EVOLUTION OF OPERATING SYSTEM – I

Evolution of Operating System:

Evolution of Operating Systems: User driven, operator driven, simple batch system, off – line batch system, directly coupled off – line system, multi- programmed spooling system, online timesharing system, multiprocessor systems, multi-computer/ distributed systems, Real time Operating Systems.

1. Serial processing
2. Batch processing
3. Multiprogramming
4. Multitasking or time sharing System

Multitasking or Time Sharing System:

- Multiprogramming didn't provide the user interaction with the computer system.
- Time sharing or Multitasking is a logical extension of Multiprogramming that provides user interaction.
- There are more than one user interacting the system at the same time
- The switching of CPU between two users is so fast that it gives the impression to user that he is only working on the system but actually it is shared among different users.
- CPU bound is divided into different time slots depending upon the number of users using the system.
- just as multiprogramming allows the processor to handle multiple batch jobs at a time, multiprogramming can also be used to handle multiple interactive jobs. In this latter case, the technique is referred to as time sharing, because processor time is shared among multiple users
- A multitasking system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time shared computer. Each user has at least one separate program in memory.
- Multitasking are more complex than multiprogramming and must provide a mechanism for jobs synchronization and communication and it may ensure that system does not go in deadlock.

Although batch processing is still in use but most of the system today available uses the concept of multitasking and Multiprogramming.

Serial Processing:

- Early computer from late 1940 to the mid 1950.
- The programmer interacted directly with the computer hardware.
- These machine are called bare machine as they don't have OS.
- Every computer system is programmed in its machine language.
- Uses Punch Card, paper tapes and language translator

These system presented two major problems.
1. Scheduling
2. Set up time:

Scheduling:
Used sign up sheet to reserve machine time. A user may sign up for an hour but finishes his job in 45 minutes. This would result in wasted computer idle time, also the user might run into the problem not finish his job in alloted time.

Set up time:
A single program involves:
- Loading compiler and source program in memory
- Saving the compiled program (object code)
- Loading and linking together object program and common function

Each of these steps involves the mounting or dismounting tapes on setting up punch cards. If an error occur user had to go the beginning of the set up sequence. Thus, a considerable amount of time is spent in setting up the program to run.

This mode of operation is turned as serial processing ,reflecting the fact that users access the computer in series.

Simple Batch Processing:

- Early computers were very expensive, and therefore it was important to maximize processor utilization.
- The wasted time due to scheduling and setup time in Serial Processing was unacceptable.
- To improve utilization, the concept of a batch operating system was developed.
- Batch is defined as a group of jobs with similar needs. The operating system allows users to form batches. Computer executes each batch sequentially, processing all jobs of a batch considering them as a single process called batch processing.

The central idea behind the simple batch-processing scheme is the use of a piece of software known

as the **monitor**. With this type of OS, the user no longer has direct access to the processor. Instead, the user submits the job on cards or tape to a computer operator, who batches the jobs together sequentially and places the entire batch on an input device, for use by the monitor. Each program is constructed to branch back to the monitor when it completes processing, at which point the monitor automatically begins loading the next program.
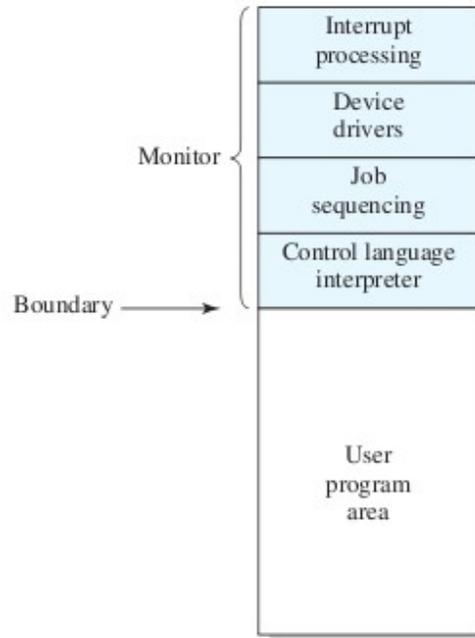

Fig.1.5: Memory Layout for resident memory

With a batch operating system, processor time alternates between execution of user programs and execution of the monitor. There have been two sacrifices: Some main memory is now given over to the monitor and some processor time is consumed by the monitor. Both of these are forms of overhead.

Multiprogrammed Batch System:

A single program cannot keep either CPU or I/O devices busy at all times. Multiprogramming increases CPU utilization by organizing jobs in such a manner that CPU has always one job to execute. If computer is required to run several programs at the same time, the processor could be kept busy for the most of the time by switching its attention from one program to the next. Additionally I/O transfer could overlap the processor activity i.e, while one program is awaiting for an I/O transfer, another program can use the processor. So CPU never sits idle or if comes in idle state then after a very small time it is again busy. This is illustrated in fig below.
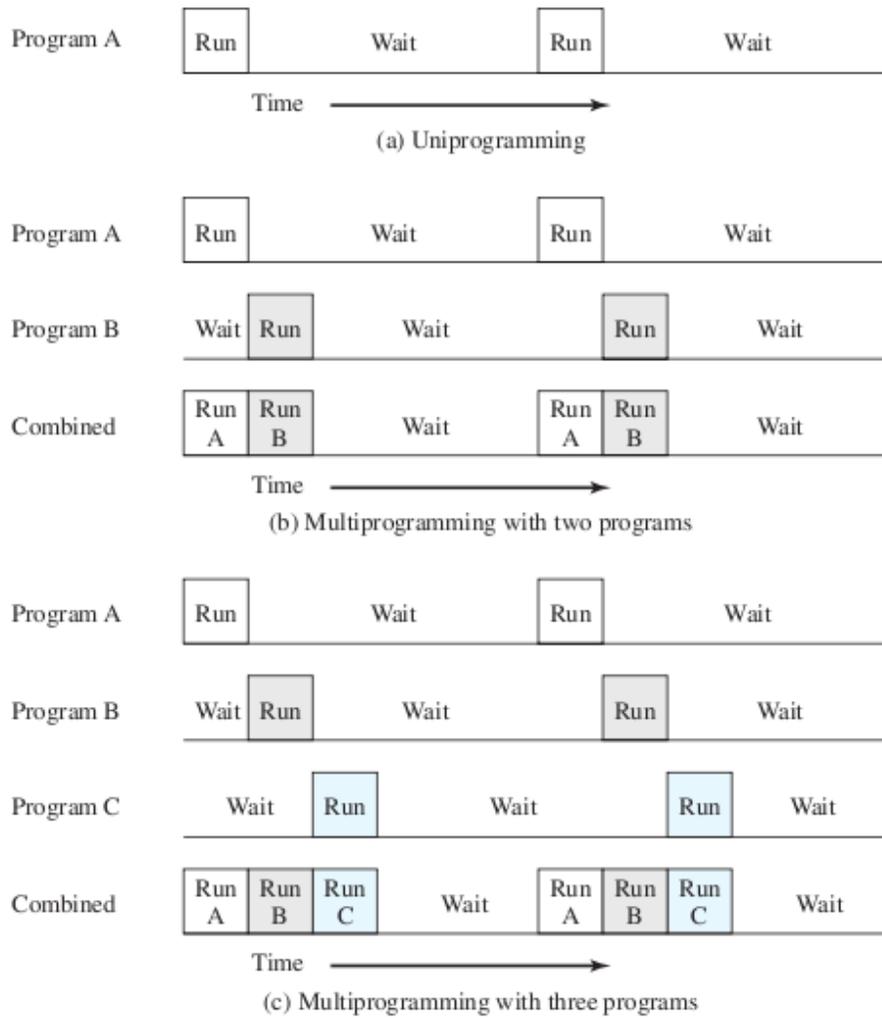
Program A | Run | Wait | Run | Wait

Time ⟶

(a) Uniprogramming

Program A | Run | Wait | Run | Wait

Program B   Wait | Run | Wait | Run | Wait

Combined | Run A | Run B | Wait | Run A | Run B | Wait

Time ⟶

(b) Multiprogramming with two programs

Program A | Run | Wait | Run | Wait

Program B   Wait | Run | Wait | Run | Wait

Program C   Wait | Run | Wait | Run | Wait

Combined | Run A | Run B | Run C | Wait | Run A | Run B | Run C | Wait

Time ⟶

(c) Multiprogramming with three programs

Fig 1.6. Multiprogramming example