MCA Part II

Paper-XIII

Topic: Introduction to Operating System

Prepared by: Dr. Kiran Pandey
(School of Computer Science)

Email-id: kiranpandey.nou@gmail.com

INTRODUCTION

An Operating System (OS) is an interface between a computer user and computer hardware. An operating system is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.

The purpose of an operating system is to provide an environment in which a user may execute the programs. Operating Systems are viewed as resource managers. The main resource is the computer hardware in the form of processors, storage, input/output devices, communication devices, and data. Some of the operating system functions are: implementing the user interface, sharing hardware among users, allowing users to share data among themselves, preventing users from interfering with one another, scheduling resources among users, facilitating input/output, recovering from errors, accounting for resource usage, facilitating

parallel operations, organising data for secure and rapid access, and handling network communications.

Operating system Concepts

An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

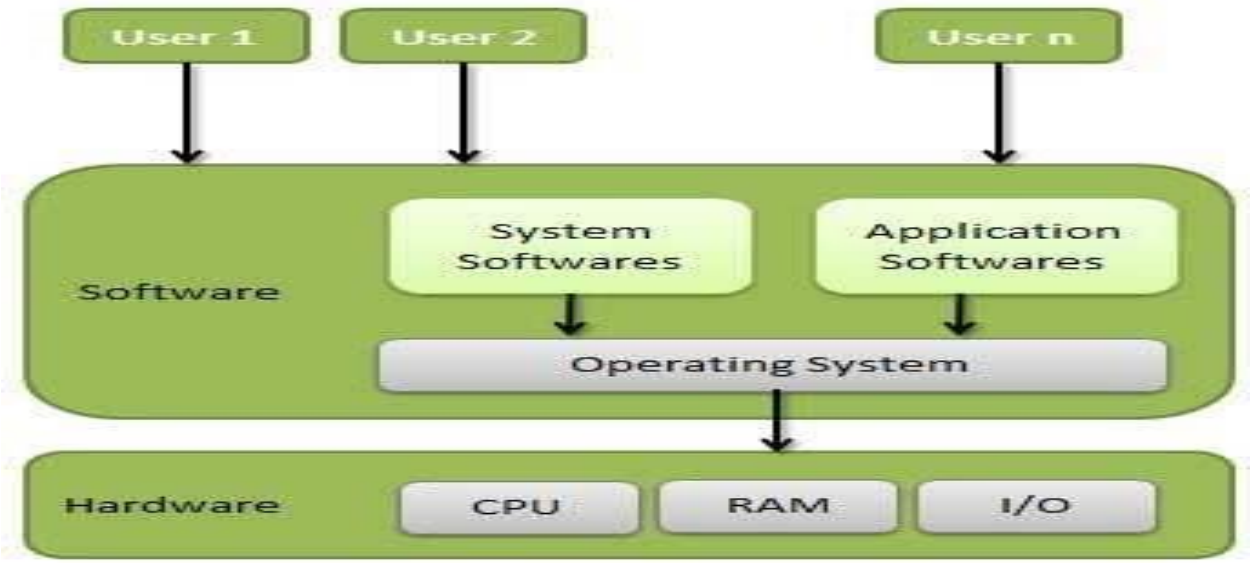

Figure 1: Components of a computer system

In a computer system as shown in figure 1, we find four main components: the hardware, the operating system, the application software and the users. In a computer system the hardware provides the basic computing resources. The applications programs define the way in which these resources are used to solve the computing problems of the users. The operating system controls and coordinates the use of the hardware among the various systems programs and application programs for the various users.

We can view an operating system as a **resource allocator**. A computer system has many resources (hardware and software) that may be required to solve a problem: CPU time, memory space, files storage space, input/output devices etc. The operating system acts as the manager of these resources and allocates them to specific programs and users as necessary for their tasks. Since there may be many, possibly conflicting, requests for resources, the operating system must decide which

requests are allocated resources to operate the computer system fairly and efficiently.

An operating system is a **control program**. This program controls the execution of user programs to prevent errors and improper use of the computer. Operating systems exist because: they are a reasonable way to solve the problem of creating a usable computing system. The fundamental goal of a computer system is to execute user programs and solve user problems.

While there is no universally agreed upon definition of the concept of an operating system, the following is a reasonable starting point: A computer's operating system is a group of programs designed to serve two basic purposes:

- To control the allocation and use of the computing system's resources among the various users and tasks, and
- To provide an interface between the computer hardware and the programmer that simplifies and makes feasible the creation, coding, debugging, and maintenance of application programs.

An effective operating system should accomplish the following functions:

- Should act as a command interpreter by providing a user friendly environment.
- Should facilitate communication with other users.
- Facilitate the directory/file creation along with the security option.
- Provide routines that handle the intricate details of I/O programming.
- Provide access to compilers to translate programs from high-level languages to machine language
- Provide a loader program to move the compiled program code to the computer's memory for execution.

- Assure that when there are several active processes in the computer, each will get fair and non-interfering access to the central processing unit for execution.
- Take care of storage and device allocation.
- Provide for long term storage of user information in the form of files.
- Permit system resources to be shared among users when appropriate, and be protected from unauthorized or mischievous intervention as necessary.

Though systems programs such as editors and translators and the various utility programs (such as sort and file transfer program) are not usually considered part of the operating system, the operating system is responsible for providing access to these system resources.

TYPES OF OPERATING SYSTEM

Batch Operating System

Batch processing is a technique in which an Operating System collects the programs and data together in a batch before processing starts. An operating system does the following activities related to batch processing – The OS defines a job which has predefined sequence of commands, programs and data as a single unit.

Batch processing is the execution of a series of jobs in a program on a computer without manual intervention (non-interactive). Strictly speaking, it is a processing mode: the execution of a series of programs each on a set or "batch" of inputs, rather than a single input (which would instead be a custom job).

The users of a batch operating system do not interact with the computer directly. Each user prepares his job on an off-line device like punch cards and submits it to the computer operator. To speed up processing, jobs with similar needs are batched together and run as a group.

The problems with Batch Systems are as follows –

- Lack of interaction between the user and the job.
- CPU is often idle, because the speed of the mechanical I/O devices is slower than the CPU.
- Difficult to provide the desired priority.

An example of batch processing is the way that credit card companies process billing. The customer does not receive a bill for each separate credit card purchase but one monthly bill for all of that month s purchases.

Time Sharing Operating System

Time sharing operating system allows many users to share the computer resources simultaneously. In other words, time sharing refers to the allocation of computer resources in time slots to several programs simultaneously. For example a mainframe computer that has many users logged on to it. Each user uses the resources of the mainframe -i.e. memory, CPU etc. The users feel that they are exclusive user of the CPU, even though this is not possible with one CPU i.e. shared among different users.

The time sharing systems were developed to provide an interactive use of the computer system. A time shared system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time-shared computer. It allows many users to share the computer resources simultaneously. As the system switches rapidly from one user to the other, a short time slot is given to each user for their executions.

The time sharing system provides the direct access to a large number of users where CPU time is divided among all the users on scheduled basis. The OS allocates a set of time to each user. When this time is expired, it passes control to the next user on the system. The time allowed is extremely small and the users are given the impression that they each have their own CPU and they are the sole owner of the CPU. This short period of time during that a user gets attention of the CPU; is known

as a *time slice or a quantum.* The Multics & UNIX operating systems are time sharing Operating Systems.

Advantages of Timesharing operating systems are as follows –

- Provides the advantage of quick response.
- Avoids duplication of software.
- Reduces CPU idle time.

Disadvantages of Time-sharing operating systems are as follows –

- Problem of reliability.
- Question of security and integrity of user programs and data.
- Problem of data communication.

Real Time Operating System

A real-time system is defined as a data processing system in which the time interval required to process and respond to inputs is so small that it controls the environment. The time taken by the system to respond to an input and display of required updated information is termed as the **response time**. So in this method, the response time is very less as compared to online processing.

Real-time systems are used when there are rigid time requirements on the operation of a processor or the flow of data and real-time systems can be used as a control device in a dedicated application. A real-time operating system must have well-defined, fixed time constraints, otherwise the system will fail. For example, scientific experiments, medical imaging systems, industrial control systems, weapon systems, robots, air traffic control systems, etc.

There are two types of real-time operating systems.

Hard real-time systems

Hard real-time systems guarantee that critical tasks complete on time. In hard real-time systems, secondary storage is limited or missing and the data is stored in ROM. In these systems, virtual memory is almost never found.

Soft real-time systems

Soft real-time systems are less restrictive. A critical real-time task gets priority over other tasks and retains the priority until it completes. Soft real-time systems have limited utility than hard real-time systems. For example, multimedia, virtual reality, Advanced Scientific Projects like undersea exploration and planetary rovers, etc.

Multiprogramming Operating System

In a multiprogramming system there are one or more programs loaded in main memory which are ready to execute. Only one program at a time is able to get the CPU for executing its instructions (i.e., there is at most one process running on the system) while all the others are waiting their turn.
The main idea of multiprogramming is to maximize the use of CPU time. Indeed, suppose the currently running process is performing an I/O task (which, by definition, does not need the CPU to be accomplished). Then, the OS may interrupt that process and give the control to one of the other in-main-memory programs that are ready to execute (i.e. *process context switching*). In this way, no CPU time is wasted by the system waiting for the I/O task to be completed, and a running process keeps executing until either it voluntarily releases the CPU or when it blocks for an I/O operation. Therefore, the ultimate goal of multiprogramming is to keep the CPU busy as long as there are processes ready to execute.

Multitasking

Multitasking has the same meaning of multiprogramming but in a more general sense, as it refers to having multiple (programs, processes, tasks, threads) running at the same time. This term is used in modern operating systems when multiple tasks share a common processing resource (e.g., CPU and Memory). At any time the CPU is executing one task only while other tasks waiting their turn. The illusion of parallelism is achieved when the CPU is reassigned to another task (i.e. *process* or *thread context switching*).
There are subtle differences between multitasking and multiprogramming. A *task* in a multitasking operating system is not a whole application program but it can also refer to a "thread of execution" when one process is divided into sub-tasks. Each smaller task does not hijack the CPU until it finishes like in the older

multiprogramming but rather a fair share amount of the CPU time called quantum. Just to make it easy to remember, both multiprogramming and multitasking operating systems are **(CPU) time sharing** systems. However, while in multiprogramming (older OSs) one program as a whole keeps running until it blocks, in multitasking (modern OSs) time sharing is best manifested because each running process takes only a fair quantum of the CPU time.

Network Operating System

A networked computing system is a collection of physical interconnected computers. The operating system of each of the interconnected computers must contain, in addition to its own stand-alone functionality, provisions for handing communication and transfer of program and data among the other computers with which it is connected.

A Network Operating System runs on a server and provides the server the capability to manage data, users, groups, security, applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), a private network or to other networks.

Examples of network operating systems include Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD.

The advantages of network operating systems are as follows –

- Centralized servers are highly stable.
- Security is server managed.
- Upgrades to new technologies and hardware can be easily integrated into the system.
- Remote access to servers is possible from different locations and types of systems.

The disadvantages of network operating systems are as follows –

- High cost of buying and running a server.
- Dependency on a central location for most operations.
- Regular maintenance and updates are required.

Distributed Operating System

A distributed computing system consists of a number of computers that are connected and managed so that they automatically share the job processing load among the constituent computers, or separate the job load as appropriate particularly configured processors. Such a system requires an operating system which, in addition to the typical stand-alone functionality, provides coordination of the operations and information flow among the component computers.

The processors communicate with one another through various communication lines (such as high-speed buses or telephone lines). These are referred as **loosely coupled systems** or distributed systems. Processors in a distributed system may vary in size and function. These processors are referred as sites, nodes, computers, and so on.

The advantages of distributed systems are as follows –

- With resource sharing facility, a user at one site may be able to use the resources available at another.
- Speedup the exchange of data with one another via electronic mail.
- If one site fails in a distributed system, the remaining sites can potentially continue operating.
- Better service to the customers.
- Reduction of the load on the host computer.
- Reduction of delays in data processing.

Operating System for Embedded Devices

As embedded systems (PDAs, cellphones, point-of-sale devices, VCR's, industrial robot control, or even your toaster) become more complex hardware-wise with every generation, and more features are put into them day-by-day, applications they run require more and more to run on actual operating system code in order to keep the development time reasonable. Some of the popular OS are:

- Nexus's Conix - an embedded operating system for ARM processors.

- Sun's Java OS - a standalone virtual machine not running on top of any other OS; mainly targeted at embedded systems.
- Palm Computing's Palm OS - Currently the leader OS for PDAs, has many applications and supporting companies.
- Microsoft's Windows CE and Windows NT Embedded OS.

OPERATING SYSTEM COMPONENTS

The basic functions of an operating system are:

- Process Management
- Memory Management
- Secondary Storage Management
- I/O Management
- File Management
- Protection
- Networking Management
- Command Interpretation.

Process Management

The CPU executes a large number of programs. While its main concern is the execution of user programs, the CPU is also needed for other system activities. These activities are called processes. A process is a program in execution. Typically, a batch job is a process. A time-shared user program is a process. A system task, such as spooling, is also a process. For now, a process may be considered as a job or a time-shared program, but the concept is actually more general.

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

- The creation and deletion of both user and system processes
- The suspension and resumption of processes.
- The provision of mechanisms for process synchronization
- The provision of mechanisms for deadlock handling.

Memory Management

Memory is the most expensive part in the computer system. Memory is a large array of words or bytes, each with its own address. Interaction is achieved through a sequence of reads or writes of specific memory address. The CPU fetches from and stores in memory.

There are various algorithms that depend on the particular situation to manage the memory. Selection of a memory management scheme for a specific system depends upon many factors, but especially upon the hardware design of the system. Each algorithm requires its own hardware support

The operating system is responsible for the following activities in connection with memory management.

- Keep track of which parts of memory are currently being used and by whom.
- Decide which processes are to be loaded into memory when memory space becomes available.
- Allocate and deallocate memory space as needed.

Secondary Storage Management

The main purpose of a computer system is to execute programs. These programs, together with the data they access, must be in main memory during execution. Since the main memory is too small to permanently accommodate all data and program, the computer system must provide secondary storage to backup main memory. Most modem computer systems use disks as the primary on-line storage of information, of both programs and data. Most programs, like compilers, assemblers, sort routines, editors, formatters, and so on, are stored on the disk until loaded into memory, and

then use the disk as both the source and destination of their processing. Hence the proper management of disk storage is of central importance to a computer system.

There are few alternatives. Magnetic tape systems are generally too slow. In addition, they are limited to sequential access. Thus tapes are more suited for storing infrequently used files, where speed is not a primary concern.

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

• Free space management

• Storage allocation

• Disk scheduling.

I/O Management

One of the purposes of an operating system is to hide the peculiarities or specific hardware devices from the user. For example, in UNIX, the peculiarities of I/O devices are hidden from the bulk of the operating system itself by the I/O system. The operating system is responsible for the following activities in connection to I/O management:

• A buffer caching system

• To activate a general device driver code

• To run the driver software for specific hardware devices as and when required

File Management

File management is one of the most visible services of an operating system. Computers can store information in several different physical forms: magnetic tape, disk, and drum are the most common forms. Each of these devices has it own characteristics and physical organization.

For convenient use of the computer system, the operating system provides a uniform logical view of information storage. The operating system abstracts from the physical properties of its storage devices to define a logical storage unit, the file. Files are mapped, by the operating system, onto physical devices.

A file is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data. Data files may be numeric, alphabetic or alphanumeric. Files may be free-form, such as text files, or may be rigidly formatted. In general a files is a sequence of bits, bytes, lines or records whose meaning is defined by its creator and user. It is a very general concept.

The operating system implements the abstract concept of the file by managing mass storage device, such as types and disks. Also files are normally organised into directories to ease their use. Finally, when multiple users have access to files, it may be desirable to control by whom and in what ways files may be accessed.

The operating system is responsible for the following activities in connection to the file management

- The creation and deletion of files.
- The creation and deletion of directory.
- The support of primitives for manipulating files and directories.
- The mapping of files onto disk storage.
- Backup of files on stable (non volatile) storage.
- Protection and security of the files.

Protection

The various processes in an operating system must be protected from each other's activities. For that purpose, various mechanisms which can be used to ensure that

the files, memory segment, CPU and other resources can be operated on only by those processes that have gained proper authorization from the operating system.

For example, memory addressing hardware ensures that a process can only execute within its own address space. The timer ensures that no process can gain control of the CPU without relinquishing it. Finally, no process is allowed to do its own I/O, to protect the integrity of the various peripheral devices. Protection refers to a mechanism for controlling the access of programs, processes, or users to the resources defined by a computer controls to be imposed, together with some means of enforcement. Protection can improve reliability by detecting latent errors at the interfaces between component subsystems. Early detection of interface errors can often prevent contamination of a healthy subsystem by a subsystem that is malfunctioning. An unprotected resource cannot defend against use (or misuse) by an unauthorized or incompetent user.

Networking

A distributed system is a collection of processors that do not share memory or a clock. Instead, each processor has its own local memory, and the processors communicate with each other through various communication lines, such as high speed buses or telephone lines. Distributed systems vary in size and function. They may involve microprocessors, workstations, minicomputers, and large general purpose computer systems.

The processors in the system are connected through a communication network, which can be configured in the number of different ways. The network may be fully or partially connected. The communication network design must consider routing and connection strategies and the problems of connection and security.

A distributed system provides the user with access to the various resources the system maintains. Access to a shared resource allows computation speed-up, data availability, and reliability.

Command Interpretation

One of the most important components of an operating system is its command interpreter. The command interpreter is the primary interface between the user and the rest of the system.

Many commands are given to the operating system by control statements. When a new job is started in a batch system or when a user logs-in to a time-shared system, a program which reads and interprets control statements is automatically executed. This program is variously called (1) the control card interpreter, (2) the command line interpreter, (3) the shell (in Unix), and so on. Its function is quite simple: get the next command statement, and execute it.

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

Operating System Structure

The design of an operating system architecture traditionally follows the *separation of concerns* principle. This principle suggests structuring the operating system into relatively independent parts that provide simple individual features, thus keeping the complexity of the design manageable.

Besides managing complexity, the structure of the operating system can influence key features such as robustness or efficiency:

- The operating system possesses various privileges that allow it to access otherwise protected resources such as physical devices or application memory. When these privileges are granted to the individual parts of the operating system that require them, rather than to the operating system as a whole, the potential for both accidental and malicious privileges misuse is reduced.
- Breaking the operating system into parts can have adverse effect on efficiency because of the overhead associated with communication between the individual parts. This overhead can be exacerbated when coupled with hardware mechanisms used to grant privileges.

The following sections outline typical approaches to structuring the operating system.

Monolithic Systems

A monolithic design of the operating system architecture makes no special accommodation for the special nature of the operating system. Although the design

follows the separation of concerns, no attempt is made to restrict the privileges granted to the individual parts of the operating system. The entire operating system executes with maximum privileges. The communication overhead inside the monolithic operating system is the same as the communication overhead inside any other software, considered relatively low.

CP/M and DOS are simple examples of monolithic operating systems. Both CP/M and DOS are operating systems that share a single address space with the applications. In CP/M, the 16 bit address space starts with system variables and the application area and ends with three parts of the operating system, namely CCP (Console Command Processor), BDOS (Basic Disk Operating System) and BIOS (Basic Input/output System). In DOS, the 20 bit address space starts with the array of interrupt vectors and the system variables, followed by the resident part of DOS and the application area and ending with a memory block used by the video card and BIOS.

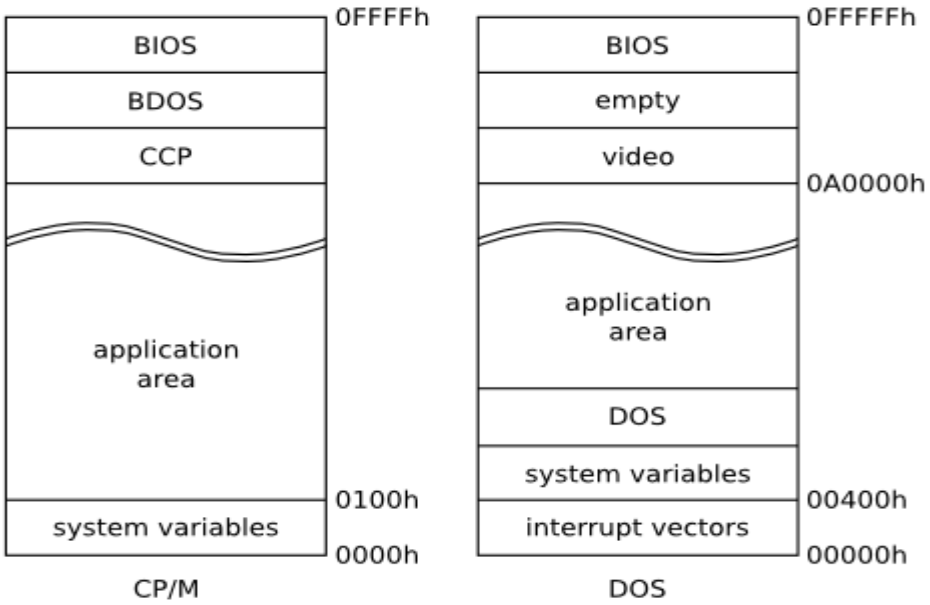

Figure 2 : Simple Monolithic Operating Systems

Most contemporary operating systems, including Linux and Windows, are also considered monolithic, even though their structure is certainly significantly different from the simple examples of CP/M and DOS.

Layered Systems

A layered design of the operating system architecture attempts to achieve robustness by structuring the architecture into layers with different privileges. The most privileged layer would contain code dealing with interrupt handling and context switching, the layers above that would follow with device drivers, memory management, file systems, user interface, and finally the least privileged layer would contain the applications.

MULTICS is a prominent example of a layered operating system, designed with eight layers formed into *protection rings*, whose boundaries could only be crossed using specialized instructions. Contemporary operating systems, however, do not use the layered design, as it is deemed too restrictive and requires specific hardware support.

Microkernel Systems

A microkernel design of the operating system architecture targets robustness. The privileges granted to the individual parts of the operating system are restricted as much as possible and the communication between the parts relies on a specialized communication mechanisms that enforce the privileges as necessary. The communication overhead inside the microkernel operating system can be higher than the communication overhead inside other software, however, research has shown this overhead to be manageable.

Experience with the microkernel design suggests that only very few individual parts of the operating system need to have more privileges than common applications. The microkernel design therefore leads to a small system kernel, accompanied by additional system applications that provide most of the operating system features.

MACH is a prominent example of a microkernel that has been used in contemporary operating systems, including the NextStep and OpenStep systems and, notably, OS X. Most research operating systems also qualify as microkernel operating systems.

Virtualized Systems

Attempts to simplify maintenance and improve utilization of operating systems that host multiple independent applications have led to the idea of running multiple operating systems on the same computer. Similar to the manner in which the operating system kernel provides an isolated environment to each hosted application, virtualized systems introduce a *hypervisor* that provides an isolated environment to each hosted operating system.

Hypervisors can be introduced into the system architecture in different ways.

- A *native* hypervisor runs on bare hardware, with the hosted operating systems residing above the hypervisor in the system structure. This makes it possible to implement an efficient hypervisor, paying the price of maintaining a hardware specific implementation.
- A *hosted* hypervisor partially bypasses the need for a hardware specific implementation by running on top of another operating system. From the bottom up, the system structure then starts with the host operating system that includes the hypervisor, and then the guest operating systems, hosted above the hypervisor.

A combination of the native and the hosted approaches is also possible. The hypervisor can implement some of its features on bare hardware and consult the hosted operating systems for its other features. A common example of this approach

is to implement the processor virtualization support on bare hardware and use a dedicated hosted operating system to access the devices that the hypervisor then virtualizes for the other hosted operating systems.

OPERATING SYSTEM SERVICES

An Operating System provides services to both the users and to the programs.

- It provides programs an environment to execute.
- It provides users the services to execute the programs in a convenient manner.

Following are a few common services provided by an operating system –

- Program execution
- I/O operations
- File System manipulation
- Communication
- Error Detection
- Resource Allocation
- Protection

Program execution

Operating systems handle many kinds of activities from user programs to system programs like printer spooler, name servers, file server, etc. Each of these activities is encapsulated as a process.

A process includes the complete execution context (code to execute, data to manipulate, registers, OS resources in use). Following are the major activities of an operating system with respect to program management –

- Loads a program into memory.
- Executes the program.

- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

I/O Operation

An I/O subsystem comprises of I/O devices and their corresponding driver software. Drivers hide the peculiarities of specific hardware devices from the users.

An Operating System manages the communication between user and device drivers.

- I/O operation means read or write operation with any file or any specific I/O device.
- Operating system provides the access to the required I/O device when required.

File system manipulation

A file represents a collection of related information. Computers can store files on the disk (secondary storage), for long-term storage purpose. Examples of storage media include magnetic tape, magnetic disk and optical disk drives like CD, DVD. Each of these media has its own properties like speed, capacity, data transfer rate and data access methods.

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Following are the major activities of an operating system with respect to file management –

- Program needs to read a file or write a file.
- The operating system gives the permission to the program for operation on file.
- Permission varies from read-only, read-write, denied and so on.

- Operating System provides an interface to the user to create/delete files.
- Operating System provides an interface to the user to create/delete directories.
- Operating System provides an interface to create the backup of file system.

Communication

In case of distributed systems which are a collection of processors that do not share memory, peripheral devices, or a clock, the operating system manages communications between all the processes. Multiple processes communicate with one another through communication lines in the network.

The OS handles routing and connection strategies, and the problems of contention and security. Following are the major activities of an operating system with respect to communication –

- Two processes often require data to be transferred between them
- Both the processes can be on one computer or on different computers, but are connected through a computer network.
- Communication may be implemented by two methods, either by Shared Memory or by Message Passing.

Error handling

Errors can occur anytime and anywhere. An error may occur in CPU, in I/O devices or in the memory hardware. Following are the major activities of an operating system with respect to error handling –

- The OS constantly checks for possible errors.
- The OS takes an appropriate action to ensure correct and consistent computing.

Resource Management

In case of multi-user or multi-tasking environment, resources such as main memory, CPU cycles and files storage are to be allocated to each user or job. Following are the major activities of an operating system with respect to resource management –

- The OS manages all kinds of resources using schedulers.
- CPU scheduling algorithms are used for better utilization of CPU.

Protection

Considering a computer system having multiple users and concurrent execution of multiple processes, the various processes must be protected from each other's activities.

Protection refers to a mechanism or a way to control the access of programs, processes, or users to the resources defined by a computer system. Following are the major activities of an operating system with respect to protection –

- The OS ensures that all access to system resources is controlled.
- The OS ensures that external I/O devices are protected from invalid access attempts.
- The OS provides authentication features for each user by means of passwords.

System Calls

- System calls provide a means for user or application programs to call upon the services of the operating system.
- Generally written in C or C++, although some are written in assembly for optimal performance.
- Figure 2 illustrates the sequence of system calls required to copy a file:

EXAMPLES OF OPERATING SYSTEM

Here we will study about some popular operating systems.

DOS

DOS (Disk Operating System) was the first widely-installed operating system for personal computers. It is a master control program that is automatically run when you start your personal computer (PC). DOS stays in the computer all the time letting you run a program and manage files. It is a single-user operating system from Microsoft for the PC. It was the first OS for the PC and is the underlying control program for Windows 3.1, 95, 98 and ME. Windows NT, 2000 and XP emulate DOS in order to support existing DOS applications.

UNIX

UNIX operating systems are used in widely-sold workstation products from Sun Microsystems, Silicon Graphics, IBM, and a environment and the client/server program model were important elements in the development of the Internet and the reshaping of computing as centered in networks rather than in individual computers. Linux, a UNIX derivative available in both "free software" and commercial versions, is increasing in popularity as an alternative to proprietary operating systems.

UNIX is written in **C**. Both UNIX and C were developed by AT&T and freely distributed to government and academic institutions, causing it to be ported to a wider variety of machine families than any other operating system. As a result, UNIX became synonymous with "open systems".

UNIX is made up of the kernel, file system and shell (command line interface). The major shells are the Bourne shell (original), C shell and Korn shell. The UNIX vocabulary is exhaustive with more than 600 commands that manipulate data and text in every way conceivable. Many commands are cryptic, but just as Windows hid the DOS prompt, the Motif GUI presents a friendlier image to UNIX users. Even with its many versions, UNIX is widely used in mission critical applications for client/server and transaction processing systems. The UNIX versions that are widely used are Sun's Solaris, Digital's UNIX, HP's HP-UX, IBM's AIX and SCO's UnixWare. A large number of IBM mainframes also run UNIX applications, because the UNIX interfaces were added to MVS and OS/390, which have obtained UNIX branding. Linux, another variant of UNIX, is also gaining enormous popularity.

WINDOWS

Windows is a personal computer operating system from Microsoft that, together with some commonly used business applications such as Microsoft Word and Excel, has become a de facto "standard" for individual users in most corporations as well as in most homes. Windows contains built-in networking, which allows users to share files and applications with each other if their PC's are connected to a network. In large enterprises, Windows clients are often connected to a network of UNIX and NetWare servers. The server versions of Windows NT and 2000 are gaining market share, providing a Windows-only solution for both the client and server. Windows is supported by Microsoft, the largest software company in the world, as well as the Windows industry at large, which includes tens of thousands of software developers.

This networking support is the reason why Windows became successful in the first place. However, Windows 95, 98, ME, NT, 2000 and XP are complicated operating environments. Certain combinations of hardware and software running together can cause problems, and troubleshooting can be daunting. Each new version of Windows has interface changes that constantly confuse users and keep support people busy, and Installing Windows applications is problematic too. Microsoft has worked hard to make Windows 2000 and Windows XP more resilient to installation of problems and crashes in general.

MACINTOSH

The Macintosh (often called "the Mac"), introduced in 1984 by Apple Computer, was the first widely-sold personal computer with a graphical user interface (GUI). The Mac was designed to provide users with a natural, intuitively under-standable and, in general, "user-friendly" computer interface. This includes the mouse, the use of icons or small visual images to represent objects or actions, the point-and-click and click-and-drag actions, and a number of window operation ideas. Microsoft was successful in adapting user interface concepts first made popular by the Mac in its first Windows operating system. The primary disadvantage of the Mac is that there are fewer Mac applications on the market than for Windows. However, all the fundamental applications are available, and the Macintosh is a perfectly useful

machine for almost everybody. Data compatibility between Windows and Mac is an issue, although it is often overblown and readily solved.

The Macintosh has its own operating system, Mac OS which, in its latest version is called Mac OS X. Originally built on Motorola's 68000 series microprocessors, Mac versions today are powered by the PowerPC microprocessor, which was developed jointly by Apple, Motorola, and IBM. While Mac users represent only about 5% of the total numbers of personal computer users, Macs are highly popular and almost a cultural necessity among graphic designers and online visual artists and the companies they work for.

In this section we will discuss some services of the operating system used by its users. Users of operating system can be divided into two broad classes: command language users and system call users. Command language users are those who can interact with operating systems using the commands. On the other hand system call users invoke services of the operating system by means of run time system calls during the execution of programs.

Questions

1. What is an operating system?

2. Describe different types of operating system.

3. Summarise the characteristics of the following operating systems:

- Time-sharing
- Real time

4. Compare and contrast between distributed operating system and network operating system.

5.What are the components of an operating system? Discuss.

6.What is the concept of monolithic operating system.

7. What are the services provides by an operating system? Explain.

8. What is System call? .

9. What is Dos? How it is different from UNIX? Explain.

10. Discuss Windows operating system.