Unit 3 Operating System

Introduction to Operating System

An operating system (OS) is the program that, after being initially loaded into the computer by a boot program, manages all the other application programs in a computer. The application programs use the OS by requesting services through a defined application program interface (API). In addition, users can interact directly with the operating system through a user interface (UI), such as a command-line interface (CLI) or a graphical user interface (GUI).

An operating system (OS) is a system software that acts as an interface between the system hardware and the user. Moreover, it handles all the interactions between the software and the hardware. All the working of a computer system depends on the OS at the base level. Further, it performs all the functions like handling memory, processes, the interaction between hardware and software, etc.

The main goal of the Operating System is to make the computer environment more convenient to use and the Secondary goal is to use the resources most efficiently.

How Operating System assists software

An operating system brings powerful benefits to computer software and software development. Without an operating system, which is system software specifically designed to run the computer, every application would need to include its own UI, as well as the comprehensive code needed to handle all low-level functionality of the underlying computer's system software, such as disk storage and network interfaces. Considering the vast array of underlying hardware available, and the number of software routines that must be run at the system software level to support computer functions, this would bloat the size of every application and make software development impractical.

Instead, the OS handles many system-level software tasks, including sending a network packet or displaying text on a standard output device, such as a display. The system software serves as an intermediary between the applications and the underlying computer and hardware functions. The OS provides a consistent and repeatable way for applications to interact with the hardware and other system-level functions without the applications needing to know any details about them.

If each application accesses the same resources and services in the same way, the OS -- and the underlying systems software with which it engages -- can service almost any number of applications. This vastly reduces the amount of time and coding required to develop and debug an application, while ensuring that users can control, configure and manage system software and hardware through a common and well-understood OS interface.

How OS Works

How does an operating system work? Once installed, an operating system relies on a large library of device drivers to tailor its services to the specific hardware environment. For example, every application could make a common call to a storage device. The OS receives that call and uses the corresponding driver to translate the call into actions, or commands, needed for the underlying hardware on that specific computer. The operating system provides a comprehensive platform that identifies, configures and manages the following:

- A range of hardware, including processors.
- Memory devices and computer memory management.
- Chipsets.
- Storage.
- Networking.
- Port communication interfaces, such as Video Graphics Array, High-Definition Multimedia Interface and Universal Serial Bus (USB).
- Subsystem interfaces, such as Peripheral Component Interconnect Express.

Function of Operating System

All major computer platforms -- both hardware and software -- typically require and include an OS that must be developed with different features to meet the specific needs of various device form factors.

From an end user's perspective, an operating system provides the following three essential capabilities:

- Offers a UI through a CLI or GUI.
- Launches applications and manages application execution.
- Identifies and determines the allocation of system hardware resources such as printers and backup disk drives to those applications, typically through a standardized API.

Operating System is used as a communication channel between the Computer hardware and the user. It works as an intermediate between System Hardware and End-User. Operating System handles the following responsibilities:

- It controls all the computer resources.
- It provides valuable services to user programs.
- It coordinates the execution of user programs.
- It provides resources for user programs.
- It provides an interface (virtual machine) to the user.
- It hides the complexity of software.
- It supports multiple execution modes.
- It monitors the execution of user programs to prevent errors.

Functions of OS

1. Memory Management

The OS manages the main or primary memory. Whatever program is executed, it has to be present in the main memory. Main memory is a quick storage area that may be accessed directly by the CPU. When the program is completed, the memory region is released and can be used by other programs. Therefore, there can be more than one program present at a time. Hence, it is required to manage the memory.

The operating system:

- Manages the allocation and deallocation of memory to various processes and ensures that the other process does not consume the memory allocated to one process.
- Keeps a record of which part of primary memory is used by whom and how much.
- In multiprogramming, the OS decides the order in which processes are granted memory access, and for how long.

2. Processor Management/Scheduling

Every software that runs on a computer, whether in the background or in the frontend, is a process. The operating system determines the status of the processor and processes, selects a job and its processor, allocates the processor to the process, and de-allocates the processor after the process is completed.

When more than one process runs on the system, the OS decides how and when a process will use the CPU. Hence, the name is also CPU Scheduling. The OS:

- Allocates and deallocates processor to the processes.
- Keeps record of CPU status.

Algorithms used for CPU scheduling

- First Come First Serve (FCFS)
- Shortest Job First (SJF)
- Round-Robin Scheduling
- Priority-based scheduling etc.

Purpose of CPU scheduling The purpose of CPU scheduling is as follows:

- The proper utilization of the CPU is necessary. Therefore, the OS makes sure that the CPU should be as busy as possible.
- Since every device should get a chance to use the processor. Hence, the OS makes sure that the devices get fair processor time.
- Increasing the efficiency of the system.

3. Device Management

An operating system identifies, configures and provides applications with common access to underlying input/output devices such as printers, keyboards and other computer hardware devices. As the OS recognizes and identifies hardware, it installs corresponding device drivers and interfaces that enable the OS and applications running on the OS to use the devices.

An operating system identifies the correct printer and installs the appropriate printer drivers so that an application needs to only make calls to the printer without having to use codes or commands that are specific to that printer. An operating system regulates device connection using device drivers. The processes may require devices for their use. This management is done by the OS. The OS:

- Allocates and deallocates devices to different processes.
- Keeps records of the connected devices to the system.
- Decides which process can use which device for how much time.

4. File Management

A file system is organized into directories for efficient or easy navigation and usage. These directories may contain other directories and other files. The operating system specifies which process receives the file and for how long. An Operating System carries out the following file management activities:

- Keeps records of the status and locations of files.
- Allocates and deallocates resources.

- Decides who gets the resources.
- An OS keeps track of information regarding the creation, deletion, transfer, copy, and storage of files in an organized way.
- It also maintains the integrity of the data stored in these files, including the file directory structure, by protecting against unauthorized access

5. I/O Management

I/O management is the important function of operating system refers to how the OS handles input and output operations between the computer and external devices, such as keyboards, mice, printers, hard drives, and monitors.

6. User Interface

The user interacts with the computer system through the operating system. Hence OS acts as an interface between the user and the computer hardware. This user interface is offered through a set of commands or a graphical user interface (GUI). Through this interface, the user makes interacts with the applications and the machine hardware.

7. Error-Detection

The operating system constantly monitors the system to detect errors and avoid malfunctioning computer systems. From time to time, the operating system checks the system for any external threat or malicious software activity. It also checks the hardware for any type of damage. This process displays several alerts to the user so that the appropriate action can be taken against any damage caused to the system.

The functions of an operating system are essential for making a computer work smoothly. It manages the hardware and software, allowing different programs to run, handling memory, files, and devices like printers or keyboards, and ensuring everything is secure. Without these core functions, we wouldn’t be able to use computers efficiently. The OS acts like the brain of the computer, coordinating all activities and making sure everything runs properly for users and applications.

8. Networking

Transparent to the user, the OS automatically connects a workstation to networkwide resources by deciphering network protocols and connecting to networks. This enables a single user to access network printers and servers that are hosted on the network.

9. Security

Using policies defined and authorized by IT, the OS enforces security access controls and encryption for users, applications and data.

10. Backup and Recovery

Data can be backed up periodically throughout the day, nightly, weekly or at whatever frequency that's defined. The OS can perform these backups automatically, without user or IT assistance. If a data outage or system failure occurs, data can easily be recovered from the latest backup.

11. Virtualization

Most OSes enable users to define multiple operating systems that run independently of each other by using software-created partitions that separate each OS from the others on a single physical workstation. This enables the user to run multiple applications, each with their own dedicated OS, simultaneously on a single workstation to optimize performance.

Types of Operating System

Although the fundamental roles of an operating system are ubiquitous, there are countless operating systems that serve a wide range of hardware and user needs, including the following:

1. Batch OS

This type of operating system were non-interactive. A user can not interact with OS. There is an operator(Person) which takes similar jobs having the same requirements and groups them into batches. It is the responsibility of the operator to sort jobs with similar needs. User prepares their jobs on **Punched Card** & submits to an operator. Batch Operating System is designed to manage and execute a large number of jobs efficiently by processing them in groups. Only after Batch1 completes, the Batch2 is inserted. The output is also in the form of punched card. The loading & unloading of punched card takes time, during which CPU remains idle. Hence, CPU utilization is low.

What is Job? Program + Input Data + Control Instruction

2. Multiprogramming OS

Multiple programs or processes will be in the main memory. It can be simply illustrated as more than one program is present in the main memory and any one of them can be kept in execution. This is basically used for better utilization of resources. The drawback is that if process P1 is in execution and it takes 1 hour but process P2 which is in queue takes 1 minute, then we can't forcefully remove P1 and allocate CPU to P2. The process P1 can leave CPU only if it needs I/O or Network operations

3. Multitasking OS

It is an extension of Multi-programming OS. It is simply a multiprogramming Operating System with having facility of a Round-Robin Scheduling Algorithm. It can run multiple programs simultaneously. Process P2 can be allocated CPU after forcefully removing the process P1 on the basis of time slot allocated to each processes.

4. Multi-processing OS

Multi-Processing Operating System is a type of Operating System in which more than one CPU is used for the execution of resources. It betters the throughput of the System.

5. Real time OS

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. When a computing device must interact with the real world within constant and repeatable time constraints, the device manufacturer might opt to use a real-time operating system (RTOS). For example, an industrial control system might direct the operations of a sprawling factory or power plant. Such a facility produces signals from myriad sensors and sends signals to operate valves, actuators, motors and countless other devices. In these situations, the industrial control system must respond quickly and predictably to changing real-world conditions -- otherwise, disaster could result. An RTOS must function without buffering, processing latencies and other delays, which are perfectly acceptable in other types of operating systems. Examples of RTOSes include FreeRTOS and Wind River VxWorks.

Types of Real-Time Operating Systems

- Hard Real-Time Systems: Hard Real-Time OSs are meant for applications where time constraints are very strict and even the shortest possible delay is not acceptable. These systems are built for saving life like automatic parachutes or airbags which are required to be readily available in case of an accident. Virtual memory is rarely found in these systems.
- Soft Real-Time Systems: These OSs are for applications where time-constraint is less strict.

6. General-purpose operating systems

A general-purpose OS can run a multitude of applications on a broad selection of hardware, enabling a user to run one or more applications or tasks simultaneously. A general-purpose OS can be installed on many different desktop and laptop models and run applications from accounting systems to databases to web browsers to games. General-purpose OSes typically focus on process, or thread, and hardware management to ensure that applications can reliably share the wide range of computing hardware present.

Common desktop operating systems include the following:

- **Windows**, Microsoft's flagship operating system, is the de facto standard for home and business computers. Introduced in 1985, the Microsoft Windows GUI-based OS has been released in many versions since then. The user-friendly Windows 95 was largely responsible for the rapid development of personal computing.
- **Mac OS** is the operating system for Apple's line of PCs and workstations.
- **Unix** is a multiuser operating system designed for flexibility and adaptability. Originally developed in the 1970s, Unix was one of the first OSes to be written in the C language.
- **Linux** is a Unix-like operating system from the open source community that was designed to provide PC users a free or low-cost OS alternative. Linux has a reputation as an efficient and fast-performing OS.

7. Mobile operating systems

Mobile operating computer systems are designed for mobile computing and communication-centric devices, such as smartphones and tablets. Mobile devices typically offer limited computing resources compared to traditional PCs, and the OS must be scaled back in size and complexity to minimize its own resource use, while ensuring adequate resources for one or more applications running on the device. Mobile operating systems tend to emphasize efficient performance, user responsiveness and close attention to data handling tasks, such as supporting media streaming. Apple iOS and Android are examples of mobile operating systems.

8. Embedded operating systems

Not all computing devices are general-purpose. A huge assortment of dedicated devices -- including home digital assistants, automated teller machines, airplane systems, retail point of sale terminals and internet of things devices -- use more customized and "slimmed down" operating systems. The principal difference between embedded OSes and general-purpose OSes is that the devices the embedded OSes are embedded on do only one major thing, so the OS is highly stripped down and dedicated to both performance and resilience. The embedded OS should run quickly, not crash, and handle all errors gracefully to continue operating in all circumstances. In most cases, the OS is provided on a chip that's incorporated into the actual device. A medical device used in a patient's life support equipment, for example, employs an embedded OS that must run reliably to keep the patient alive. Embedded Linux is one example of an embedded OS.

9. Network operating systems

A network operating system (NOS) is another specialized OS intended to facilitate communication between devices operating on a local area network. A NOS provides the communication stack needed to understand network protocols to create, exchange and decompose network packets. Today, the concept of a specialized NOS is largely obsolete because other OSes handle network communication. Windows 10 and Windows Server 2019, for example, include comprehensive networking capabilities. The concept of a NOS is still used for some networking devices, such as routers, switches and firewalls, and manufacturers employ proprietary NOSes, including Cisco Internetwork Operating System and the open source NOS RouterOS from MikroTik.

10. Distributed operating systems

A distributed OS installed on a network can provide service to a multitude of workstations, especially thin-client computers that have little or no resident applications or data on them. Multiple users access and share applications and resources that are hosted on these larger, distributed network servers, and the OSes on these servers manage the access requests and resource consumption from multiple user workstations. Examples of distributed OSes are Microsoft Windows Server and various distributions of open source Linux for servers.

11. Cluster operating systems

Cluster operating systems are OSes designed to run a cluster of computers that work together on a single system. Artificial intelligence processing is a prime example of cluster computing, as it requires the rapid, simultaneous, parallel processing of data. High-performance computing systems in which thousands of transactions must be processed simultaneously and in real time -- such as brokerage systems -- are another use case for cluster computing. Rocks Cluster Distribution and open source Open MPI are two examples of cluster computing OSes.

The differences between operating system types aren't absolute, and some OSes can share characteristics of others. For example, general-purpose OSes routinely include the networking capabilities found in a traditional NOS. Similarly, an embedded operating system commonly includes attributes of an RTOS, while a mobile operating system can typically run numerous apps simultaneously like other general-purpose OSes.

Open Source Operating System