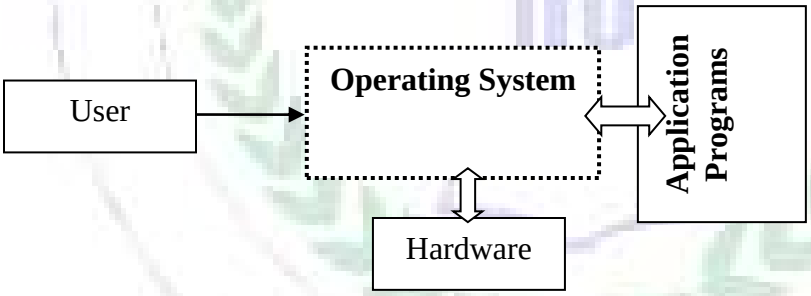
Lesson 1, 2

Objectives

- What is OS?
- Simple Batch systems
- Multi-programmed batch systems
- Time Sharing systems
- Parallel systems
- Distributed systems
- Real time systems

WHAT IS AN OPERATING SYSTEM?

It is a program (system software) that acts as an intermediary (**interface**) between users of a computer and the computer hardware. OR It is system software which is responsible of over all computer system **supervision** and make the best use of its hardware. It simply works like a **government**. It simply provides an *environment* in which other programs (application software) can do useful work. This all is depicted in the figure given below.



This diagram shows that user can not directly communicate with hardware (there are some cases but common users even advance users can't)

Components of a Computer System are:

- Hardware (Peripherals)
- Data
- Software

There are two types of software.

1. Application Software (MS. Office, MS. Paint, etc)
2. System Software (operating system)

A computer system as a whole has certain resources generally named above, so it can also be viewed as a ***resource allocator***, which allocates the resources among users and application programs. Like a manager it monitors all the activities. It is easy to define an operating system by what they do instead of what they are?

***Goals of an operating system*:**

- Users' convenience
- Efficient operation of computer system

The history of operating system is as old as computer architecture and both are highly influenced by each other. Types of computer systems are given in a little brief.

In short, **operating systems are developed for two main purposes:**

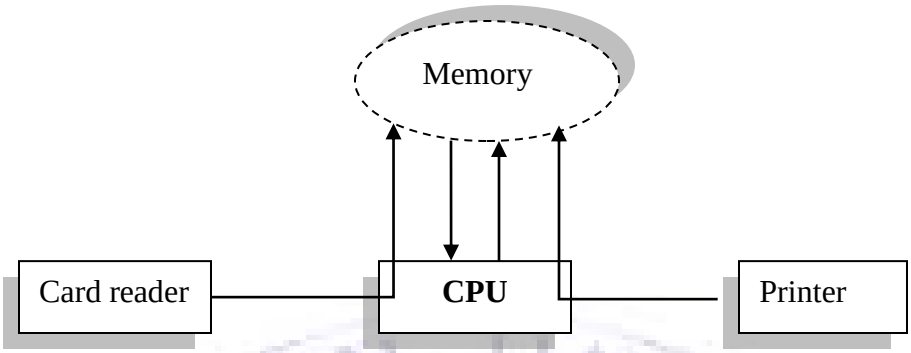
1. To schedule computational activities to ensure good performance of computer system
2. To provide convenient environment for the development and execution of programs

SIMPLE BATCH SYSTEMS

In this system jobs (programs, requirements, request) were punched over a computer readable card (like a table with certain rows and columns). These cards were fetched to the CPU (input) and CPU after a considerable time generated output sent to some printer or to punch again on some card. To save the CPU time operator took cards from programmers and put them together in such a way that similar jobs kept together. These groups were called ***batches***. Those days operating system was very simple.

Spooling

Latterly, to make the things further sophisticated card readers were attached to the memory and memory interacted with CPU and with output devices. Since the speed of card reader was much lower than CPU speed and this difference is even significant in modern era. This process was called *simultaneous peripheral operations online* or *Spooling*. It was helpful to process the data at remote sites. Card images were written on magnetic or tape devices and read accordingly.



MULTIPROGRAMMED BATCH SYSTEMS

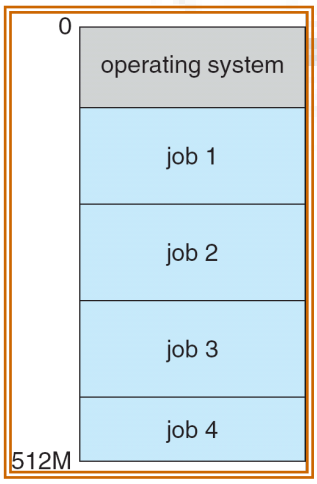
One of the drawbacks of batch processing systems was that a single job was to run on CPU and after that it sits idle and to wait for the next. This limitation was overcome by advent of spooling. In this way OS creates a *job pool* (consists of all the jobs need to be executed) on storage device, a subset of these jobs is kept in main memory to be executed **sequentially**. In this way CPU need not to sit idle and jobs one after the other run. Some terminologies relevant to this procedure are:

Job scheduling: How when and how many jobs should kept in main memory (since it is limited), which to execute first.

CPU scheduling: How much time should be given to a single job in order to make rest of the jobs in lesser waiting state?

Memory Management: What if the size of a job exceeds from available size of main memory?

Note: These are the topics we will discuss in detail subsequently.



TIME-SHARING SYSTEMS

The problems with above systems are less interaction of a running job and user, and hard to debug. To overcome these problems *multitasking* or *time sharing* systems came into being. In these systems a user is involved during the job processing. Also when a process is in wait to get some instruction from user, CPU gets busy with remaining jobs. In this manner each user thinks that he is utilizing CPU, though it is sort of a time division fashion. Each user is allocated CPU at a random, so these systems give an interactive use.

PARALLEL SYSTEMS

Most of the systems we used to deal with are single processor systems. However there is trend towards *multiprocessor systems*. Such systems have more than one CPU in close communication, sharing the available resources like common bus, the clock and sometime memory and peripherals. These systems are referred to as *tightly coupled* systems. Such systems are capable of running both CPU in parallel without any interference.

Benefits

- Job sharing
- Less time to execute multiple jobs
- If one CPU fails, second can backup, this property is sometime referred as *graceful degradation* or *fault tolerance*
- Fewer resources needed like power, space, wiring etc.
- Faster execution
- Feasible in batch processing systems e.g., one CPU for card reading (remote job entries), one for execution

Drawbacks

- Hard to manage more than one CPU
- Sometime separate operating system needed for individual CPU
- Sharing violation occurs
- Duplication of software and hardware needed
- *N* processes does not give *N* time of single CPU throughput
- Job scheduling and CPU scheduling and inter CPU communication is a tough job

DISTRIBUTED SYSTEMS

The recent trend in computer systems is to distribute computation among several processors. In contrast to parallel systems, these do not require shared memory or clock; these systems are connected via some communication line, like high speed busses or telephone lines. These systems are referred as *loosely coupled systems* or *distributed* systems. The processors in distributed systems may vary in size and function. They may include small microprocessors, workstations etc. Following are benefits to use distributed systems.

Resource sharing

In this way a computer can access the resources available to some other computer. For example, computer B is connected with A, with A contains a printer but B does not. Then B can share this from A.

Computation Speedup

In this context computation can be partitioned into a number of sub-computations that can run concurrently, and then a distributed system may allow us to distribute the computation among the various sites. Thus load is shared and computation just goes speed up. This movement of jobs is called *job sharing*.

Reliability

If one site fails in a distributed system, remaining sites can potentially continue operating without effecting rest of the operations.

Communication

Distributed systems are by virtue of a network (LAN), so the exchange of a data among systems even at a distance is possible.

REAL-TIME SYSTEMS

These are the systems configured to be in phase with real time scenarios. For example, operating system to control a robot, missile or radar, is dedicated as real time system, since it depends upon real situations. These are basically constrained systems; main constraint is time and accuracy. There are two types of real-time systems.

- Hard real-time systems: These are the systems where the time constraint is very stringent; there is no concept of time-sharing. This is because the real time tasks

are unpredictable and couldn't be in wait, even super computers goes impractical for such situations

- Soft real-time systems: These are the systems where time is not that much stringent constraint. Where we can afford some kind of latency. Like weather forecasting, online seat reservation or air-traffic control system.