## Operating System Services

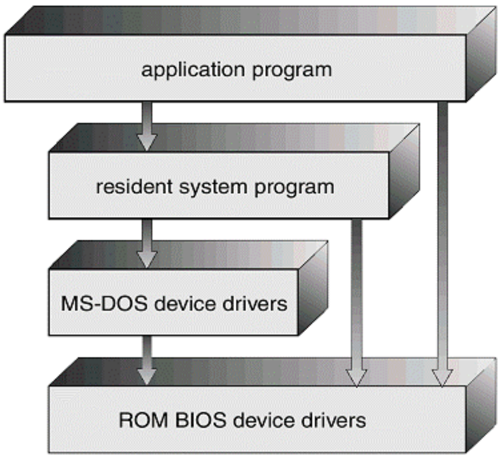
An operating system provides an environment for the execution of the program. It provides some services to the programs. The various services provided by an operating system are as follows:

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 terminate this execution either normally or abnormally.
2. **I/O Operation:** A running program may require I/O. This I/O may involve a file or a I/O device for specific device. Some special function can be desired. Therefore, the operating system must provide a means to do I/O.
3. **File System Manipulation:** The programs need to create and delete files by name and read and write files. Therefore the operating system must maintain each and every files correctly.
4. **Communication:** The communication is implemented via shared memory or by the technique of message passing in which packets of information are moved between the processes by the operating system.
5. **Error detection:** The operating system should take the appropriate actions for the occurrences of any type like arithmetic overflow, access to the illegal memory location and too large user CPU time.
6. **Research Allocation:** When multiple users are logged on to the system the resources must be allocated to each of them. For current distribution of the resource among the various processes the operating system uses the CPU scheduling run times which determine which process will be allocated with the resource.
7. **Accounting:** The operating system keep track of which users use how many and which kind of computer resources.
8. **Protection:** The operating system is responsible for both hardware as well as software protection. The operating system protects the information stored in a multiuser computer system.

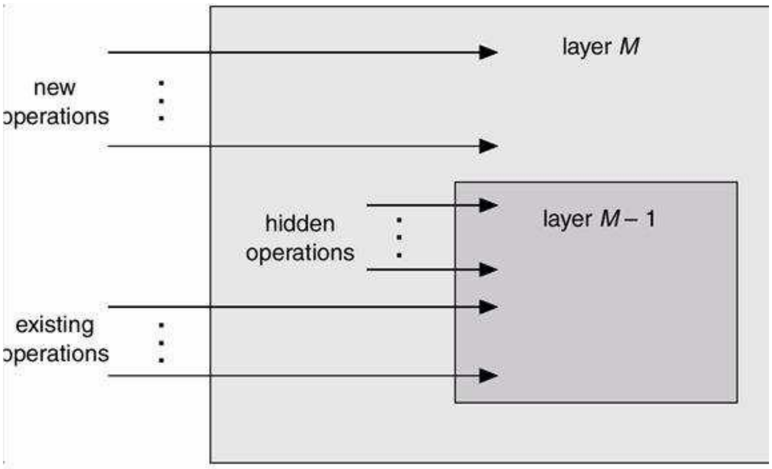
## System structure:

1. **Simple structure:** There are several commercial system that don’t have a well- defined structure such operating systems begins as small, simple & limited systems and then grow

beyond their original scope. MS-DOS is an example of such system. It was not divided into modules carefully. Another example of limited structuring is the UNIX operating system.




2. **Layered approach:** In the layered approach, the OS is broken into a number of layers (levels) each built on top of lower layers. The bottom layer (layer o ) is the hardware & top most layer (layer N) is the user interface. The main advantage of the layered approach is modularity.




- The layers are selected such that each users functions (or operations) & services of only lower layer.
- This approach simplifies debugging & system verification, i.e. the first layer can be debugged without concerning the rest of the system. Once the first layer is debugged, its correct functioning is assumed while the 2nd layer is debugged & so on.

- If an error is found during the debugging of a particular layer, the error must be on that layer because the layers below it are already debugged. Thus the design & implementation of the system are simplified when the system is broken down into layers.
- Each layer is implemented using only operations provided by lower layers. A layer doesn't need to know how these operations are implemented; it only needs to know what these operations do.
- The layer approach was first used in the operating system. It was defined in six layers.

| Layers | Functions |
|---|---|
| **5** | User Program |
| **4** | I/O Management |
| **3** | Operator Process Communication |
| **2** | Memory Management |
| **1** | CPU Scheduling |
| **0** | Hardware |

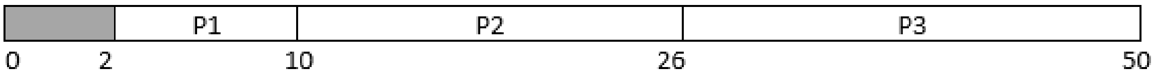
## CPU Scheduling Algorithm:

CPU Scheduling deals with the problem of deciding which of the processes in the ready queue is to be allocated first to the CPU. There are four types of CPU scheduling that exist.

**Q1. Three process P1, P2 and P3 arrive at time zero. The total time spent by the process in the system is 10ms, 20ms, and 30ms respectively. They spent first 20% of their execution time in doing I/O and the rest 80% in CPU processing. What is the percentage utilization of CPU using FCFS scheduling algorithm?**

### Solution:

| Process | Execution time | I/O time | CPU time |
|---|---|---|---|
| P1 | 10 | 2 | 8 |
| P2 | 20 | 4 | 16 |
| P3 | 30 | 6 | 24 |




Since all the processes will first do I/O and then CPU processing, process P1 spends first 0% i.e. 2ms in doing I/O. So CPU is not utilized for the first 2ms. Then P1 spends next 80% i.e. 8ms on processing. By the time P1 finishes with CPU processing P2 has finished its I/O 4ms) and then it gets turn for CPU processing which it does for 16ms and similarly P3 does processing for next 24ms.

Total time = 50ms

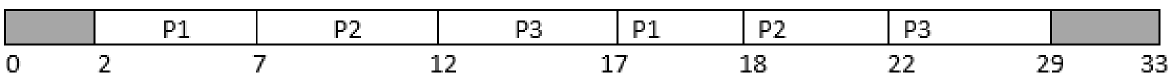
CPU utilized for 48ms (starting from 2 and ending at 50)

Therefore utilization = (48/50)*100 = 96%

**Q2. Three process p1, P2 and P3 arrive at time zero. Their total execution time is 10ms, 15ms, and 20ms respectively. They spent first 20% of their execution time in doing I/O, next 60% in CPU processing and the last 20% again doing I/O. For what percentage of time was the CPU free? Use Round robin algorithm with time quantum 5ms.**

## Solution:

| Process | Execution time | I/O burst | CPU burst | I/O Burst |
|---|---|---|---|---|
| P1 | 10 | 2 | 6 | 2 |
| P2 | 15 | 3 | 9 | 3 |
| P3 | 20 | 4 | 12 | 4 |




**Q 3 Assume you have the following jobs to execute with one processor, with the jobs arriving in the order listed here:**

| Process id | Burst time | Arrival time |
|---|---|---|
| 0 | **80** | **0** |
| 1 | **20** | **10** |
| 2 | **10** | **10** |
| 3 | **20** | **80** |
| 4 | **50** | **85** |

Suppose a system uses RR scheduling with a quantum of 15.

1. Create Gantt chart illustrating the execution of these processes using FCFS, SJF, priority, and Round Robin scheduling algorithms?
2. What is the turnaround time for process p3?
3. What is the average wait time for the processes?