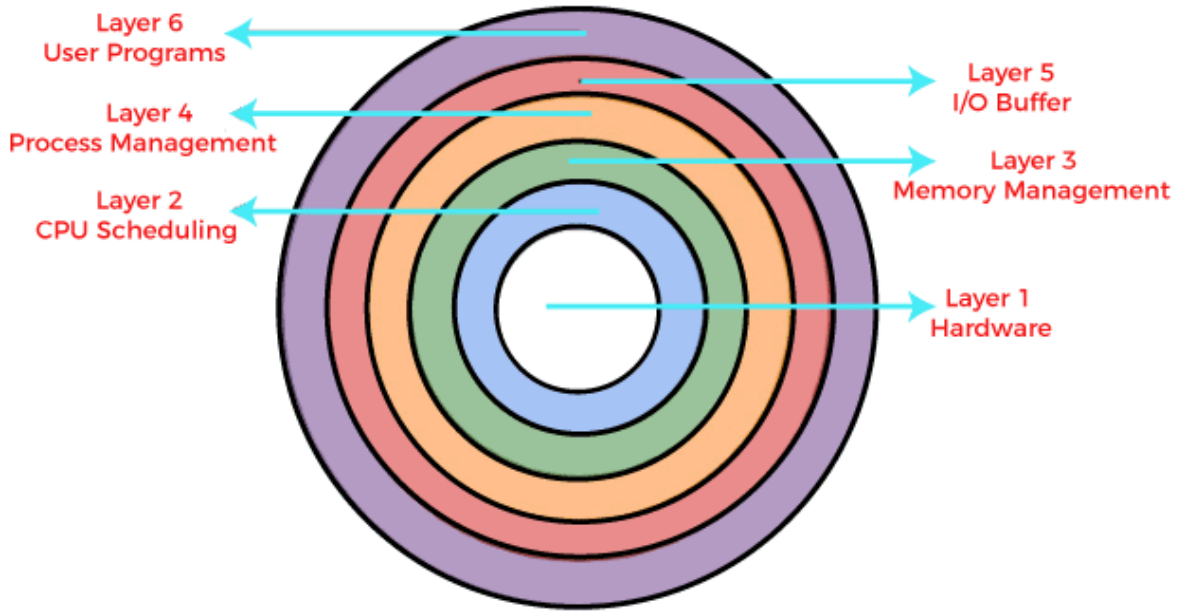
LAYERED STRUCTURE OF OPERATING SYSTEM

The operating system can be implemented with the help of various structures. The structure of the OS depends mainly on how the various common components of the operating system are interconnected and melded into the kernel. Depending on this, we have to follow the structures of the operating system.

The layered structure approach breaks up the operating system into different layers and retains much more control on the system. The bottom layer (layer 0) is the hardware, and the topmost layer (layer N) is the user interface. These layers are so designed that each layer uses the functions of the lower-level layers only. It simplifies the debugging process as if lower-level layers are debugged, and an error occurs during debugging. The error must be on that layer only as the lower-level layers have already been debugged



WHY LAYERING IN OPERATING SYSTEM

Layering provides a distinct advantage in an operating system. All the layers can be defined separately and interact with each other as required. Also, it is easier to create, maintain and update the system if it is done in the form of layers. Change in one layer specification does not affect the rest of the layers. Each of the layers in the operating system can only interact with the above and below layers. The lowest layer handles the hardware, and the uppermost layer deals with the user applications.

ARCHITECTURE OF LAYERED STRUCTURE

This type of operating system was created as an improvement over the early monolithic systems. The operating system is split into various layers in the layered operating system,

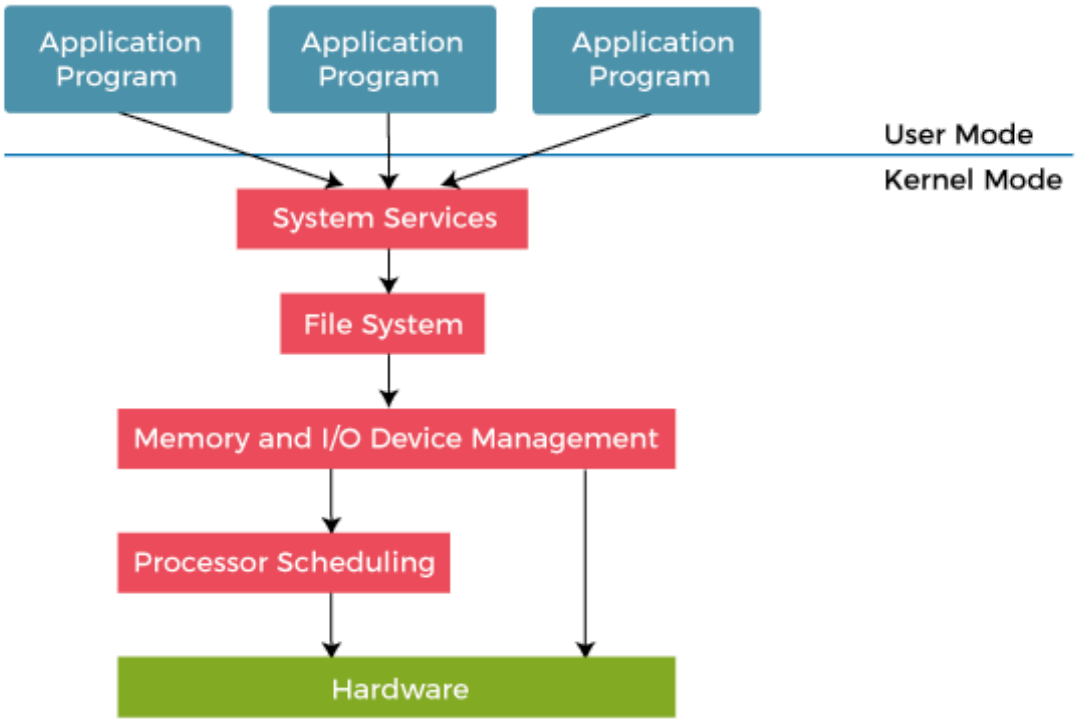
and each of the layers has different functionalities. There are some rules in the implementation of the layers as follows.

A particular layer can access all the layers present below it, but it cannot access them. That is, layer n-1 can access all the layers from n-2 to 0, but it cannot access the nth

Layer 0 deals with allocating the processes, switching between processes when interruptions occur or the timer expires. It also deals with the basic multiprogramming of the CPU.

Thus if the user layer wants to interact with the hardware layer, the response will be traveled through all the layers from n-1 to 1. Each layer must be designed and implemented such that it will need only the services provided by the layers below it.

There are six layers in the layered operating system. A diagram demonstrating these layers is as follows:



Hardware:

This layer interacts with the system hardware and coordinates with all the peripheral devices used, such as a printer, mouse, keyboard, scanner, etc. These types of hardware devices are managed in the hardware layer.

The hardware layer is the lowest and most authoritative layer in the layered operating system architecture. It is attached directly to the core of the system.

CPU Scheduling:

This layer deals with scheduling the processes for the CPU. Many scheduling queues are used to handle processes. When the processes enter the system, they are put into the job queue.

The processes that are ready to execute in the main memory are kept in the ready queue. This layer is responsible for managing how many processes will be allocated to the CPU and how many will stay out of the CPU.

Memory Management:

Memory management deals with memory and moving processes from disk to primary memory for execution and back again. This is handled by the third layer of the operating system. All memory management is associated with this layer. There are various types of memories in the computer like RAM, ROM.

If you consider RAM, then it is concerned with swapping in and swapping out of memory. When our computer runs, some processes move to the main memory (RAM) for execution, and when programs, such as calculator, exit, it is removed from the main memory.

Process Management:

This layer is responsible for managing the processes, i.e., assigning the processor to a process and deciding how many processes will stay in the waiting schedule. The priority of the processes is also managed in this layer. The different algorithms used for process scheduling are FCFS (first come, first served), SJF (shortest job first), priority scheduling, round-robin scheduling, etc.

I/O Buffer:

I/O devices are very important in computer systems. They provide users with the means of interacting with the system. This layer handles the buffers for the I/O devices and makes sure that they work correctly.

Suppose you are typing from the keyboard. There is a keyboard buffer attached with the keyboard, which stores data for a temporary time. Similarly, all input/output devices have some buffer attached to them. This is because the input/output devices have slow processing or storing speed. The computer uses buffers to maintain the good timing speed of the processor and input/output devices.

User Programs:

This is the highest layer in the layered operating system. This layer deals with the many user programs and applications that run in an operating system, such as word processors, games, browsers, etc. You can also call this an application layer because it is concerned with application programs.

Advantages of Layered Structure

1. Modularity: This design promotes modularity as each layer performs only the tasks it is scheduled to perform.
2. Easy debugging: As the layers are discrete so it is very easy to debug. Suppose an error occurs in the CPU scheduling layer. The developer can only search that

particular layer to debug, unlike the Monolithic system where all the services are present.

3. Easy update: A modification made in a particular layer will not affect the other layers.
4. No direct access to hardware: The hardware layer is the innermost layer present in the design. So a user can use the services of hardware but cannot directly modify or access it, unlike the Simple system in which the user had direct access to the hardware.
5. Abstraction: Every layer is concerned with its functions. So the functions and implementations of the other layers are abstract to it.

Disadvantages of Layered Structure

1. Complex and careful implementation: As a layer can access the services of the layers below it, so the arrangement of the layers must be done carefully. For example, the backing storage layer uses the services of the memory management layer. So it must be kept below the memory management layer. Thus, with great modularity comes complex implementation.
2. Slower in execution: If a layer wants to interact with another layer, it requests to travel through all the layers present between the two interacting layers. Thus, it increases response time, unlike the Monolithic system, which is faster than this. Thus, an increase in the number of layers may lead to a very inefficient design.
3. Functionality: It is not always possible to divide the functionalities. Many times, they are interrelated and can't be separated.
4. Communication: No communication between non-adjacent layers.