1

INTRODUCTION TO OPERATION SYSTEM

SYNOPSIS

***

1.1 Introduction

An Operating System (O.S) is an important part of computer system. It is heart of computer and without this computer is like a dead body. In this chapter we will discuss what is an operating system. At present, there are many operating systems available like Windows NT, Windows 2000, Windows XP, DOS, UNIX and LINUX. We will discuss what is the purpose of an operating systems and how operating systems are developed.

1.2 Operating system (O.S)

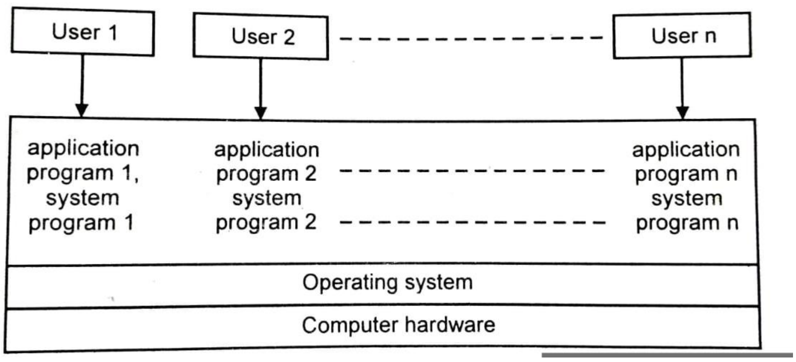
Definition :

An operating system is a set of system programs that act as an interface between computer user and the computer hardware.

Purpose of an Operating System :

- Provides an environment for running user programs.
- Provides an interface to the user to communicate with the system.
- It manage the computer resources in an efficient manner.
- Uses hardware of the systems in efficient manner.
- To execute user programs and to make solving user problems easier.
- Provides an overall control to the system.

To carry out above mentioned, as well as many more tools, operating system occupies a very special place in the computer system. Fig. 1.1 shows the abstract view of the components of a computer system.



Every computer system is divided into four parts.

- the hardware.
- the systems programs(O.S).
- the application programs.
- the users.

An operating systems uses hardware resources in efficient manner to run the application programs like complier, text editors, database system, banking system which are used to solve the computing problems of the users. The operating systems controls and co-ordinates the use of the hardware among the various application programs for the various users.

An operating systems controls and manages all the resources (e.g. I/O devices, CPU, memory). An operating system is the one program running all times on the computer(called kernal), with all else being application programs.

1.3 TYPES OF OPERATING SYSTEM

The operating systems are evolved from batch operating systems to distributed operating system and clustered operating system.

1.3.1 Early batch systems

In early days, the computers were physically very large machines. There was no software for loading and executing the programs.

To run a program, a programmer had to sign the machine (way of booking the computer systems) first. He had to manually feed his program to the computer system by adjusting number of switches. Appropriate buttons were pushed to load the starting address of the program, and start execution of the program.

Running program could be monitored by programmer by display lights on the control.

In case of any error, user had to halt the program. Examine contents of the memory and different registers. Debugging was done directly from the console. Output was first printed on tapes or punched onto cards for future printing.

A programmer had to carry out all these number of steps manually, to execute any program on the machine. All this time of loading a program was called as setup time.

There was two major problems with these systems :

- While signing up the computer, programmer was suppose to specify the duration, for which he need the computer. Specifying this in advance was very difficult. Too short duration was leaving his job incomplete. Too big duration was making him pay for a costly computer time.

- During the setup time, the CPU was idle. Setup time was directly affecting performance of the computer system.

As days passed, number of inventions were made in software and hardware of the computer e.g. card reader, line printers, tape, assemblers, compilers, libraries of common functions were developed.

Initially, programmer had to write a program for each of the input/output device which was present. Because each input/output device had different properties. To avoid this tedious task, device drivers for different devices were developed.

Now the setup time required was for :

- Loading proper compiler for the program.
- Loading device drivers for number of input/output devices.
- Loading the libraries of common functions.
- Loading the actual program.
- Loading other related softwares.
- Loading data required for program and unloading all this in case of completion or error.

For running any program on the computer, major time wasted was the set up time.

Efforts were taken to reduce the setup time. Some of the solutions are as follows:

(1) Sequential processing: In this case jobs submitted sequentially. Suppose there were four jobs need CPU time. First user would like to execute COBOL program (job), second user would like to execute FORTRAN program (job); third user want to execute COBOL and forth one want to execute FORTRAN again. If jobs are process sequentially, there would be so much repetition is done. First load COBOL compiler then FORTRAN compiler, then again COBOL compiler and FORTRAN compiler so it increase set-up time and hence CPU idle time is also increases as shown in Fig 1.2.

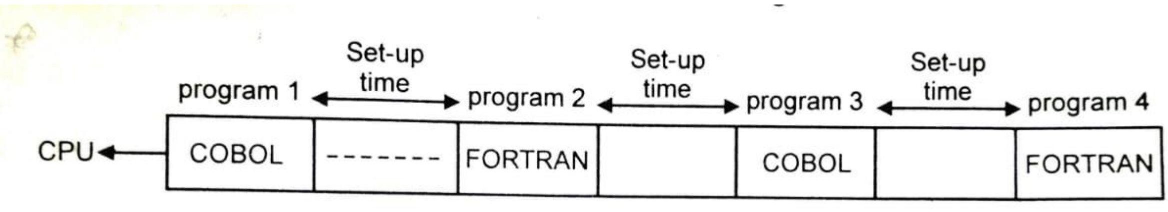


Fig. 1.2 : Sequential processing

(2) Batch processing : Major of the set-up time included, the time for loading and unloading common software like compilers, assemblers, etc. So programs which required common needs for execution were batched together and run as group. Hence, the set-up time is less.

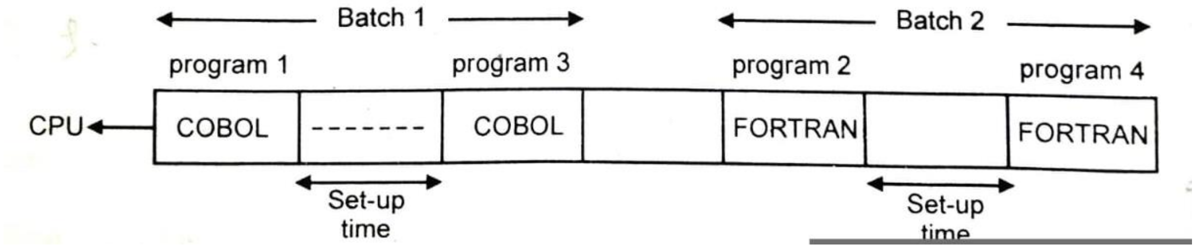


Fig. 1.3: Batch Processing

The Fig. 1.3 shows that the COBOL programs are group in a batch and the run together. So FORTRAN compiler is loaded only once. Similarly, to execute for FORTRAN programs, FORTRAN compiler loaded once.

In batch processing, during the transition from one program to another program, CPU sat idle. To avoid this CPU idle time, automatic job sequencing was introduced. Computer could start with next job, without user interference and this was manage by a special system program called resident moniter. The resident monitor is always resident in the memory and could transfer the control from one job to another automatically.

(3) Resident monitor : When the computer was turned on, the resident monitor was activated and it would transfer control to a program. When the program terminates, the control would return to the resident monitor and which would then go on to the next program. A control and interpreter is major part of resident monitor that is responsible for reading and carrying out the instructions on the cards at the point of execution.

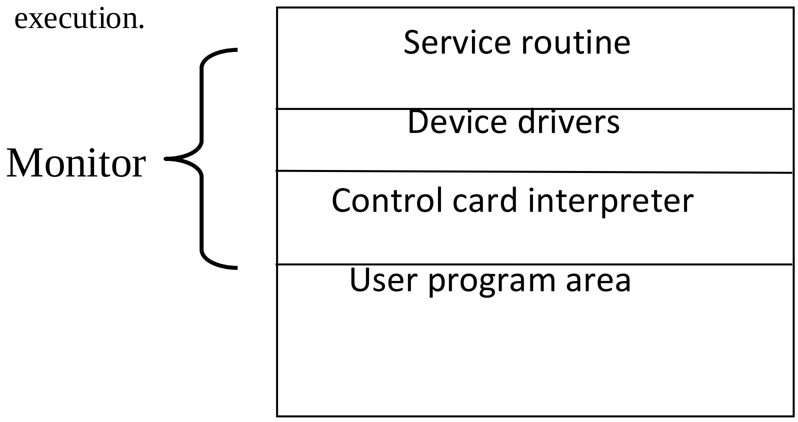


Fig. 1.4 : Resident monitor

Resident monitor needed some extra information about programs as,

- When program to execute.
- Which cards are belonging to which programs i. e. in a batch of few programs, at which card one program is over and the other one is started.

To solve this problem of resident monitor, control cards were introduced.

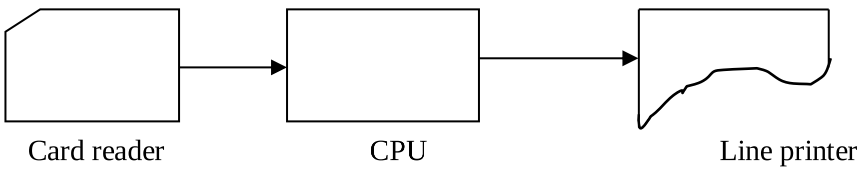
Control cards had same physical appearance like program cards. But it contained some special mark at some special position, so that it could be identified by resident monitor. Control cards not only separated the programs in the batch but also contain some information about program and programmer.

All the above mentioned solutions helped a lot to reduce setup time. But still CPU idle time problem was not yet solved.

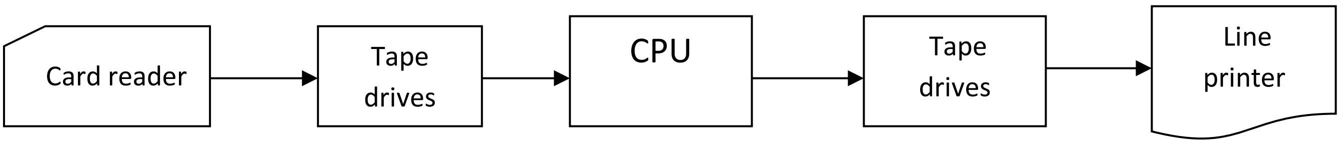
CPU was remaining idle for most of the time, because of slow input/output devices. Therefore, efforts were taken to improve CPU utilisation.

(Though batch monitor improved the performance of the system, once the batch was submitted to computer, while execution was going on, user could not interact with his program. One could get output of his program only after the total batch was over, resulting in increased turn around time. Throughput was goof by batching of jobs.)

(4) On-line and off-line operation : The difference in speed of CPU and I/O devices results in CPU idle time. When I/O was performed, if I/O device is slow then CPU has to wait for I/O. if CPU is processing, the I/O devices were idle.



(a) On-line



(b) Off-line

Fig. 1.5

Instead of CPU reading from slow card reader and printing to slow line printer, faster tapes were for both purposes.

All cards were copied onto the tapes first and all output was printed to tape and further tape was printed using the printer.

Though CPU idle time was reduced, this scheme had few drawbacks :

- For input and output, two different tapes were required. Since two ends of the same tape cannot be accessed simultaneously.
- Only sequential access was possible.

- User's application program was expecting an input from a cards. But the cards were copied onto tape. Now there was need of having program that could run with different input/output devices. This property is called as device independence. This can be made possible by having an operating system identify the device, a program is expecting.

Another method to achieve off-line was by means of a dedicated fast terminal. This terminal was used for the purpose of input/output. This terminal was called as satellite to main computer.

(5) Buffering :

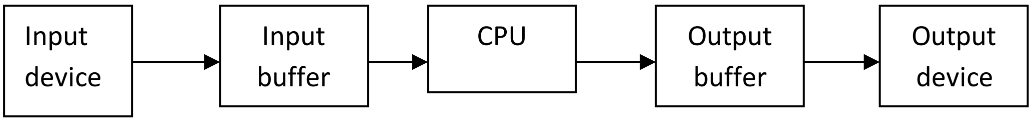


Fig. 1.6

Buffering keeps input/output devices and CPU busy simultaneously. As input is slow, it starts reading before CPU. After, some of input is done, CPU starts processing. CPU gets the input from input buffer.

On the other hand CPU stores the output in the output buffer and output device takes it from there.

In the worst case with very slow input/output devices CPU will always find an empty input buffer and a full output buffer. In both the situations or in either of it CPU will have to wait.

When input finishes with one record then only it can start reading next record. Interrupts are used to implement this situation. After input is over an interrupt is generated. An interrupt is a signal to the microprocessor that its immidiate attention is needed. The moment the interrupt draws the microprocessor's attention, the microprocessor's normal work is interrupted and hence the name, interrupt.

How the interrupt functions :

Interrupt are important part of the computer architecture. To manage the interrupts generated by different devices, each kind of interrupt is identified by a unique number. IVT (Interrupt Vector Table) is maintained for all interrupts in the system. IVT is nothing but an array of addresses. Each location of it can be accessed by taking interrupt number as an index. At this address position is the ISR (Interrupt Service Routine), which is nothing but the action part for that interrupt. An ISR saves the current status of the system.

Then the service routine is executed and control returns back to CPU. CPU resumes its work by taking help of reserved status.

In short, interrupt generated → CPU stops whatever it is doing → interrupt is → identified taking interrupt number as an index → IVT accessed → preserves status → service routine executed → control returns → CPU resumes.

Whenever input/output device finishes reading/writing one unit of input, it interrupts the CPU. If an interrupt is generated after each character is read or written, CPU will consume more time in attending the interrupt rather than actual reading or writing.

To solve this problem Direct Memory Access (DMA) is used for high-speed input/output devices. In this case, only one interrupt is generated for one block rather than for one byte.

(6) Spooling : Spooling stands for Simultaneous Peripheral Operation Online.

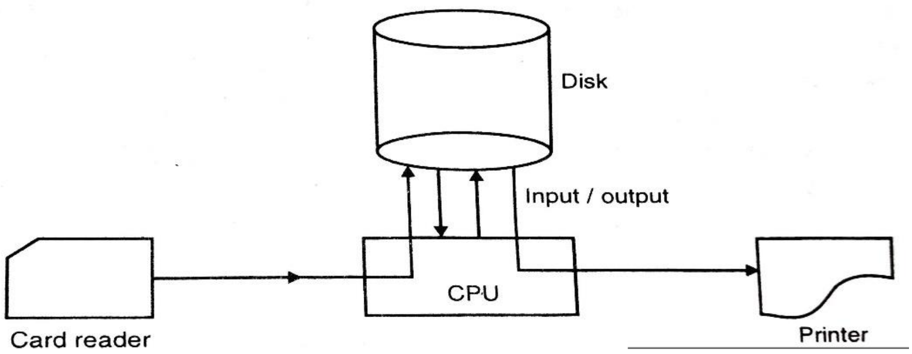


Fig.1.7

Disk is used for the storing the input as well as output. Single disk is enough for both purposes. All the cards are copied into the disk. A table is maintained which preserves the address of each card image. Each job is entered into the table as it is read in.

Advantages :

1. Spooling overlaps input/output of one job with computation of other. (Even in simple spooler, it may read data of one process and may computer data of other process.)
2. A spooler can manage number of jobs from the job pool. They all can be read into the disk. As disk provides direct access, spooler can change the sequence of the jobs in which they can be executed. This was not possible with taps. Because of this property, spooling can allow scheduling of the jobs.

Disadvantages :

1. There is an extra overhead of maintaining table of card images.
2. An extreme of spooling is called as staging of tapes. To avoid the wear of the magnetic tape, it is copied onto the disk and CPU will access the disk and not the tape.

1.3.2 Multiprogramming system

In multiprogramming, more than one programs or jobs are present in the memory at a time. Multiprogramming, in the simplest sense, means to maintain two or more jobs concurrently in execution or states of execution. In serial execution or batch system, to execute any program, it has to be brought into memory. When execution of this job is over, other is brought into the memory. There are two main reasons in this scheme by which CPU remains idle :

- the job in the memory first performs input/output when CPU has nothing to do.
- time is wasted between swapout of executed and swapin of job to be executed.

The basic idea of multiprogramming is as follows. The job pool consists of number of jobs. The operating system picks jobs from job pool. Many jobs are keep in memory simultaneously. The number of jobs present in memory at time is called **degree of multiprogramming.** The operating system start execution of a job. (Only one job is executed at a time). If a job require I/O, then operating system switches to another job and executes it. CPU would not sat idle. When that job requires to wait for an I/O, the operating system is switches to another job and so on. The Fig. 1.8 shows the multiprogramming system with job execution

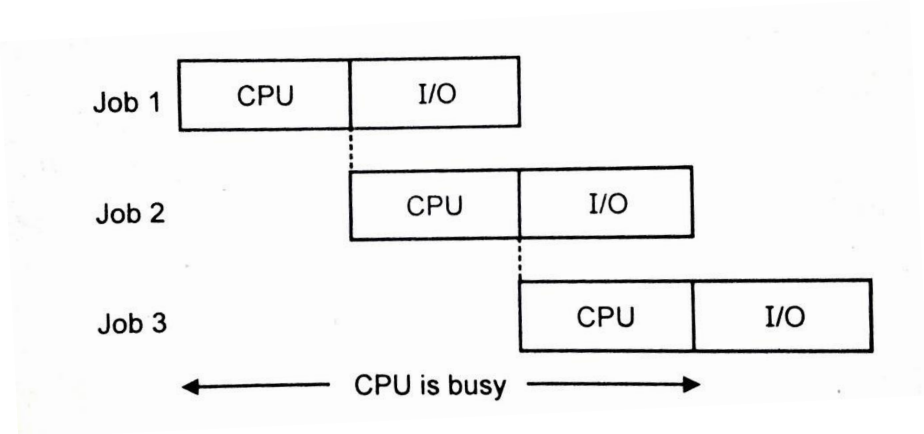


Fig.1.8 Job execution in multiprogramming

Fig. 1.8 shows that job 1 started its execution and if it requires I/O then, operating system switch to job 2 which is executed next and CPU would not remain idle.

Fig. 1.9 shows that several jobs are ready to run, the system must keep them in memory at a time. As there will always be several jobs in memory all the time, operating system requires some form of memory management. If many jobs are ready to run, then operating system must select among them. Operating system has to perform some CPU scheduling to schedule only one job at time.

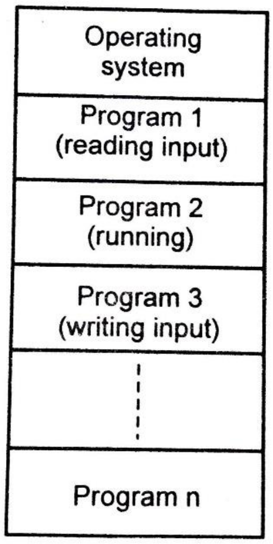


Fig. 1.9 Memory layout in multiprogramming

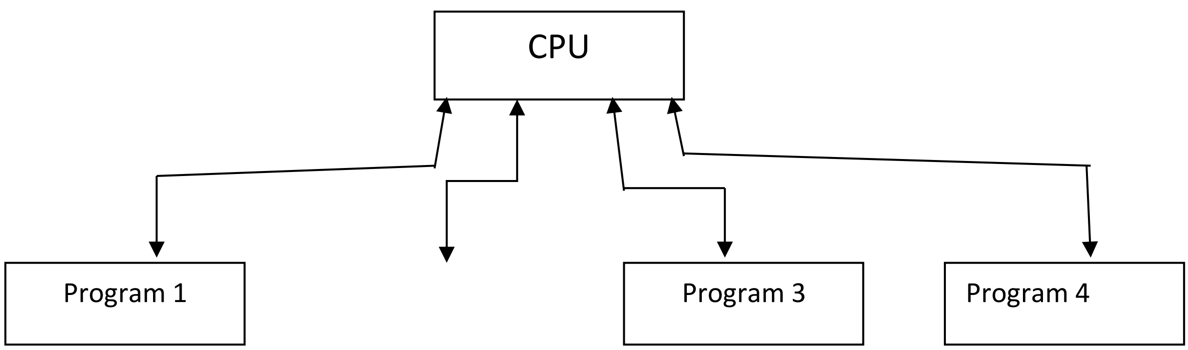
Advantages :

1. Because of this scheme above mentioned idle time is reduced a lot.
2. Multiprogram to give good result should be given proper job mix (of input/output bound and CPU bound jobs).
3. Multiprogramming results in improved in CPU utilization and in turn good throughput. As more than one job resides in the memory, turnaround time is also good.

1.3.3 Time-Sharing Systems

It is an interactive system, CPU is scheduled for the short time for each user. Each user is allowed to use the CPU for this short slot. After this slot is over, the status of this user program is preserved and CPU switches to next user. This process is called as context switching. After the time slot of last user is over, CPU switches again to the first terminal of the first user. This continues in Round Robin till CPU time of each of the user is over. In between if any user's CPU time is over, he is executed for the next round and so on. Every time when CPU switches to any user program, it continue execution (with the help of preserved status), wherever it was stopped last time. This switching of CPU from one user program to other user program happens so fast, that each user think he is the only program executed by CPU.

Fig. 1.10 shows the time sharing system, which provide a mechanism, for concurrent execution.



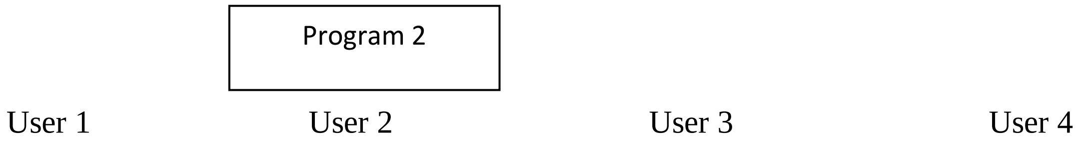


Fig. 1.10 : Time-sharing system

Advantages :

It is multitasking and CPU utilitsation is very effective. Many user share the computer simultaneously.

1.3.4 Desktop Systems

Instead of maximizing CPU and resource utilisation, the operating system use for the user convenience having better GUI (Graphical User Interface). These system includes :

- Microsoft Windows - Multitasking
- OS/2 - Multitasking
- Apple Macintosh - Virtual memory and multitasking
- Linux - Multiuser and multitasking

The computer uses this operating system having low cost of hardware but security required is more since viruses can be easily spread.

1.3.5 Multiprocessor System (Parallel System)

In this system, more than one processors are sharing computers bus or CPU, the clock, memory and I/O devices. Fig. 1.11 shows multiprocessing architecture.

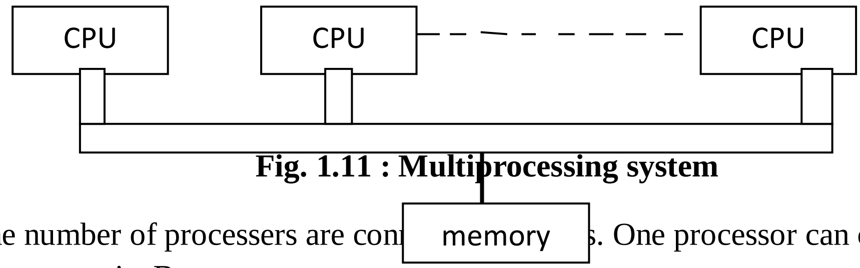


Fig. 1.11 : Multiprocessing system

The number of processers are con s. One processor can communicate to other processor via. Bus.

These are of two types :

(1) Symmetric multiprocessing : In this each processer, runs on identical copy of the operating system. These copies are communicate with one another when required.

(2) Asymmetric multiprocessing : In this each processor is assigned a specific task. A master processor controls the system and other (slave) processors communicate to master for further instruction.

In symmetric multiprocessing, there is no master-slave relationship between processors. Since there are many processors running simultaneously, the control is necessary on data which is received from input device and also send to output device. Also, CPU's are separate, so it should not happen that one CPU is idle and other is overloaded. To avoid this multiprocessing uses sharing memory or resources. These memory or resources are dynamically shared among the various processes. The operating systems like Linux, Windows NT, OS/2, UNIX, are provide support for such multiprocessing technique.

Advantages :

- Number of processors are more, so more work done in less time i.e. throughput is increases.
- Since sharable memory and resources are used, cost is less.
  - More reliable (if any of the processor fails that job is done by another processor and system will not halt).

1.3.6 Distributed systems

Distributed operating system was developed during 1980's. These depends on networks. It consists of many processors through networking. The networking are of different types and depends on the distances between their nodes, such as : LAN (local Area Network), exists for long distance and MAN (Metropolitan Area Network), could link building within the city.

In distributed operating system, uses are not aware of were the programs are located and processed, that should all be handled automatically by the operating system. There are basically two schemes for building such systems :

(1) Tightly coupled, in which the processor share the memory and a clock. A communication between the processors usually through the shared memory.

(2) Loosely coupled, in which the processor do not share memory or a clock. However, each processor has its own local memory and the processor communicate with one another through various communication lines such as telephone lines or high speed buses.

Speed several processors are running at the same time, the distributed operating system requires more complex processor scheduling algorithm.

Some of the reasons for using distributed operating system :

- **Reliability :** More reliable, if no node fails, the remaining nodes (sites) are continue operating.
- **Communication :** The several sites are connected to one another in the network, the information can be exchange among different site or users.
- **Speed :** Distributed operating system allow to share the computation among various nodes to run it concurrently so execution speed increases.
- **Resource sharing :** It allows to share resources computational task and provides a rich set of features to users.

Examples : UNIX and Windows operating system supports networking throughTCP/IP protocol.

1.3.7 Client Server Systems

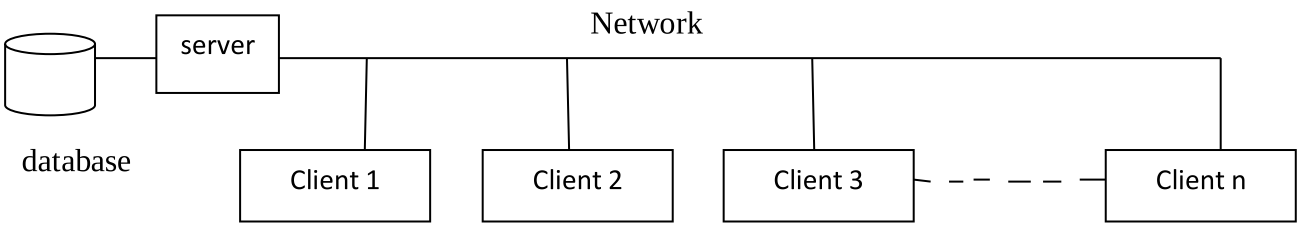


Fig.1.12 : Client-server architecture

Today, client-server system is popularly used to satisfy the user needs. In this, we access and use different database or services via. a network. It uses four components: client, server, database and network.

The **client** is a single user workstation which is used to access to data. It performs task such as :

- validation of data.
- send user requests and receives required information from the server.
- gives the interface.

The **server** is a multi-user processor that manage the database and provide information to the user. It performs the following tasks such as :

- manage database of several users.
- security to database is maintained.
- process user requests.

Server systems are of two types :

Computer-server : The request from the client is served and results is send back to the client.

File-server : Clients can create, delete, update, read the files using file-system interface.

Example : Internet

Advantages :

- reduce total execution time.
- increase client CPU usage.
- Use of client memory is reduces.
- network traffic is reduces.

1.3.8 Clustered systems

Clustered systems keep multiple CPUs together like parallel systems. The difference in these two is that clustered system composed of two or more individual systems coupled together. In this each node can monitor one or more of the other node in LAN. If the monitored machine fails, the monitoring machine can store its data and restart the applications that were running on the failed machine. In asymmetric clustering, one machine is standby machine while the other running the applications. The standby machine is monitored the other and if server fails, it becomes active server. In symmetric clustering, two or more hosts are running applications, and they are monitoring each other. This clustering is more efficient.

Example : Oracle parallel server is version of Oracle's database which run on parallel clusters.

1.3.9 Real-Time Systems

It is used when immediate response is required.

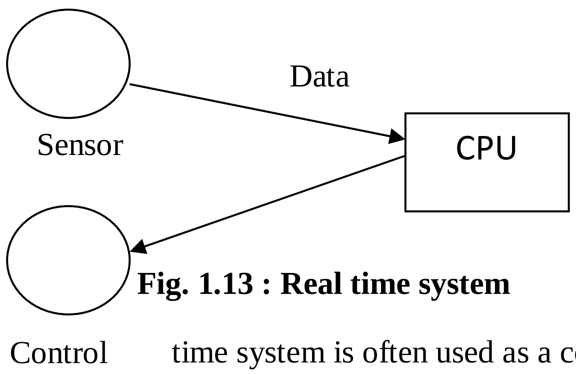


Fig. 1.13 : Real time system

A real time system is often used as a control device in a dedicated application, where response time is very critical. It has well-defined fixed constraints. If processing is not done within the defined constraint, system fails.

It is always used as control device in a dedicated system. Data is brought by sensors. After data is analysed, controls are adjusted to modify sensor inputs.

1.5.2 Handled systems.

It includes personal digital assistants (PDAs) such as Palm Pilots or Cellular telephones with

connectivity to a network such as the Internet. PDA is very small device uses virtual memory. Wireless technology is used for some handheld devices.

1.4. System Structures

1.4.1 INTRODUCTION

Operating system manages system resources. It schedules CPU, memory and many other functions. This chapter discuss about the services provided by operating system and how it is provided.

1.4.2 SYSTEM COMPONENTS

A system is a very large and complex structure. An operating system divide the system into smaller components and manages it. These system components are as follows.

1.4.2.1 Memory Management

Any instruction is stored in memory which is read by CPU. To perform read and write operation the data or instruction must be in memory. So memory is the main storage device, it is stored in memory by the CPU and then I/O calls are generated. When the program is executed, its object code is stored in memory along with data. When the program terminates, the memory which was allocated by the program, now available and next program is loaded from disk.

As we know in multiprogramming several programs are kept in memory at a time. Selection of a program to be executed is done by the operating system. The operating system manages the memory. Management, operating system various tasks as follows :

- When memory is available, operating system decides which processes are to be loaded in memory.
- Allocation and deallocation of memory whenever required.
- Keeping track of which parts of memory are currently used and by which processor.

1.4.2.2 Process Management

A process is a program in execution state. CPU executes a one process at a time instruction by instruction. Any process required many resources such as I/O devices, memory, CPU time, files. Operating system manages all these resources by allocating and deallocating resources to the process whenever necessary. E.g. if process P1 is running and it requires printer, then operating system allocate it if the printer is free in the multiuser system. If process P1 is terminated, then resources allocated to it are free or available to other processes.

The operating system can perform the following task with process management.

- Allocation and deallocation of the resources.
- Maintain process state i.e. running, suspending or ready.
- Providing mechanism for process communication.
- Providing mechanism for process synchronization.
- Handling deadlocked.

All these tasks we will discuss in further chapters.

1.4.2.3 File Management

File is a abstract or virtual storage. The operating system maps the files into physical media, and accesses these files via. the storage devices. Every time, to read file from the secondary storage and write the file onto the storage media these operations are required. Operating system selects the proper file from the storage for reading purpose and write it. Files are organize into the directories by operating system. When one file is shared among many user, the operating system controls the access to the file. It checks what purpose the file is needed and by which user.

The operating system performs the following tasks with file management.

- Creating and deleting directories and files.
- File manipulation.
- Mapping files onto secondary storage that is logical record is converted into physical record.
- Backing up files on storage media.

We will discuss these techniques in chapter 8.

1.4.2.4 Input/Output System Management

Since many input/output devices are involved, operating system manages all these devices very effectively.

The I/O sub-system like buffering, caching, spooling device-driver, hardware devices are managed by operating system. (Refer Chapter 9).

1.4.2.5 Secondary-storage Management

The programs are stored on secondary storage devices like disks. Secondary storage is used as source and destination for program processing. Hence the proper management of disk storage is required. The operating system performs the following task with disk management.

- Disk scheduling.
- Storage allocation using compaction.
- Free-space management.

1.4.2.6 Networking

A networking means many processors are connected through a communication network. An operating system usually generalize network access as a form of file access, with the details of networking which contain in network device driver. The networking is used when many applications are inherently distributed. The distributed operating system for reliability, resource sharing and speedup.

With a distributed system, nothing has to be done explicitly, it is all automatically done by the operating system without the user's knowledge. So, distributed system is a software system built on top of a network.

1.4.2.7 Protection System

If a computer system has multiple users, multiple processes then protection mechanism is required to protect one from the others activities. There was need of protecting monitor memory being accessed by user as well as one user accessing memory or data of the other users. Many kinds of protection are provided such as:

I/O Protection: one user cannot read/write the data of another user.

Memory Protection: Since in multiuser system, many program reside in the main memory. Each program is enclosed in two special fence registers, lower bound register and upper bound registers.

CPU protection: The program should not go n the infinite loop. So timer/counter is set.

1.4.2.8 Command-Interpreter System

Operating system provides number of services to the user. Generally, the services which are provided by the operating system vary system-wise. But there are some services which are provided by the operating systems.

Some such services are:

1. **Program execution:** To get any work done from a computer user writes a program. Operating system must let the user to load his program into memory and execute it. Execution of the program must end either normally or abnormally.
2. **Input/Output operations:** Operating system must provide means to perform input/output for any of user program, may be from a file or input/output device.
3. **File system manipulation:** User need files for different purposes. There should be some means to read/write file, create/delete files.
4. **Error detection:** Operating system must be able to be aware of error in CPU ans memory hardware, user programs (due to wrong/illegal memory access/ operations), error in input/output device (e.g. keyboard not connected) properly, printer out of paper etc).

Few other functions n addition to above mentioned functions are :

5. **Resource allocation:** In multiprogramming, multiuser environment, there are many resources to each of the process. These resources are managed by operating system e.g. status of resources (available/not available), whether a process request for a resource is valid or not. Update the status of resource in case it is released. Check if a resources can be shared or not. All these things are managed by operating system.
6. **Accounting:** Information about available resources, users is maintained by operating system.

7. **Protection:** When several jobs belonging to different users are executed, at that time program or data of one job is protected other jobs by operating system.

1.5 SYSTEM CALLS

A system call is a requested by the user to the operating system to do something on user's behalf. System call provides on interface between a running program and an operating system.

Some systems may allow system calls to be written in higher-level language program, in which the calls normally like predefined function or subroutine calls. They may be generated call to a special run-time routine that makes the system calls, or the system call may be generated directly in line.

To really understand what operating system does, we must examine these calls closely.

The system calls can categorized into five major categories : process and job control, file management, device management, information maintenance and communications. The system calls vary from operating system to operating system.

(a) Process and job control:

The types of system calls normally provided by the operating system with process and job control are :

- End, abort.
- Load, execute.
- Create process (FORK), terminate process.
- Get process attributes, set process attributes.
- Wait for time.
- Allocate memory, free memory.

When a program is running its execution is stop either normally (end) or abnormally (abort). It error occurs input data of the program, then error message is generated. In both normal and abnormal halt, the operating system transfer control to the command interpreter. The command interpreter then reads the next command. In multiuser system, command interpreter continues with the next command, while in batch system, it terminates the entire job and continue with the next job. We should able to control the execution to the newly created jobs or processes by assigning the job priority, or reset the attribute of the processes or sometimes terminates a created process if we find that it is wrong.

When the job is created, we should wait for its execution. For a specific event (wait event).

For debugging a program many system provide system calls to dump memory. This provision is helpful both for assembly language and machine-language debugging, particularly in a batch system. A program trace lists each instruction in the way it is executed. In microprocessor a trap is executed by the CPU after every instruction, known as single step.

In short, a process –

- May want to run or call another process.
- After called program finishes its execution, the control should return back to original caller, for the which its status must be preserved.
- The caller should be able to continue its own further processing.
- After a process is created, the processor has to wait till other gets over so wait time is there, or signal event will be generated to inform that process in over.
- After execution is over process may end normally/abnormally or abort.

For all these tasks the above system calls are used by operating system.

(b) File Management:

Major of the user activities are related with files. He need to be able to do number of operations on file. System calls for the purpose of the file manipulation may be :

- Create file, delete file
- Open, close
- Read, write, reposition
- Get file attribute, set file attribute. (attributes are file name, file type, accounting information and so on).

(c) Device management:

Many system calls used for files are used for devices.

- Request device, release device.
- Read and write operations.
- Get device attributes, set device attributes.
- Logically attach or detach devices.

Many operating system such as UNIX, merge the I/O devices and files into a combined file device structure. File can be thought of as abstract or virtual devices. If many users are using the system, however, we must request the device, to ensure that we have exclusive use of it. After work is over with the device, we must release it. These functions are analogous to the open/close system calls for files. After device is granted to us, we can read, write and reposition the device just as we can with files. In this case, I/O devices are recognized by special file names.

(d) Information Maintenance:

Different kinds of system information can be accessed by following system calls:

- Get time or date, set time or date.

- Get system data, set system data.
- Get process file, or device attribute.
- Set process file, or device attributes.

The systems have a system call to return the current time and date. Some system calls provides the information about number of users, version of operating system, free disk space, free memory and so on. Some system calls give the job or process information.

(e) Communication:

There are two common models of communication:

(i) **Message passing model:** In this approach, two or more processes accept to share their data exchange information in the shared area. Sharing of data is achieved by reading/writing data into shared area. e.g. P1 process will write the data to shared area, which P2 can read from the same place. The location of memory is decided by process (in the shared area) for writing purpose. They have to take care of more than one process, not writing at the same locations at the same time.

For the these activities, following system calls are used :

- Create, delete communication connection
- Send, receive messages
- Transfer status information
- Attach or detach remote devices.

1.5.1 System Call Implementation

Implementation of system call vary on different systems.

Generally, a unique number identifies the type of system call and a system call is made by calling the particular number.

Sometimes there is a need of sending some additional information along with, when the call s made. That is called as, parameter or parameter list of the system call [e.g. to open file, we need to specify file name and mode in which to open the file along with system call fopen ()].

Special register is kept aside for the purpose of sending parameters.

If parameters are too many in number and are beyond the capacity of the register to hold, parameters are stored in a block or table in the memory and the base address of it is passed as parameter through the register.

Most of the systems calls are available in assembly language.

Some programming languages like C permit system calls through program.

1.5.2 SYSTEM PROGRAMS

Most of the system provide the environment by a set of very efficient system programs.

System programs depending upon the kind of service, they provide, are categorized into several types.

- **File manipulations:**
  These programs manipulate files and directories. All functions related to file and directory manipulations are taken care by these programs e.g. rename, copy, print etc.
- **Status information:**
  Status information like date, time, memory available, number of users can be given by some system programs falling in this category.
- **File modification:**
  File creation as well as modification is made possible by means of programs like text editors.
- **Programming language support:**

Language processor like compilers, assemblers/interpreters for common programming languages are provided by system.

- **Program loading and execution:**
  Assembled or compiled programs are not of standalone nature. To make them execute, systems programs like loaders, linkers are provided by system.
- **Applications programs:**
  Programs to fulfill common requirement of the user fall in this category e.g. text formatters, plotters, database packages.

One of the most important system program is command interpreter. In simple words command interpreter is a program that translates a command into an action.

It is the system program through which the user communicates with the operating system. The commands we give, are nothing but programs.When we give any command, it is analysed by the command interpreter. Command interpreter gives call to appropriate program (for all valid commands only) to carry out the task.

The programs are present in command interpreter in either of the forms.

1. The command interpreter itself contains the code to execute the command. In this approach, size of the command interpreter is vet large. More the number of commands, bigger the command interpreter.
2. In the other approach, a separate file for each available command is present. Command interpreter just gives call to appropriate file. Addition of new commands in the system is very easy in this approach.

QUESTION

1. Define terms :
   (a) buffering (c) spooling
   (b) off-line operation (d) on-line operation
2. What are the purposes of operating system?
3. State the different types of operating system.
4. Differentiate batch operating system and multiprogramming operating system.
5. Why is spooling necessary for batch programming?
6. What is multiprogramming and time-sharing operation system? Is it possible to combine both these together? Justify.
7. Define multiprocessor system? What are the advantages of it?
8. State the types of distributed systems.
9. Define client-server architecture?
10. Explain clustered system in brief.

Multiple Choice Questions:

(i) Many programs are in memory at a same time in ____________ system.
(a) Multiprogramming (b) time-sharing (c) batch (d) none.

(ii) Batch system is not used in _______ .
(a) buffering (b)spooling (c) real-time (d)resident monitor.

(iii) Internet is an example of __________ system.
(a) client-server (b) multiprogramming (c) batch (d) handheld.

Ans. : (i) a, (ii) c, (iii) a.

State true or false :

(i) Multiprogramming also allows time-sharing.
(ii) In time-sharing system several processors are used.
(iii) In asymmetric multiprocessing master processer controls the system.
(iv) Today client-server is more popular system used.
(v) Operating system is a hardware component.

Ans. : (i) True, (ii) False, (iii) True, (iv) True, (v) False.

1. State and explain in brief, different services provided by an operating system.
2. Discuss the system services offered for the process and Job control.
3. What is a system call? How it is implemented?
4. Explain various types of system programs.
5. Explain the systems calls used for file and device manipulation.
6. What is the purpose of the command interpreter?

Multiple Choice Questions :

(i) …………….is a system program through which the user communicates with the operating system.

(a) command interpreter (c) status information

(b) system call (d) all

(ii) The operating system provide services of …………..

(a) Error detection (c) file management

(b) protection (d) all

(iii) Following is not a file manipulation system call.

(a) create (c) get time or date

(b) read (d) set attributes

(iv) To execute compiler ………………….is provided.

(a)operating system (c) communication

(b) system program (d) command interpreter

Ans. (i) a, (ii) d, (iii)c, (iv)b, a.

State true or false :

(i) Command interpreter is used when user first log in.

(ii) System calls are provided by system program.

(iii) An operating system can terminates the program abnormally.

Ans. (i) True, (ii) False, (iii) True.

***