Class- 4CE1
Year-2015-16

UNIT-I

OS-2140702

Introduction

Prof. Jayna Donga

M. Imp. * **Definition** :- An OS is a Program that acts as an interface between the computer users and the computer hardware.

- The purpose of OS is to provide an environment in which a user can execute Application programs in a convenient and efficient manner.

Note:- computer software/Program can be divided into two kinds :- System programs, which manage the operation of computer itself.
while Application programs, which performs the actual work the user wants.
- The most fundamental system program is OS, which controls all the computer's resources and provide base upon which the application programs can be written and executed.

Ex- system s/w → OS ; Ex - APPⁿ s/w → word processor, spreadsheets, compiler & web browsers.

* Generation of OS / Evolution of OS

M. Imp. - The OS may process its work serially or concurrently.
- That is it can dedicate all the computer resources to a single program untill the program finishes or can dynamically assign the resources to various currently active programs.
- The execution of multiple programs in an interleaved manner is known as multiprogramming.
- In this section we will see Evolution of OS from serial to multiprogramming systems.

1) Serial processing :- Early computers were enormous machines run from a console.

- The common input devices were card readers, and tapedrives.
- The common output devices were line printers, tape drives and card punches.
- The user did not interact directly with the computer systems.
- The user prepared a Job (machine language) - which consisted of the program, the data, and some control information about the nature of Job and submitted it to the computer operator.
- The Job was usually in form of punchcards.
- During the late 1940s to 1950s, there were no os.
- programmers used to interact directly with the computer hardware by writing machine language.
- Programmer enter program through input devices.
- If the execution of programs were completed successfully, the output appeared (after minutes, hours, or days) in printed form using the printers attached to the machine.
- whenever the programmer wants to execute a program, first of all he had to first reserve machine time slote.
- In given time he can enter into the machine room, execute his program.
- Once, the time slot of one programmer was over, the next programmer was supposed to perform the same procedure.
- Thus, all the users were allowed to enter the computer room sequentially, hence, this was known as serial processing.
- The main problem associated with serial processing was that in some cases, a programmer might sign up for two hours but finished his work in one and half hours in such situations, the computer processing time (CPU time) would be wasted, since no other Programmer was allowed to enter the machine room during that time.
- so, serial Processing resulted in low utilization of resources.

2) Batch Processing :- In the mid 1950s, the mainframe systems were introduced, which was the first computer used to tackle many commercial and scientific applications.

- These machines were generally operated by professional operators.
- However they were so expensive that only major government agencies and big corporations could afford to buy them.
- A clear distiction was made between computer operators and programmers.
- Programmers used to prepare a job that consisted of instructions, data & some control infoⁿ about nature of Job, and submit it to the computer operator.
- The Jobs were generally in form of punchcards.
- when currently running job was finished, the operator would take off the output using a printer (it might be kept in another room), and the programmer could collect the output later at any time.
- The oper^r performed the same process for all the card decks submitted by the programmers.
- much computer time was wasted while the operator was moving from one room to another.
- To reduce this wasted time and speed up the processing, the operator used to batch together the jobs with similar requirements, and run these batches one by one.
- This system was known as batch processing.

EX:- the Jobs that need FORTRAN compiler can be batched together so that the FORTran compiler can be loaded only once to process all these Job.

- Note that the Jobs in a batch are independent of each other and belong to different users.

3) multi programming : The drawback of batch processing is, it dedicates all resources to a single Job at a time.

- The execution of a single Job cannot keep the CPU and I/O devices busy at all the time because during execution, the Jobs sometimes require CPU & sometimes require I/O devices but not both at one point of time.
- Hence, when the Job is busy with CPU, the I/O devices have to wait, and when the Job is busy with I/O devices, the CPU remains idle.
  - For Example consider two Jobs P_1 & P_2, both of which require CPU time and I/O time alternatively.
  - The serial execution of P_1 and P_2 is shown in Fig-I(a).
  - The shaded boxes show the CPU activity of the Jobs, and white boxes show their I/O activity.
  - It is clear from the figure that when P_1 is busy in it I/O activities the CPU is idle even if P_2 is ready for execution.

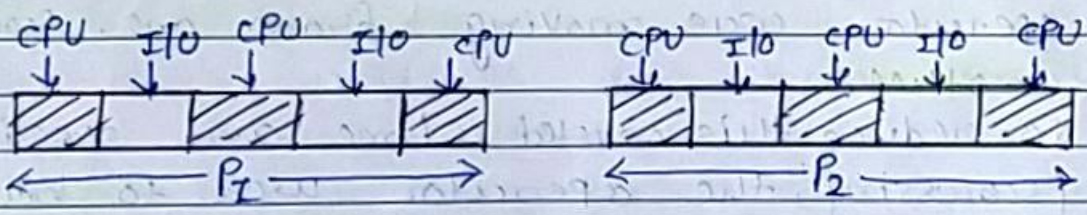

Fig-I(a) serial execution of P_1 & P_2

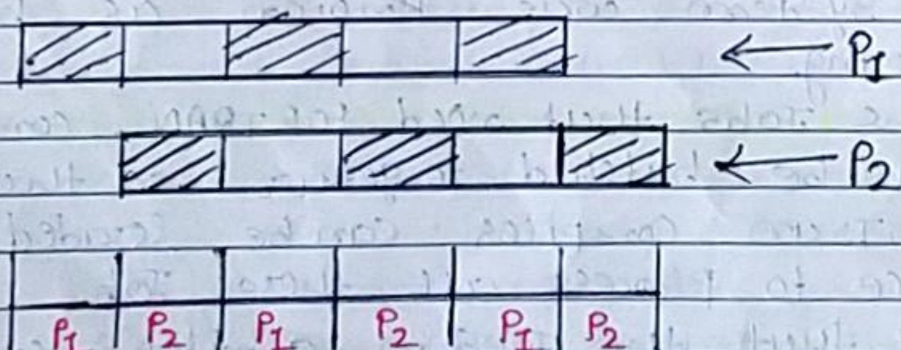

Fig-I(b) multiprogrammed execution of P_1 & P_2

- The idle time of CPU and I/O devices can be reduced by employing multiprogramming which allows multiple Jobs; to reside in main memory at the same time.
- If one of the Job is busy with I/O devices, the CPU can pick another Job & start executing it.
- To implement the multiprogramming, the memory is partitioned into several Partitions, where each partition can hold only one Job.
- The Jobs are organized in such a way that the CPU always has one Job to execute.
- This increases the amount of CPU utilization by minimizing the CPU idle time.
- The basic idea behind multiprogramming is that the OS loads multiple jobs into the memory from the job pool on the disk.
- It then pick up one Job from the pool and starts executing it.
- When this Job needs to perform I/O activity, the OS simply picks up another job and starts executing it.
- Again when this job requires I/O activity, the OS switches to third job & so on.
- When the I/O activity of job gets finished, it gets CPU back.
- Therefore, as long as there is at least one Job to execute, the CPU will never remain idle.
- Fig 1(b) shows the multiprogrammed execution of Job P_1 & P_2.
- Both are assumed to be in memory and waiting to get CPU time.
- Further assume that job P_1 gets the CPU time first.
- When P_1 needs to perform Its I/O activity, the CPU starts executing P_2. When P_2 needs to perform I/O activity, the CPU again switches to P_1, and so on.
- This type of execution of multi Processes is known as Concurrent execution.

M. Imp. * **Types of operating system**

→ Following are some of the most widely used types of OS.

1) Batch OS :- The user of batch OS cannot interact with the computer directly.
- Each user prepares his Job and submit it to the operator.
- The programs (such as payroll, forecasting, statistical analysis, and large scientific applications) that dont require interaction are well-served by batch Operating system.
- To speed up processing, Jobs with similar needs are batched together and run as a group.

Note: [Run one Job at a time] (FCFS) scheduling

* Problems → Lack of interaction b/w user and job.
→ CPU is often idle, because the speeds of the mechanical I/O devices is slower than CPU.
→ Difficult to provide the desired priority.

2) multiprogramming OS :- multiprogramming systems allow concurrent execution of multiple programs, and hence multiprogramming OS require more sophisticated scheduling algorithms.
- The programs should be scheduled in such a way that the CPU remains busy for maximum amount of time.
- memory management must provide isolation and protection of multiple programs residing simultaneously in the main memory

Note: [Run more than one program at a time]

- since multi programming os allow sharing of I/o devices among multiple users, more sophisticated I/o management is required.
- File management in multiprogramming os must provide advanced protection, and concurrency control methods.

→ Time-sharing os :- Time sharing system is an extention of multi programmed system.
- Processor's time which is shared among multiple users simultaneously is termed as time sharing.
- Time sharing systems require more complicated CPU scheduling algorithms.
- Most time-sharing systems make use of a round-robin scheduling algorithm, in which each program is given a system-defined time slot for its execution.
- when this time slot gets over, and program still requires CPU for its execution, it is interrupted by the os and is placed at the end of the queue of waiting programs.

3) Real-time os :- In real-time systems, the correctness of the computations depends not only on the output of the computation but also on the time at which the output is generated.
- A Realtime system has well-defined, fixed time constraints.
- If these time constraints are not met, the system is said to have failed inspite of producing the correct output.

- Thus, the main aim of real-time system is to generate the correct result within its time constraints.
- Consider an example of a car running on an assembly line.
- certain actions are to be taken at certain instants of time.
- If the actions are taken too early or too late, the car will be collapsed.
- ∴ For such systems, the deadlines must be met in order to produce the correct result.
- Some examples of real time systems are: air-traffic control systems, fuel-injection systems, robotics, home-appliance controllers etc.

→ Real-time systems are of two types:

1) Hard real-time system: the actions must be taken within the specified timeline; otherwise undesirable results may be produced.
   - Industrial control and robotics are two examples of hard real-time systems.

2) soft real-time system: In this case, it is not mandatory to meet the deadline.
   - A real-time task always gets the priority over other tasks, and retains the priority until its completion.
   - If the deadline could not be met due to any reason, it is then possible to reschedule the task and complete it.
   - multi media, virtual reality, and advanced scientific applications come under the category of soft-real-time systems.

4) Distributed OS :- A distributed computer system is basically a computer network in which two or more autonomous computers are connected via their hardware and software interconnections, to facilitate communication and cooperation.

- The computers can be interconnected by telephone lines, coaxial cables, satellite links, radio waves, etc.
- The main objective of a distributed OS is to provide transparency to its users.
- That is, users should not be bothered about how various components and resources are distributed in the system.
- A distributed system is generally designed to support system-wide sharing of resources, such as I/O devices, files and computational capacity.

5) Network OS :- Network OS runs on a server and provides server the capability to manage data, users, groups, security, applications, and other networking functions.

- The primary purpose of network OS is to allow shared file and printer access among multiple computers in a network, typically a LAN, a private network or to other Networks.

M.Imp. * **OS services**

- One of the major responsibility of an OS is to provide an environment for the efficient execution of user programs.
- For this, it provides certain services to the programs and users.
- These services provide an abstract view of the system to the users so that they only have to concentrate on the functional part of the programs without bothering about the internal details of the system.
- Ex- programmer need not to bother about how memory is allocated, to their programs, where their programs are loaded in memory during execution, how multiple programs are managed and executed, how their programs are organized in files to reside on disk etc., while writing programs.

- Inspite of some specific services that may differ from one OS to another, all the OS provide some common services.
- These services include the following:
  → User Interface: providing a user interface to interact with ~~users~~ b/w is essential for an OS.
  - This interface can be in one of the several forms.
  - One of the UI is command-line interface, in which user interact with the OS by typing commands.
  - Another is batch interface, in which several commands & directives to control those commands are collected into files which are then executed.
  - The most commonly used interface is graphical user interface (GUI), in which users interact with the system with a pointing device, such as a mouse.

→ Program execution: The system must allocate memory to the user programs, and then load these programs into memory so that they can be executed.
- The programs must be able to terminate either normally or abnormally.

→ I/O operations: Almost all the programs require I/O involving a file or an I/O device.
- For efficiency and protection, the OS must provide a means of performing I/O instead of leaving it for users to handle I/O devices directly.

→ File-system manipulation: Often, programs require to manipulate files & directories, such as creating new file, writing contents to an existing file, deleting or searching a file by providing its name etc.
- Some programs may also need to manage permissions for files to allow or deny other program's request to access these files or directories.

→ Communication: A process executing in one computer may need to exchange info^n with the process executing on the same computer or on a different computer connected via a computer network.
- The info^n is moved b/w processes by the OS.

→ Error detection: There is always a possibility of occurrence of errors in the computer system.
- Error may occur in the CPU,

memory, I/o devices or in user program. Examples of errors include an attempt to access an illegal memory location, power failure, link failure on a network, too long use of CPU by a user program etc.
- The OS must be constantly aware of possible errors, and should take appropriate action in the event of occurrence of error to ensure correct and consistent computing.

→ Resource Allocation :- In case of multi programming many programs execute concurrently, each of which require many different types of resources, such as CPU, memory, I/o devices etc., Therefore, in such a situation OS must $\overset{\text{allocate}}{}$ resources such that resources are utilized efficiently, and no program should wait forever for other programs to complete their execution.

→ Protection & security :- Protection involves ensuring controlled access to the system resources. In multiuser system, the owner of info^n may want to protect info^n. When several processes executes concurrently, a process should not be allowed to interfere with the other processes, or with OS itself. Security involves protecting the system from unauthorized users.

→ Accounting :- We may want to keep track of usage of system resources by each individual user.
- This info^n may be used for accounting so that users can be billed, or for accumulating usage statistics, something which is often of value to researchers.