Operating System Principles

❖ Operating System Components –

➢ Process management –

- A process can be thought of as a program in execution. Process is the unit of work in most system.
- A process will need certain resources such as CPU time, memory, files and I/O devices to accomplish its task. These resources are allocated to the process either when it is created or while it is executing.
- A process is the unit of work in the most systems. Such a system consists of a collection of processes. OS processes execute system code and user processes execute user code.
- All these processes may execute concurrently.
- Most modern OS now support processes that have multiple threads.
- The OS is responsible for the following activities in connection with process and thread management. The creation and deletion of both user and system processes, the scheduling of processes and the provision of mechanism for synchronization, communication and deadlock handling for processes.

- Now a days computers run multitasking OS. The systems run multiple programs at a time.
- Because of multitasking environment different processes or different jobs are created for multiple programs.
- When one program or job runs and if it does not require CPU then it may be assigned to other devices like I/O devices.
- Till that time CPU should not be free so may be another process is assigned to CPU.
- So, for this kind of processing the process management is required.
- Process management includes creating, running, terminating and assigning different processes to different devices.
- Even process scheduling is a part of process management where the sequence and priorities of the processes are defined.
- The process manager is one of the main parts of the four major parts of the OS. It implements the process abstraction. Process abstraction hides the complexity of the processes from the user.

➢ Main memory management –

- Main memory is central to the operation of a modern computer system.
- Main memory is a large array of words or bytes ranging in size from hundreds of thousands to billions.
- Main memory is a repository of quickly accessible data shared by the CPU and I/O devices.
- The main memory is generally the only large storage device that the CPU is able to address and access directly.

- For a program to be executed, it must be mapped to absolute addresses and loaded into memory.
- As the program executes, it accesses program installation and data from memory by generating their absolute addresses.
- Eventually, the program terminates its memory space is declared available, and the next program can be loaded and executed.
- To improve both the utilization of the CPU and the speed of the computer's response to its users, we must keep several programs in memory.
- Many different memory management schemes are available and the effectiveness of the different algorithms depends on the particular situation.
- Selection of a memory-management scheme for a specific system depends on many factors especially in the hardware designs of the system. Each algorithm requires its own hardware system post.

The OS is responsible for the following activities in connection with memory management –

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

➢ File management –

- File management is one of the most visible components of an OS. Computers can store information on several different types of physical media.
- Each media is controlled by a device, such as a disk drive or tape drive, that also has unique characteristics.
- These properties include access speed, capacity, data transfer rate and access method.

- 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.
- A file consists of a sequence of bits, bytes, lines or records whose meaning are defined by their creators.
- The OS implements the abstract concept of a file by managing man storage media, such as disk and tapes, and the device that control them.
- Also, files are normally organized into directories to ease their use.
- Finally, when multiple users have access to files, we may want to control by when and in what ways (e.g., Read, write, append) files may be accessed.

The OS is responsible for the following activities in connection with file management –

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

➢ I/O system management –

- One of the purposes of an OS is to hide the peculiarities of specific hardware devices from the user.
- e.g., in UNIX, the peculiarities of I/O devices are hidden from the bulk of the OS itself by the I/O subsystem.

The I/O subsystem consists of –

- A memory management component that includes buffering occulting and spoofing.
- A general device drive information.
- Drives for specific hardware devices.

- Only the device driver known the peculiarities of the specific device to which it is assigned.

➢ Secondary storage management –

- The main purpose of a computer system is to execute program.
- These programs, with the data they access, must be in main memory or primary storage during execution.
- Because main memory is too small to accommodate all data and programs, and because the data that it holds are lost when power is lost, the computer system must provide secondary storage to back up main memory.
- Most modern computer systems use disks as the principal on-line storage medium, for both programs and data.
- Most programs-including compilers, assemblers, sort routines, editors and formalities are stored on a 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.

The OS is responsible for following activities in connection with disk management –

- Free space management
- Storage allocation
- Disk scheduling

- Because secondary storage is used frequently, it must be used efficiently.
- The entire speed of operation of a computer may huge on the speeds of the disk subsystem and of the algorithms that manipulate that subsystem.

❖ Operating System Services –

The OS services are provided for the convenience of the programming to make the programming task faster.

1. Program execution –

   The system must be able to load a program into memory and to run that program. The program must be able to end its execution, either normally or abnormally (indicating error).

2. I/O operations –

   A running program may require I/O. This I/O may involve a file or an I/O device. For specific devices, special functions may be desired (such as to rewind a tape drive or to blank a CRT screen). For efficiency and protection, users usually cannot control I/O devices directly. Therefore, the OS must provide a means to do I/O.

3. File system manipulation –

   The file system is of particular interest programs need to read and write files. Programs also need to create and delete files by name.

4. Communications –

   In many circumstances, one process needs to exchange information with other process. Such communication can occur in two major ways. The first takes between processes that are executing on the same computer, the second takes place between processes that are executing on different computer systems that are tied together by a computer network. Communication may be implemented via shared memory by the technique of message passing in which packets of information are moved between processes by the OS.

5. Error detection –

   The OS constantly needs to be aware of possible errors. Errors may occur in the CPU and memory hardware (such as a memory error or a power failure), in I/O devices and in the user program. For each type of error, the OS should take the appropriate action to ensure correct and consistent computing.

6. Resource allocation –

   When multiple users are logged on the system or multiple jobs are running at the same time, resources must be allocated to each of them. Many different types of resources are managed by the OS.

7. Accounting –

   We want to keep track of which users use how many and which kinds of computer resources. This record keeping may be used for accounting (so that users can be filled) or simply for accumulating usage statistics. Usage statistics may be a valuable tool for researchers who wish to reconfigure the system to improve computing services.

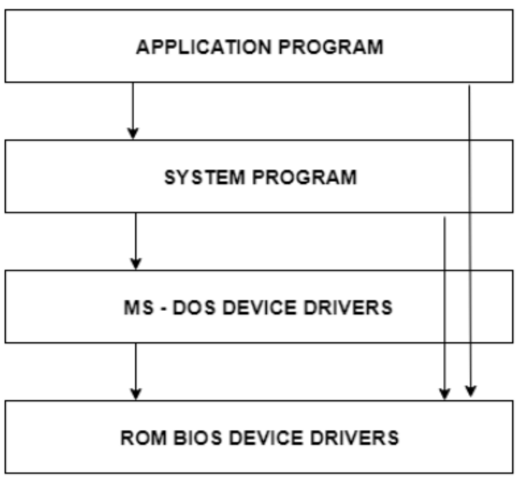
8. Protection –

   The owners of information stored in a multiuser computer system may want to control use of that information. When several disjointed processes execute concurrently, it should not be possible for one process to interface with the others, or with the OS itself. Protection involves ensuring that all access to system resources is controlled. Security of the system from outsides is also important. Such security starts with each user having to authenticate himself to the system, usually by means of a password, to be allowed access to the resources.

❖ Operating System Structure -

➢ 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 −

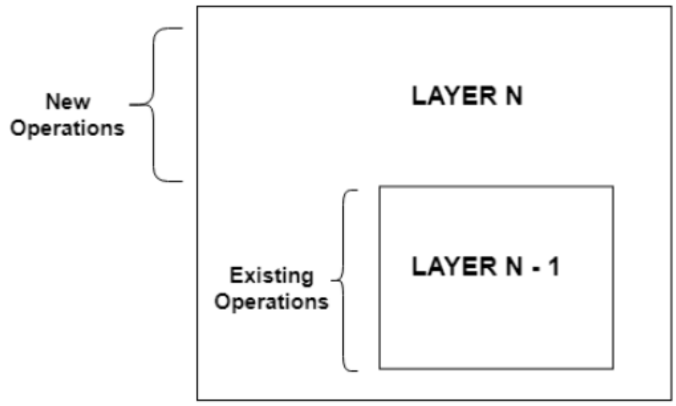


MS-DOS STRUCTURE

- 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 −



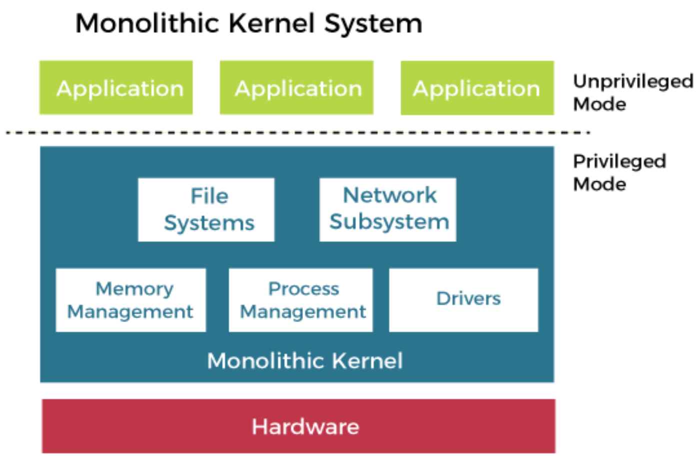
Layered Structure of Operating System

- 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.

➢ Monolithic Structure –

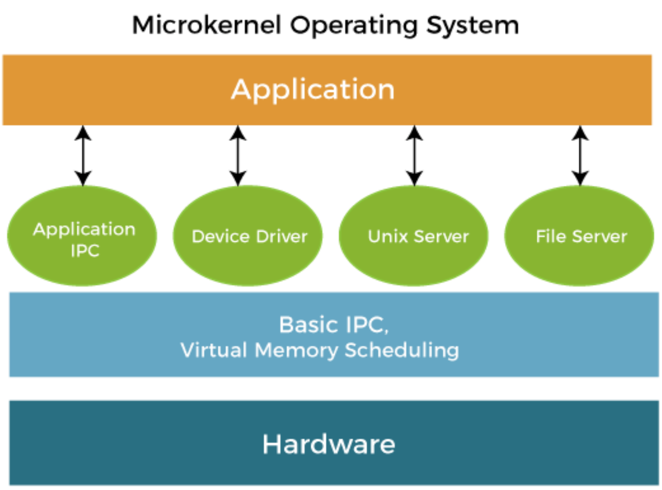
- The monolithic operating system is a very basic operating system in which file management, memory management, device management, and process management are directly controlled within the kernel. The kernel can access all the resources present in the system. In monolithic systems, each component of the operating system is contained within the kernel. Operating systems that use monolithic architecture were first time used in the 1970s.
- The monolithic operating system is also known as the monolithic kernel. This is an old operating system used to perform small tasks like batch processing and time-sharing tasks in banks. The monolithic kernel acts as a virtual machine that controls all hardware parts.
- It is different from a microkernel, which has limited tasks. A microkernel is divided into two parts, *kernel space*, and *user space*. Both parts communicate with each other through IPC (Inter-process

communication). Microkernel's advantage is that if one server fails, then the other server takes control of it.



➢ Microkernel Structure –

- The microkernel is one of the kernel's classifications. Being a kernel, it handles all system resources. On the other hand, the user and kernel services in a microkernel are implemented in distinct address spaces. User services are kept in user address space, while kernel services are kept in kernel address space. It aids to reduce the kernel and OS's size.
- It provides a minimal amount of process and memory management services. The interaction between the client application and services running in user address space is established via message passing that helps to reduce the speed of microkernel execution. The OS is unaffected because kernel and user services are isolated, so if any of the user services fails, the kernel service is unaffected. It is extendable because new services are added to the user address space, hence requiring no changes in kernel space. It's also lightweight, secure, and reliable.
- Microkernels and their user environments are typically used in C++ or C languages with a little assembly. On the other hand, other implementation programming languages may be possible with some high-level code.



- System Calls –

Concept –

- System calls provide the if between a process and the OS. These calls are generally available as assembly language installations and they are usually listed in the various manuals used by the assembly language programs.

- A system call is a way for programs to interact with the OS.
- A computer program makes a system call when it makes a request to the OS kernel.
- System call provides the services of the OS to the user programs via application program interface (API).
- System calls are the only entry points into the kernel system.
- All programs needing resources must use system calls.

System calls can be grouped roughly into fine major categories –

Process control, file management, device management, information maintenance and communication.

- Process control –

- End, abort
- Load, execute
- Create process, terminate process
- Get process attributes, set process attributes
- Wait for time
- Wait event, single event
- Allocate and free memory

- File management –

- Create file, delete file
- Open, close
- Read, write, reposition
- Get file attributes, set file attributes

- Device management –

- Request device, release device
- Read, write, reposition
- Get device attributes, set device attributes
- Logically attach or detach devices

- Information maintenance –

- 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

- Communication –

- Create, delete communication connection
- Send, receive messages
- Transfer status information
- Attach or detach remote devices

- System boot –

- The procedure of starting a computer by loading the kernel is known as booting the system.
- Hence it needs a special program, stored in ROM to do this job known as the bootstrap loader. E.g., BIOS
- Typically, the BIOS will allow the user to configure a boot order.
- If the boot order is set to: CD Drive, Hard Disk Drive, Network then the BIOS will try to boot from the CD drive first and if that fails then it will try to boot from the hard disk drive and if that fails then it will try to boot from the network, and if that fails then it won't boot at all.
- Booting is a startup sequence that starts the OS of a computer when it is turned on.
- A boot sequence is the initial set of operation that the computer performs when it is switched on.

- Boot loader –

- Bootloader is a piece of code that runs before any OS is running.
- Bootloader are used to boot other OS, usually each OS has a set of bootloaders specific for it.
- Boot loaders usually contain several ways to boot the OS kernel and also contain commands for debugging and for modifying the kernel environment.
- Bootloader is program written to load a more complex kernel.

The bootloader ultimately has to:

- Bring the kernel into memory.
- Prove the kernel with the information it needs to work correctly.
- Switch to an environment that the kernel will like.
- Transfer control to the kernel.