multiprogramming, appropriate job scheduling and CPU scheduling are required which are handled by OS.

Time-Sharing Systems

In time sharing (or multitasking) systems, CPU executes multiple jobs by switching among them, but the switches occur so frequently that the users can interact with each program while it is running. Each user has the impression that the entire computer system is dedicated to this use, even though it is being shared among many users.

Time sharing operating system are more complex than multiprogramming operating systems. In both types, several jobs reside simultaneously in memory. Jobs may have to be swapped in and out of main memory to the disk to obtain a reasonable response time. Virtual memory is a technique that allows the execution of a job that may not be present completely in memory.

Time sharing systems must also provide an appropriate file system, disk management, CPU scheduling for concurrent execution, job synchronization and communication for orderly execution and to avoid deadlock.

(2) Desktop Systems: Personal computers are micro computers, they are smaller and less expensive than mainframe systems. The goals of these systems are not to maximize CPU and peripheral utilization but to opt for maximizing user convenience and responsiveness.

Previously file protection, security etc. design issues were not important on a personal machine. But now these machines may be connected to local area networks or other Internet connection, security has become an important issue.

First important operating system was MS DOS. Now, Microsoft windows, OS/2, Mac OSX, Unix are used having rich GUI. Linux, a unix-like operating system available for PCs, has also become popular recently.

(3) Multiprocessor Systems: These systems are also known as parallel systems or tightly coupled systems. Such systems have more than one processor in close communication sharing the computer bus, the clock and sometimes memory and peripheral devices.

Multiprocessor systems have the following advantages:

(i) Increased throughput: By increasing the number of processors, we can get more work done in less time. If there are N processors, speed-up ratio should be N but due to some overhead and shared resources it is less than N.

(ii) Economy of Scale: Multiprocessor systems can save money compared to multiple single processor

systems because they can share peripherals, mass storage and power supplies. For example, several programs have to operate on same data. Two ways are there. One — many computers with local disks and many copies of the data.

Second - data on one disk and all processors sharing the data. Second option is cheap having one disk with data shared by all processors.

(iii) Increased Reliability:

If functions can be distributed properly among several processors, then the failure of one processor will not halt the system, only slow it down. This is known as graceful degradation of the systems and such type of systems are known as fault tolerant.

If one processor fails, other processors will share the work of failure processor, thus system will continue to work with little less performance.

Types of Multiprocessing

1) Symmetric Multiprocessing (SMP):

Each processor runs an identical copy of operating system, and these copies communicates with one another as needed. All processors are peers. The benefit of this model is that many proceses can run simultaneously without causing a significant deterioration of performance. However, we must carefully control I/O to ensure that the data reach the appropriate processor. To avoid the inefficiencies, processors can share certain data structures and memory dynamically.

2) Asymmetric multiprocessing: Each processor is assigned a specific task. A master processor controls the system, the other processors either look to the master for instruction or have predefined tasks. This is known as master-slave relationship. The master processor schedules and allocates work to slave processors.

The difference between symmetric and asymmetric multiprocessing can be implemented either by hardware and software. Special hardware can differentiate the multiple processors or the software can be written to allow one master and multiple slaves e.g. Sun's Operating system SunOS version 4 provides asymmetric multiprocessing.

* Distributed Systems: Distributed systems depend on networking for their functionality. Distributed systems are able to share computational tasks. They are able to communicate and provide a rich set of features to users.

(1) Client-Server Systems: Today centralized systems act as server systems to satisfy requests generated by client systems. As in old centralized system, terminals were connected to the server. Terminals had either no or very less functionality. Now-a-days, due to powerful PCs, all user interface functionality is handled by PCs only.

Servers systems can be broadly categorized as compute servers and file servers.

- Compute-server systems provide an interface to which clients can send requests to perform an action, in response to which they execute the action and send back results to the client
- File-server systems provide an interface to ~~which~~ where clients can create, update, read and delete files.

2) Peer-to-Peer Systems: In beginning (1970s), PCs were used only as standalone computers. In 1980s with the intensive use of Internet PCs were connected to computer networks and network connectivity became an essential components of a ~~an network~~ connectivity computer system.

Operating system includes the system software (such as TCP/IP and PPP) that enables a computer to access the Internet via a local area network or telephone connection. Several operating systems include the web browser itself, as well as electronic mail, remote login and file-transfer clients and servers.

The computer networks used in these applications consist of a collection of processors that do not share memory or a clock, each processor has its own local memory. The processors communicate with one another through various communication lines, such as high-speed buses or telephone lines. These systems are usually referred as loosely coupled systems or distributed systems.

(5) Clustered Systems Clustered systems are the combination of multiple CPUs to accomplish computational work. These systems share storage and are closely linked via LAN networking.

Clustering is performed to provide high availability. A layer of cluster software runs on the cluster nodes. Each node can monitor one or more computers. If the machine which was monitored fails, the monitoring machine can take ownership and restarts the application. Thus, user can continue his work. Two types of asymmetric clustering are possible:

(i) Asymmetric clustering - In this type, one machine is in hot standby mode while the other is running the applications. The hot-standby host monitor the active server. If the server fails, the hot standby host becomes the active server.

(ii) Symmetric Clustering - In this type, two or more hosts are running the applications and they are monitoring each other. If one host fails, other can share the work. This is more efficient because it uses all of the available hardware.

Parallel clusters allow multiple hosts to access the same data on the shared storage. To perform this task parallel clusters have special versions of software and special releases of applications e.g. Oracle Parallel Server is a version of Oracle's database that has been designed to run on parallel clusters. Cluster technology is rapidly changing. Current clusters are limited to two or four hosts due to the complexity of connecting the hosts to shared storage.

(6) Real-Time Systems - Real-time operating systems are used in environments where a large number of events must be accepted and processed in a short time or within certain deadlines. e.g. industrial control, telephone switching equipment, flight control and real-time simulations and military applications.

A primary objective of real-time systems is to provide quick event-response times, and thus meet the scheduling deadlines. User convenience and resource utilization are of secondary concern to real-time system designers.

A real-time system has well-defined, fixed time constraints. Processing must be done within the defined constraints, or the system will fail. A real-time system functions correctly only if it returns the correct result within its time constraints. There are two types of real-time systems:

(i) Hard real-time system :- It guarantees that critical tasks should be completed on time. Memory management is less demanding. There is little moving of programs between primary and secondary storage. The process population in real-time system is static. Processes in real-time systems have to cooperate closely.

(ii) Soft real-time system - A critical real-time task gets priority over other tasks, and retains that priority until it completes. They are useful in several areas including multimedia, virtual reality and advanced scientific projects. Major versions of UNIX provide soft real-time facility.

7. Handheld Systems

Personal digital assistants (PDAs) are handheld systems e.g. Palm-Pilots or cellular telephones with connectivity to a network such as the Internet. Due to the limited size, most handheld devices have slow processors, small display screens and a small amount of memory.

Many handheld devices have very small amount of memory (512 KB – 8 MB). Thus, the operating system must manage memory efficiently. Faster processors require more power i.e. large battery. But to minimize the size of most handheld devices smaller, slower processors which consume less power are used. Therefore, OS should be designed for efficient utilization of processors.

Due to small size of monitor, e-mail, browsing web pages must be condensed onto smaller displays. Sometimes, only a small subset of a web page is delivered and displayed on the handheld device through web clipping.

Some handheld machine (cellular telephone) use wireless technology such as Blue tooth and allow remote access to e-mail and web browsing. Many PDAs do not use wireless technology; they may have data from Internet through indirectly.

Having so many limitations, handheld devices are very popular due to convenience and portability. They may have other functionalities like camera and MP3 players.

Computing Environments :

Different systems are used in different types of computing environment which are as follows:

1. Traditional Computing :

Early traditional computing was comprised of PCs connected to a small network with file server and print server placed in a small area. Remote access was not common and was limited upto the mainframes with fewer terminals. Portability is generally achieved by laptop computers.

The current technology provides for more ways to access these environments. Companies implement portals which provide web accessibility to their internal servers. Handheld computers are used as portable computers and can be connected to wireless networks to use the company's web portal. Previously network connectivity was possible only at high cost and now it is available at low cost with high speed. At home, users can attach their single PC with Internet with help of modem connection, even firewall is possible at home.

2. Web-based Computing :

Web computing has increased the emphasis on networking. Previous standalone devices are now networked and can have wired or wireless access. PCs, handheld PDAs and even cell phones are also possible to connect with Internet. Network connectivity is faster either by improved network technology, optimized network implementation code, or both. Operating Systems like Windows 95 which acted as web clients,

have evolved into Windows ME and Windows 2000 which can act as web servers as well as clients.

3. Embedded Computing:

Embedded computers run embedded real-time operating systems. These devices have to perform specific tasks, thus OSs also provides limited features i.e. according to the requirements. These OSs have little or no user interface, spending their time to monitor and manage the hardware devices. Disk management, file management, memory management etc features are not required in this type of computing. These devices are automobile engines, robots, VCRs, microwave ovens, washing machines etc. These devices can be standalone units and as the members of networks and web

multitasking

Questions:

1. What is an operating system? (2) or (3)
2. Define an operating system. (2) or (3)
3. Justify the statement "Operating system can be viewed as a government, resource allocator and a control program". (5) or (6)
4. Distinguish between:
   (i) Multiprogramming and multiprocessing (3) or (4)
   (ii) Batch system and time-sharing system (3) or (4)
5. Explain briefly each of the following.
   (i) Batch systems
   (ii) Multiprogramming systems
   (iii) Time sharing systems
   (iv) Multiprocessing systems
   (v) Distributed systems
   (vi) Clustered systems
   (vii) Real-time systems
6. Comment on the statement: A time sharing system always improves the system throughput.
7. Define how multi-programming improves the performance.
8. Differentiate between tightly coupled and loosely coupled system.

(1)

Part 1

Unit - 1 Introduction (ch-3)

System Components

An operating system performs large number of functions, and operating system can be considered as a collection of many programs or components. Each function is carried out by a component of the operating system. These components are as follows:

1. Process management
2. Main-memory management
3. file management
4. I/O system management
5. Secondary storage management
6. Networking
7. Protection System
8. Command - Interpreter System.

Process Management:

A process is a program in execution and one unit of work in system. A program is a passive entity such as contents of a file whereas a process is an active entity with a program counter which specify the next executable instruction for execution. A process may be text editor in execution, system task etc.

A process needs certain resources including CPU time, memory, files, I/O devices, required data to accomplish its task. These resources may be given at the time of creation of process or when process is running. When the process is running terminates, these resources should be reclaimed by OS. Thus, the function of OS is to keep track of all the processes, their creation and deletion, required resources and their allocation etc.

Neelam Bawane

In a single user, multiprogramming environment, the process management module of the operating system is far less complicated and less significant than in a multi-user scheme. In a multiuser and multi-programming environment, function of OS is to keep track of the competing processes, schedule them and dispatch them for execution one after another, but each user should have an impression that he has full control of the CPU and other resources.

The operating system is responsible for the following activities in connection with process management:

- Creating and deleting both user and system processes
- Suspending and resuming processes.
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanism for deadlock handling

Main Memory Management:

Memory management is primarily concerned with allocation of physical memory of finite capacity to requesting processes. No process may be activated before a certain amount of memory can be allocated to it.

Main memory is a large array of words or bytes, ranging in size from hundreds of thousands to billions. Each word or byte has its own address. The main memory is generally the only storage device that the CPU is able to address and access directly. The ~~main memory~~ CPU reads instructions from main memory during instruction-fetch cycle and it reads and writes data from

main memory during data-fetch cycle.

A program must be mapped to absolute addresses and loaded into memory for execution. CPU accesses instructions and data from memory by generating these absolute addresses and when program terminates its memory space is declared available.

Under this service OS keeps track of memory locations and allocating/deallocating memory to it various processes

In single user and uniprogramming environment memory management module is less complicated but in multi user and multi-programming environment this module will be complex.

OS should keep several programs in memory for effective and efficient utilization of CPU with the help of different memory management schemes

The OS's functions regarding memory management are as follows:

- ~~Keeping track of which parts of memory are currently being used and by whom.~~
- Allocating and deallocating memory space as needed
- Keeping track of the parts of memory used by different processes.
- Deciding which processes are to be loaded into memory.

File Management: Computers can store information on several different types of physical media such as magnetic tape, magnetic disk and optical disk. Each medium is controlled by a device, such as a disk drive or tape drive. Each of these media and its drive have their own characteristics and physical

organization. These characteristics may be access speed, capacity, data transfer rate, and access method. A file is a collection of related information such as program and data defined by its creator.

The operating system implements the abstract concept of a file by managing mass storage media and devices that control them. files are normally organized into directories to ease their use.
In a multi-user system file management is more complex because OS has to keep track of all the files in use, used by whom, access rights, the way files may be accessed. The responsibilities of operating system regarding file management are

- Creating and deleting files
- Creating and deleting directories
- Supporting primitives (commands and instructions) for manipulating files and directories
- Mapping files onto secondary storage
- Backing up files on stable storage media.

I/O Management: I/O management system is concerned with the proper implementation of I/O operations in the system. A running program may require I/O which may be file or an I/O device. I/O devices vary so widely in their function and speed, a variety of methods are needed to control them. To encapsulate the details and functionalities of different devices, the kernel of an OS is structured to use device-driver modules. The I/O subsystem consists of:

- A general device-driver interface

- Drivers for specific hardware devices
- A memory-management component that includes buffering, caching, and spool spooling

Secondary-Storage Management

At the time of execution, program with relevant data is in main memory. Most programs including compilers, assemblers, sort routines, editors and formatters are stored on a disk until loaded into memory. They use the disk as both the source and destination of their processing.

Main memory is small and volatile, thus secondary storage is necessary to provide backup and permanent storage. Hence the proper management of disk storage is very important for a computer system. The entire speed of operation of a computer system depends on the speed of disk subsystem and algorithms that manipulate that subsystem.

The operating system is responsible for the following activities in connection with disk management:

- Storage allocation/deallocation
- Disk scheduling
- Free space management

Networking -

If the processors in the system are connected through a communication network, network can be configured in a number of ways. The purpose of the distributed system is to provide an efficient and convenient environment for such sharing of resources, computation speed-up, reliability and communication.

A network operating system provides an environment in which users, who are aware of the multiplicity of machines, can access remote resources by either logging into the appropriate remote machine or transferring data from the remote machine to their own machines. To increase the system's utility, there are ~~dif~~ many different protocols and world wide web. World wide web works with a new protocol hypertext transfer protocol (http).

Protection System

Protection refers to a mechanism for controlling the access of programs, processes or users to the resources defined by a computer system. A protection-oriented system provides means to distinguish between authorized and unauthorized usage

We need to provide protection for several reasons The most obvious is the need to prevent mischievous, intentional violation of an access restriction by a user. ~~If~~ a In a multiuser system and multiprogramming system, various processes must be protected from one another's activities. The mechanisms ensure that the files, memory segments, CPU, and other resources can be operated on by only those processes that have gained proper authorization from the operating system.

Command-Interpreter System

Command interpreter is an interface between the user and the operating System. Some operating systems such as MS-DOS and UNIX treat the command interpreter as a special program that is running when a job is initiated or when a user first logs on.

Some operating systems are frequently differentiated in the area of the shell with a user-friendly command interpreter such as Macintosh, Microsoft Windows. The mouse is moved to position the mouse pointer on images, or icons, on the screen that represent programs, files and system functions.

In some complex, powerful shells, commands are typed on a keyboard and displayed on a screen with the enter key signaling. These shells are MS-DOS and UNIX.

The command statements deal with process creation and management, I/O handling, secondary storage management, main-memory management, file-system access, protection and networking.

Operating System Services

The operating system provides an user friendly environment for the creation and execution of programs and provides different services to the user. Services may differ from one OS to another, but there are some common services which are as follows:

1. Program execution
2. I/O operation
3. File system
4. Communication
5. Error detection
6. Resource allocation
7. Accounting
8. Protection

Program execution: The primary function of an OS is to organize the execution of user computations in a computer system. A number of tasks need to be performed to execute a program. Instructions and data must be loaded into main memory. I/O devices and files must be initialized, and other resources must be prepared. The operating system handles these scheduling duties for the user.

I/O operation: A running program may require input and output. Each I/O device requires its own set of instructions or control signals for operation. The programmer or user can not control I/O devices directly. Thus OS must provide a uniform interface to hide the hardware details of I/O

File System Manipulation: file system is the most visible aspect of an OS. files are mapped by the OS onto physical devices. OS can provide system calls to create, write, read, reposition, delete and truncate files.

Communication: Two or more processes may exchange the information in two major ways — shared memory and message system. Communication through shared memory takes place between the processes that are executing on the same computer. In this case OS needs to provide only shared memory, other responsibilities are attached with application programmers. In message system, different CSs are tied together by Computer Network, and communication takes place between two processes in form of packets by OS.

Error detection: A variety of errors can occur while a computer system is running. These include internal and external hardware errors such as a memory error or a device failure or malfunction, and software errors such as arithmetic overflow, attempt to access illegal memory location etc. for each type of errors, the operating system must provide a response that clears the error condition with the least impact on running applications. The response may range from ending the program that caused the error, to retrying the operation, to simply reporting the error to the application.

Resource allocation: Many different types of resources such as CPU cycles, memory, file storage etc, are managed by operating system, specially in multiprogramming and multiuser system. To achieve this task efficiently, operating system have different routines such as CPU scheduling, job scheduling or general request and release code.

Accounting: A good operating system will collect ~~used~~ usage statistics for various resources and monitor performance parameters such as response time. On any system, this information is useful in anticipating the need for future enhancements and in tuning the system to improve performance. On a multiuser ~~sys~~ system, the information can be used for billing purposes.

Protection: Protection involves ensuring that all access to system resources are controlled. One process should not interfere in execution of other processes or with OS itself. Security of system provide authenticated

operations to the system by means of password, privilages etc. Invalid access attempts of session, external I/O devices etc. also implemented with OS

System Calls

System calls provide the interface between a process and the operating system, or system and applications programmers often invoke services of OS from their programs by means of system calls. These calls are generally available as assembly-language instructions and they are usually listed in the various manuals used by the assembly language programmers.

System commands issued by command-language users are normally converted into, and execute a series of system calls. In addition to providing most of the functionalities available to command-language users, system calls usually allow finer control over system operations and more direct access to hardware facilities, especially the input/output devices.

Certain systems allow system calls to be made directly from a higher level language program. Several languages such as C, C++, and Perl fall under this category e.g. UNIX system calls may be invoked directly from a C or C++ program. Every program makes heavy use of operating system by system calls.

Example: a program to read data from one file and to copy them to another file. Each and every operation which is required to finish this program is carried out by sequence of system calls.

Following operations are required.

1. Program needs names of two files : Input and output files. These names can be specified in many ways depending on the operating-system design.
   (i) Program ~~needs names~~ asks the user to give two files names and this is performed by a sequence of system calls in an interactive system
   (ii) User can select from a window in window and menu systems.
2. Operations like opening input file, creating output file, reading input file, writing the output file all require system calls.
3. If copying is over, closing the files and normal termination of program also requires system calls.
4. Errors may occur at any step which is displayed on monitor; program may be aborted abnormally, again system calls are required.

System calls occur in many ways, depending on the computer in use. More information is required other than identity of system which depends on OS and system call.

Three general methods are used to pass parameters to OS.

(i) To pass the parameters in registers.
(ii) If parameters are more than registers These are stored in a block or table in memory, and the address of block is passed as a parameter in a register. e.g. Linux.
(iii) Parameters can be pushed onto the stack by the program, and popped off the stack by the OS

Categories of System Calls :-

- process control
- file management
- device management
- information maintenance
- Communication.

↓ Process Control :

A running program needs to be able to halt its execution normally (end) or abnormally (abort). A process or job executing one program may want to load and execute another program. To create a process, create process, terminate process system calls are required. To control the execution of new process, it is required to determine and reset the attributes of a job or process including the job's priority, its maximum allowable execution time etc - get process attributes, set process attributes. After creation of new job, it is required to wait for a certain amount of time (wait time), to wait for a specific event to occur (wait event). Then the job or processes should signal when that event has occurred (signal event).

2. File management - We should be able to create and delete the files. After creation we need to open it for use. We may read, write or reposition. Finally, we need to close the file. To determine and reset the values of various attributes such as file name, file type, protection codes, accounting information etc, two systems are required which are get file attributes and set file attributes. Some OS may provide more system calls.

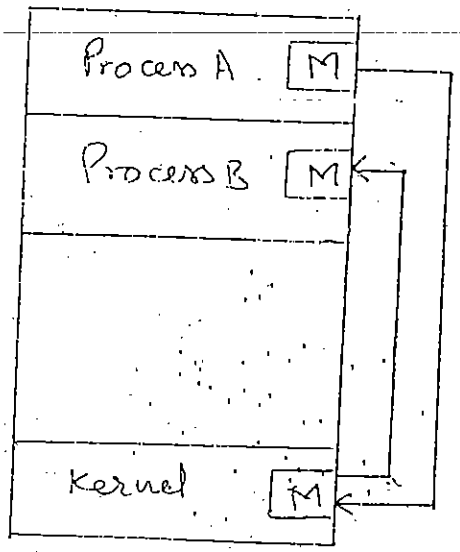
3. Device Management: A process may need additional resources to proceed such as more memory, tape drives, access to files etc. Many of the system calls for files are also needed for device — request device, release device. These functions are similar to the open and close system calls for files. We can read, write and reposition the device.

4 Information Maintenance: Many systems calls exist only for the purpose of transferring information between the user program and the O.S. Such as time, date, no. of users etc.
get time or date, set time or date;
get system data, set system data;
get process, file, or device attributes;
set process, file, or device attributes

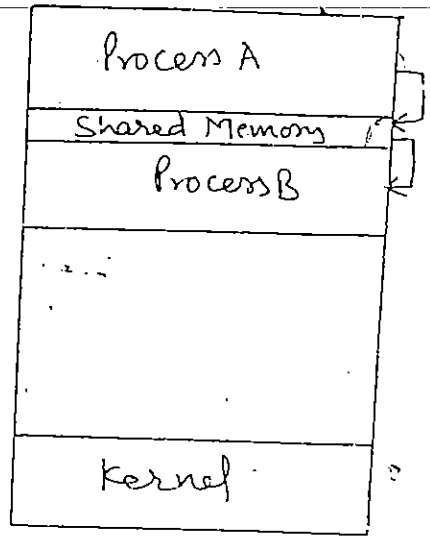
5. Communication: There are two common models of communication. In the message-passing model, information is exchanged through an interprocess communication facility provided by the O.S. Before communication takes place, a connection must be opened. Necessary system calls are get hostid, get processid, open connection, close connection, read message, write message etc.

In the shared-memory model, processes use map memory system calls to gain access to regions of memory owned by other processes. Processes are responsible for ensuring that they are not writing to the same location simultaneously.

Both of these methods are common in OS. Message-passing is useful when smaller numbers of data need to be exchanged, because no conflicts can arise. Shared memory allows maximum speed and convenience of communication, as it can be done at memory speeds when within a computer. Two problems exist: protection and synchronization.



Message Passing



Shared Memory

System Programs:

System programs provide a convenient environment for program development and execution. Some of them are simply user interfaces to system calls, some are more complex e.g. compiler, interpreter, loader, text editor etc. Most important system program for an OS is the command Interpreter. The main function of command interpreter is to get and execute the user specified command such as create, delete, list, print, copy, execute etc.

These commands can be implemented in two general ways: ① Command interpreter itself contains the code to execute the command. In this case, the number of commands determines the size of command interpreter, since each command requires its own implementing code. ② Command interpreter does not understand the command and it uses the command to identify a file to be loaded into memory and executed. Thus, UNIX command rm would search for the file named rm, load the file into memory and executed with given parameter. The function associated with rm would be defined completely by the code in the file rm.

New programmer can add new commands by creating new files with the proper names. The command interpreter is not changed for this addition. But to execute a command as a separate program the OS must provide a mechanism to pass parameters to that command, load the program etc. Interpretation of the parameters depends on the ~~sys~~ programmers which can cause the inconsistency across the ~~parame~~ programs.

Categories of System programs: These system programs can be divided into following categories:

1. File management - These programs create, delete, copy, rename, print, dump, list ~~em~~ edit and manipulate the files and directories e.g. cp, cat, rm in UNIX.

2. Status Information: Some programs ask the system for the date, time, amount of memory available, disk space, number of users etc.

3. File modification: Several text editors may be available to create and modify the contents of files stored on disk or tape. e.g. vi editor in UNIX.

4. Programming Language Support: Compilers, assemblers, and interpreters for common programming languages (such as C, C++, Java, Visual Basic and PERL) are provided with the OS.

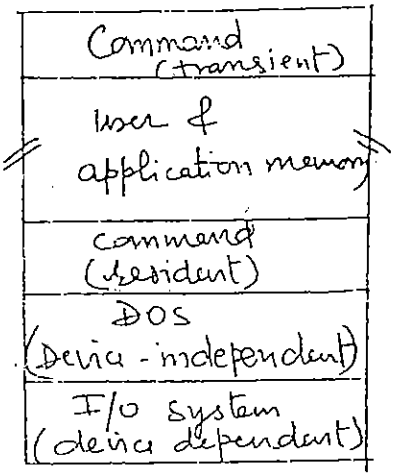
5. Program loading and execution: Once a program is assembled or compiled it must be loaded into memory to be executed. The system may provide absolute loaders, relocatable loaders, linkage editors, overlay loaders etc.

6. Communication: These programs provide the mechanism for creating virtual connections among processes, users, and different computer systems. They allow users to send messages to one another's screens, to browse message pages, to send e-mails to login remotely, to transfer files from one machine to another.

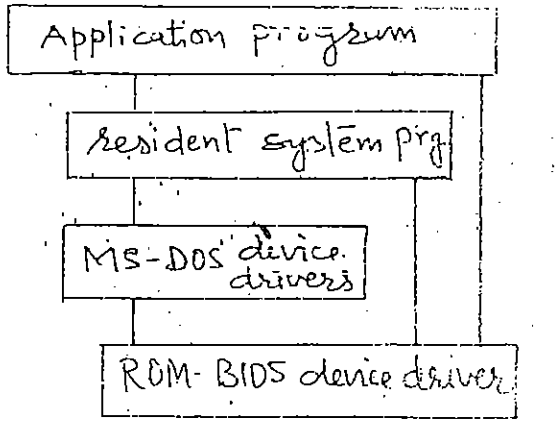
System Structure: Olden days the software code for OS was very small, more and more features were added, the code of OS became larger. So it was difficult to manage the OS with huge amount of code. The software was developed as a modular structure which was easy to develop and debug. For large OSs, modular programming alone was not sufficient, hierarchical layers were required.

1. Simple Structure: Operating systems started as small, simple, and limited systems and grew larger and larges.

Example: ① MS-DOS was written to provide the most functionality in the least space because of the limited hardware, so it was not divided into modules carefully.

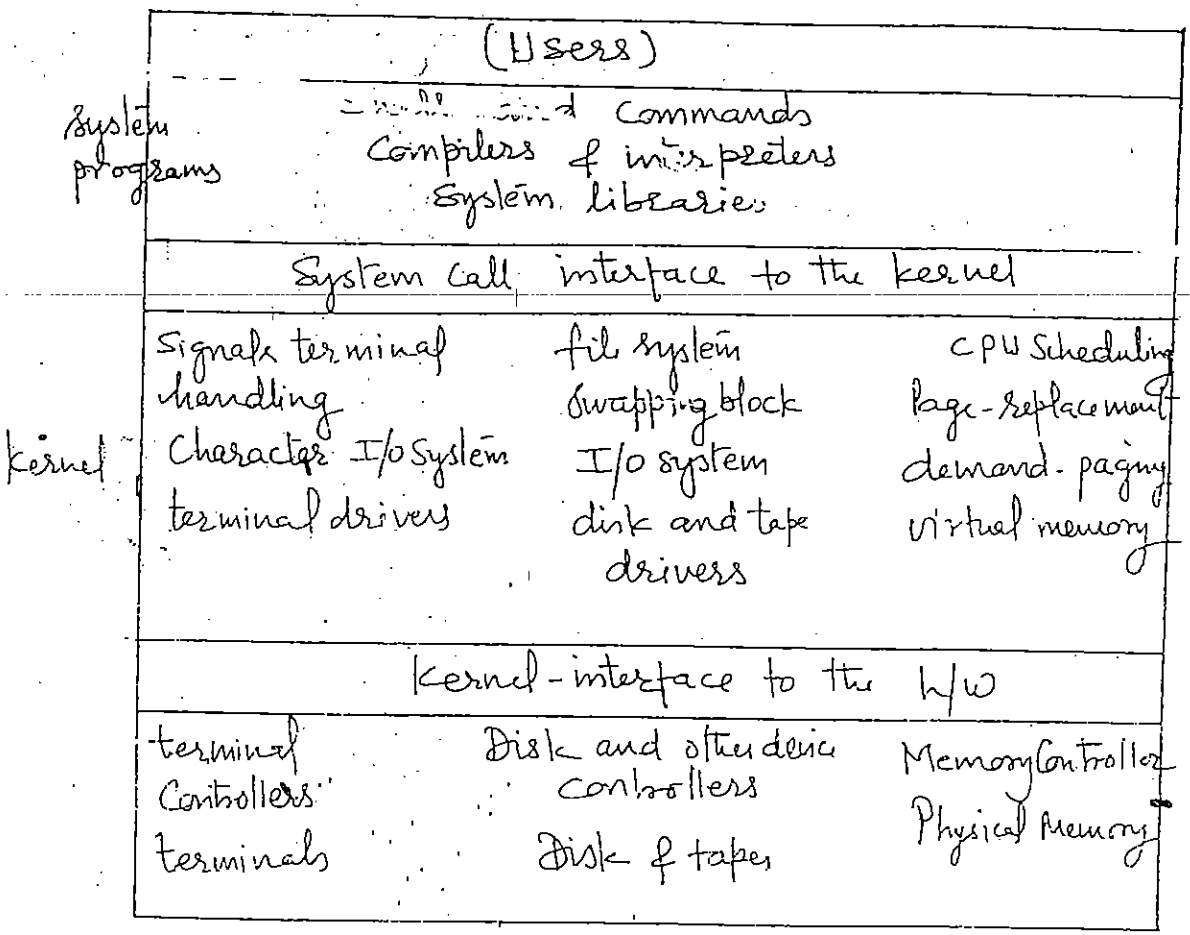


DOS - Memory Map



MS-DOS layer Structure

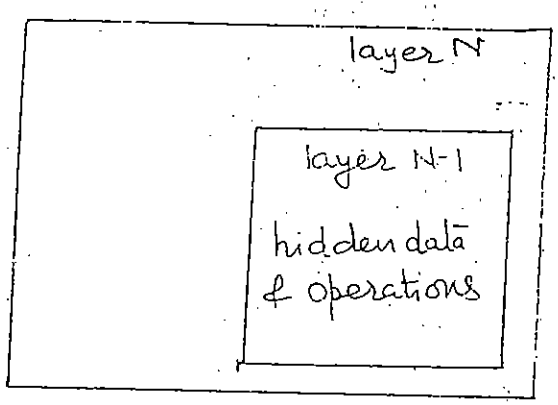
② UNIX was initially limited by hardware functionality. It consists of two separable parts: the kernel and the system programs. The kernel is further separated into a series of interfaces and device drivers, which were added and expanded over the years as UNIX evolved. Every thing below the system call interface and above the physical hardware is the kernel. The kernel provides the file system, CPU scheduling, memory management and other operating-system functions through system calls. System calls define the API to UNIX, the set of system programs commonly available defines the user interface.



UNIX - System Structure

2/ Layered Approach :

Modularization of a system can be achieved by layered approach. The operating system is broken up into a number of layers. Each layer is built on top of lower layers. The layer 0 i.e. bottom layer is the hardware and the layer N i.e. highest layer is the user interface. Each layer consists of data structures and a set of routines. Each layer can invoke operations of next lower level layer. Each layer hides implementation of data structures and operations, but provides the service to next higher level layer. Upper layer knows only what next layer can do but how, it does not know.



Benefits of layered approach :

1. Modularity : The design and implementation of the system are simplified when the system is broken down into layers.
2. Since each layer uses services of next lower layer debugging and system verification. The first layer can be debugged without any concern for rest of the system

Neelam Bawane

Limitations of layered approach: ①

This approach inv... the careful definition of the layers, because a la... can use only those operations functions given by lower layers e.g. the device driver for the disk space used by virtual memory algorithms must be a lower layer than that of memory-management routines, because memory management requires the ability to use the disk space.

② This approach is less efficient. e.g. when a user program executes an I/O operation, it executes a system that is trapped to the I/O layer which calls the memory management layer, which in turn calls the CPU-scheduling layer which is then passed to the hardware. At each layer, the parameters may be modified & each layer adds overhead to the system call, thus a system call takes more time.

Modified approach to overcome limitations :-

Fewer layers with more functionalities are being designed, providing most of the advantages of modularized code while avoiding the difficult problems of layer definition and interaction.

e.g. ① OS/2 (descendant of MS-DOS) was implemented in a more layered fashion than MS-DOS due to multitasking, dual mode operation and other features. OS/2 allows OS with more control over hardware

② The first release of Windows NT had a highly layer-oriented organization, but performance was less than that of Windows 95. Windows NT 4.0 was released in which layers were moved more user space to kernel space and integrated more closely.

Microkernels

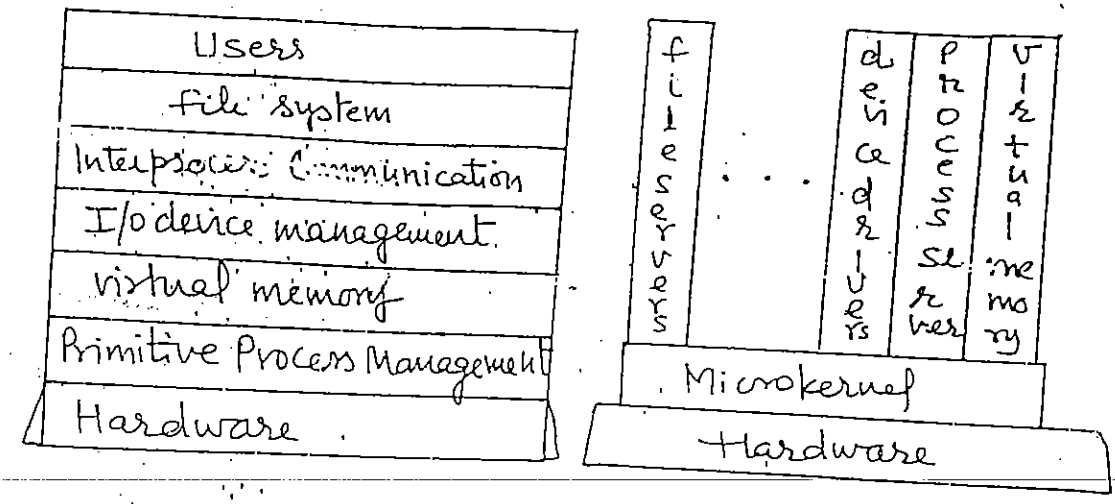
The microkernel approach was popularized by its use in the Mach operating system. Layered operating systems were developed in which functions are organized hierarchically and interaction only takes place between adjacent layers. Most or all of the layers execute in kernel mode. ~~In micro kernel approach —~~

Only absolutely essential core operating system functions should be in the kernel. Less essential services and applications are built on the microkernel and execute in user mode.

Many services that traditionally have been part of the operating system are now external subsystems that interact with the kernel and with each other, these include device drivers, file systems, virtual memory manager, windowing system, and security services.

A microkernel approach replaces the traditional ~~vertical~~ horizontal, layered stratification of an OS with a ~~horizontal~~ vertical one. Operating system components external to the microkernel are implemented as server processes, these interact with each other on a ~~peer~~ peer basis, typically by means of messages passed through the microkernel. Thus, the microkernel validates and passes the messages between components, and grants access to hardware.

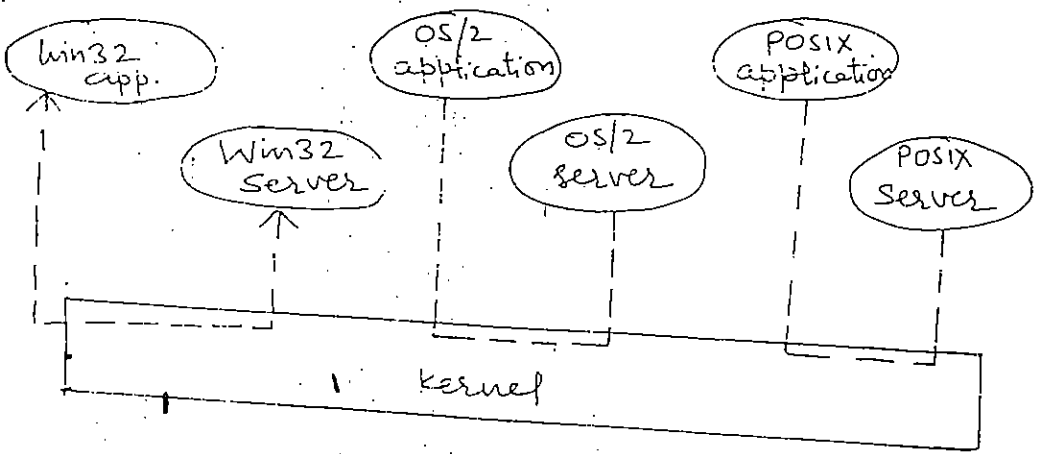
e.g. Tru 64 UNIX - UNIX interface & Mach kernel
Windows NT - hybrid structure
QNX real-time OS
Apple Mac OS X server OS - Mach kernel



Layered kernel

Microkernel approach

Windows NT: Windows NT is designed to run various applications such as Win32 (native windows applications), OS/2 and POSIX. It provides a server that runs in user space for each application type. Client programs for each application type also run in user space. The kernel coordinates the message passing between client applications and application servers.



Windows NT

2. ~~Simple Structure~~

Benefits of Microkernel Organization

1. Uniform Interfaces: Microkernel design imposes a uniform interface on requests made by a process. Processes need not distinguish between kernel-level and user-level services because all such services are provided by means of message passing.

2. Extensibility: It allows the addition of new services and multiple services in the functional area. When a new feature is added, only selected servers need to be modified or added.

3. Flexibility: Existing features can be subtracted to produce a smaller, more efficient implementation.

4. Portability: All processor-specific code is in the microkernel. Thus, changes needed to port the system to a new processor are fewer.

5. Reliability: A small microkernel can be rigorously tested. Its use of a small number of application programming interfaces (APIs) improves the chance of producing quality code for the OS services outside the kernel.

Virtual Machines

Normally, a computer system is made up of layers. The system programs above the kernel are able to use either system calls or hardware instructions.

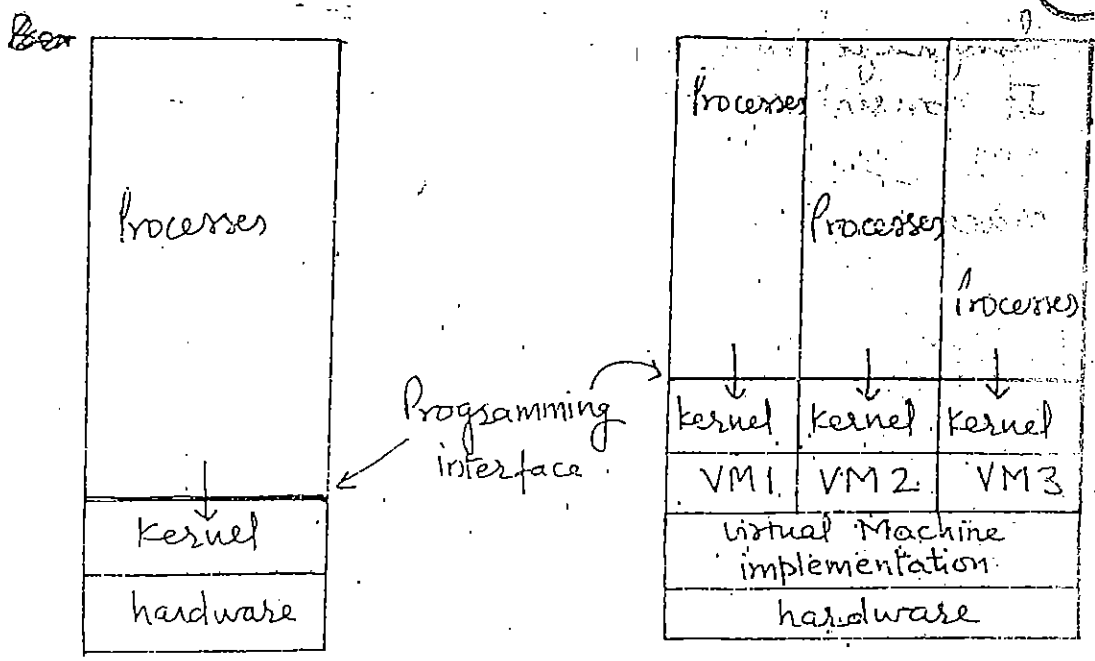
By using CPU scheduling and virtual memory techniques, an OS can create the illusion that a process has its own processor with its own (virtual) memory. The virtual machine approach does not provide any additional functionality, but rather provides an interface that is identical to the underlying bare hardware. Each process is provided with a (virtual) copy of the underlying computer.

The physical computer shares resources to create the virtual machines. CPU scheduling can share out the CPU to create the appearance that users have their own processors.

Spooling and file system can create virtual card readers and virtual line printers. A normal user time-sharing terminal provides the function of the virtual-machine operator's console.

A major difficulty is disk system. It cannot allocate a disk drive to virtual machine because no. of virtual machines is more than no. of disk drives. Virtual-machine software also needs disk space to provide virtual memory. The solution is virtual disks (or minidisks). The system implements each minidisk by allocating as many tracks or on the physical disks as the minidisk needs.

The virtual machine software is concerned with multiprogramming multiple virtual machines onto a physical machine.



Non virtual Machine

Virtual Machine

Benefits :

1. Security :- The VM provides a robust level security. Each virtual machine is completely isolated from all other virtual machines, so we have no security problems as the various system resources are completely protected.

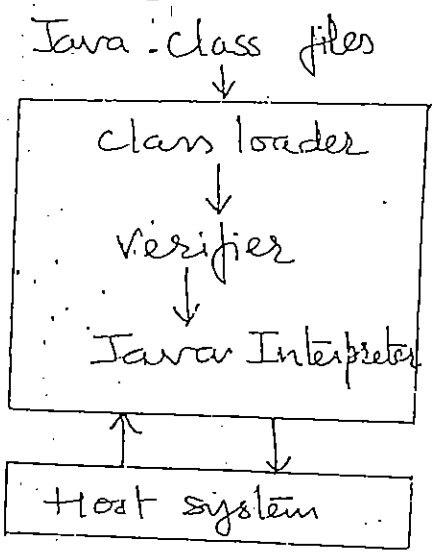
2. System development : The VM allows system development to be done without disrupting normal system operation. System programmers are given their own virtual machine, and system development is done on the virtual machine instead of on the actual physical machine.

3. Sharing of resources is not direct. It is possible through sharing of minidisk which is implemented by a software. It is also possible to define a virtual communication network implemented by software.

Java Virtual Machine:

Java is an object-oriented language introduced by Sun Microsystems in 1995. It consists of a language specification, a large API library and a specification for Java virtual machine (JVM).

The JVM is a specification for an abstract computer. The implementation of the JVM is specific for each system - such as Windows or UNIX - and it abstracts the system in a standard way to the Java program, providing a clean, architecture-neutral interface. The JVM consists of a class loader, a class verifier, and a Java interpreter (or just-in-time (JIT))



Java virtual Machine

Execution:

Java objects are specified with the class construct, a Java program consists of one or more classes. For each Java class, the Java compiler produces an architecture-neutral bytecodes (.class) file. The class loader loads .class files from both the Java program and the Java API for execution by the Java interpreter. The verifier

Checks the validity of Java bytecode, overflow & underflow of stack, no pointer arithmetic. Then Java interpreter interprets the bytecode one at a time. There may be a just-in-time (JIT) compiler that turns the architecture-neutral bytecodes into native machine language for the host computer. Most implementations of the JVM use a JIT compiler for enhanced performance.

Advantages:

1. Garbage Collection: JVM automatically manages memory by performing garbage collection – the practice of reclaiming memory from objects no longer in use and returning it to the system.

2. A complete environment: Its virtual-machine design provides a secure, efficient, object-oriented, portable and architecture-neutral platform.