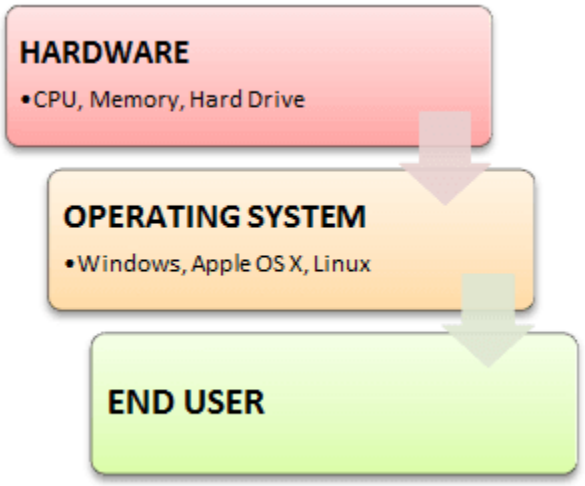
What is an Operating System?

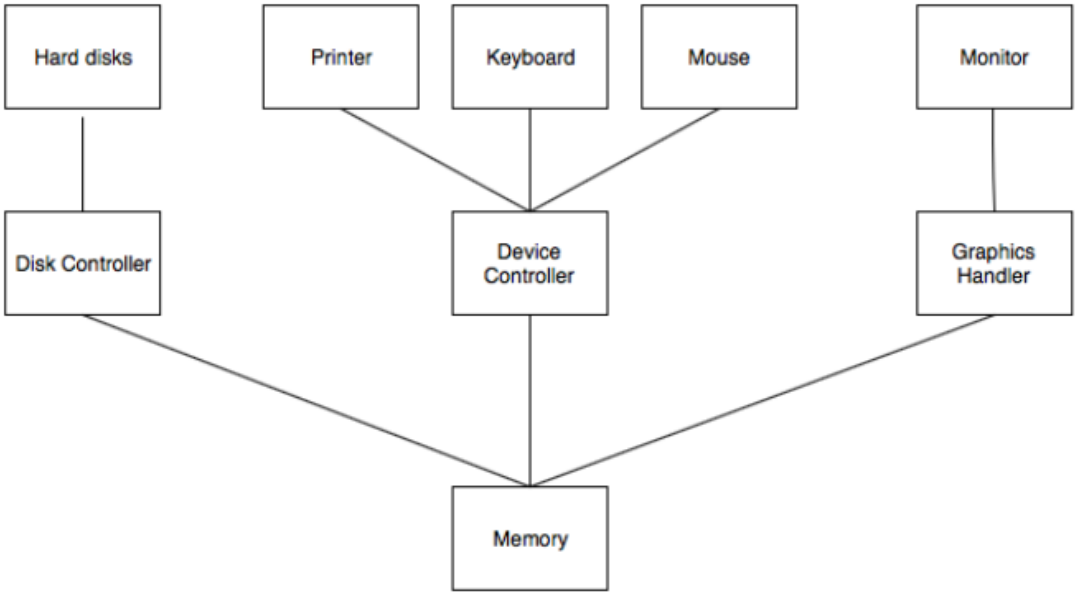
An **Operating System (OS)** is a software that acts as an interface between computer hardware components and the user. Every computer system must have at least one operating system to run other programs. Applications like Browsers, MS Office, Notepad Games, etc., need some environment to run and perform its tasks.
The OS helps you to communicate with the computer without knowing how to speak the computer's language. It is not possible for the user to use any computer or mobile device without having an operating system.



History Of OS

- Operating systems were first developed in the late 1950s to manage tape storage
- The General Motors Research Lab implemented the first OS in the early 1950s for their IBM 701
- In the mid-1960s, operating systems started to use disks
- In the late 1960s, the first version of the Unix OS was developed
- The first OS built by Microsoft was DOS. It was built in 1981 by purchasing the 86-DOS software from a Seattle company
- The present-day popular OS Windows first came to existence in 1985 when a GUI was created and paired with MS-DOS.

Computer System Organization



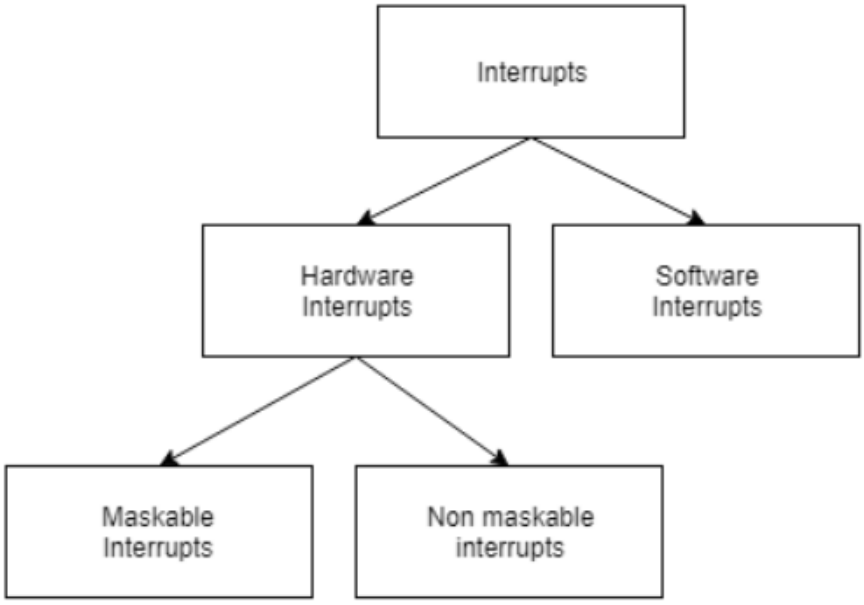
The salient points about the above figure displaying Computer System Organisation is −

- The I/O devices and the CPU both execute concurrently. Some of the processes are scheduled for the CPU and at the same time, some are undergoing input/output operations.
- There are multiple device controllers, each in charge of a particular device such as keyboard, mouse, printer etc.
- There is buffer available for each of the devices. The input and output data can be stored in these buffers.
- The data is moved from memory to the respective device buffers by the CPU for I/O operations and then this data is moved back from the buffers to memory.
- The device controllers use an interrupt to inform the CPU that I/O operation is completed.

Interrupt Handling

An interrupt is a necessary part of Computer System Organisation as it is triggered by hardware and software parts when they need immediate attention.

An interrupt can be generated by a device or a program to inform the operating system to halt its current activities and focus on something else. The types of interrupts are better explained using the following diagram −

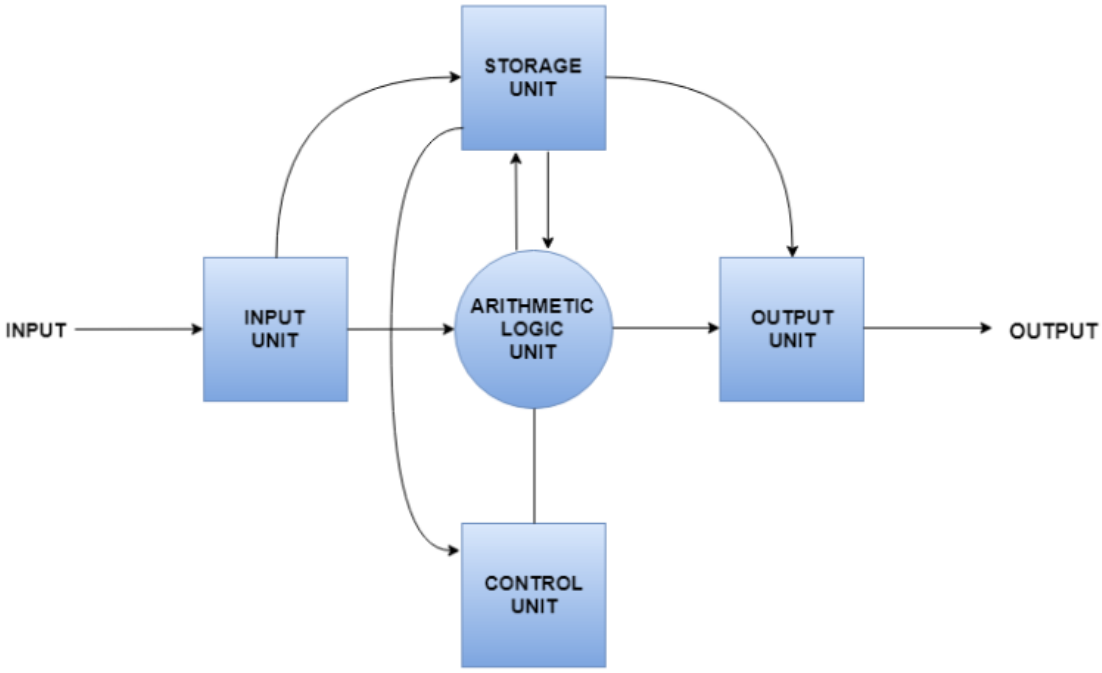


Hardware and software interrupts are two types of interrupts. Hardware interrupts are triggered by hardware peripherals while software interrupts are triggered by software function calls.

Hardware interrupts are of further two types. Maskable interrupts can be ignored or disabled by the CPU while this is not possible for non maskable interrupts.

Computer System Architecture

A computer system is basically a machine that simplifies complicated tasks. It should maximize performance and reduce costs as well as power consumption.The different components in the Computer System Architecture are Input Unit, Output Unit, Storage Unit, Arithmetic Logic Unit, Control Unit etc.



The input data travels from input unit to ALU. Similarly, the computed data travels from ALU to output unit. The data constantly moves from storage unit to ALU and back again. This is because stored data is computed on before being stored again. The control unit controls all the other units as well as their data.

Details about all the computer units are −

- **Input Unit**
  The input unit provides data to the computer system from the outside. So, basically it links the external environment with the computer. It takes data from the input devices, converts it into machine language and then loads it into the computer system. Keyboard, mouse etc. are the most commonly used input devices.
- **Output Unit**
  The output unit provides the results of computer process to the users i.e it links the computer with the external environment. Most of the output data is the form of audio or video. The different output devices are monitors, printers, speakers, headphones etc.
- **Storage Unit**
  Storage unit contains many computer components that are used to store data. It is traditionally divided into primary storage and secondary storage.Primary storage is also known as the main memory and is the memory directly accessible by the CPU. Secondary or external storage is not directly accessible by the CPU. The data from secondary storage needs to be brought into the primary storage before the CPU can use it. Secondary storage contains a large amount of data permanently.

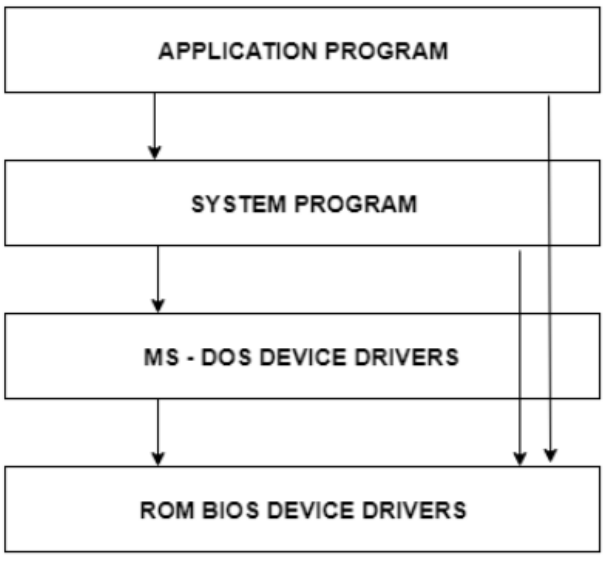
- **Arithmetic Logic Unit**
  All the calculations related to the computer system are performed by the arithmetic logic unit. It can perform operations like addition, subtraction, multiplication, division etc. The control unit transfers data from storage unit to arithmetic logic unit when calculations need to be performed. The arithmetic logic unit and the control unit together form the central processing unit.
- **Control Unit**
  This unit controls all the other units of the computer system and so is known as its central nervous system. It transfers data throughout the computer as required including from storage unit to central processing unit and vice versa. The control unit also dictates how the memory, input output devices, arithmetic logic unit etc. should behave.

Operating System Structure

An operating system is a construct that allows the user application programs to interact with the system hardware. Since the operating system is such a complex structure, it should be created with utmost care so it can be used and modified easily. An easy way to do this is to create the operating system in parts. Each of these parts should be well defined with clear inputs, outputs and functions.

Simple Structure

There are many operating systems that have a rather simple structure. These started as small systems and rapidly expanded much further than their scope. A common example of this is MS-DOS. It was designed simply for a niche amount for people. There was no indication that it would become so popular.



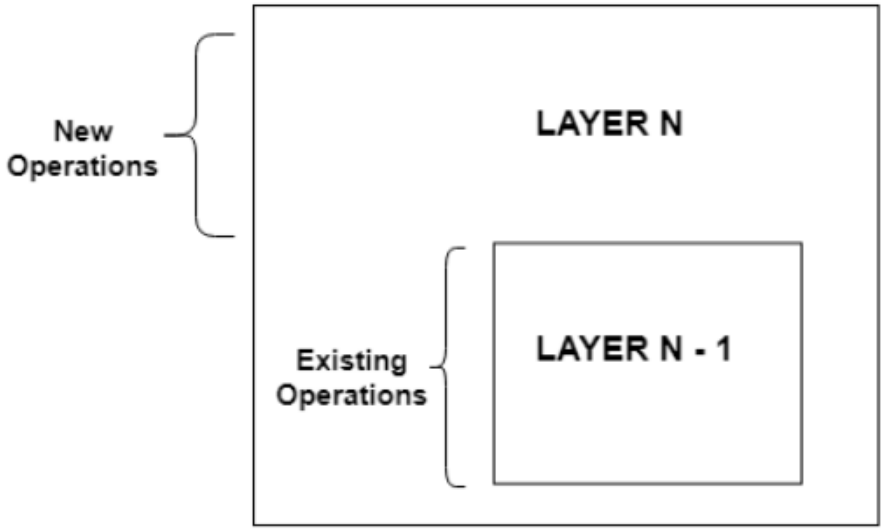
MS-DOS STRUCTURE

It is better that operating systems have a modular structure, unlike MS-DOS. That would lead to greater control over the computer system and its various applications. The modular structure would also allow the programmers to hide information as required and implement internal routines as they see fit without changing the outer specifications.

Layered Structure

One way to achieve modularity in the operating system is the layered approach. In this, the bottom layer is the hardware and the topmost layer is the user interface.

An image demonstrating the layered approach is as follows



Layered Structure of Operating System

As seen from the image, each upper layer is built on the bottom layer. All the layers hide some structures, operations etc from their upper layers.

One problem with the layered structure is that each layer needs to be carefully defined. This is necessary because the upper layers can only use the functionalities of the layers below them.

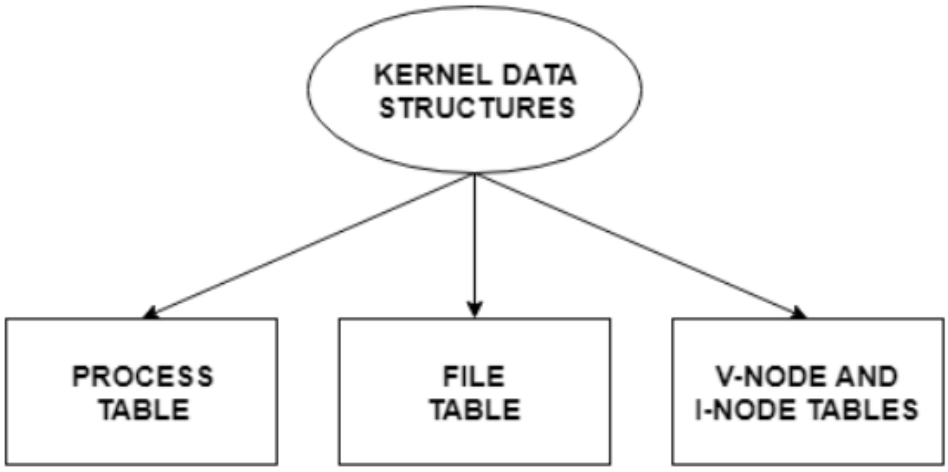
Kernel Data Structures

The kernel data structures are very important as they store data about the current state of the system. For example, if a new process is created in the system, a kernel data structure is created that contains the details about the process.

Most of the kernel data structures are only accessible by the kernel and its subsystems. They may contain data as well as pointers to other data structures.

Kernel Components

The kernel stores and organizes a lot of information. So it has data about which processes are running in the system, their memory requirements, files in use etc. To handle all this, three important structures are used. These are process table, file table and v node/ i node information.



Process Table

The process table stores information about all the processes running in the system. These include the storage information, execution status, file information etc.

When a process forks a child, its entry in the process table is duplicated including the file information and file pointers. So the parent and the child process share a file.

File Table

The file table contains entries about all the files in the system. If two or more processes use the same file, then they contain the same file information and the file descriptor number.

Each file table entry contains information about the file such as file status (file read or file write), file offset etc. The file offset specifies the position for next read or write into the file.

The file table also contains v-node and i-node pointers which point to the virtual node and index node respectively. These nodes contain information on how to read a file.

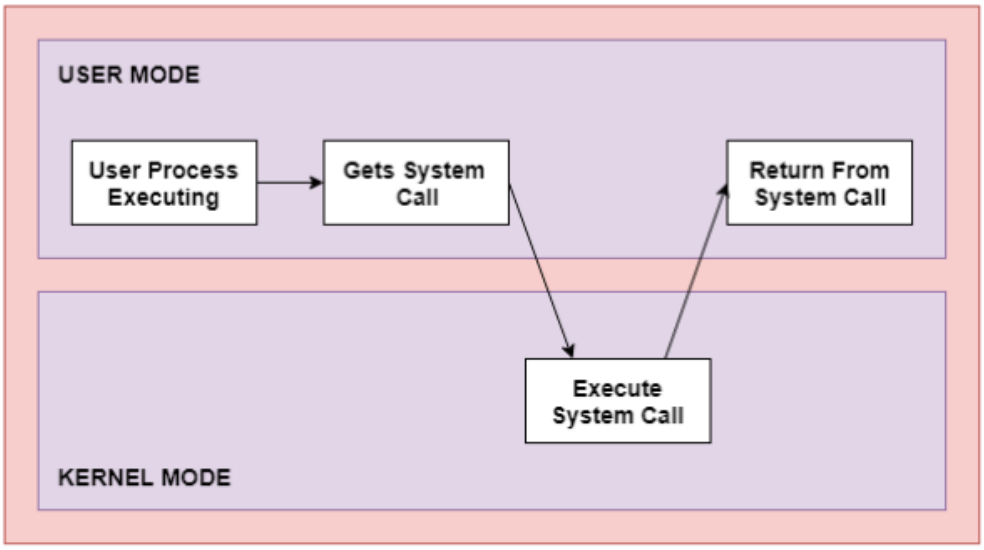
V-Node and I-Node Tables

Both the v-node and i-node are references to the storage system of the file and the storage mechanisms. They connect the hardware to the software.

The v-node is an abstract concept that defines the method to access file data without worrying about the actual structure of the system. The i-node specifies file access information like file storage device, read/write procedures etc.

What are system calls in Operating System?

The interface between a process and an operating system is provided by system calls. In general, system calls are available as assembly language instructions. They are also included in the manuals used by the assembly level programmers. System calls are usually made when a process in user mode requires access to a resource. Then it requests the kernel to provide the resource via a system call.



As can be seen from this diagram, the processes execute normally in the user mode until a system call interrupts this. Then the system call is executed on a priority basis in the kernel mode. After the execution of the system call, the control returns to the user mode and execution of user processes can be resumed.

In general, system calls are required in the following situations −

- If a file system requires the creation or deletion of files. Reading and writing from files also require a system call.
- Creation and management of new processes.

- Network connections also require system calls. This includes sending and receiving packets.
- Access to a hardware devices such as a printer, scanner etc. requires a system call.

Types of System Calls

There are mainly five types of system calls. These are explained in detail as follows −

Process Control

These system calls deal with processes such as process creation, process termination etc.

File Management

These system calls are responsible for file manipulation such as creating a file, reading a file, writing into a file etc.

Device Management

These system calls are responsible for device manipulation such as reading from device buffers, writing into device buffers etc.

Information Maintenance

These system calls handle information and its transfer between the operating system and the user program.

Communication

These system calls are useful for interprocess communication. They also deal with creating and deleting a communication connection.

Some of the examples of all the above types of system calls in Windows and Unix are given as follows −

Types of System Calls	**Windows**
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()
File Management	CreateFile() ReadFile() WriteFile() CloseHandle()
Device Management	SetConsoleMode() ReadConsole() WriteConsole()

Types of System Calls	Windows
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()

There are many different system calls as shown above. Details of some of those system calls are as follows −

open()

The open() system call is used to provide access to a file in a file system. This system call allocates resources to the file and provides a handle that the process uses to refer to the file. A file can be opened by multiple processes at the same time or be restricted to one process. It all depends on the file organisation and file system.

read()

The read() system call is used to access data from a file that is stored in the file system. The file to read can be identified by its file descriptor and it should be opened using open() before it can be read. In general, the read() system calls takes three arguments i.e. the file descriptor, buffer which stores read data and number of bytes to be read from the file.

write()

The write() system calls writes the data from a user buffer into a device such as a file. This system call is one of the ways to output data from a program. In general, the write system calls takes three arguments i.e. file descriptor, pointer to the buffer where data is stored and number of bytes to write from the buffer.

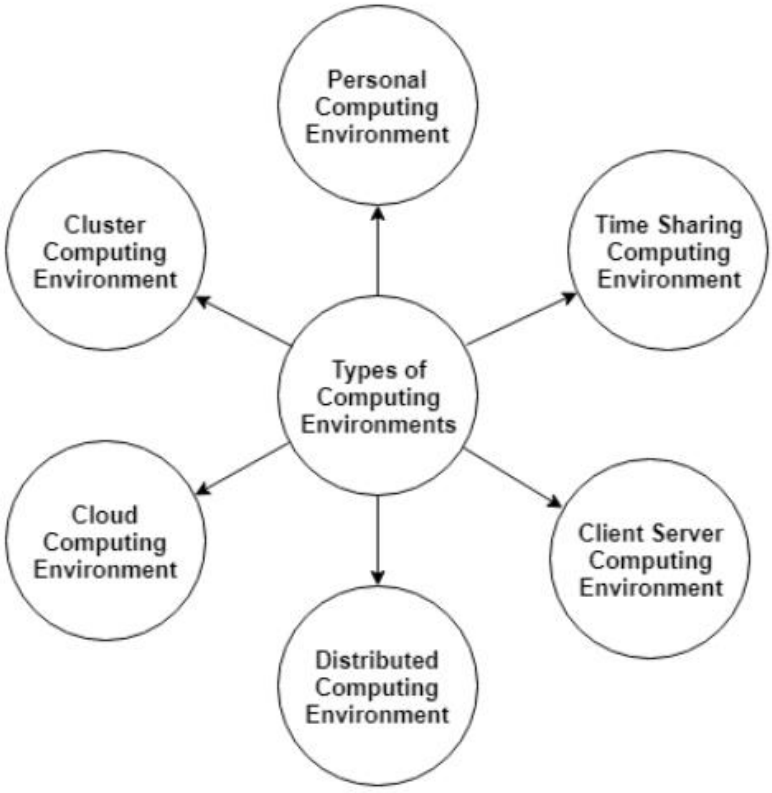
close()

The close() system call is used to terminate access to a file system. Using this system call means that the file is no longer required by the program and so the buffers are flushed, the file metadata is updated and the file resources are de-allocated.

Types of Computing Environments

A computer system uses many devices, arranged in different ways to solve many problems. This constitutes a computing environment where many computers are used to process and exchange information to handle multiple issues.

The different types of Computing Environments are −



Personal Computing Environment

In the personal computing environment, there is a single computer system. All the system processes are available on the computer and executed there. The different devices that constitute a personal computing environment are laptops, mobiles, printers, computer systems, scanners etc.

Time Sharing Computing Environment

The time sharing computing environment allows multiple users to share the system simultaneously. Each user is provided a time slice and the processor switches rapidly among the users according to it. Because of this, each user believes that they are the only ones using the system.

Client Server Computing Environment

In client server computing, the client requests a resource and the server provides that resource. A server may serve multiple clients at the same time while a client is in contact with only one server. Both the client and server usually communicate via a computer network but sometimes they may reside in the same system.

Distributed Computing Environment

A distributed computing environment contains multiple nodes that are physically separate but linked together using the network. All the nodes in this system communicate with each other and handle processes in tandem. Each of these nodes contains a small part of the distributed operating system software.

Cloud Computing Environment

The computing is moved away from individual computer systems to a cloud of computers in cloud computing environment. The cloud users only see the service being provided and not the internal details of how the service is provided. This is done by pooling all the computer resources and then managing them using a software.

Cluster Computing Environment

The clustered computing environment is similar to parallel computing environment as they both have multiple CPUs. However a major difference is that clustered systems are created by two or more individual computer systems merged together which then work parallel to each other.

Open Source Operating Systems

Open Source operating systems are released under a license where the copyright holder allows others to study, change as well as distribute the software to other people. This can be done for any reason. The different open source operating system available in the market are −

Cosmos

This is an open source operating system written mostly in programming language C#. Its full form is C# Open Source Managed Operating System. Till 2016, Cosmos did not intend to be a fully

fledged operating system but a system that allowed other developers to easily build their own operating systems. It also hid the inner workings of the hardware from the developers thus providing data abstraction.

FreeDOS

This was a free operating system developed for systems compatible with IBM PC computers. FreeDOS provides a complete environment to run legacy software and other embedded systems. It can booted from a floppy disk or USB flash drive as required. FreeDos is licensed under the GNU General Public license and contains free and open source software. So there is no license fees required for its distribution and changes to the system are permitted.

Genode

Genode is free as well as open source. It contains a microkernel layer and different user components. It is one of the few open source operating systems not derived from a licenced operating system such as Unix. Genode can be used as an operating system for computers, tablets etc. as required. It is also used as a base for virtualisation, interprocess communication, software development etc. as it has a small code system.

Ghost OS

This is a free, open source operating system developed for personal computers. It started as a research project and developed to contain various advanced features like graphical user interface, C library etc. The Ghost operating system features multiprocessing and multitasking and is based on the Ghost Kernel. Most of the programming in Ghost OS is done in C++.

ITS

The incompatible time-sharing system was developed by the MIT Artificial Intelligence Library. It is principally a time sharing system. There is a remote login facility which allowed guest users to informally try out the operating system and its features using ARPAnet. ITS also gave out many new features that were unique at that time such as device independent graphics terminal, virtual devices, inter machine file system access etc.

OSv

This was an operating system released in 2013. It was mainly focused on cloud computing and was built to run on top of a virtual machine as a guest. This is the reason it doesn't include drivers for bare hardware. In the OSv operating system, everything runs in the kernel address space and there is no concept of a multi-user system.

Phantom OS

This is an operating system that is based on the concepts on persistent virtual memory and is code oriented. It was mostly developed by Russian developers. Phantom OS is not based on concepts

of famous operating systems such as Unix. Its main goal is simplicity and effectiveness in process management.