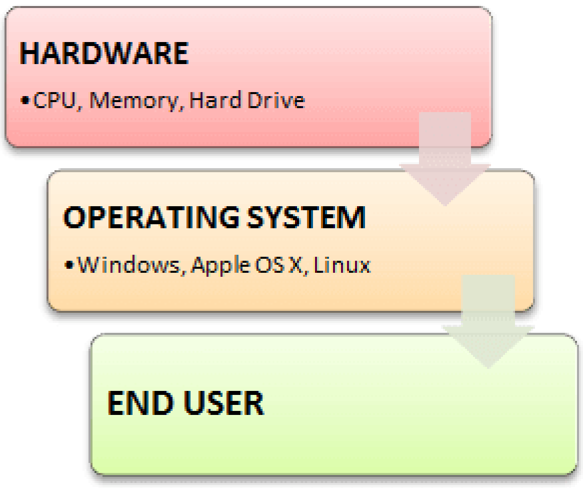
Unit -1

Introduction

What is an Operating System?

An **Operating System (OS)** is 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.



Introduction to Operating System

What is Operating System?

An Operating System (OS) is an interface between computer user and computer hardware. An operating system is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.

Generally, a **Computer System** consists of the following components:

- **Computer Users** are the users who use the overall computer system.
- **Application Softwares** are the softwares which users use directly to perform different activities. These softwares are simple and easy to use like Browsers, Word, Excel, different Editors, Games etc. These are usually written in high-level languages, such as Python, Java and C++.
- **System Softwares** are the softwares which are more complex in nature and they are more near to computer hardware. These software are usually written in low-

level languages like assembly language and includes **Operating Systems** (Microsoft Windows, macOS, and Linux), Compiler, and Assembler etc.
- **Computer Hardware** includes Monitor, Keyboard, CPU, Disks, Memory, etc.

Operating System - Examples

There are plenty of Operating Systems available in the market which include paid and unpaid (Open Source). Following are the examples of the few most popular Operating Systems:

- **Windows:** This is one of the most popular and commercial operating systems developed and marketed by Microsoft. It has different versions in the market like Windows 8, Windows 10 etc and most of them are paid.
- **Linux** This is a Unix based and the most loved operating system first released on September 17, 1991 by Linus Torvalds. Today, it has 30+ variants available like Fedora, OpenSUSE, CentOS, UBuntu etc. Most of them are available free of charges though you can have their enterprise versions by paying a nominal license fee.
- **MacOS** This is again a kind of Unix operating system developed and marketed by Apple Inc. since 2001.
- **iOS** This is a mobile operating system created and developed by Apple Inc. exclusively for its mobile devices like iPhone and iPad etc.
- **Android** This is a mobile Operating System based on a modified version of the Linux kernel and other open source software, designed primarily for touchscreen mobile devices such as smartphones and tablets.

Some other old but popular Operating Systems include Solaris, VMS, OS/400, AIX, z/OS, etc.

Introduction of Operating System

An operating system acts as an intermediary between the user of a computer and computer hardware. The purpose of an operating system is to provide an environment in which a user can execute programs conveniently and efficiently.

An operating system is software that manages computer hardware. The hardware must provide appropriate mechanisms to ensure the correct operation of the computer system and to prevent user programs from interfering with the proper operation of the system.

Operating System: Definition:

- An operating system is a program that controls the execution of application programs and acts as an interface between the user of a computer and the computer hardware.
- A more common definition is that the operating system is the one program running at all times on the computer (usually called the kernel), with all else being application programs.
- An operating system is concerned with the allocation of resources and services, such as memory, processors, devices, and information. The operating system correspondingly includes programs to manage these

resources, such as a traffic controller, a scheduler, a memory management module, I/O programs, and a file system.

Features of Operating system: Operating system has the following features:

1. **Convenience:** An OS makes a computer more convenient to use.
2. **Efficiency:** An OS allows the computer system resources to be used efficiently.
3. **Ability to Evolve:** An OS should be constructed in such a way as to permit the effective development, testing, and introduction of new system functions at the same time without interfering with service.
4. **Throughput:** An OS should be constructed so that It can give maximum **throughput**(Number of tasks per unit time).

Major Functionalities of Operating System:

- **Resource Management:** When parallel accessing happens in the OS means when multiple users are accessing the system the OS works as Resource Manager, Its responsibility is to provide hardware to the user. It decreases the load in the system.
- **Process Management:** It includes various tasks like **scheduling and termination** of the process. It is done with the help of **CPU Scheduling** algorithms.
- **Storage Management:** The **file system** mechanism used for the management of the storage. **NIFS**, **CFS**, **CIFS**, **NFS**, etc. are some file systems. All the data is stored in various tracks of Hard disks that are all managed by the storage manager. It included **Hard Disk**.
- **Memory Management:** Refers to the management of primary memory. The operating system has to keep track of how much memory has been used and by whom. It has to decide which process needs memory space and how much. OS also has to allocate and deallocate the memory space.
- **Security/Privacy Management:** Privacy is also provided by the Operating system by means of passwords so that unauthorized applications can’t access programs or data. For example, Windows uses **Kerberos** authentication to prevent unauthorized access to data.

The process operating system as User Interface:

1. User
2. System and application programs
3. Operating system
4. Hardware

Every general-purpose computer consists of hardware, an operating system(s), system programs, and application programs. The hardware consists of memory, CPU, ALU, I/O devices, peripheral devices, and storage devices. The system program consists of compilers, loaders, editors, OS, etc. The application program consists of business programs and database programs.

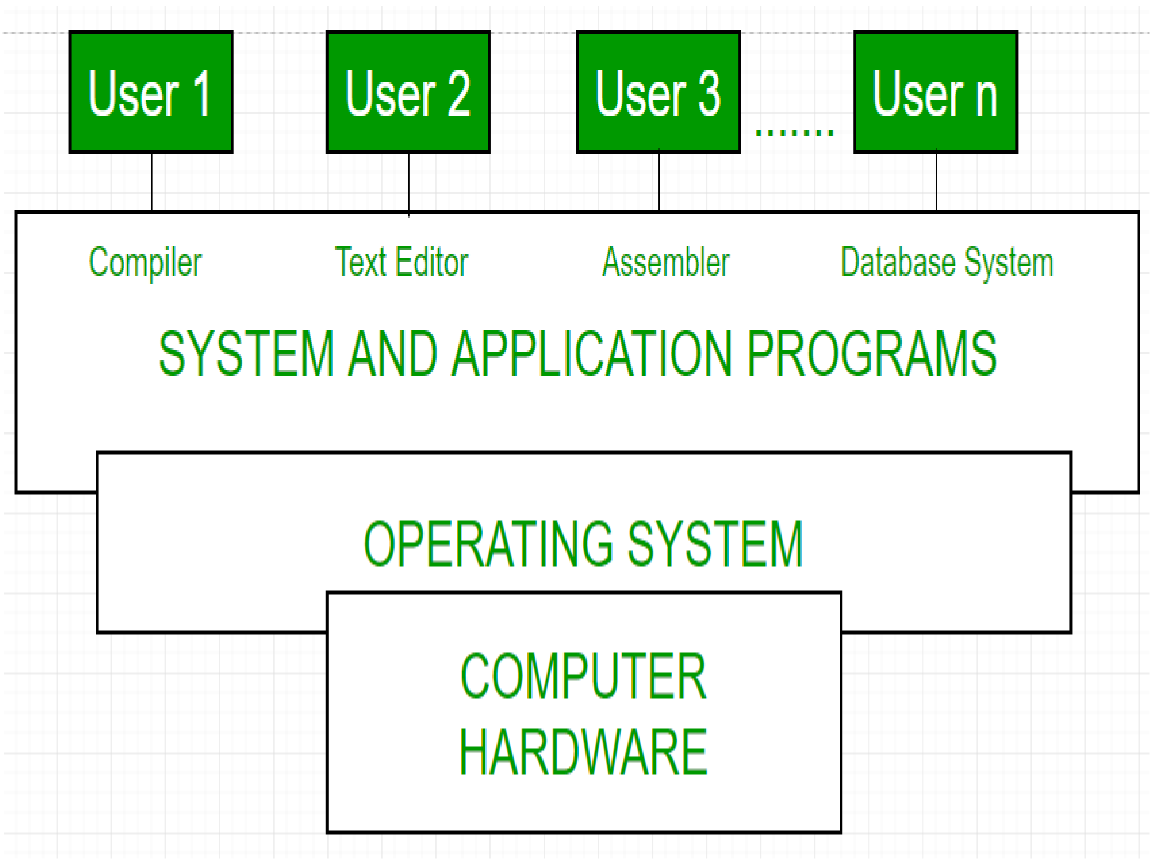


Fig1: Conceptual view of a computer system

Every computer must have an operating system to run other programs. The operating system coordinates the use of the hardware among the various system programs and application programs for various users. It simply provides an environment within which other programs can do useful work.

The operating system is a set of special programs that run on a computer system that allows it to work properly. It performs basic tasks such as recognizing input from the keyboard, keeping track of files and directories on the disk, sending output to the display screen, and controlling peripheral devices.

OS is designed to serve two basic purposes:

1. It controls the allocation and use of the computing System's resources among the various user and tasks.
2. It provides an interface between the computer hardware and the programmer that simplifies and makes it feasible for coding and debugging of application programs.

The Operating system must support the following tasks. The tasks are:

1. Provides the facilities to create and modify of programs and data files using an editor.
2. Access to the compiler for translating the user program from high-level language to machine language.

3. Provide a loader program to move the compiled program code to the computer's memory for execution.
4. Provide routines that handle the details of I/O programming.

I/O System Management: The module that keeps track of the status of devices is called the I/O traffic controller. Each I/O device has a device handler that resides in a separate process associated with that device. The I/O subsystem consists of

- A memory Management component that includes buffering caching and spooling.
- A general device driver interface.

Drivers for specific hardware devices.

Assembler: The input to an assembler is an assembly language program. The output is an object program plus information that enables the loader to prepare the object program for execution. At one time, the computer programmer had at his disposal a basic machine that interpreted, through hardware, certain fundamental instructions. He would program this computer by writing a series of ones and Zeros (Machine language), and place them into the memory of the machine. Examples of assembly languages include

Compiler and Interpreter: The High-level languages- examples are C, C++, Java, Python etc (around 300+ famous high level languages) are processed by compilers and interpreters. A compiler is a program that accepts a source program in a "high-level language "and produces machine code in one go. Some of the compiled languages are FORTRAN, COBOL, C, C++ , Rust and Go. An interpreter is a program that does the same thing but converts high-level code to machine code line-by-line and not all at once. Example of interpreted languages are Python, Perl and Ruby.

Loader: A Loader is a routine that loads an object program and prepares it for execution. There are various loading schemes: absolute, relocating, and direct-linking. In general, the loader must load, relocate and link the object program. The loader is a program that places programs into memory and prepares them for execution. In a simple loading scheme, the assembler outputs the machine language translation of a program on a secondary device and a loader places it in the core. The loader places into memory the machine language version of the user's program and transfers control to it. Since the loader program is much smaller than the assembler, those make more core available to the user's program.

History of Operating system: The operating system has been evolving through the years. The following table shows the history of OS.

Generation	Year	Electronic device used	Types of OS Devices
First	1945-55	Vacuum Tubes	Plug Boards
Second	1955-65	Transistors	Batch Systems

Third	**1965-80**	**Integrated Circuits(IC)**	**Multiprogramming**
Fourth	**Since 1980**	**Large Scale Integration**	**PC**

Types of Operating Systems

- Batch Operating System- Sequence of jobs in a program on a computer without manual interventions.
- Time-sharing operating System- allows many users to share computer resources. (Max utilization of the resources).
- Distributed operating System- Manages a group of different computers and makes appear to be a single computer.
- Network operating system- computers running in different operating systems can participate in a common network (It is used for security purposes).
- Real-time operating system – used where any or all the jobs have strict time constraints.

Features of operating systems:

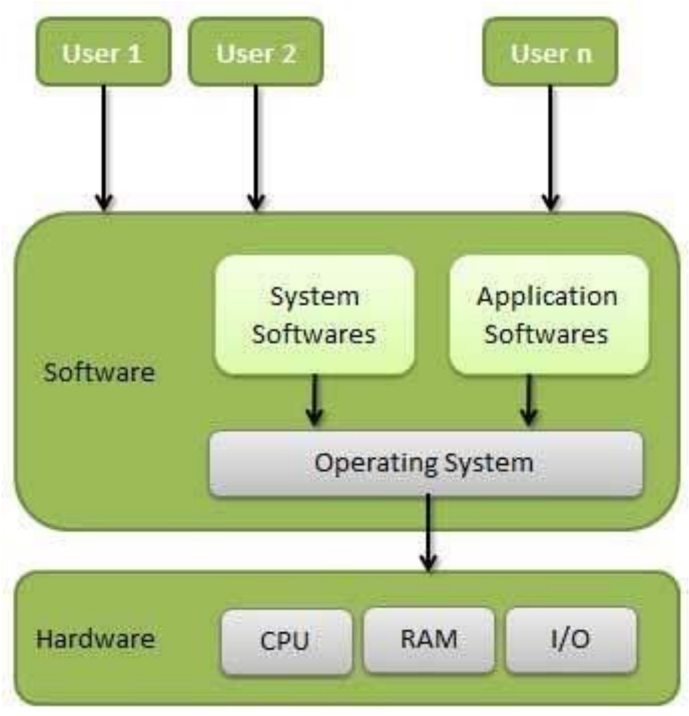
Memory Management: The operating system manages the computer's memory, ensuring that programs have the necessary memory to run and allocating memory efficiently to maximize performance.
Process Management: The operating system is responsible for managing the processes running on the computer. It schedules processes to run on the CPU, allocates system resources to processes, and terminates processes when they are no longer needed.
File System Management: The operating system manages the file system, which is responsible for storing and retrieving files on the computer's hard disk or other storage devices.
Device Management: The operating system manages the computer's input and output devices, such as the keyboard, mouse, and printer, ensuring that they are properly configured and functioning correctly.
Security: The operating system provides security features to protect the computer from unauthorized access, viruses, and other types of malware.
User Interface: The operating system provides a graphical user interface (GUI) or a command-line interface (CLI) to interact with the computer, making it easier for users to access and use the computer's resources.
Networking: The operating system provides networking features that enable computers to communicate with each other over a network, allowing users to share resources and collaborate with others.
Examples of Operating Systems are as follows:

- Windows (GUI-based, PC)
- GNU/Linux (Personal, Workstations, ISP, File, and print server, Three-tier client/Server)

- macOS (Macintosh), used for Apple’s personal computers and workstations (MacBook, iMac).
- Android (Google’s Operating System for smartphones/tablets/smartwatches)
- iOS (Apple’s OS for iPhone, iPad, and iPod Touch)

Architecture

We can draw a generic architecture diagram of an Operating System which is as follows:



Operating system and functions:

What is an Operating System?
An operating system is a program on which application programs are executed and acts as a communication bridge (interface) between the user and the computer hardware.
The main task an operating system carries out is the allocation of resources and services, such as the allocation of memory, devices, processors, and information. The operating system also includes programs to manage these resources, such as a traffic controller, a scheduler, memory management module, I/O programs, and a file system.

- It controls all of computer resources.
- It provides valuable services to user programs.
- It coordinates the execution of user programs.
- It provides resources of user programs.

- It provides an interface(virtual machine) to the user.
- It hides the complexity of software.
- It supports the multiple execution modes.
- It monitors the execution of user programs to prevent errors.

Important functions of an operating System:

1. **Security –**
   The operating system uses password protection to protect user data and similar other techniques. it also prevents unauthorized access to programs and user data.

2. **Control over system performance –**
   Monitors overall system health to help improve performance. records the response time between service requests and system response to having a complete view of the system health. This can help improve performance by providing important information needed to troubleshoot problems.

3. **Job accounting –**
   Operating system Keeps track of time and resources used by various tasks and users, this information can be used to track resource usage for a particular user or group of users.

4. **Error detecting aids –**
   The operating system constantly monitors the system to detect errors and avoid the malfunctioning of a computer system.

5. **Coordination between other software and users –**
   Operating systems also coordinate and assign interpreters, compilers, assemblers, and other software to the various users of the computer systems.

6. **Memory Management –**
   The operating system manages the Primary Memory or Main Memory. Main memory is made up of a large array of bytes or words where each byte or word is assigned a certain address. Main memory is fast storage and it can be accessed directly by the CPU. For a program to be executed, it should be first loaded in the main memory. An Operating System performs the following activities for memory management:
   It keeps track of primary memory, i.e., which bytes of memory are used by which user program. The memory addresses that have already been allocated and the memory addresses of the memory that has not yet been used. In multiprogramming, the OS decides the order in which processes are granted access to memory, and for how long. It Allocates the memory to a process when the process requests it and deallocates the memory when the process has terminated or is performing an I/O operation.

7. **Processor Management –**
   In a multi-programming environment, the OS decides the order in which processes have access to the processor, and how much processing time each process has. This function of OS is called process scheduling. An Operating System performs the following activities for processor management.
   Keeps track of the status of processes. The program which performs this task is known as a traffic controller. Allocates the CPU that is a processor to a process. De-allocates processor when a process is no more required.

8. **Device Management –**
   An OS manages device communication via their respective drivers. It performs the following activities for device management. Keeps track of all devices connected to the system. designates a program responsible for every device known as the Input/Output controller. Decides which process gets access to a certain device and for how long. Allocates devices in an effective and efficient way. Deallocates devices when they are no longer required.

9. **File Management –**
   A file system is organized into directories for efficient or easy navigation and usage. These directories may contain other directories and other files. An Operating System carries out the following file management activities. It keeps track of where information is stored, user access settings and status of every file, and more… These facilities are collectively known as the file system.

Moreover, Operating System also provides certain services to the computer system in one form or the other.
The Operating System provides certain services to the users which can be listed in the following manner:

1. **Program Execution**: The Operating System is responsible for the execution of all types of programs whether it be user programs or system programs. The Operating System utilizes various resources available for the efficient running of all types of functionalities.
2. **Handling Input/Output Operations**: The Operating System is responsible for handling all sorts of inputs, i.e, from the keyboard, mouse, desktop, etc. The Operating System does all interfacing in the most appropriate manner regarding all kinds of Inputs and Outputs.
   For example, there is a difference in the nature of all types of peripheral devices such as mice or keyboards, the Operating System is responsible for handling data between them.
3. **Manipulation of File System**: The Operating System is responsible for making decisions regarding the storage of all types of data or files, i.e,

floppy disk/hard disk/pen drive, etc. The Operating System decides how the data should be manipulated and stored.

4. **Error Detection and Handling**: The Operating System is responsible for the detection of any type of error or bugs that can occur while any task. The well-secured OS sometimes also acts as a countermeasure for preventing any sort of breach to the Computer System from any external source and probably handling them.
5. **Resource Allocation:** The Operating System ensures the proper use of all the resources available by deciding which resource to be used by whom for how much time. All the decisions are taken by the Operating System.
6. **Accounting:** The Operating System tracks an account of all the functionalities taking place in the computer system at a time. All the details such as the types of errors that occurred are recorded by the Operating System.
7. **Information and Resource Protection:** The Operating System is responsible for using all the information and resources available on the machine in the most protected way. The Operating System must foil an attempt from any external resource to hamper any sort of data or information.

All these services are ensured by the Operating System for the convenience of the users to make the programming task easier. All different kinds of Operating systems more or less provide the same services.

characteristics of operating systems:

- Virtualization: Operating systems can provide virtualization capabilities, allowing multiple operating systems or instances of an operating system to run on a single physical machine. This can improve resource utilization and provide isolation between different operating systems or applications.
- Networking: Operating systems provide networking capabilities, allowing the computer system to connect to other systems and devices over a network. This can include features such as network protocols, network interfaces, and network security.
- Scheduling: Operating systems provide scheduling algorithms that determine the order in which tasks are executed on the system. These algorithms prioritize tasks based on their resource requirements and other factors to optimize system performance.
- Interprocess communication: Operating systems provide mechanisms for applications to communicate with each other, allowing them to share data and coordinate their activities.
- Performance monitoring: Operating systems provide tools for monitoring system performance, including CPU usage, memory usage, disk usage, and network activity. This can help identify performance bottlenecks and optimize system performance.
- Backup and recovery: Operating systems provide backup and recovery mechanisms to protect data in the event of system failure or data loss.

- Debugging: Operating systems provide debugging tools that allow developers to identify and fix software bugs and other issues in the system.

Classification of Operating systems:

Operating systems can be classified as follows:

Multi-user: is the one that concede two or more users to use their programs at the same time. Some of O.S permits hundreds or even thousands of users simultaneously.

Single-User: just allows one user to use the programs at one time.

Multiprocessor: Supports opening the same program more than just in one CPU.

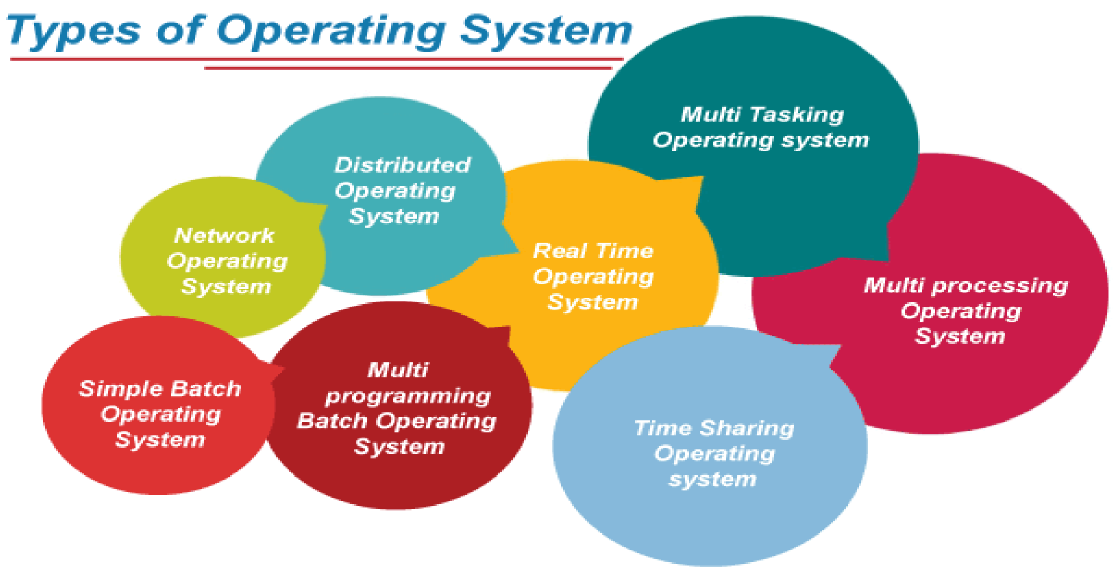
Multitasking: Allows multiple programs running at the same time.

Single-tasking: Allows different parts of a single program running at any one time.

Real time: Responds to input instantly. Operating systems such as DOS and UNIX, do not work in real time.

Types of Operating Systems (OS)

An operating system is a well-organized collection of programs that manages the computer hardware. It is a type of system software that is responsible for the smooth functioning of the computer system.

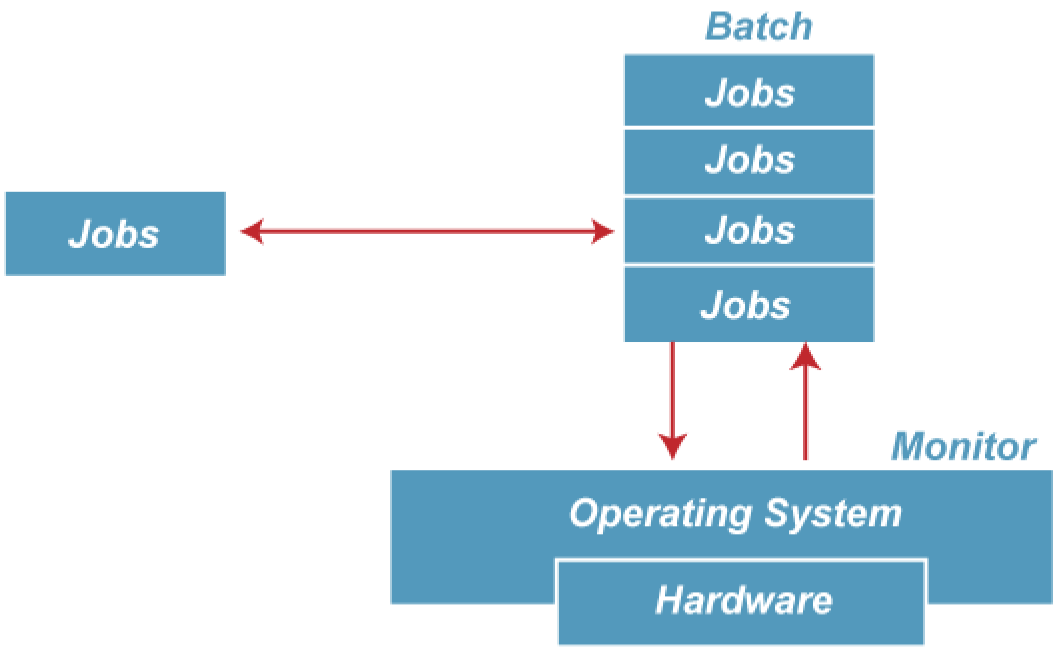

Types of Operating System
Multi Tasking Operating system
Distributed Operating System
Network Operating System
Real Time Operating System
Multi processing Operating System
Simple Batch Operating System
Multi programming Batch Operating System
Time Sharing Operating system

Batch Operating System

In the 1970s, Batch processing was very popular. In this technique, similar types of jobs were batched together and executed in time. People were used to having a single computer which was called a mainframe.

In Batch operating system, access is given to more than one person; they submit their respective jobs to the system for the execution.

The system put all of the jobs in a queue on the basis of first come first serve and then executes the jobs one by one. The users collect their respective output when all the jobs get executed.



The purpose of this operating system was mainly to transfer control from one job to another as soon as the job was completed. It contained a small set of programs called the resident monitor that always resided in one part of the main memory. The remaining part is used for servicing jobs.

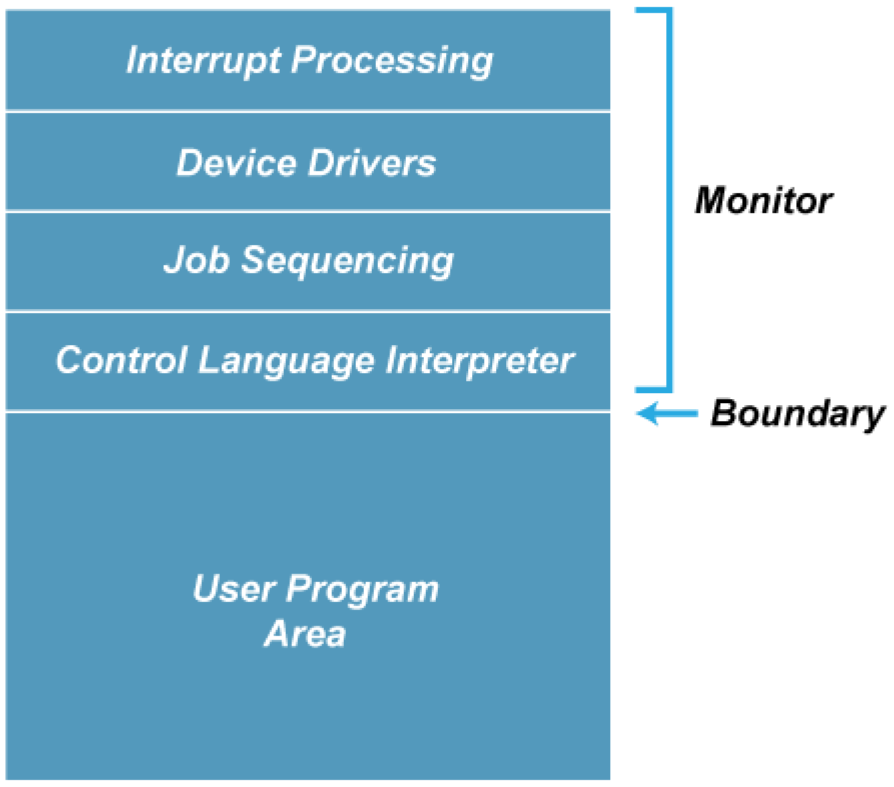


Figure: Memory Layout of the resident monitor

Advantages of Batch OS

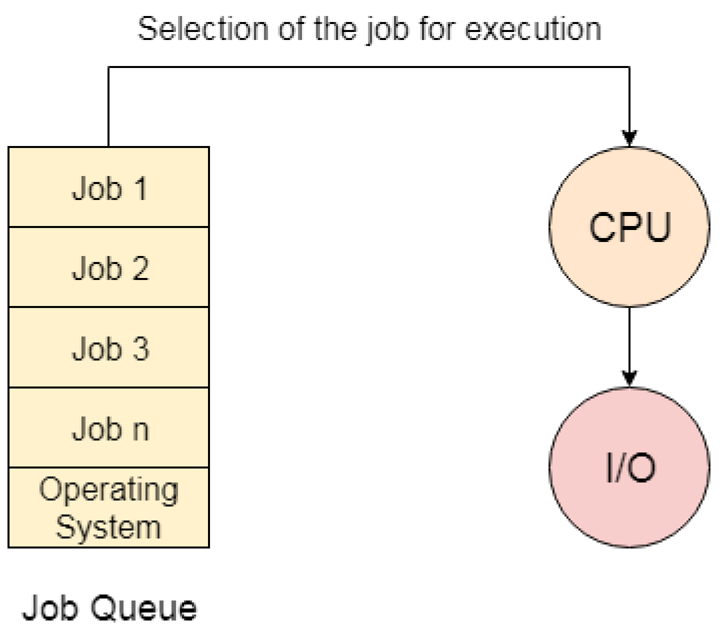
- The use of a resident monitor improves computer efficiency as it eliminates CPU time between two jobs.

Disadvantages of Batch OS

1. Starvation

Batch processing suffers from starvation.

For Example:



There are five jobs J1, J2, J3, J4, and J5, present in the batch. If the execution time of J1 is very high, then the other four jobs will never be executed, or they will have to wait for a very long time. Hence the other processes get starved.

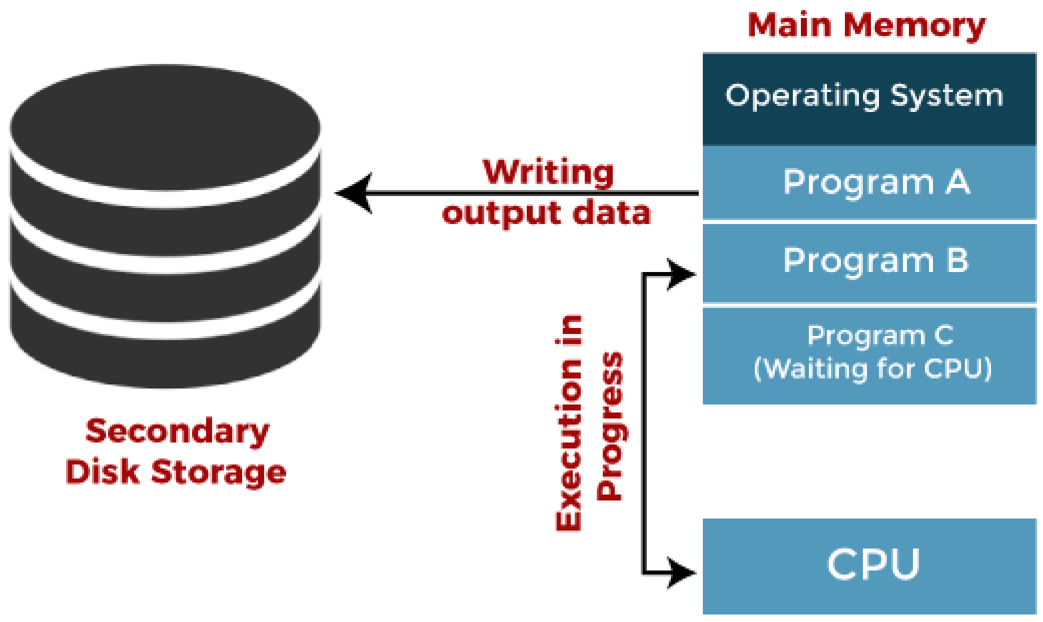
2. Not Interactive

Batch Processing is not suitable for jobs that are dependent on the user's input. If a job requires the input of two numbers from the console, then it will never get it in the batch processing scenario since the user is not present at the time of execution.

Multiprogramming Operating System

Multiprogramming is an extension to batch processing where the CPU is always kept busy. Each process needs two types of system time: CPU time and IO time.

In a multiprogramming environment, when a process does its I/O, The CPU can start the execution of other processes. Therefore, multiprogramming improves the efficiency of the system.



Jobs in multiprogramming system

Advantages of Multiprogramming OS

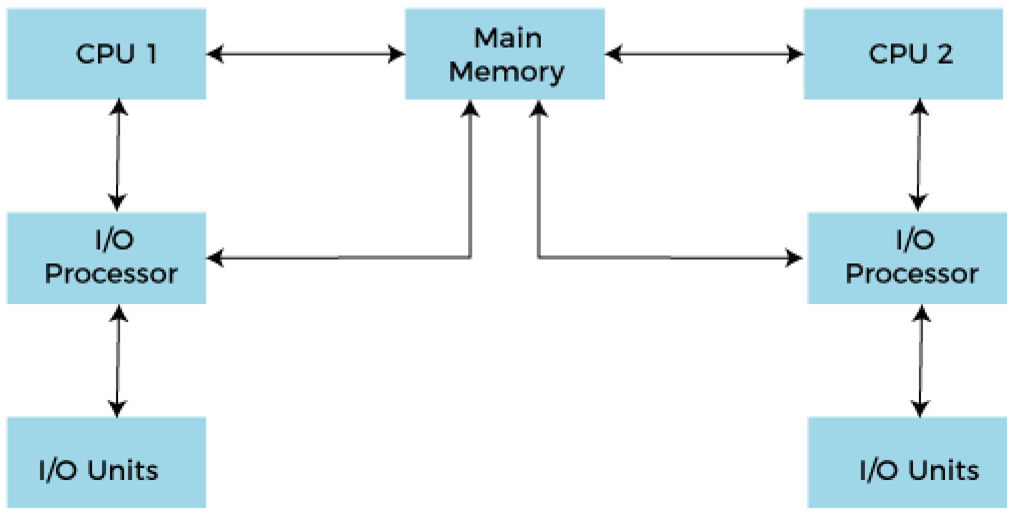
- Throughout the system, it increased as the CPU always had one program to execute.
- Response time can also be reduced.

Disadvantages of Multiprogramming OS

- Multiprogramming systems provide an environment in which various systems resources are used efficiently, but they do not provide any user interaction with the computer system.

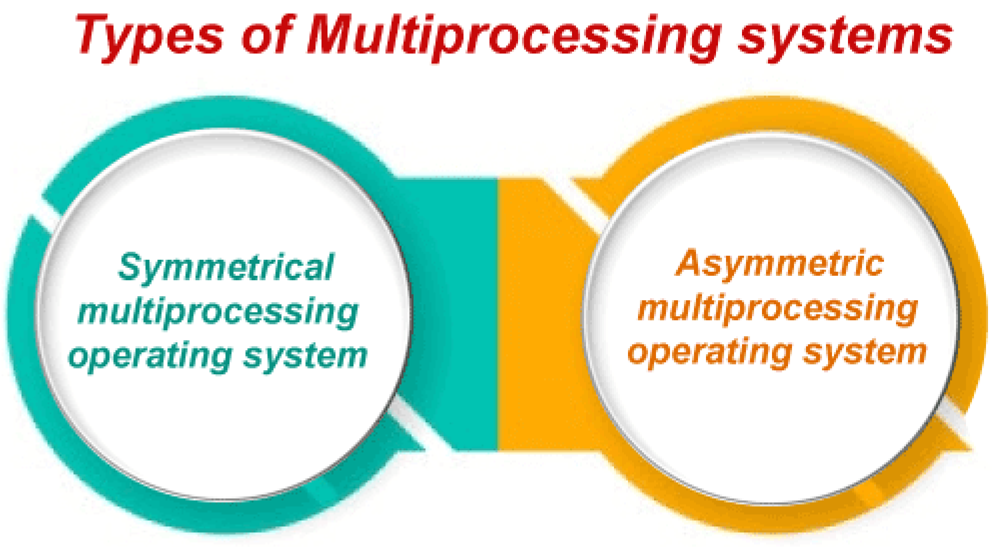
Multiprocessing Operating System

In Multiprocessing, Parallel computing is achieved. There are more than one processors present in the system which can execute more than one process at the same time. This will increase the throughput of the system.



Working of Multiprocessor System

In Multiprocessing, Parallel computing is achieved. More than one processor present in the system can execute more than one process simultaneously, which will increase the throughput of the system.


Types of Multiprocessing systems
Symmetrical multiprocessing operating system
Asymmetric multiprocessing operating system

Advantages of Multiprocessing operating system:

- **Increased reliability:** Due to the multiprocessing system, processing tasks can be distributed among several processors. This increases reliability as if one processor fails, the task can be given to another processor for completion.
- **Increased throughout:** As several processors increase, more work can be done in less.

Disadvantages of Multiprocessing operating System

Batch Operating System

In the beginning, computers were extremely large machines that ran from a console. In general, tape drivers or card readers were used for input, and tape drives, punch cards, and line printers were used for output. Users had no direct interface with the system, and job execution was done in a batch system. These systems are known as batched operating systems, and users have to prepare a job separately to execute it.

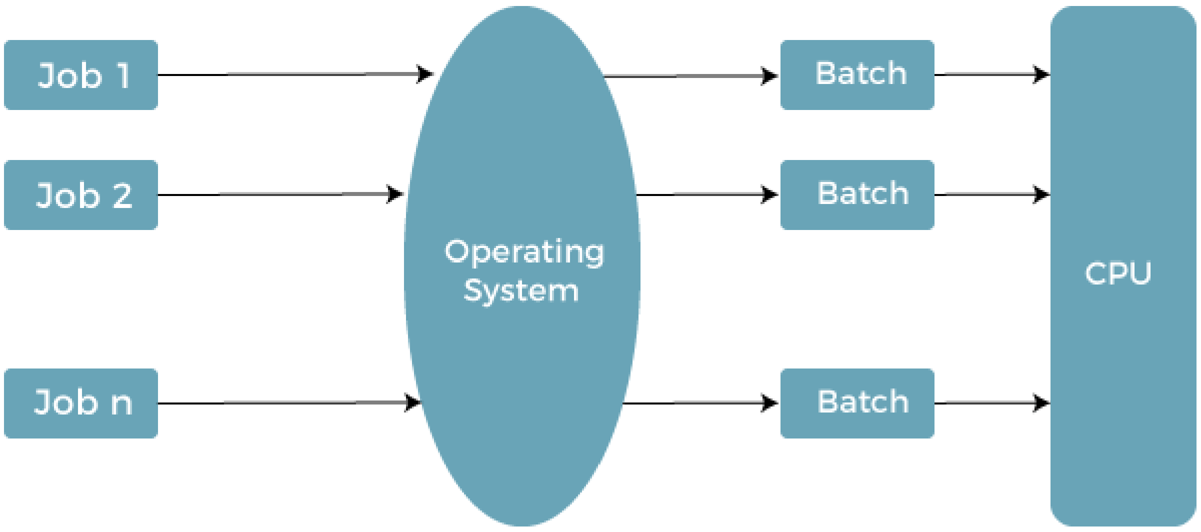
There were no developed operating systems, complex computer architecture, or secondary memory devices in the **1950s** and **1960s**. Instead, large mainframe computers were used to process data, with punched cards or magnetic tapes serving as input and output. The primary problem at the time was a lack of hard disks.

In the early **1950s, General Motors Research Laboratories (GMRL)** introduced the first **Single-Stream** batch processing systems. It only executed one job at a time, and data was sent in batches or groups. The batch operating system eliminates the setup time issue.

In this article, you will learn about the batch operating system, types, working, and its advantages and disadvantages.

What is Batch Operating System?

Batch processing was very popular in the **1970s**. The jobs were executed in batches. People used to have a single computer known as a **mainframe**. Users using batch operating systems do not interact directly with the computer. Each user prepares their job using an offline device like a punch card and submitting it to the computer operator. Jobs with similar requirements are grouped and executed as a group to speed up processing. Once the programmers have left their programs with the operator, they sort the programs with similar needs into batches.



The batch operating system grouped jobs that perform similar functions. These job groups are treated as a batch and executed simultaneously. A computer system with this operating system performs the following batch processing activities:

1. A job is a single unit that consists of a preset sequence of commands, data, and programs.
2. Processing takes place in the order in which they are received, i.e., first come, first serve.
3. These jobs are stored in memory and executed without the need for manual information.
4. When a job is successfully run, the operating system releases its memory.

Types of Batch Operating System

There are mainly two types of the batch operating system. These are as follows:

1. **Simple Batched System**
2. **Multi-programmed batched system**

Simple Batched System

The user did not directly interact with the computer system for job execution in a simple batch operating system. However, the user was required to prepare a job that included the program, control information, and data on the nature of the job on control cards. The job was then submitted to the computer operator, who was usually in the form of a punch card. The program's output included results and registers and memory dumps in the event of a program error. The output appeared after some time that could take days, hours, and minutes.

Its main role was to transfer control from one job to another. Jobs with similar requirements were pooled together and processed through the processor to improve processing speed. The operators were used in the program to create batches with similar needs. The computer runs the batches one by one when they became available. This system typically reads a sequence of jobs, each with its control cads and predefined job tasks.

Multi-programmed batched system

Spooling deals with many jobs that have already been read and are waiting to run on disk. A disk containing a pool of jobs allows the operating system to choose which job to run next to maximize CPU utilization. Jobs that come on magnetic tape or cards directly cannot be run in a different order. Jobs run sequentially because they are executed in a first-come, first-served manner. When various jobs are stored on a direct access device, job scheduling becomes possible like a disk. Multi-programming is an important feature of job scheduling. For overlapped I/O, spooling and offline operations have their limitations. Generally, a single user could not maintain all of the input/output devices, and CPU buys at all times.

In the multi-programmed batched system, jobs are grouped so that the CPU only executes one job at a time to improve CPU utilization. The operating system maintains various jobs in memory at a time. The operating system selects one job and begins executing it in memory. Finally, the job must wait for a task to complete, such as mounting a tape on an I/O operation.

In a multiprogramming system, do not sit idle because the operating system switches to another task. When a job is in the wait state, and the current job is completed, the CPU is returned.

Why are Batch Operating Systems used?

Batch operating systems load less stress on the CPU and include minimal user interaction, and that is why you can still use them nowadays. Another benefit of batch operating systems is that huge repetitive jobs may be done without interacting with the computer to notify the system that you need to perform after you finish that job.

Old batch operating systems weren't interactive, which means that the user did not interact with the program while executing it. Modern batch operating systems now support interactions. For example, you may schedule the job, and when the specified time arrives, the computer acknowledges the processor that the time is up.

How does Batch Operating System work?

The operating system keeps the number of jobs in memory and performs them one at a time. Jobs are processed in a first-come, first-served manner. Each job set is defined as a batch. When a task is finished, its memory is freed, and the work's output is transferred into an output spool for later printing or processing. User interaction is limited in the batch operating system. When the system takes the task from the user, user is free. You may also use the batch processing system to update data relating to any transactions or records.

Role of Batch Operating System

A batch operating system's primary role is to execute jobs in batches automatically. The main task of a batch processing system is done by the **'Batch Monitor'**, which is located at the low end of the main memory. This technique was made possible by the development of hard disk drives and card readers. The jobs can now be stored on a disk to form a pool of jobs for batch execution. After that, they are grouped with similar jobs being placed in the same batch. As a result, the batch operating system automatically ran the batched jobs one after the other, saving time by performing tasks only once. It resulted from a better system due to reduced turnaround time.

Characteristics of Batch Operating System

There are various characteristics of the Batch Operating System. Some of them are as follows:

1. In this case, the CPU executes the jobs in the same sequence that they are sent to it by the operator, which implies that the task sent to the CPU first will be executed first. It's also known as the **'first come, first serve'**

2. The word job refers to the command or instruction that the user and the program should perform.
3. A batch operating system runs a set of user-supplied instructions composed of distinct instructions and programs with several similarities.
4. When a task is successfully executed, the OS releases the memory space held by that job.
5. The user does not interface directly with the operating system in a batch operating system; rather, all instructions are sent to the operator.
6. The operator evaluates the user's instructions and creates a set of instructions having similar properties.

Advantages and Disadvantages of Batch Operating System

There are various advantages and disadvantages of the Batch Operating System. Some of them are as follows:

Advantages

There are various advantages of the Batch Operating System. Some of them are as follows:

1. It isn't easy to forecast how long it will take to complete a job; only batch system processors know how long it will take to finish the job in line.
2. This system can easily manage large jobs again and again.
3. The batch process can be divided into several stages to increase processing speed.
4. When a process is finished, the next job from the job spool is run without any user interaction.
5. CPU utilization gets improved.

Disadvantages

There are various disadvantages of the Batch Operating System. Some of them are as follows:

1. When a job fails once, it must be scheduled to be completed, and it may take a long time to complete the task.
2. Computer operators must have full knowledge of batch systems.

3. The batch system is quite difficult to debug.
4. The computer system and the user have no direct interaction.
5. If a job enters an infinite loop, other jobs must wait for an unknown period of time.

Interactive Operating System

Interactive operating systems are computers that accept human inputs. Users give commands or some data to the computers by typing or by gestures. Some examples of interactive systems include MS Word and Spreadsheets, etc. They facilitate interactive behavior. Mac and Windows OS are some examples of interactive operating systems.

What is an Interactive Operating System?

- An interactive operative system is an operating system that allows the execution of interactive programs. All PC operating systems are interactive operating systems only.
- An interactive operating system gives permission to the user to interact directly with the computer. In an Interactive operating system, the user enters some command into the system and the work of the system is to execute it.
- Programs that allow users to enter some data or commands are known as Interactive Operating Systems. Some commonly used examples of Interactive operating systems include Word Processors and Spreadsheet Applications.
- A non-interactive program can be defined as one that once started continues without the need for human interaction. A compiler can be an example of a non-interactive program.

Properties of Interactive Operating System:

1. **Batch Processing:** It is defined as the process of gathering programs and data together in a batch before performing them. The job of the operating system is to define the jobs as a single unit by using some already defined sequence of commands or data, etc. Before they are performed or carried out, these are stored in the memory of the system and their processing depends on a FIFO basis. The operating system releases the memory and then copies the output into an output spool for later printing when the job is finished. Its use is that it basically improves the system performance because a new job begins only when the old one is completed without any interference from the user. One disadvantage is that there is a small chance that the jobs will enter an infinite loop. Debugging is also somewhat difficult with batch processing.
2. **Multitasking:** The CPU can execute many tasks simultaneously by switching between them. This is known as Time- Sharing System and also it has a very fast response time. They switch so Fastly that the users can very easily interact with each running program.

3. **Multiprogramming:** Multiprogramming happens when the memory of the system stores way too many processes. The job of the operating system here is that runs these processes in parallel on the same processor. Multiple processes share the CPU, thus increasing CPU utilization. Now, the CPU only performs one job at a particular time while the rest wait for the processor to be assigned to them. The operating system takes care of the fact that the CPU is never idle by using its memory management programs so that it can monitor the state of all system resources and active programs. One advantage of this is that it gives the user the feeling that the CPU is working on multiple programs simultaneously.
4. **Distributive Environment:** A distributive environment consists of many independent processors. The job of the operating system here is that distribute computation logic among the physical processors and also at the same time manage communication between them. Each processor has its own local memory, so they do not share a memory.
5. **Interactivity:** Interactivity is defined as the power of a user to interact with the system. The main job of the operating system here is that it basically provides an interface for interacting with the system, manages I/O devices, and also ensures a fast response time.
6. **Real-Time System:** Dedicated embedded systems are real-time systems. The main job of the operating system here is that reads and react to sensor data and then provide a response in a fixed time period, therefore, ensuring good performance.
7. **Spooling:** Spooling is defined as the process of pushing the data from different I/O jobs into a buffer or somewhere in the memory so that any device can access the data when it is ready. The operating system here handles the I/O device data spooling because the devices have different data access rates in order to maintain the spooling buffer. Now the job of the buffer here is that acts like a waiting station for the data to rest while the devices which are slower can catch up.

Example of an Interactive Operating System:

Some examples of Interactive operating systems are as follows:

1. Unix Operating System
2. Disk Operating System

Advantages of Interactive Operating System:

The advantages of an Interactive Operating System are as follows:

1. **Usability:** An operating system is designed to perform something and the interactiveness allows the user to manage the tasks more or less in real-time.
2. **Security:** Simple security policy enhancement. In non-interactive systems, the user virtually always knows what their programs will do during their lifetime, thus allowing us to forecast and correct the bugs.

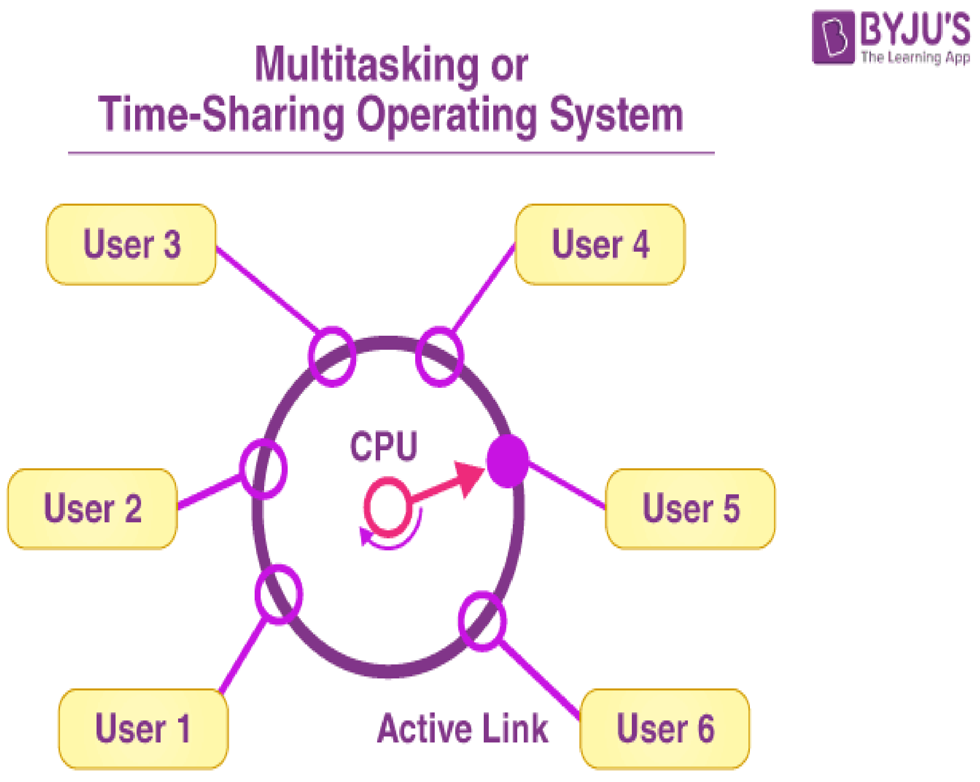
Disadvantages of Interactive Operating System:

1. **Tough to design:** Depending on the target device, interactivity might be proved challenging to design because the user must be prepared for every input. What about having many inputs? The state of a program can alternate at any particular time, all the programs should be handled in some way, and also it doesn't always work out properly.

Time Sharing Operating System:

What is the Time-Sharing Operating System?

Time-sharing is a method of allowing multiple individuals at different terminals to access the same computer system simultaneously. Multiprogramming is logically followed by time-sharing. Many processes are assigned to computer resources in time slots in this time-sharing operating system. The processor's time is split among numerous users in this scenario. It's dubbed a time-sharing OS for this reason. The individual processes have a set time slice. Its primary goal is to improve interactive reaction time.



The CPU changes between numerous jobs to complete them, but the shifts are frequent. As a result, the user can expect a quick answer. To give each user a little amount of time, the operating system utilises CPU scheduling and multiprogramming. Computer systems that were originally built for batch processing have been converted to time-sharing systems.

The time-sharing operating system uses a CPU scheduling and multi-programming mechanism to provide a modest amount of operational time to each user. Batch processing systems are related to the time-sharing system method. The primary distinction between Multiprogrammed Batch Systems and Time-Sharing Systems is that with multiprogrammed batch systems, the goal is to maximise processor utilisation. In Time-Sharing Systems, on the other hand, the goal is to reduce reaction time.

This type of operating system provides direct access to numerous users while dividing CPU time among them according to a scheduling system, which assigns a set of times to each user. When this time slot reaches the end of its session, it transfers control to another user. The time slot is shrinking, and all users are given the notion that they are the sole owners of their own CPUs. The “Time Slice or Quantum” is a short period of time during which all users compete for the CPU’s attention.

What is Time-Sharing in OS?

In data processing, time-sharing is a technique of operation in which numerous users with different programmes interact with the CPU (central processing unit) of a large-scale digital computer nearly concurrently.

The CPU has enough time to address multiple distinct problems during the input/output process since it runs far quicker than most peripheral equipment, for example, video display terminals and printers. Even though the CPU tackles each user’s problem, in turn, remote terminals perceive access to as well as retrieval from the time-sharing system as instantaneous since the solutions are available to them as soon as the problem is fully typed.

In the late 1950s and early 1960s, time-sharing was created to make better use of expensive processor time. The parallel operation, multiprocessing, and multiprogramming are all common time-sharing strategies. Furthermore, time-sharing systems are at the heart of many computer networks created for the purpose of transferring data and resources.

Examples of Time-Sharing OS

For instance, in a transaction processing system, all sorts of processors have the ability to run each user programme in tiny bursts or quantum of computation, such that if n users exist, each user can seize a temporal quantum. Other examples include:

- Multics
- UNIX
- Linux
- TOPS-20 (DEC)
- Windows NT server
- Windows 2000 server
- TOPS-10 (DEC)

Features of Time-Sharing OS

Users can benefit from the following advantages provided by the time-sharing OS:

- For all operations, each user sets aside time.
- At the same time, multiple online users can utilise the same computer.
- End users believe they have complete control over the computer system.
- Interaction among users and computers is improved.
- User inquiries can result in quick responses.
- It is no longer necessary to wait for the previous task to complete before using the processor.
- It can do a large number of tasks quickly.

Pros of Time-Sharing OS

The following are some of the benefits of the time-sharing operating system:

- It has a quick response time.
- CPU idle time is reduced.
- Each task is assigned a certain time limit.
- Reduced likelihood of program duplication improves reaction time.
- User-friendly and simple to use.

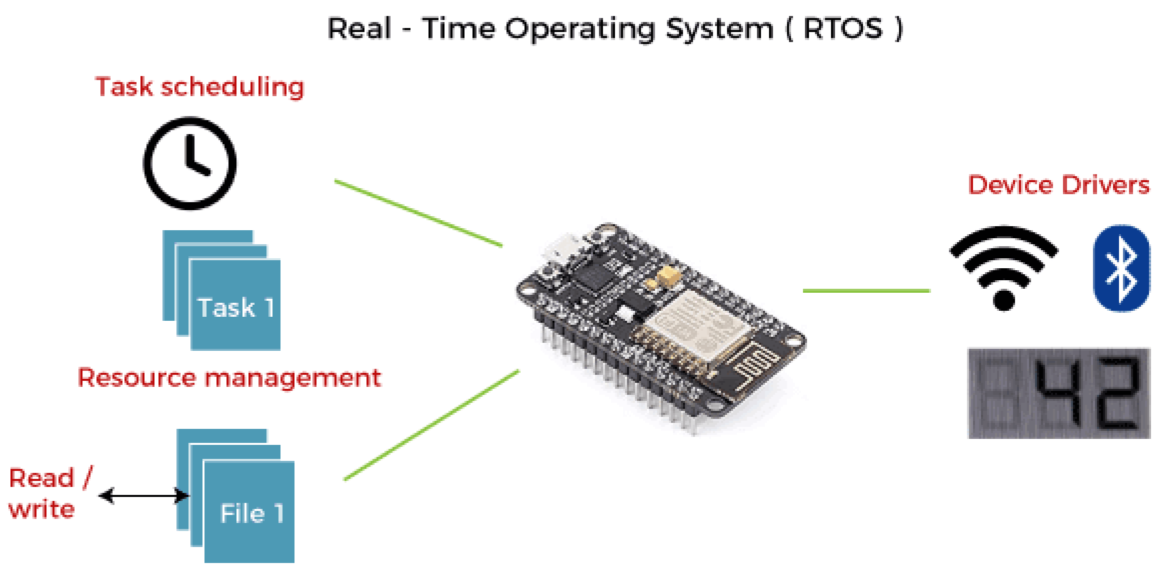
Cons of Time-Sharing OS

The following are some of the downsides of the time-sharing operating system:

- It uses a lot of resources.
- Hardware with high quality is required.
- It has difficulty with consistency.
- A security and integrity problem with user programs and data.
- Data communication problem probability.

Real Time Operating System (RTOS)

A real-time operating system (RTOS) is a special-purpose operating system used in computers that has strict time constraints for any job to be performed. It is employed mostly in those systems in which the results of the computations are used to influence a process while it is executing. Whenever an event external to the computer occurs, it is communicated to the computer with the help of some sensor used to monitor the event. The sensor produces the signal that is interpreted by the operating system as an interrupt. On receiving an interrupt, the operating system invokes a specific process or a set of processes to serve the interrupt.



This process is completely uninterrupted unless a higher priority interrupt occurs during its execution. Therefore, there must be a strict hierarchy of priority among the interrupts. The interrupt with the highest priority must be allowed to initiate the process , while lower priority interrupts should be kept in a buffer that will be handled later. Interrupt management is important in such an operating system.

Real-time operating systems employ special-purpose operating systems because conventional operating systems do not provide such performance.

The various examples of Real-time operating systems are:

- MTS
- Lynx
- QNX
- VxWorks etc.

Applications of Real-time operating system (RTOS):

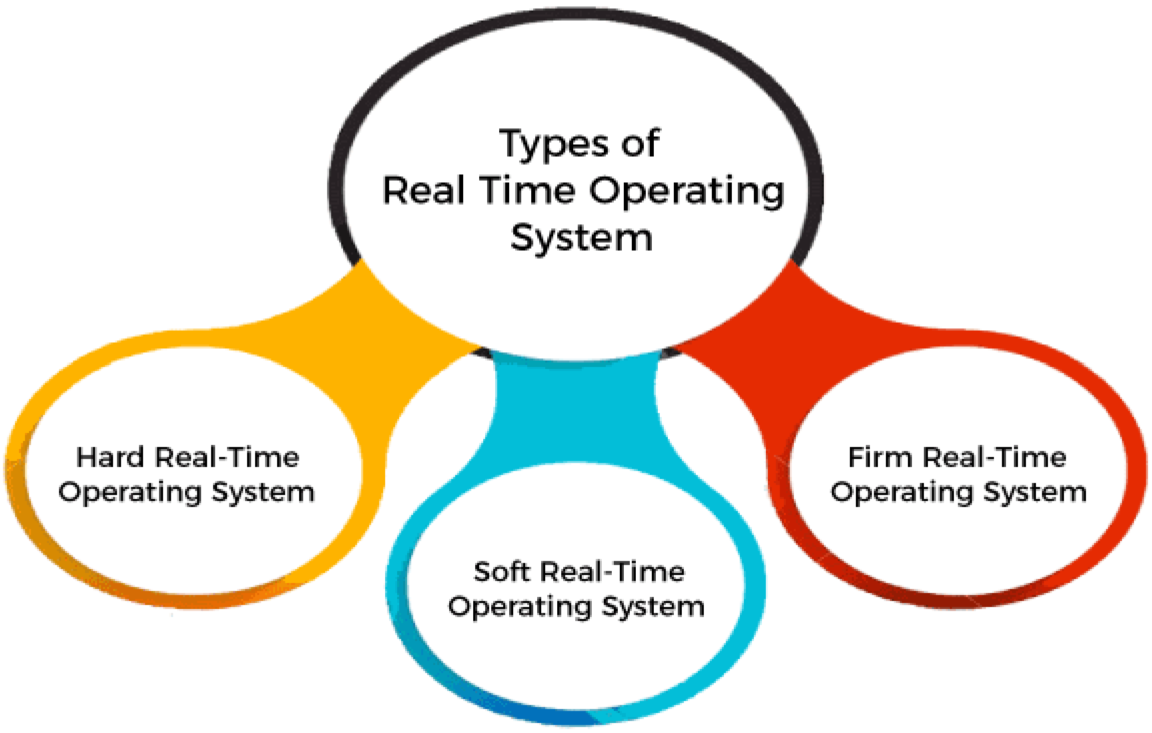
RTOS is used in real-time applications that must work within specific deadlines. Following are the common areas of applications of Real-time operating systems are given below.

- Real-time running structures are used inside the Radar gadget.
- Real-time running structures are utilized in Missile guidance.
- Real-time running structures are utilized in on line inventory trading.
- Real-time running structures are used inside the cell phone switching gadget.
- Real-time running structures are utilized by Air site visitors to manipulate structures.

- Real-time running structures are used in Medical Imaging Systems.
- Real-time running structures are used inside the Fuel injection gadget.
- Real-time running structures are used inside the Traffic manipulate gadget.
- Real-time running structures are utilized in Autopilot travel simulators.

Types of Real-time operating system

Following are the three types of RTOS systems are:



Hard Real-Time operating system:

In Hard RTOS, all critical tasks must be completed within the specified time duration, i.e., within the given deadline. Not meeting the deadline would result in critical failures such as damage to equipment or even loss of human life.

For Example,

Let's take an example of airbags provided by carmakers along with a handle in the driver's seat. When the driver applies brakes at a particular instance, the airbags grow and prevent the driver's head from hitting the handle. Had there been some delay even of milliseconds, then it would have resulted in an accident.

Similarly, consider an on-stock trading software. If someone wants to sell a particular share, the system must ensure that command is performed within a given critical time. Otherwise, if the market falls abruptly, it may cause a huge loss to the trader.

Soft Real-Time operating system:

Soft RTOS accepts a few delays via the means of the Operating system. In this kind of RTOS, there may be a closing date assigned for a particular job, but a delay for a small amount of time is acceptable. So, cut off dates are treated softly via means of this kind of RTOS.

For Example,

This type of system is used in Online Transaction systems and Livestock price quotation Systems.

Firm Real-Time operating system:

In Firm RTOS additionally want to observe the deadlines. However, lacking a closing date might not have a massive effect, however may want to purposely undesired effects, like a massive discount within the fine of a product.

For Example, this system is used in various forms of Multimedia applications.

Advantages of Real-time operating system:

The benefits of real-time operating system are as follows-:

- Easy to layout, develop and execute real-time applications under the real-time operating system.
- The real-time working structures are extra compact, so those structures require much less memory space.
- In a Real-time operating system, the maximum utilization of devices and systems.
- Focus on running applications and less importance to applications that are in the queue.
- Since the size of programs is small, RTOS can also be embedded systems like in transport and others.
- These types of systems are error-free.
- Memory allocation is best managed in these types of systems.

Disadvantages of Real-time operating system:

The disadvantages of real-time operating systems are as follows-

- Real-time operating systems have complicated layout principles and are very costly to develop.
- Real-time operating systems are very complex and can consume critical CPU cycles.

Multiprocessor Systems

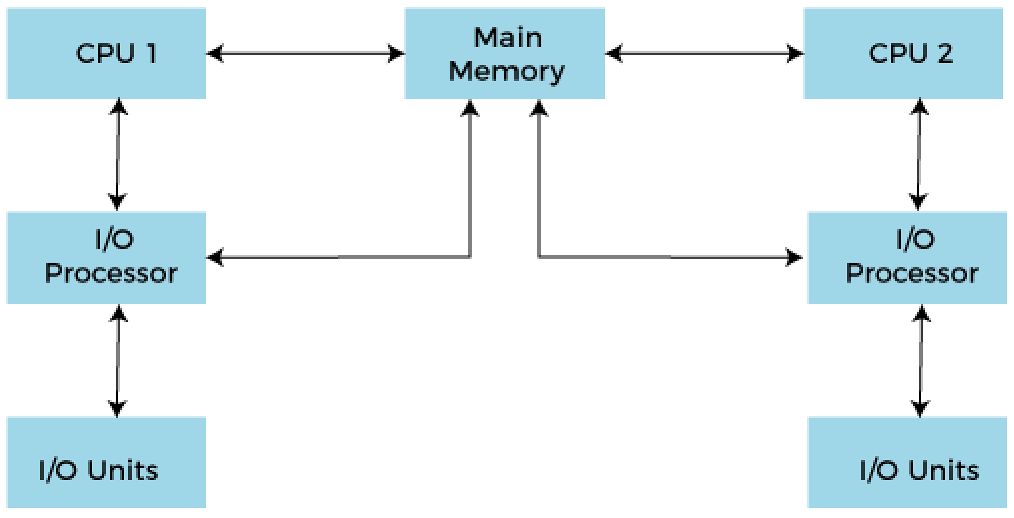
In operating systems, to improve the performance of more than one CPU can be used within one computer system called Multiprocessor operating system.

Multiple CPUs are interconnected so that a job can be divided among them for faster execution. When a job finishes, results from all CPUs are collected and compiled to give the final output. Jobs needed to share main memory and they may also share other system resources among themselves. Multiple CPUs can also be used to run multiple jobs simultaneously.

For Example: UNIX Operating system is one of the most widely used multiprocessing systems.

The basic organization of a typical multiprocessing system is shown in the given figure.

Backward Skip 10sPlay VideoForward Skip 10s



Working of Multiprocessor System

To employ a multiprocessing operating system effectively, the computer system must have the following things:

- A motherboard is capable of handling multiple processors in a multiprocessing operating system.
- Processors are also capable of being used in a multiprocessing system.

Advantages of multiprocessing operating system are:

- **Increased reliability:** Due to the multiprocessing system, processing tasks can be distributed among several processors. This increases reliability as if one processor fails; the task can be given to another processor for completion.
- **Increased throughout:** As several processors increase, more work can be done in less
- **The economy of Scale:** As multiprocessors systems share peripherals, secondary storage devices, and power supplies, they are relatively cheaper than single-processor systems.

Disadvantages of Multiprocessing operating System

- Operating system of multiprocessing is more complex and sophisticated as it takes care of multiple CPUs at the same time.

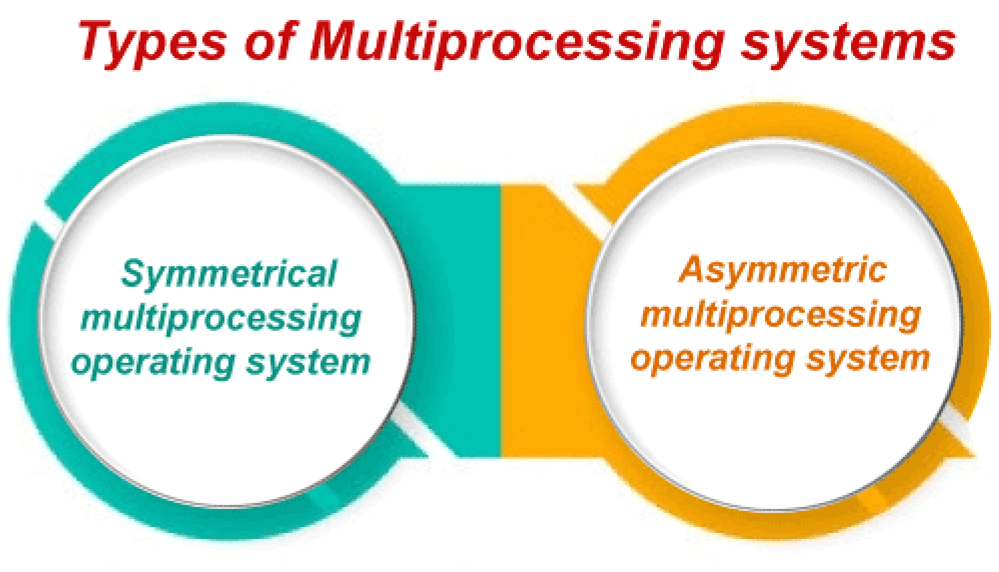
Types of multiprocessing systems

- Symmetrical multiprocessing operating system
- Asymmetric multiprocessing operating system

Symmetrical multiprocessing operating system:

In a Symmetrical multiprocessing system, each processor executes the same copy of the operating system, takes its own decisions, and cooperates with other processes to smooth the entire functioning of the system. The CPU scheduling policies are very simple. Any new job submitted by a user can be assigned to any processor that is least burdened. It also results in a system in which all processors are equally burdened at any time.

The symmetric multiprocessing operating system is also known as a "shared every-thing" system, because the processors share memory and the Input output bus or data path. In this system processors do not usually exceed more than 16.


Types of Multiprocessing systems
Symmetrical multiprocessing operating system
Asymmetric multiprocessing operating system

Characteristics of Symmetrical multiprocessing operating system:

- In this system, any processor can run any job or process.
- In this, any processor initiates an Input and Output operation.

Advantages of Symmetrical multiprocessing operating system:

- These systems are fault-tolerant. Failure of a few processors does not bring the entire system to a halt.

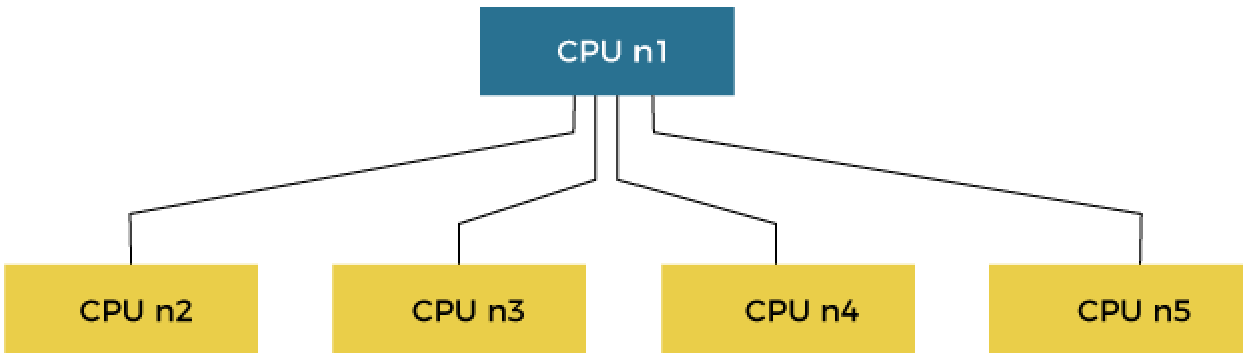
Disadvantages of Symmetrical multiprocessing operating system:

- It is very difficult to balance the workload among processors rationally.
- Specialized synchronization schemes are necessary for managing multiple processors.

Asymmetric multiprocessing operating system

In an asymmetric multiprocessing system, there is a master slave relationship between the processors.

Further, one processor may act as a master processor or supervisor processor while others are treated as shown below.



Asymmetric Multiprocessor System

In the above figure, the asymmetric processing system shows that CPU n1 acts as a supervisor whose function controls other following processors.

In this type of system, each processor is assigned a specific task, and there is a designated master processor that controls the activities of other processors.

For example, we have a math co-processor that can handle mathematical jobs better than the main CPU. Similarly, we have an MMX processor that is built to handle multimedia-related jobs. Similarly, we have a graphics processor to handle the graphics-related job better than the main processor. When a user submits a new job, the OS has to decide which processor can perform it better, and then that processor is assigned that newly arrived job. This processor acts as the master and controls the system. All other processors look for masters for instructions or have predefined tasks. It is the responsibility of the master to allocate work to other processors.

Advantages of Asymmetric multiprocessing operating system:

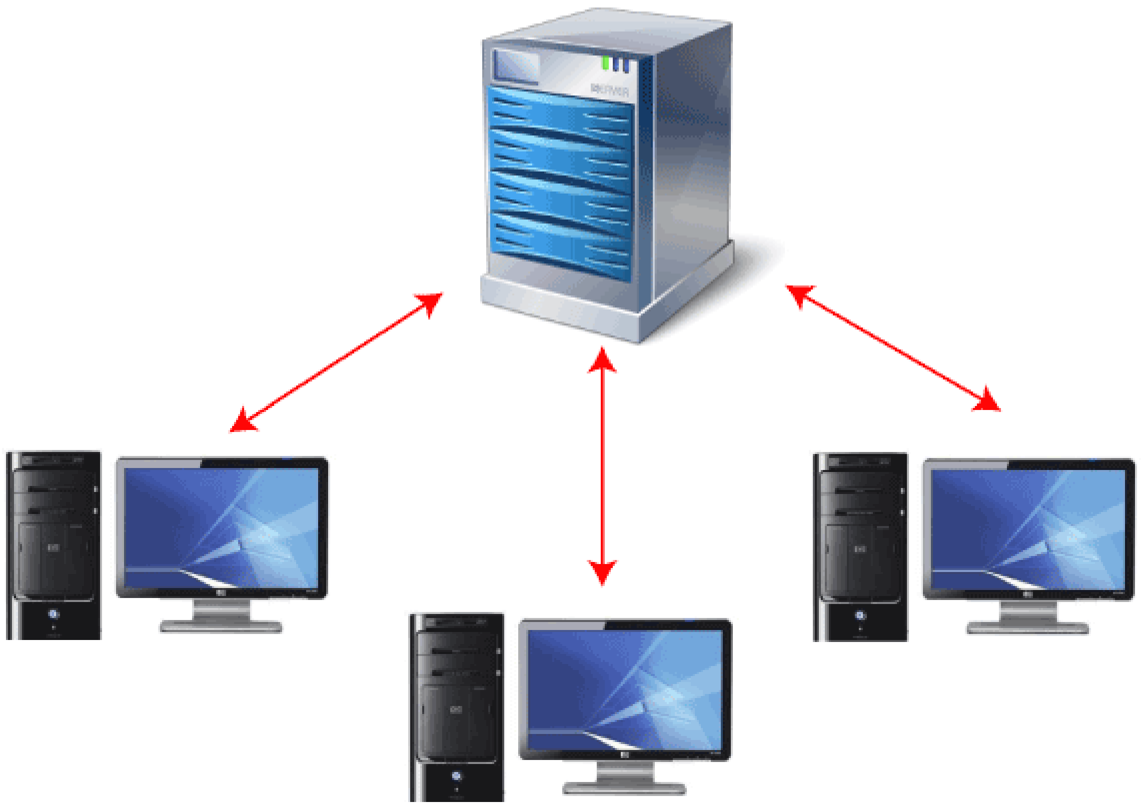
- In this type of system execution of Input and Output operation or an application program may be faster in some situations because many processors may be available for a single job.

Disadvantages of Asymmetric multiprocessing operating system:

- In this type of multiprocessing operating system the processors are unequally burdened. One processor may be having a long job queue, while another one may be sitting idle.
- In this system, if the process handling a specific work fails, the entire system will go down.

What is Multi-User Operating System?

A multi-user operating system is an operating system that permits several users to access a single system running to a single operating system. These systems are frequently quite complex, and they must manage the tasks that the various users connected to them require. Users will usually sit at terminals or computers connected to the system via a network and other system machines like printers. A multi-user operating system varies from a connected single-user operating system in that each user accesses the same operating system from different machines.



The main goal of developing a multi-user operating system is to use it for time-sharing and batch processing on mainframe systems. This multi-user operating system is now often used in large organizations, the government sector, educational institutions like large universities, and on servers' side such as Ubuntu Server or Windows Server. These servers allow several users to access the operating system, kernel, and hardware at the same time.

It is usually responsible for handling memory and processing for other running programs, identifying and using system hardware, and efficiently handling user interaction and data requests. It's especially important for an operating system, a multi-user operating system because several users rely on the system to function properly at the same time.

Components of Multi-User Operating System

There are various components of a multi-user operating system. Some of them are as follows:

Memory

The physical memory present inside the system is where storage occurs. It is also known as **Random Access Memory (RAM).** The system may rectify the data that is present in the main memory. So, every executed program should be copied from physical storage like a hard disk. Main memory is determined as an important part of OS because it specifies how many programs may be executed simultaneously.

Kernel

A multi-user operating system makes use of the Kernel component, which is built in a low-level language. This component is embedded in the computer system's main memory and may interact directly with the system's H/W.

Processor

The **CPU (Central Processing Unit)** of the computer is sometimes known as the computer's brain. In large machines, the CPU would necessitate more ICS. On smaller computers, the CPU is mapped in a single chip known as a microprocessor.

User Interface

The user interface is the way of interaction between users and all software and hardware processes. It enables the users to interact with the computer system in a simple manner.

Device Handler

Each input and output device needs its device handler. The device handler's primary goal is to provide all requests from the whole device request queue pool. The device handler operates in continuous cycle mode, first discarding the I/O request block from the queue side.

Spooler

Spooler stands for **'Simultaneous Peripheral Output on Line'.** The Spooler runs all computer processes and outputs the results at the same time. Spooling is used by a variety of output devices, including printers.

Types of Multi-User Operating System

There are various types of multi-user operating systems. Some of them are as follows:

Distributed System

A distributed system is also known as distributed computing. It is a collection of multiple components distributed over multiple computers that interact, coordinate, and seem like a single coherent system to the end-user. With the aid of the network, the end-user would be able to interact with or operate them.

Time-Sliced Systems

It's a system in which each user's job gets a specific amount of CPU time. In other words, each work is assigned to a specific time period. These time slices look too small to the user's eyes. An internal component known as the 'Scheduler' decides to run the next job. This scheduler determines and executes the job that must perform based on the priority cycle.

Multiprocessor System

Multiple processors are used in this system, which helps to improve overall performance. If one of the processors in this system fails, the other processor is responsible for completing its assigned task.

How to work the Multi-User operating system?

The single master system is contained within the multi-user system. All network users can access the master system anytime and from any place and open their local version of the system. The local version is also known as a 'working model'. All users can update, delete, and create new files on their local working model, but this model will not be available to other users until it is saved to the master system.

Characteristics of Multi-User Operating System

There are various characteristics of a multi-user operating system. Some of them are as follows:

Resource Sharing

Several devices, like printers, fax machines, plotters, and hard drives, can be shared in a multi-user operating system. Users can share their own documents using this functionality. All users are given a small slice of CPU time under this system.

Multi-Tasking

Multi-user operating systems may execute several tasks simultaneously, and several programs may also execute at the same time.

Background Processing

Background processing is a term that refers to when commands are not processed but rather executed "in the background". Usually, other programs interact with the system in real-time.

Time-Sharing

A strategy used by multi-user operating systems to operate on several user requests at the same time by switching between jobs at very short periods of time.

System

The operating system must handle a computer's combination of hardware and software resources.

Invisibility

Various functions of the multi-user operating system are hidden from users. It is due to factors such as the OS being instinctive or happening at the lower end, such as disk formatting, etc.

Examples of Multi-User Operating System

There are various examples of multi-user operating systems. Some of them are as follows:

Unix

A highly dependable open system architecture for small and medium-scale business computing systems. Because it is based on Open System Architecture, tech giants including AIX, Solaris, and even Mac OS have their own version of Unix. For example, the Hospitality industry, Healthcare, etc.

Multiple Virtual Storage

IBM develops an operating system for use on mainframe systems. It's commonly utilized in enterprise computing, where high-intensity I/O is required. For example, Banking, Insurance, Aviation business, etc.

Shared Computing

A multi-user OS is a software that operates the servers that support most webmail apps. A typical webmail application may require the utilization of hundreds of computers. Each one runs a multi-user operating system capable of supporting various users at the same time. Because these systems have millions, if not billions, of users who constantly log on to check their messages, they require operating systems that can handle a high number of users at once.

Advantages and Disadvantages of Multi-User Operating System

There are various advantages and disadvantages of a multi-user operating system. These are as follows:

Advantages

There are various advantages of a multi-user operating system. Some of the advantages are as follows:

1. A multi-user operating system can be used in the printing process to allow multiple users to access the same printer, which a normal operating system may not do.
2. On a single computer system, several users can access the same copy of a document. For instance, if a PPT file is kept on one computer, other users can see it on other systems.
3. Multi-user operating systems are very useful in offices and libraries because they can efficiently handle printing jobs.
4. If one computer fails in its own network system, the entire system does not come to a halt.
5. Airlines use multi-user operating systems for some of their functions.
6. The ticket reservation system uses a multi-user operating system.
7. Each user can access the same document on their own computer.

Disadvantages of Multi-User Operating System

There are various disadvantages of a multi-user operating system. Some of the disadvantages are as follows:

1. Virus attacks occur simultaneously on all of them as the computers are shared. As a result, if one machine is affected, the others will be as well.
2. If a virus hits one computer, it spreads to the entire network system simultaneously, and finally, all computer systems fail.
3. All computer information is shared publicly, and your personal information is accessible to everyone on the network.
4. Multiple accounts on a single computer may not be suitable for all users. Thus, it is better to have multiple PCs for each user.

Multi-Threading

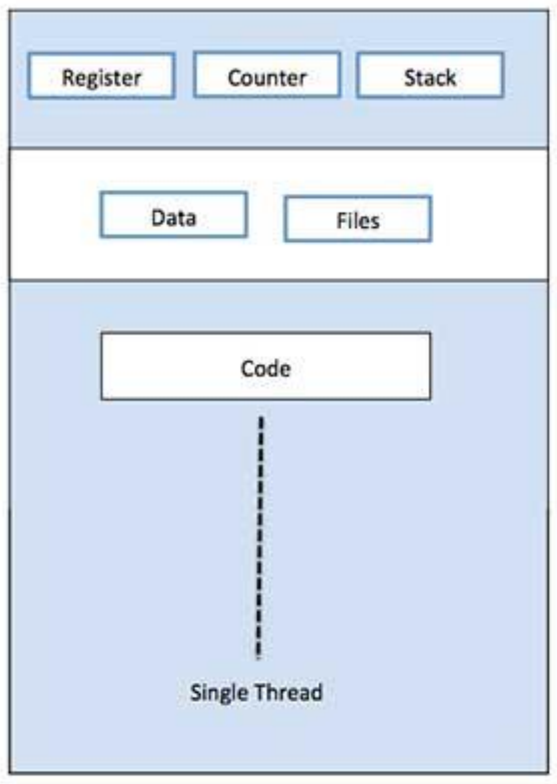
What is Thread?

A thread is a flow of execution through the process code, with its own program counter that keeps track of which instruction to execute next, system registers which hold its current working variables, and a stack which contains the execution history.

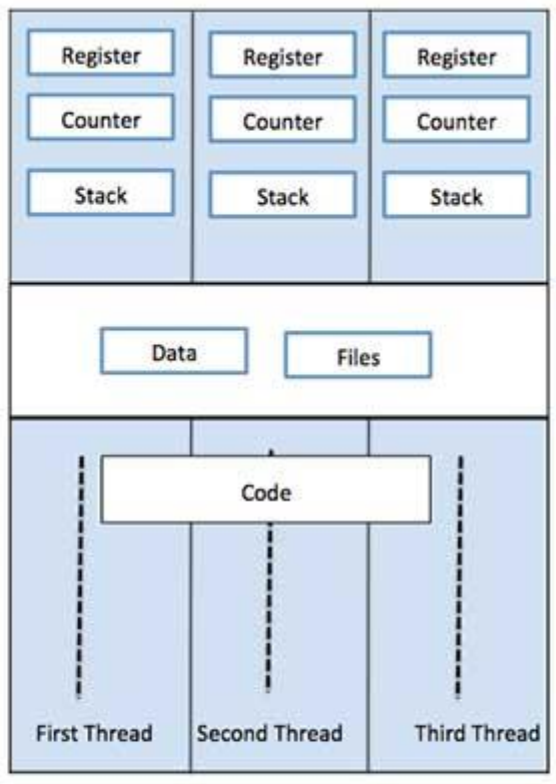
A thread shares with its peer threads few information like code segment, data segment and open files. When one thread alters a code segment memory item, all other threads see that.

A thread is also called a **lightweight process**. Threads provide a way to improve application performance through parallelism. Threads represent a software approach to improving performance of operating system by reducing the overhead thread is equivalent to a classical process.

Each thread belongs to exactly one process and no thread can exist outside a process. Each thread represents a separate flow of control. Threads have been successfully used in implementing network servers and web server. They also provide a suitable foundation for parallel execution of applications on shared memory multiprocessors. The following figure shows the working of a single-threaded and a multithreaded process.



Single Process P with single thread



Single Process P with three threads

Difference between Process and Thread

S.N.	Process	Thread
1	Process is heavy weight or resource intensive.	Thread is light weight, taking lesser resources than a process.
2	Process switching needs interaction with operating system.	Thread switching does not need to interact with operating system.
3	In multiple processing environments, each process executes the same code but has its own memory and file resources.	All threads can share same set of open files, child processes.
4	If one process is blocked, then no other process can execute until the first process is unblocked.	While one thread is blocked and waiting, a second thread in the same task can run.
5	Multiple processes without using threads use more resources.	Multiple threaded processes use fewer

		resources.
6	In multiple processes each process operates independently of the others.	One thread can read, write or change another thread's data.

Advantages of Thread

- Threads minimize the context switching time.
- Use of threads provides concurrency within a process.
- Efficient communication.
- It is more economical to create and context switch threads.
- Threads allow utilization of multiprocessor architectures to a greater scale and efficiency.

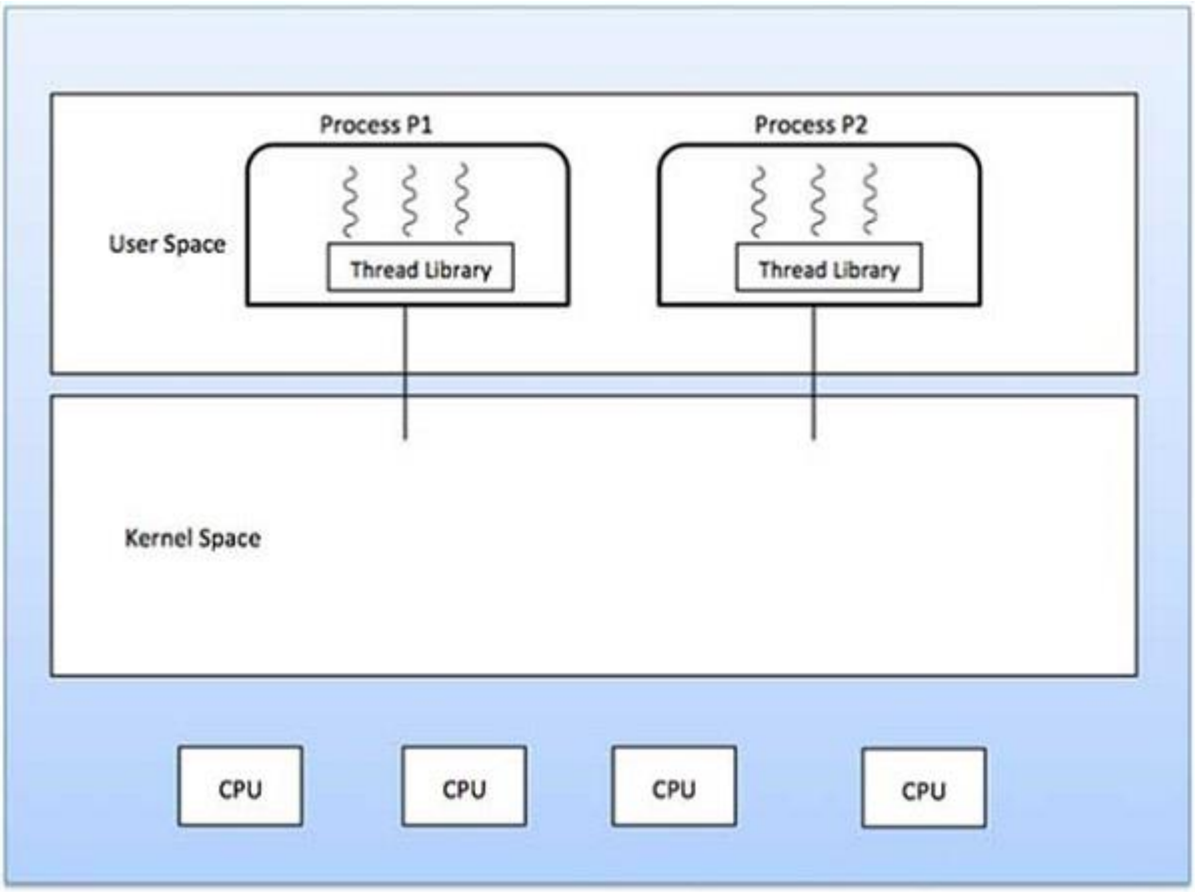
Types of Thread

Threads are implemented in following two ways −

- **User Level Threads** − User managed threads.
- **Kernel Level Threads** − Operating System managed threads acting on kernel, an operating system core.

User Level Threads

In this case, the thread management kernel is not aware of the existence of threads. The thread library contains code for creating and destroying threads, for passing message and data between threads, for scheduling thread execution and for saving and restoring thread contexts. The application starts with a single thread.



Advantages

- Thread switching does not require Kernel mode privileges.
- User level thread can run on any operating system.
- Scheduling can be application specific in the user level thread.
- User level threads are fast to create and manage.

Disadvantages

- In a typical operating system, most system calls are blocking.
- Multithreaded application cannot take advantage of multiprocessing.

Kernel Level Threads

In this case, thread management is done by the Kernel. There is no thread management code in the application area. Kernel threads are supported directly by the operating system. Any application can be programmed to be multithreaded. All of the threads within an application are supported within a single process.

The Kernel maintains context information for the process as a whole and for individuals threads within the process. Scheduling by the Kernel is done on a thread basis. The Kernel performs thread creation, scheduling and management in Kernel space. Kernel threads are generally slower to create and manage than the user threads.

Advantages

- Kernel can simultaneously schedule multiple threads from the same process on multiple processes.
- If one thread in a process is blocked, the Kernel can schedule another thread of the same process.
- Kernel routines themselves can be multithreaded.

Disadvantages

- Kernel threads are generally slower to create and manage than the user threads.
- Transfer of control from one thread to another within the same process requires a mode switch to the Kernel.

Multithreading Models

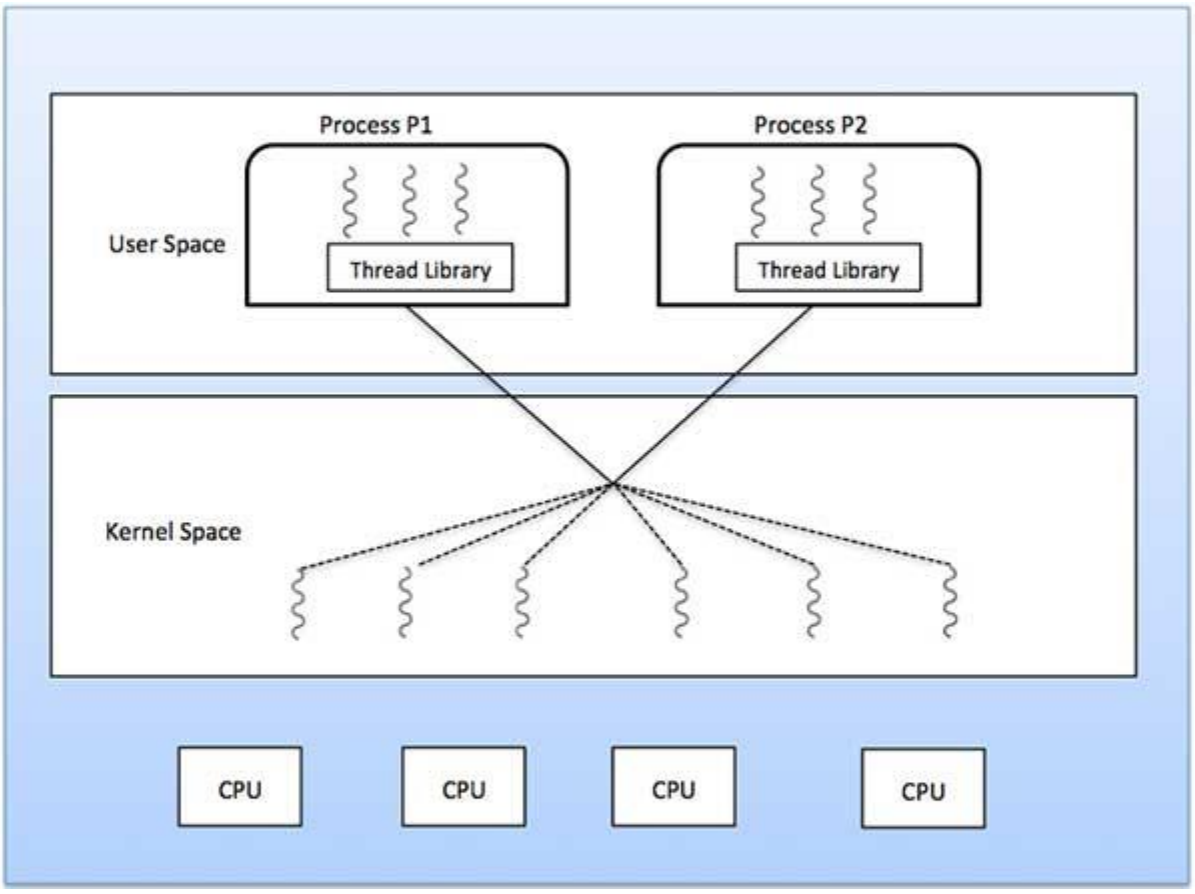
Some operating system provide a combined user level thread and Kernel level thread facility. Solaris is a good example of this combined approach. In a combined system, multiple threads within the same application can run in parallel on multiple processors and a blocking system call need not block the entire process. Multithreading models are three types

- Many to many relationship.
- Many to one relationship.
- One to one relationship.

Many to Many Model

The many-to-many model multiplexes any number of user threads onto an equal or smaller number of kernel threads.

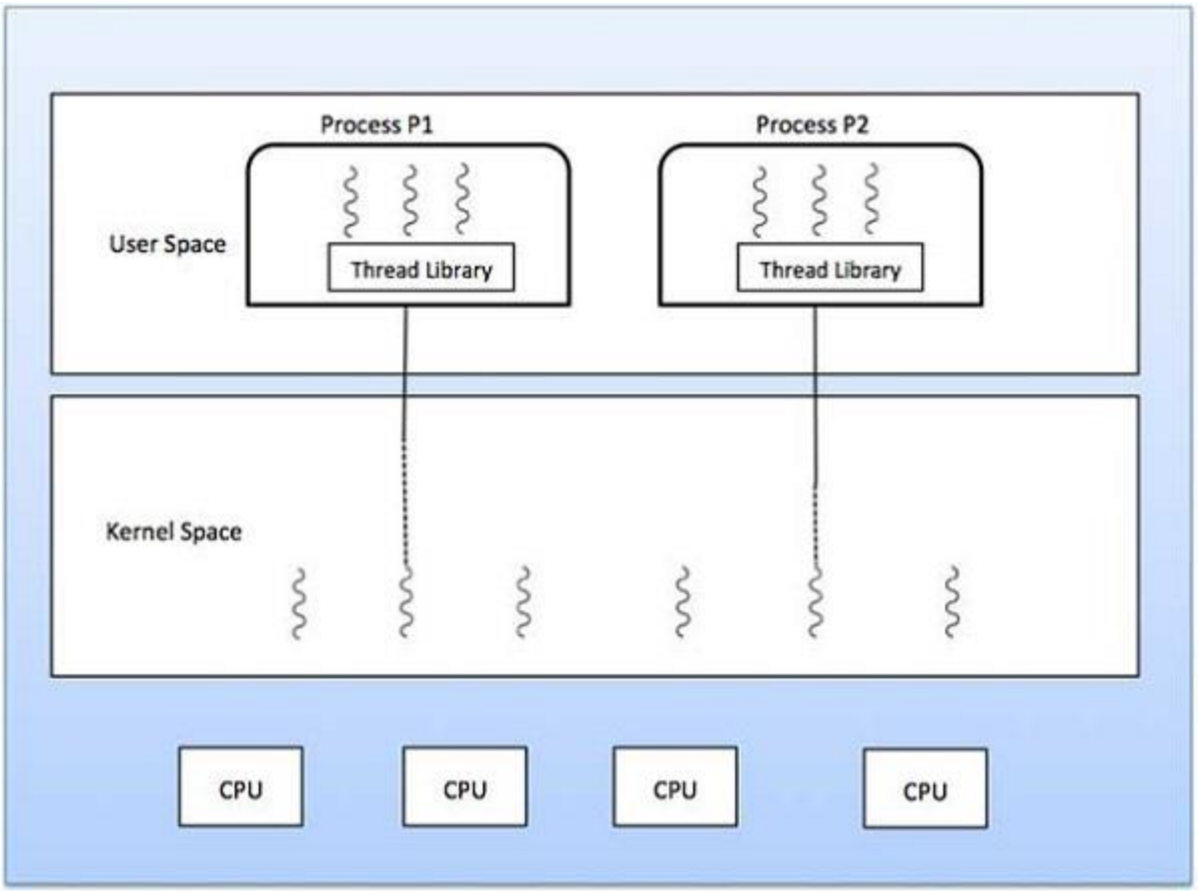
The following diagram shows the many-to-many threading model where 6 user level threads are multiplexing with 6 kernel level threads. In this model, developers can create as many user threads as necessary and the corresponding Kernel threads can run in parallel on a multiprocessor machine. This model provides the best accuracy on concurrency and when a thread performs a blocking system call, the kernel can schedule another thread for execution.



Many to One Model

Many-to-one model maps many user level threads to one Kernel-level thread. Thread management is done in user space by the thread library. When thread makes a blocking system call, the entire process will be blocked. Only one thread can access the Kernel at a time, so multiple threads are unable to run in parallel on multiprocessors.

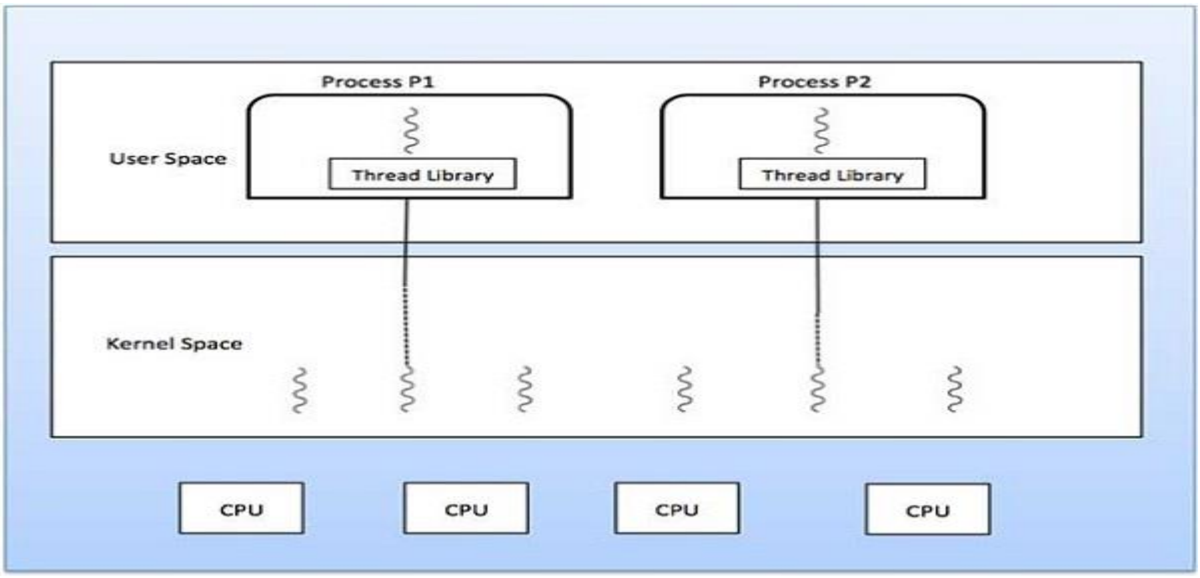
If the user-level thread libraries are implemented in the operating system in such a way that the system does not support them, then the Kernel threads use the many-to-one relationship modes.



One to One Model

There is one-to-one relationship of user-level thread to the kernel-level thread. This model provides more concurrency than the many-to-one model. It also allows another thread to run when a thread makes a blocking system call. It supports multiple threads to execute in parallel on microprocessors.

Disadvantage of this model is that creating user thread requires the corresponding Kernel thread. OS/2, windows NT and windows 2000 use one to one relationship model.



Difference between User-Level & Kernel-Level Thread

S.N.	User-Level Threads	Kernel-Level Thread
1	User-level threads are faster to create and manage.	Kernel-level threads are slower to create and manage.
2	Implementation is by a thread library at the user level.	Operating system supports creation of Kernel threads.
3	User-level thread is generic and can run on any operating system.	Kernel-level thread is specific to the operating system.
4	Multi-threaded applications cannot take advantage of multiprocessing.	Kernel routines themselves can be multithreaded.

Layered Operating System

Layered Structure is a type of system structure in which the different services of the operating system are split into various layers, where each layer has a specific well-defined task to perform. It was created to improve the pre-existing structures like the Monolithic structure (UNIX) and the Simple structure (MS-DOS).

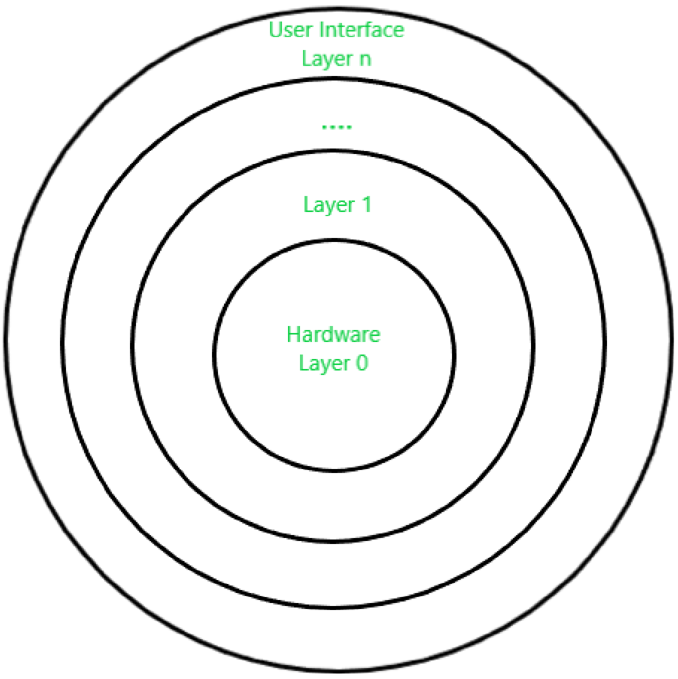
Example – The Windows NT operating system uses this layered approach as a part of it.

Design Analysis :

The whole Operating System is separated into several layers (from 0 to n) as the diagram shows. Each of the layers must have its own specific function to perform. There are some rules in the implementation of the layers as follows.

1. The outermost layer must be the User Interface layer.
2. The innermost layer must be the Hardware layer.
3. A particular layer can access all the layers present below it but it cannot access the layers present above it. That is layer n-1 can access all the layers from n-2 to 0 but it cannot access the nth layer.

Thus if the user layer wants to interact with the hardware layer, the response will be traveled through all the layers from n-1 to 1. Each layer must be designed and implemented such that it will need only the services provided by the layers below it.



Layered OS Design

Advantages :
There are several advantages to this design :

1. **Modularity :**
   This design promotes modularity as each layer performs only the tasks it is scheduled to perform.
2. **Easy debugging :**
   As the layers are discrete so it is very easy to debug. Suppose an error occurs in the CPU scheduling layer, so the developer can only search that particular layer to debug, unlike the Monolithic system in which all the services are present together.
3. **Easy update :**
   A modification made in a particular layer will not affect the other layers.
4. **No direct access to hardware :**

   The hardware layer is the innermost layer present in the design. So a user can use the services of hardware but cannot directly modify or access it, unlike the Simple system in which the user had direct access to the hardware.
5. **Abstraction :**
   Every layer is concerned with its own functions. So the functions and implementations of the other layers are abstract to it.

Disadvantages :
Though this system has several advantages over the Monolithic and Simple design, there are also some disadvantages as follows.

1. **Complex and careful implementation :**

   As a layer can access the services of the layers below it, so the arrangement of the layers must be done carefully. For example, the backing storage layer uses the services of the memory management layer. So it must be kept below the memory management layer. Thus with great modularity comes complex implementation.
2. **Slower in execution :**

   If a layer wants to interact with another layer, it sends a request that has to travel through all the layers present in between the two interacting layers. Thus it increases response time, unlike the Monolithic system which is faster than this. Thus an increase in the number of layers may lead to a very inefficient design.

Components of Operating System

There are various components of an Operating System to perform well defined tasks. Though most of the Operating Systems differ in structure but logically they have similar components. Each component must be a well-defined portion of a system that appropriately describes the functions, inputs, and outputs.

There are following 8-components of an Operating System:

1. Process Management
2. I/O Device Management
3. File Management
4. Network Management
5. Main Memory Management
6. Secondary Storage Management
7. Security Management
8. Command Interpreter System

Following section explains all the above components in more detail:

Process Management

A process is program or a fraction of a program that is loaded in main memory. A process needs certain resources including CPU time, Memory, Files, and I/O devices to accomplish its task. The process management component manages the multiple processes running simultaneously on the Operating System.

A program in running state is called a process.

The operating system is responsible for the following activities in connection with process management:

- Create, load, execute, suspend, resume, and terminate processes.
- Switch system among multiple processes in main memory.
- Provides communication mechanisms so that processes can communicate with each others
- Provides synchronization mechanisms to control concurrent access to shared data to keep shared data consistent.
- Allocate/de-allocate resources properly to prevent or avoid deadlock situation.

I/O Device Management

One of the purposes of an operating system is to hide the peculiarities of specific hardware devices from the user. I/O Device Management provides an abstract level of H/W devices and keep the details from applications to ensure proper use of devices, to prevent errors, and to provide users with convenient and efficient programming environment.

Following are the tasks of I/O Device Management component:

- Hide the details of H/W devices
- Manage main memory for the devices using cache, buffer, and spooling
- Maintain and provide custom drivers for each device.

File Management

File management is one of the most visible services of an operating system. Computers can store information in several different physical forms; magnetic tape, disk, and drum are the most common forms.

A file is defined as a set of correlated information and it is defined by the creator of the file. Mostly files represent data, source and object forms, and programs. Data files can be of any type like alphabetic, numeric, and alphanumeric.

A files is a sequence of bits, bytes, lines or records whose meaning is defined by its creator and user.

The operating system implements the abstract concept of the file by managing mass storage device, such as types and disks. Also files are normally organized into directories to ease their use. These directories may contain files and other directories and so on.

The operating system is responsible for the following activities in connection with file management:

- File creation and deletion
- Directory creation and deletion
- The support of primitives for manipulating files and directories
- Mapping files onto secondary storage
- File backup on stable (nonvolatile) storage media

Network Management

The definition of network management is often broad, as network management involves several different components. Network management is the process of managing and administering a computer network. A computer network is a collection of various types of computers connected with each other.

Network management comprises fault analysis, maintaining the quality of service, provisioning of networks, and performance management.

Network management is the process of keeping your network healthy for an efficient communication between different computers.

Following are the features of network management:

- Network administration
- Network maintenance
- Network operation
- Network provisioning
- Network security

Main Memory Management

Memory is a large array of words or bytes, each with its own address. It is a repository of quickly accessible data shared by the CPU and I/O devices.

Main memory is a volatile storage device which means it loses its contents in the case of system failure or as soon as system power goes down.

The main motivation behind Memory Management is to maximize memory utilization on the computer system.

The operating system is responsible for the following activities in connections with memory management:

- Keep track of which parts of memory are currently being used and by whom.
- Decide which processes to load when memory space becomes available.
- Allocate and deallocate memory space as needed.

Secondary Storage Management

The main purpose of a computer system is to execute programs. These programs, together with the data they access, must be in main memory during execution. Since the main memory is too small to permanently accommodate all data and program, the computer system must provide secondary storage to backup main memory.

Most modern computer systems use disks as the principle on-line storage medium, for both programs and data. Most programs, like compilers, assemblers, sort routines, editors, formatters, and so on, are stored on the disk until loaded into memory, and then use the disk as both the source and destination of their processing.

The operating system is responsible for the following activities in connection with disk management:

- Free space management
- Storage allocation

  Disk scheduling

Security Management

The operating system is primarily responsible for all task and activities happen in the computer system. The various processes in an operating system must be protected from each other's activities. For that purpose, various mechanisms which can be used to ensure that the files, memory segment, cpu and other resources can be operated on only by those processes that have gained proper authorization from the operating system.

Security Management refers to a mechanism for controlling the access of programs, processes, or users to the resources defined by a computer controls to be imposed, together with some means of enforcement.

For example, memory addressing hardware ensure that a process can only execute within its own address space. The timer ensure that no process can gain control of the CPU without relinquishing it. Finally, no process is allowed to do it’s own I/O, to protect the integrity of the various peripheral devices.

Command Interpreter System

One of the most important component of an operating system is its command interpreter. The command interpreter is the primary interface between the user and the rest of the system.

Command Interpreter System executes a user command by calling one or more number of underlying system programs or system calls.

Command Interpreter System allows human users to interact with the Operating System and provides convenient programming environment to the users.

Many commands are given to the operating system by control statements. A program which reads and interprets control statements is automatically executed. This program is called the shell and few examples are Windows DOS command window, Bash of Unix/Linux or C-Shell of Unix/Linux.

Other Important Activities

An Operating System is a complex Software System. Apart from the above mentioned components and responsibilities, there are many other activities performed by the Operating System. Few of them are listed below:

- **Security** − By means of password and similar other techniques, it prevents unauthorized access to programs and data.
- **Control over system performance** − Recording delays between request for a service and response from the system.
- **Job accounting** − Keeping track of time and resources used by various jobs and users.
- **Error detecting aids** − Production of dumps, traces, error messages, and other debugging and error detecting aids.
- **Coordination between other softwares and users** − Coordination and assignment of compilers, interpreters, assemblers and other software to the various users of the computer systems.

Operating System - Services

An Operating System provides services to both the users and to the programs.

- It provides programs an environment to execute.
- It provides users the services to execute the programs in a convenient manner.

Following are a few common services provided by an operating system −

- Program execution
- I/O operations
- File System manipulation
- Communication
- Error Detection
- Resource Allocation
- Protection

Program execution

Operating systems handle many kinds of activities from user programs to system programs like printer spooler, name servers, file server, etc. Each of these activities is encapsulated as a process.

A process includes the complete execution context (code to execute, data to manipulate, registers, OS resources in use). Following are the major activities of an operating system with respect to program management −

- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

I/O Operation

An I/O subsystem comprises of I/O devices and their corresponding driver software. Drivers hide the peculiarities of specific hardware devices from the users.

An Operating System manages the communication between user and device drivers.

- I/O operation means read or write operation with any file or any specific I/O device.
- Operating system provides the access to the required I/O device when required.

File system manipulation

A file represents a collection of related information. Computers can store files on the disk (secondary storage), for long-term storage purpose. Examples of storage media include magnetic tape, magnetic disk and optical disk drives like CD, DVD. Each of these media has its own properties like speed, capacity, data transfer rate and data access methods.

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Following are the major activities of an operating system with respect to file management −

- Program needs to read a file or write a file.
- The operating system gives the permission to the program for operation on file.
- Permission varies from read-only, read-write, denied and so on.
- Operating System provides an interface to the user to create/delete files.
- Operating System provides an interface to the user to create/delete directories.
- Operating System provides an interface to create the backup of file system.

Communication

In case of distributed systems which are a collection of processors that do not share memory, peripheral devices, or a clock, the operating system manages communications between all the processes. Multiple processes communicate with one another through communication lines in the network.

The OS handles routing and connection strategies, and the problems of contention and security. Following are the major activities of an operating system with respect to communication −

- Two processes often require data to be transferred between them
- Both the processes can be on one computer or on different computers, but are connected through a computer network.
- Communication may be implemented by two methods, either by Shared Memory or by Message Passing.

Error handling

Errors can occur anytime and anywhere. An error may occur in CPU, in I/O devices or in the memory hardware. Following are the major activities of an operating system with respect to error handling −

- The OS constantly checks for possible errors.

- The OS takes an appropriate action to ensure correct and consistent computing.

Resource Management

In case of multi-user or multi-tasking environment, resources such as main memory, CPU cycles and files storage are to be allocated to each user or job. Following are the major activities of an operating system with respect to resource management −

- The OS manages all kinds of resources using schedulers.
- CPU scheduling algorithms are used for better utilization of CPU.

Protection

Considering a computer system having multiple users and concurrent execution of multiple processes, the various processes must be protected from each other's activities.

Protection refers to a mechanism or a way to control the access of programs, processes, or users to the resources defined by a computer system. Following are the major activities of an operating system with respect to protection −

- The OS ensures that all access to system resources is controlled.
- The OS ensures that external I/O devices are protected from invalid access attempts.
- The OS provides authentication features for each user by means of passwords.

Reentrant Kernel

Reentrant Kernel: In kernel mode, a reentrant kernel allows processes (or, more precisely, their corresponding kernel threads) to give up the CPU. They have no effect on other processes entering kernel mode. Multiple processor systems may be scheduled together in the case of single-processor systems.

Example:

A disc read is an example of this. When a user program requests a disc read, the scheduler will delegate the CPU to another process (kernel thread) until the disc controller issues an interrupt indicating that the data is accessible and our thread can be resumed. This process can still access I/O, such as user input, which requires kernel functions. The system remains responsive, and the amount of CPU time wasted as a result of IO delays is reduced. The original function (whatever requested data) would be blocked in a non-reentrant kernel until the disc read was completed.

If a computer program or routine can be safely called again before its previous invocation has been completed, it is said to be reentrant (i.e it can be safely executed concurrently). A computer program or routine that is reentrant:

- There cannot be any non-constant data that is static (or global).
- The address must not be returned to static (or global) non-constant data.
- It must only work with the data provided by the caller.
- Locks should not be used to protect singleton resources, a variable that is only referenced once
- It must not change its own code (unless executing in its own unique thread storage).
- Non-reentrant computer programs or routines must not be called.

Non-reentrant functions can still be executed by reentrant kernels if locks are used to ensure that only one process can run the non-reentrant function. Even though the current process is operating in kernel mode, hardware interrupts can suspend it (allowing things like Ctrl+c to cease execution).

Kernel Control Path: A set of instructions that the kernel executes in order to handle a system call. Normally, instructions are executed in order, but some activities force the CPU to interleave control routes. In user mode, the following system call is made: The scheduler chooses a new process to run and switches it on. On behalf of two separate processes, two control pathways are executed.

Monolithic Structure of Operating System

The monolithic operating system is a very basic operating system in which file management, memory management, device management, and process management are directly controlled within the kernel. The kernel can access all the resources present in the system. In monolithic systems, each component of the operating system is contained within the kernel. Operating systems that use monolithic architecture were first time used in the 1970s.

The monolithic operating system is also known as the monolithic kernel. This is an old operating system used to perform small tasks like batch processing and time-sharing tasks in banks. The monolithic kernel acts as a virtual machine that controls all hardware parts.

It is different from a microkernel, which has limited tasks. A microkernel is divided into two parts, ***kernel space***, and ***user space***. Both parts communicate with each other through IPC (Inter-process communication). Microkernel's advantage is that if one server fails, then the other server takes control of it.

Monolithic kernel

A monolithic kernel is an operating system architecture where the entire operating system is working in kernel space. The monolithic model differs from other operating system architectures, such as the microkernel architecture, in that it alone defines a high-level virtual interface over computer hardware.

A set of primitives or system calls implement all operating system services such as process management, concurrency, and memory management. Device drivers can be added to the kernel as modules.

Advantages of Monolithic Kernel

Here are the following advantages of a monolithic kernel, such as:

- The execution of the monolithic kernel is quite fast as the services such as memory management, file management, process scheduling, etc., are implemented under the same address space.
- A process runs completely in single address space in the monolithic kernel.
- The monolithic kernel is a static single binary file.

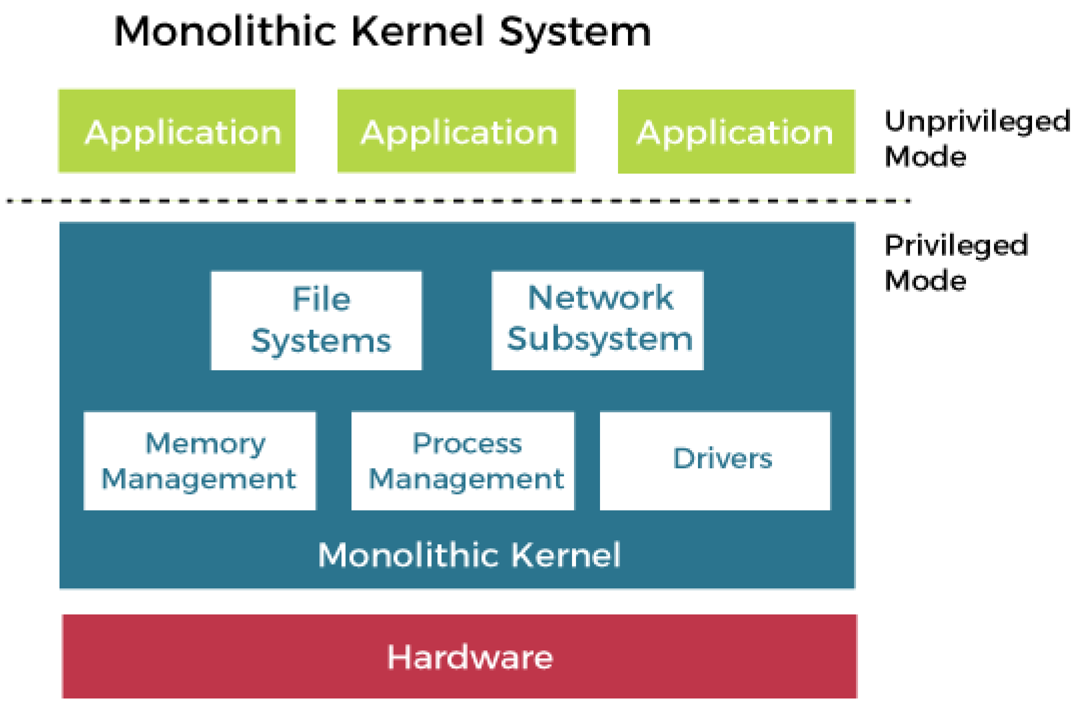
Disadvantages of Monolithic Kernel

Here are some disadvantages of the monolithic kernel, such as:

- If any service fails in the monolithic kernel, it leads to the failure of the entire system.
- The entire operating system needs to be modified by the user to add any new service.

Monolithic System Architecture

A monolithic design of the operating system architecture makes no special accommodation for the special nature of the operating system. Although the design follows the separation of concerns, no attempt is made to restrict the privileges granted to the individual parts of the operating system. The entire operating system executes with maximum privileges. The communication overhead inside the monolithic operating system is the same as that of any other software, considered relatively low.



CP/M and DOS are simple examples of monolithic operating systems. Both CP/M and DOS are operating systems that share a single address space with the applications.

- In CP/M, the 16-bit address space starts with system variables and the application area. It ends with three parts of the operating system, namely CCP (Console Command Processor), BDOS (Basic Disk Operating System), and BIOS (Basic Input/Output System).
- In DOS, the 20-bit address space starts with the array of interrupt vectors and the system variables, followed by the resident part of DOS and the application area and ending with a memory block used by the video card and BIOS.

Advantages of Monolithic Architecture:

Monolithic architecture has the following advantages, such as:

- Simple and easy to implement structure.

- Faster execution due to direct access to all the services

Disadvantages of Monolithic Architecture:

Here are some disadvantages of monolithic architecture:

- The addition of new features or removal of obsolete features is very difficult.
- Security issues are always there because there is no isolation among various servers present in the kernel.

Modular Monolithic Systems

Modular operating systems and most modern monolithic operating systems such as OS-9 OpenVMS, Linux, BSD, SunOS, AIX, and MULTICS can dynamically load (and unload) executable modules at runtime.

This modularity of the operating system is at the binary level and not at the architecture level. Modular monolithic operating systems are not confused with the architectural level of modularity inherent in server-client operating systems (and its derivatives sometimes marketed as the hybrid kernel) that use microkernels and servers.

Limitations of Monolithic System

The monolithic operating system has the following limitations, such as:

- Code written in this operating system (OS) is difficult to port.
- Monolithic OS has more tendencies to generate errors and bugs. The reason is that user processes use the same address locations as the kernel.
- Adding and removing features from monolithic OS is very difficult. All the code needs to be rewritten and recompiled to add or remove any feature.

Features of Monolithic System

The monolithic operating system provides the following features to the users, such as:

1. **Simple structure:** This type of operating system has a simple structure. All the components needed for processing are embedded into the kernel.
2. **Works for smaller tasks:** It works better for performing smaller tasks as it can handle limited resources.
3. **Communication between components:** All the components can directly communicate with each other and also with the kernel.

4. **Fast operating system:** The code to make a monolithic kernel is very fast and robust.

Difference between Monolithic Kernel and Microkernel

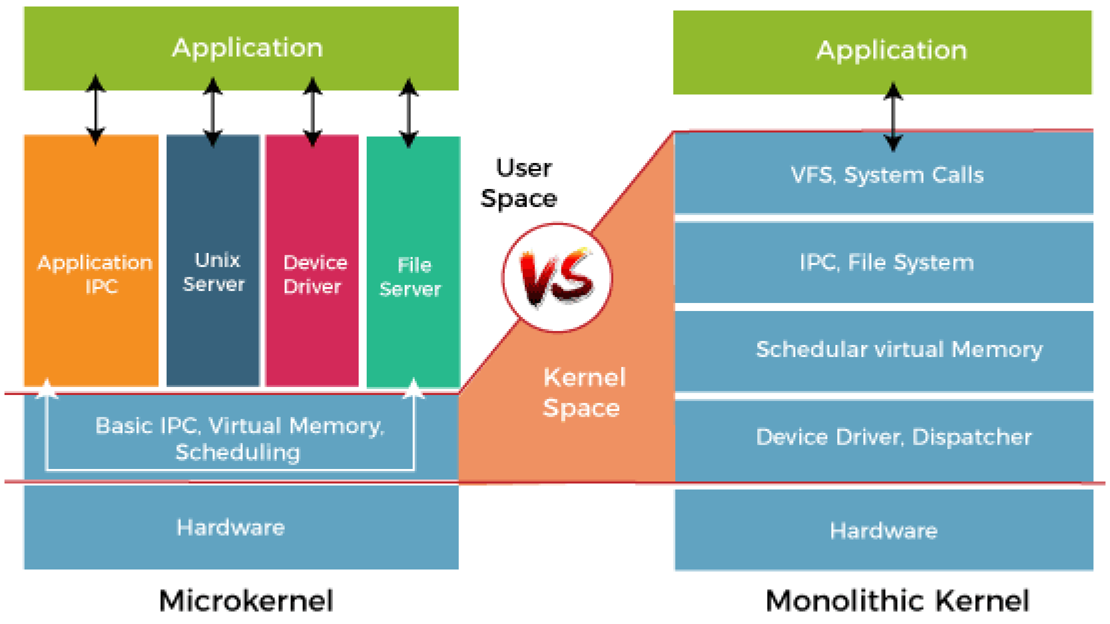
A kernel is the core part of an operating system, and it manages the system resources. A kernel is like a bridge between the application and hardware of the computer. The kernel can be classified further into two categories, Microkernel and Monolithic Kernel.

Terms	Monolithic Kernel	microkernel
Definition	A monolithic kernel is a type of kernel in operating systems where the entire operating system works in the kernel space.	A microkernel is a kernel type that provides low-level address space management, thread management, and interprocess communication to implement an operating system.
Address space	In a monolithic kernel, both user services and kernel services are kept in the same address space.	In microkernel user services and kernel, services are kept in separate address spaces.
Size	The monolithic kernel is larger than the microkernel.	The microkernel is smaller in size.
Execution	It has fast execution.	It has slow execution.
OS services	In a monolithic kernel system, the kernel contains the OS services.	In a microkernel-based system, the OS services and kernel are separated.
Extendible	The monolithic kernel is quite complicated to extend.	The microkernel is easily extendible.
Security	If a service crashes, then the whole system crashes in a monolithic kernel.	If a service crashed, it does not affect the working of the microkernel.
Customization	It is difficult to add new functionalities to the monolithic kernel. Therefore, it is not customizable.	It is easier to add new functionalities to the microkernel. Therefore, it is more customizable.
Code	Less coding is required to write a	A microkernel is required more coding.

	monolithic kernel.	
Example	Linux, FreeBSD, OpenBSD, NetBSD, Microsoft Windows (95, 98, Me), Solaries, HP-UX, DOS, OpenVMS, XTS-400, etc.	QNX, Symbian, L4L.inux, Singularity, K42, Mac OS X, Integrity, PikeOS, HURD, Minix, and Coyotos.

The ***microkernel*** is a type of kernel that allows customization of the operating system. It runs on privileged mode and provides low-level address space management and Inter-Process Communication (IPC). Moreover, OS services such as file system, virtual memory manager, and CPU scheduler are on top of the microkernel. Each service has its own address space to make them secure. Besides, the applications also have their own address spaces. Therefore, there is protection among applications, OS Services, and kernels.

A *monolithic kernel* is another classification of the kernel. In monolithic kernel-based systems, each application has its own address space. Like microkernel, this one also manages system resources between application and hardware, but ***user services*** and ***kernel services*** are implemented under the same address space. It increases the size of the kernel, thus increases the size of the operating system as well.



This kernel provides CPU scheduling, memory management, file management, and other system functions through system calls. As both services are implemented under the same address space, this makes operating system execution faster. Below are some more differences between Microkernel and Monolithic kernel, such as: