Unit1. Introduction (3)

1.1 What is operating system?

An **Operating System** (OS) is an interface between computer user and computer hardware. An operating system is 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.

An operating system is software that enables applications to interact with a computer's hardware. The software that contains the core components of the operating system is called the **kernel**.

The primary purposes of an **Operating System** are to enable applications (software) to interact with a computer's hardware and to manage a system's hardware and software resources.

Some popular Operating Systems include Linux Operating System, Windows Operating System, VMS, OS/400, AIX, z/OS, etc.

Following are some of important functions of an operating System.

- Memory Management
- Processor Management
- Device Management
- File Management
- Network Management
- Security
- Control over system performance
- Job accounting
- Error detecting aids
- Coordination between other software and users

1.2 Two Views Of Operating System

The operating system may be observed from the viewpoint of the user or the system. It is known as the user view and the system view. There are mainly two types of views of the operating system. These are as follows:

1. **User View**
2. **System View**

User View

The user view depends on the system interface that is used by the users. Some systems are designed for a single user to monopolize the resources to maximize the user's task. In these cases, the OS is designed primarily for ease of use, with little emphasis on quality and none on resource utilization.

The user viewpoint focuses on how the user interacts with the operating system through the usage of various application programs. In contrast, the system viewpoint focuses on how the hardware interacts with the operating system to complete various tasks.

1. Single User View Point

Most computer users use a monitor, keyboard, mouse, printer, and other accessories to operate their computer system. In some cases, the system is designed to maximize the output of a single user. As a result, more attention is laid on accessibility, and resource allocation is less important. These systems are much more designed for a single user experience and meet the needs of a single user, where the performance is not given focus as the multiple user systems.

2. Multiple User View Point

Another example of user views in which the importance of user experience and performance is given is when there is one mainframe computer and many users on their computers trying to interact with their kernels over the mainframe to each other. In such circumstances, memory allocation by the CPU must be done effectively to give a good user experience. The client-server architecture is another good example where many clients may interact through a remote server, and the same constraints of effective use of server resources may arise.

3. Handled User View Point

Moreover, the touchscreen era has given you the best handheld technology ever. Smartphones interact via wireless devices to perform numerous operations, but they're not as efficient as a computer interface, limiting their usefulness. However, their operating system is a great example of creating a device focused on the user's point of view.

4. Embedded System User View Point

Some systems, like embedded systems that lack a user point of view. The remote control used to turn **on** or **off** the tv is all part of an embedded system in which the electronic device communicates with another program where the user viewpoint is limited and allows the user to engage with the application.

System View

The OS may also be viewed as just a resource allocator. A computer system comprises various sources, such as hardware and software, which must be managed effectively. The operating system manages the resources, decides between competing demands, controls the program execution, etc. According to this point of view, the operating system's purpose is to maximize performance. The operating system is responsible for managing hardware resources and allocating them to programs and users to ensure maximum performance.

1. Resource Allocation

The hardware contains several resources like registers, caches, RAM, ROM, CPUs, I/O interaction, etc. These are all resources that the operating system needs when an application program demands them. Only the operating system can allocate resources, and it has used several tactics and strategies to maximize its processing and memory space. The operating system uses a variety of strategies to get the most out of the hardware resources, including paging, virtual memory, caching, and so on. These are very important in the case of various user viewpoints because inefficient resource allocation may affect the user viewpoint, causing the user system to lag or hang, reducing the user experience.

2. Control Program

The control program controls how input and output devices (hardware) interact with the operating system. The user may request an action that can only be done with I/O devices; in this case, the operating system must also have proper communication, control, detect, and handle such devices.

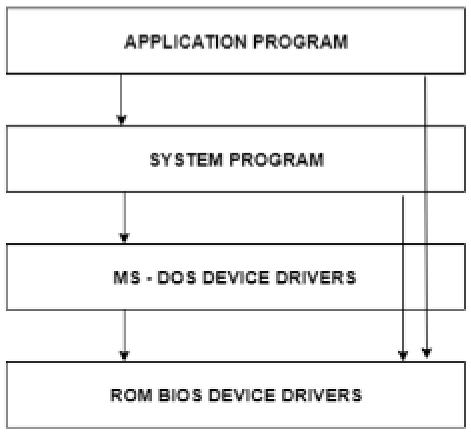
1.3 Operating System Structures

An operating system is a construct that allows the user application programs to interact with the system hardware. Since the operating system is such a complex structure, it should be created with utmost care so it can be used and modified easily. An easy way to do this is to create the operating system in parts. Each of these parts should be well defined with clear inputs, outputs and functions.

Operating System Structures:

Simple structure:

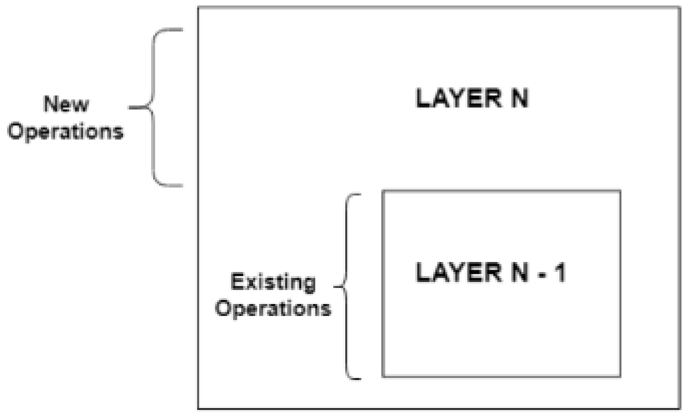
- There are many operating systems that have a rather simple structure. These started as small systems and rapidly expanded much further than their scope. A common example of this is MS-DOS. It was designed simply for a niche amount for people. There was no indication that it would become so popular.
- An image to illustrate the structure of MS-DOS is as follows –



MS-DOS STRUCTURE

- It is better that operating systems have a modular structure, unlike MS-DOS. That would lead to greater control over the computer system and its various applications. The modular structure would also allow the programmers to hide information as required and implement internal routines as they see fit without changing the outer specifications.

  Layered Structure
- One way to achieve modularity in the operating system is the layered approach. In this, the bottom layer is the hardware and the topmost layer is the user interface.
- An image demonstrating the layered approach is as follows –



Layered Structure of Operating System

- As seen from the image, each upper layer is built on the bottom layer. All the layers hide some structures, operations etc from their upper layers.
- One problem with the layered structure is that each layer needs to be carefully defined. This is necessary because the upper layers can only use the functionalities of the layers below them.

1.4 Types of Operating Systems

An Operating System performs all the basic tasks like managing files, processes, and memory. Thus operating system acts as the manager of all the resources, i.e. **resource manager**. Thus, the operating system becomes an interface between user and machine.
Types of Operating Systems: Some widely used operating systems are as follows-
1. Batch Operating System –
This type of operating system does not interact with the computer directly. There is an operator which takes similar jobs having the same requirement and group them into batches. It is the responsibility of the operator to sort jobs with similar needs.

2. Time-Sharing Operating Systems –
Each task is given some time to execute so that all the tasks work smoothly. Each user gets the time of CPU as they use a single system. These systems are also known as Multitasking Systems. The task can be from a single user or different users also. The time that each task gets to execute is called quantum. After this time interval is over OS switches over to the next task.

3. Distributed Operating System –
These types of the operating system is a recent advancement in the world of computer technology and are being widely accepted all over the world and, that too, with a great pace. Various autonomous interconnected computers communicate with each other using a shared communication network. Independent systems possess their own memory unit and CPU. These are referred to as **loosely coupled systems** or distributed systems. These system's processors differ in size and function. The major benefit of working with these types of the operating system is that it is always possible that one user can access the files or software which are not actually present on his system but some other system connected within this network i.e., remote access is enabled within the devices connected in that network.

4. Network Operating System –
These systems run on a server and provide the capability to manage data, users, groups, security, applications, and other networking functions. These types of operating systems allow shared access of files, printers, security, applications, and other networking functions over a small private network. One more important aspect of Network Operating Systems is that all the users are well aware of the underlying configuration, of all other users within the network, their individual connections, etc. and that's why these computers are popularly known as **tightly coupled systems**.

5. Real-Time Operating System –
These types of OSs serve real-time systems. The time interval required to process and respond to inputs is very small. This time interval is called **response time**.
Real-time systems are used when there are time requirements that are very strict like missile systems, air traffic control systems, robots, etc.

1.5 system call

System calls serve as the interface between an operating system and a process. System calls can typically be found as assembly language instructions. They are also covered in the manuals that the programmers working at the assembly level use. When a process in user mode needs access to a resource, system calls are typically generated. The resource is then requested from the kernel via a system call.

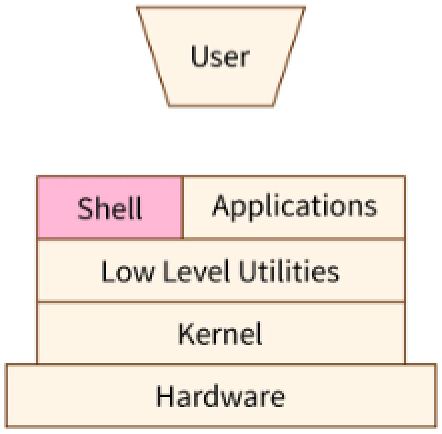
System calls are always carried out in the operating system's kernel mode. We will talk about what is system call in OS? and how a system call works in this article. Its operation, types, and how to pass parameters to such calls will be covered.

1.6 shell

What is a Shell in an Operating System?

Shell is a computer application software that the services of an operating system to an external user or another program. Depending on the computer's particular operation and role, its shell may have either a command line interface (C.L.I.) or a graphical user interface (G.U.I.). As suggested by the name, the shell is the outermost layer in an operating system.

Shells in most operating systems are not direct interfaces to the kernel that lies beneath, even if the shell communicates with the program or user through peripheral devices directly attached to the computer system. A shell is precisely software that uses the kernel API in a manner the same as is used by other software programs. The shell also manages the interaction between the system and the user by taking user input, evaluating it, and eventually dealing with the output. Since it is an application program, in most operating systems the shell can easily be replaced with some other similar application.



1.7 Open Source Operating System

Linux is a community of open-source Unix like operating systems that are based on the Linux Kernal. It is a free and open-source operating system and the source code can be modified and distributed to anyone commercially or non-commercially under the GNU(General Public License). Let us consider the following points relating to the Unix Operating System −

- Multitasking features.
- Multiuser computer operating systems.
- Provides stimulus to the open source movement.
- Comparative complex functionality.
- It does not give notice or warn about the consequences of a user's action