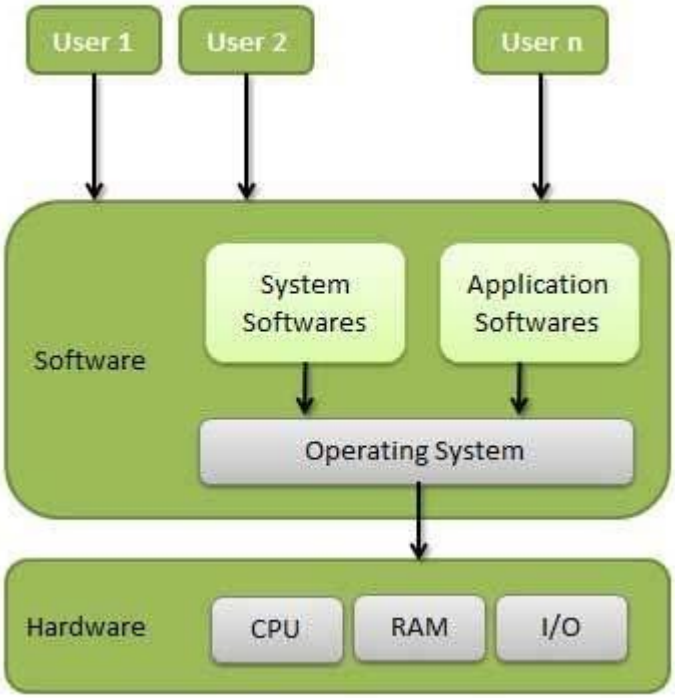
MODULE -1

Introduction of Operating System

An operating system (OS) is the program that, after being initially loaded into the computer by a boot program, manages all of the other application programs in a computer. The application programs make use of the operating system by making requests for services through a defined application program interface (API). In addition, users can interact directly with the operating system through a user interface, such as a command-line interface (CLI) or a graphical UI (GUI).

Definition

An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.



The operating system's job

Your computer's **operating system** (**OS**) manages all of the **software** and **hardware** on the computer. Most of the time, there are several different computer programs running at the same time, and they all need to access your computer's **central processing unit (CPU)**, **memory**, and **storage**. The operating system coordinates all of this to make sure each program gets what it needs.

Functions of Operating System

Some typical operating system functions may include managing memory, files, processes, I/O system & devices, security, etc.

Functions of Operating System

In an operating system software performs each of the function:

1. **Process management:-** Process management helps OS to create and delete processes. It also provides mechanisms for synchronization and communication among processes.

2. **Memory management:-** Memory management module performs the task of allocation and de-allocation of memory space to programs in need of this resources.

3. **File management:-** It manages all the file-related activities such as organization storage, retrieval, naming, sharing, and protection of files.

4. **Device Management:** Device management keeps tracks of all devices. This module also responsible for this task is known as the I/O controller. It also performs the task of allocation and de-allocation of the devices.

5. **I/O System Management:** One of the main objects of any OS is to hide the peculiarities of that hardware devices from the user.

6. **Secondary-Storage Management:** Systems have several levels of storage which includes primary storage, secondary storage, and cache storage.

Instructions and data must be stored in primary storage or cache so that a running program can reference it.

7. **Security:-** Security module protects the data and information of a computer system against malware threat and authorized access.

8. **Command interpretation:** This module is interpreting commands given by the and acting system resources to process that commands.

9. **Networking:** A distributed system is a group of processors which do not share memory, hardware devices, or a clock. The processors communicate with one another through the network.

10. **Job accounting:** Keeping track of time & resource used by various job and users.

11. **Communication management:** Coordination and assignment of compilers, interpreters, and another software resource of the various users of the computer systems.

Features of Operating System (OS)

Here is a list important features of OS:

1. Protected and supervisor mode
2. Allows disk access and file systems Device drivers Networking Security
3. Program Execution
4. Memory management Virtual Memory Multitasking
5. Handling I/O operations
6. Manipulation of the file system
7. Error Detection and handling
8. Resource allocation
9. Information and Resource Protection

Advantage of using Operating System

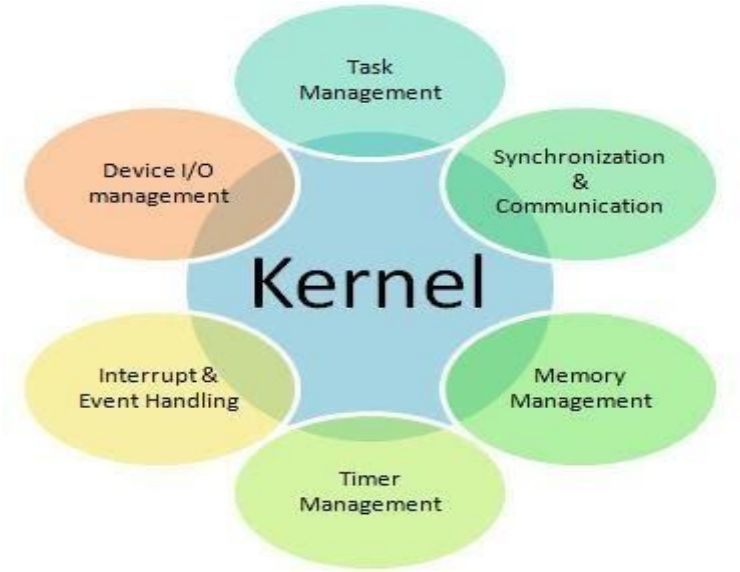
- Allows you to hide details of hardware by creating an abstraction
- Easy to use with a GUI
- Offers an environment in which a user may execute programs/applications
- The operating system must make sure that the computer system convenient to use
- Operating System acts as an intermediary among applications and the hardware components
- It provides the computer system resources with easy to use format
- Acts as an intermediator between all hardware's and software's of the system

Disadvantages of using Operating System

- If any issue occurs in OS, you may lose all the contents which have been stored in your system
- Operating system's software is quite expensive for small size organization which adds burden on them. Example Windows
- It is never entirely secure as a threat can occur at any time

What is Kernel in Operating System?

The kernel is the central component of a computer operating systems. The only job performed by the kernel is to the manage the communication between the software and the hardware. A Kernel is at the nucleus of a computer. It makes the communication between the hardware and software possible. While the Kernel is the innermost part of an operating system, a shell is the outermost one.



Function of kernel

The kernel is the core component of an operating system which acts as an interface between applications, and the data is processed at the hardware level.

When an OS is loaded into memory, the kernel is loaded first and remains in memory until the OS is shut down. After that, the kernel provides and manages the computer resources and allows other programs to run and use these resources. The kernel also sets up the memory address space for applications, loads the files with application code into memory, and sets up the execution stack for programs.

The kernel is responsible for performing the following tasks:

- Input-Output management
- Memory Management
- Process Management for application execution.
- Device Management
- System calls control

Earlier, all the basic system services like process and memory management, interrupt handling, etc., were packaged into a single module in the kernel space.

This type of kernel was called the Monolithic Kernel. The problem with this approach was that the whole kernel had to be recompiled for even a small change.

In a modern-day approach to monolithic architecture, a microkernel contains different modules like device management, file management, etc. It is dynamically loaded and unloaded. With this modern-day approach, the kernel code size was reduced while its stability increased.

Types of Kernels

There are many types of kernels that exists, but among them, the two most popular kernels are:

1. Monolithic

A monolithic kernel is a single code or block of the program. It provides all the required services offered by the operating system. It is a simplistic design which creates a distinct communication layer between the hardware and software.

2. Microkernels

Microkernel manages all system resources. In this type of kernel, services are implemented in different address space. The user services are stored in user address space, and kernel services are stored under kernel address space. So, it helps to reduce the size of both the kernel and operating system.

Difference between Firmware and Operating System

Firmware	Operating System
Define Firmware: Firmware is one kind of programming that is embedded on a chip in the device which controls that specific device.	Define Operating System: OS provides functionality over and above that which is provided by the firmware.

Firmware is programs that been encoded by the manufacture of the IC or something and cannot be changed.	OS is a program that can be installed by the user and can be changed.
It is stored on non-volatile memory.	OS is stored on the hard drive.

Difference between 32-Bit vs. 64 Bit Operating System

SR. No.	**Parameters**	**32. Bit**	**64. Bit**
1.	Architecture and Software	Allow 32 bit of data processing simultaneously	Allow 64 bit of data processing simultaneously
2.	Compatibility	32-bit applications require 32-bit OS and CPUs.	64-bit applications require a 64-bit OS and CPU.
3.	Systems Available	All versions of Windows 8, Windows 7, Windows Vista, and Windows XP, Linux, etc.	Windows XP Professional, Vista, 7, Mac OS X and Linux.
4.	Memory Limits	32-bit systems are limited to 3.2 GB of RAM.	64-bit systems allow a maximum 17 Billion GB of RAM.

Operating System as a Resource Manager

A computer has many resources (Hardware and Software), which may be required to complete a task. The commonly required resources are Input/output devices, Memory file storage space, CPU (Central Processing Unit) time and so on.

The operating system acts as the manager of these resources and allocates them to specific programs and users as necessary for their tasks. Therefore we can say an operating system is a **resource allocator**. This is the main features of an operating system.

When a number of computers connected through a network more than one computers trying for a computer print or a common resource, then the operating system follow same order and manage the resources in an efficient manner.

Generally, resources sharing in two ways "in time" and "in space". When a resource is a time-sharing resource, first one of the tasks get the resource for some time, then another and so on.

For example, a CPU in the time-sharing system. In the time-sharing system, OS fixes the time slot for the CPU. First one of the processes gets the CPU, when the time slot expires, the CPU switch to the next process in the ready queue. In this example, the CPU is a time resource. CPU Scheduling in Operating System

The other kind of sharing is the "**space sharing**". In this method, the users share the space of resource. For example, the main memory consisting of several processes at a time, so the process of sharing the resources.

The main difference between "in time" sharing resources and "in space" sharing resources is "in time" resources are not divided into units, whereas "in space" sharing resources are divided into units.

Memory Management

Memory management refers to management of Primary Memory or Main Memory. Main memory is a large array of words or bytes where each word or byte has its own address.

Main memory provides a fast storage that can be accessed directly by the CPU. For a program to be executed, it must in the main memory. An Operating System does the following activities for memory management −

- Keeps tracks of primary memory, i.e., what part of it are in use by whom, what part are not in use.
- In multiprogramming, the OS decides which process will get memory when and how much.
- Allocates the memory when a process requests it to do so.
- De-allocates the memory when a process no longer needs it or has been terminated.

Processor Management

In multiprogramming environment, the OS decides which process gets the processor when and for how much time. This function is called **process scheduling**. An Operating System does the following activities for processor management −

- Keeps tracks of processor and status of process. The program responsible for this task is known as **traffic controller**.
- Allocates the processor (CPU) to a process.
- De-allocates processor when a process is no longer required.

Device Management

An Operating System manages device communication via their respective drivers. It does the following activities for device management −

- Keeps tracks of all devices. Program responsible for this task is known as the **I/O controller**.
- Decides which process gets the device when and for how much time.
- Allocates the device in the efficient way.
- De-allocates devices.

File Management

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions.

An Operating System does the following activities for file management −

- Keeps track of information, location, uses, status etc. The collective facilities are often known as **file system**.
- Decides who gets the resources.
- Allocates the resources.
- De-allocates the resources.

Other Important Activities

Following are some of the important activities that an Operating System performs −

- **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 software's and users** − Coordination and assignment of compilers, interpreters, assemblers and other software to the various users of the computer systems.

Operating system types and examples

Although the fundamental roles of an operating system are ubiquitous, there are countless operating systems that serve a wide range of hardware and user needs.

General-purpose operating system. A general-purpose OS represents an array of operating systems intended to run a multitude of applications on a broad selection of hardware, enabling a user to run one or more applications or tasks simultaneously. A general-purpose OS can be installed on many different desktop and laptop models and run applications from accounting systems to databases to web browsers to games. General-purpose operating systems typically focus on process (thread) and hardware management to ensure that applications can reliably share the wide range of computing hardware present.

Common desktop operating systems include the following:

- Windows is Microsoft's flagship operating system, the de facto standard for home and business computers. Introduced in 1985, the GUI-based OS has been released in many versions since then. The user-friendly Windows 95 was largely responsible for the rapid development of personal computing.
- Mac OS is the operating system for Apple's Macintosh line of PCs and workstations.
- Unix is a multiuser operating system designed for flexibility and adaptability. Originally developed in the 1970s, Unix was one of the first operating systems to be written in the C language.

- Linux is a Unix-like operating system that was designed to provide PC users a free or low-cost alternative. Linux has a reputation as an efficient and fast-performing system.

Mobile operating system. Mobile operating systems are designed to accommodate the unique needs of mobile computing and communication-centric devices, such as smartphones and tablets. Mobile devices typically offer limited computing resources compared to traditional PCs, and the OS must be scaled back in size and complexity in order to minimize its own resource use, while ensuring adequate resources for one or more applications running on the device. Mobile operating systems tend to emphasize efficient performance, user responsiveness and close attention to data handling tasks, such as supporting media streaming. Apple iOS and Google Android are examples of mobile operating systems.

Embedded operating system. Not all computing devices are general purpose. A huge assortment of dedicated devices -- including home digital assistants, automated teller machines (ATMs), airplane systems, and retail point of sale (POS) terminals and internet of things (IoT) devices -- includes computers that require an operating system. The principal difference is that the associated computing device only does one major thing, so the OS is highly stripped down and dedicated to both performance and resilience. The OS should run quickly, not crash, and handle all errors gracefully in order to continue operating in all circumstances. In most cases, the OS is provided on a chip that is incorporated into the actual device. A medical device used in a patient's life support equipment, for example, will employ an embedded OS that must run reliably in order to keep the patient alive. Embedded Linux is one example of an embedded OS.

Evolution of Operating System

Generations Duration	1st (1946-1959)	2nd (1959-1965)	3rd (1965-1970)	4th (1970-1981)	5th (1981-onwards)
Major Innovation	Vacuum Tubes	Transistors as Main component	Integrated Circuit (ICs) as basic electronic component	LSIC & VLSIC (Microprocessor)	ULSIC (Ultra Large Scale Integrated Circuit)
Main Memory	Magnetic Drums	RAM and ROM	PROM & DRAM	EPROM & SRAM	EEPROM, SIMM & DIMM
External Storage	Punched cards	Magnetic capes and magnetic disk	Improve disk (Floppy disk)	Floppy disk & Hard disk	Modified magnetic and optical disks
Input/Output devices	Punched cards & paper	Magnetic tape, Punched card, paper for output	Keyboard for input and Monitor for output	Monitor for Output	Keyboard, Pointing device, Scanner as input and Monitor as main output
Languages	Low level machine language	Assembly-language, some high level languages for example BASIC, COLOL, FORTRAN	More high level languages	Languages and application software	AI (Artificial intelligence) Expert systems
Operating system	No operating system, human operators to set switches	Human handles punched card	Complete operating system were introduced	MS-DOS & PC-DOS	GUI based e.g. Windows 95 and Windows NI
size	Main frame for example ENLAC, EDVAC, UNIVAC	Main frame for example IBM-1401, NCR-300, IBM-600 etc	Mini, for example IBM system/360 ICH-360, Honey well 316 etc	Micro computer e.g. IBM-PC, Apple, Macintosh etc	Very small in size e.g. Laptop, Note book, Digital diary, Palm top and Pocket PC

Subject	1st generation	2nd generation	3rd generation	4th generation	5th generation
Period	1940-1956	1956-1963	1964-1971	1971-present	present & beyond
Circuitry	Vacuum tube	Transistor	Integrated chips (IC)	Microprocessor (VLSI)	ULSI (Ultra Large Scale Integration) technology
Memory Capacity	20 KB	128KB	1MB	Magnetic core memory, LSI and VLSI. High Capacity	ULSI
Processing Speed	300 IPS instructions Per sec.	300 IPS	1MIPS (1 million inst. Per sec.)	Faster than 3rd generation	Very fast
Programming Language	Machine, Language	Assembly language & early high-level languages(FORTRAN, COBOL, ALGOL)	C,C++	Higher level languages,C,C++,Java	All the Higher level languages,,Neural networks,
Example of computers	UNIVAC, EDVAC	IBM 1401, IBM 7094, CDC 3600,D UNIVAC 1108	IBM 360 series, 1900 series	Pentium series,Multimedia,	Artificial Intelligence, Robotics

- The first computer, Z1, was made in 1936 – 1938. Unfortunately, this computer ran without an operating system.
- Twenty years later, the first-ever operating system was made in 1956.
- In the 1960s, bell labs started working on building UNIX, the first multitasking operating system.
- In 1977 the apple series came into existence. Apple Dos 3.3 was the first disk operating system.
- In 1981, Microsoft built the first operating system called DOS by purchasing 86 – DOS software from a Seattle company.
- The most famous Microsoft windows came into existence in 1985 when MS-DOS was paired with GUI, a graphics environment.

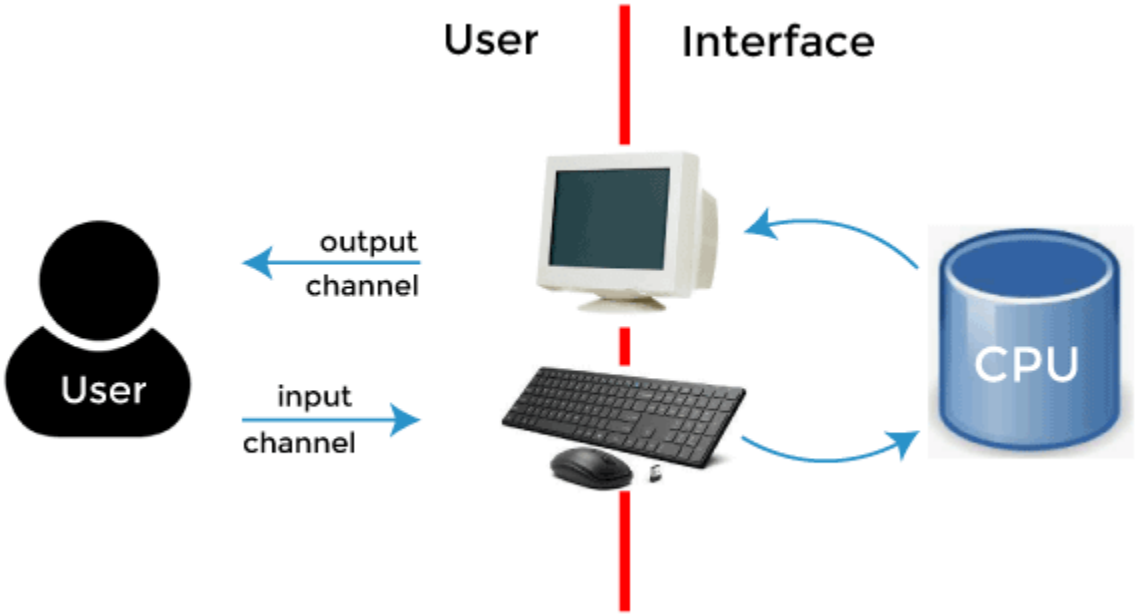
Types of operating system

1. Single User operating system
2. Batch operating system
3. Multiprogramming operating system

4. Multiprocessor operating system
5. Multiuser operating system
6. Multitasking operating system
7. Multithreading operating system
8. Real Time operating system
9. Time Sharing operating system
10. Network operating system
11. Distributed operating system

1. Single User operating system:- A single-user operating system is a type of operating system developed and intended for use on a computer or similar machine that will only have a single user at any given time. This type of OS is typically used on devices like wireless phones and two-way messaging devices.

The operating system is responsible for handling many different tasks and is typically one of the most important programs used on a computer. It manages memory usage and other resources, hardware connectivity and the proper execution of other applications. A single task operating system can only run one program or application at a time. So it is not as useful for a computer or other device intended to run multiple programs at once.



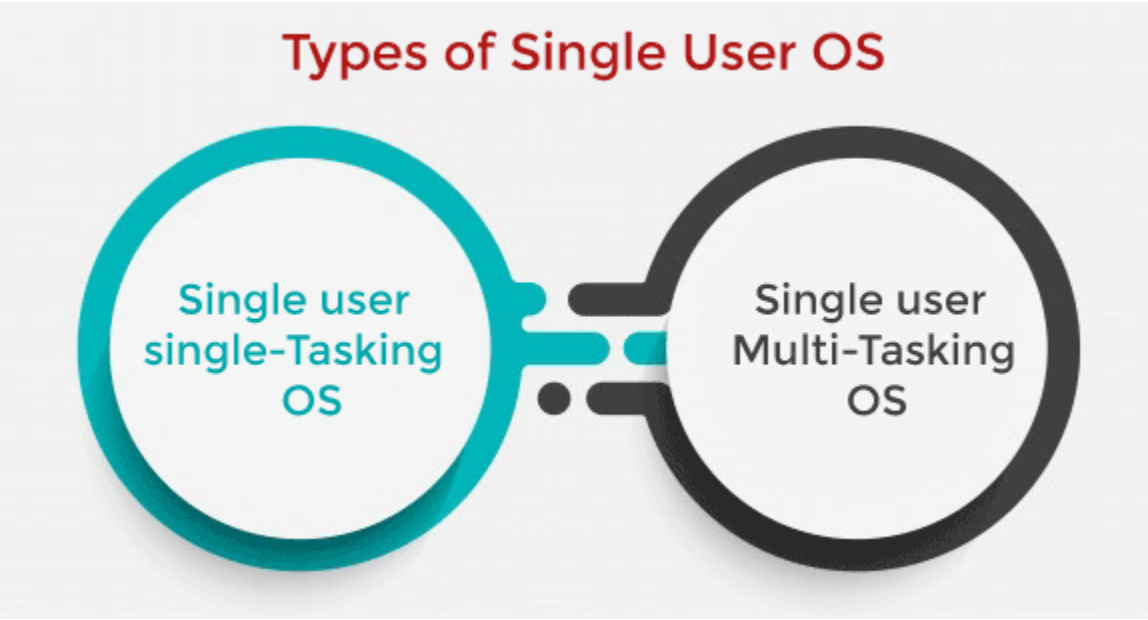
This is where a multitasking single-user operating system is used instead. A multitasking OS can run multiple applications and programs at once. This is often

used on computers where someone may want to navigate the internet, run a graphics editing program, play music through a media playing program, and type in notes in a simple word processing program all at the same time. A single task OS could not do this, but the multitasking systems can handle all these processes.

Even though this type of operating system can connect to other computers through a network, it is still only being used by a single user. As long as the computer only has one monitor, keyboard and other input devices, then it is a single-user system.

Types of Single-User Operating System

Single user operating system can be classified into two parts, such as:



1. Single-User Single-Tasking operating system

In the Single-User Single-Tasking operating system, only one user is permitted for performing a single task at a time. Some functions such as printing a document and downloading images and videos are performed in one given frame. This operating system is designed especially for wireless phones as well as two-way messaging devices. For example, MS-DOS, Palm OS, etc. It has the following advantages:

- This operating system occupies less space in memory.
- It is cost-effective.

2. Single-User Multi-Tasking operating system

Single-User Multi-Tasking operating system is developed especially for one user, but this single user can perform multiple tasks simultaneously. For example, you can write any text while surfing the internet, downloading images, watching movies, etc., on Windows, Linux, Mac O/S. It has the following advantages:

- Time-saving.
- High productivity in less time frame.
- Less memory is used for performing multiple tasks.

Features of Single User Operating System

Single user operating system provides the following features to the user, such as:

- It does not use the scheduling process for I/O.
- It uses less scheduling for the users.
- It is only dedicated to single-use.
- It is not intended for several tasks at the same given time.
- It does not use MMU.

Functions of a Single User Operating System

Description and exemplification are the main functions of a single user operating system. Below are some more functions of single-user OS, such as:

1. **Interpreting users commands:** It takes the instructions from the user and passes them on to the rest of the operating system. This is the part of the OS that the user interacts with to give instructions to the computer. It can be command-driven, menu-driven or a GUI, which may use a WIMP system.
2. **File management:** It controls the organization of backing storage. It uses part of the disc as a file catalogue, which holds details of where data is stored on the disc. Files are organized into a hierarchical filing system storing files in separate directories and subdirectories.
3. **Memory management:** It controls where programs and data are placed in the main memory. Many OS's allow more than one program to be in memory at the same time. Memory management ensures that any program does not overwrite any other program (including the OS itself).
4. **Input/output management:** It communicates between the computer system and the input and output devices. It handles the transfer of data between peripherals and the CPU.

5. **Resource Allocation:** The OS allocates resources to other programs such as applications. For example:
    - Scheduler to share processor, e.g. batch, foreground or background, timesharing.
    - Memory management - e.g. Virtual memory paging, swapping memory contents to hard disc.
    - Ensures drivers are given access to hardware devices.
    - Mechanisms for accounting; limit enforcement - e.g. no. of pages printed, file space per user etc.
6. Managing Processes: It controls processes and handles interrupt**ions. The kernel** manages processes (when a program runs, it starts a process, which may start further processes) and handles interrupts.

Examples of Single User Operating System

There are some operating system examples like DOS, Windows 3x, Windows 95, 97, 98. Also, in mobile phones, Symbian OS is single-user OS.

These systems are also known are single application OS as one application is concentrated at a time. These systems require fewer resources and can do processing in a queue. The recurring tasks are performed well in these systems, such as making payroll or employees' salaries.

Advantages of Single User Operating System

Single-user operating is that in which one user works on one interface. In these systems, no other operating system is interrupting the processing. Single-user operating systems work on processing that requires fewer resources. Below are some more important advantages of single-user OS, Such as:

- **Supports one user at a time:** In these systems, one user is only active at a time. So there will be no other user interfering with the applications. And in these systems, all computer resources are used by user requests.
- **Easy to maintain:** These systems use fewer resources, and their complexity is less, making them easy to maintain and debug. Higher resources are needed in a multi-user operating system, and resources are used most of the time.

- **Less chance to damage:** These systems include fewer requests to hardware and software at a time, so they have less chance to damage. These systems do not make higher load time also.
- **Concentrate on one task:** In a modern operating system, there are running multiple tasks at a time. Like many applications and tasks are running simultaneously, but in single-user OS, only one task runs at a time. So these systems sometimes give less output result at a time.

Disadvantages of Single User Operating System

Single user operating system also has some disadvantages, such as:

- **Tasks take longer to complete:** As you know, many tasks are waiting for the CPU if no multiple tasks run at a time. So these systems respond to processes at a higher time. This will make the system slow, and response time is higher.
- **Idle time is higher:** If only one task is running and this task doesn't require memory or I/O use, these devices remain idle. But other tasks need those devices. So only one task is run at a time, then other tasks have to wait till the first task is finished. So CPU, memory and disk I/O are not used properly.

Difference between Single User and Multi-User OS

The difference between single-user operating systems and multi-user operating systems are given below:

Terms	Single User Operating System	Multi-User Operating System
Definition	A Single-User Operating System is a system in which only one user can access the computer system at a time.	A Multi-User Operating System is a system that allows more than one user to access a computer system at one time.
Bootloader	The bootloader loads all the resources of the CPU in the profile of a single superuser.	The bootloader distributes the resources of the CPU based on the priority of the users.

Super User	A superuser gets all the powers of maintaining the system and making changes to ensure the system runs smoothly.	Superuser does not exist for a multi-user operating system as each entity has control over their working.
Complexity	Single-User Operating System is simple and easy to design.	The Multi-User Operating systems are complicated as they have to apply complex techniques to share the resources among various users.
Performance	Only one task at one time gets performed.	Schedules different tasks for performance at any rate.
Example	MS-DOS, Windows 95, Windows NT, Windows 2000, Personal Computers, etc.	Mainframes, IBM AS400, Linux & Unix Distributed OS, etc.

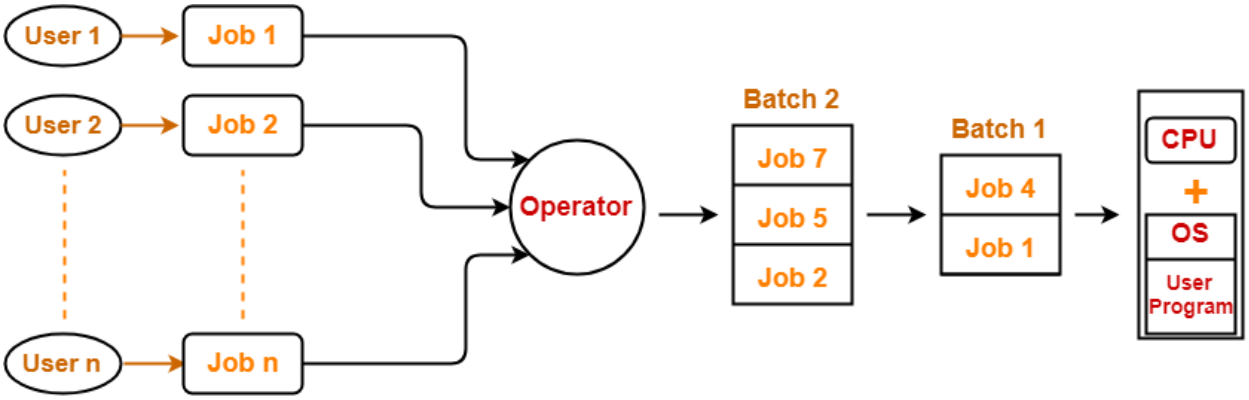
2. Batch Operating System: - In the beginning, computers were extremely large machines that ran from a console. In general, tape drivers or card readers were used for input, and tape drives, punch cards, and line printers were used for output. Users had no direct interface with the system, and job execution 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.

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.



Batch Operating System

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.

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.

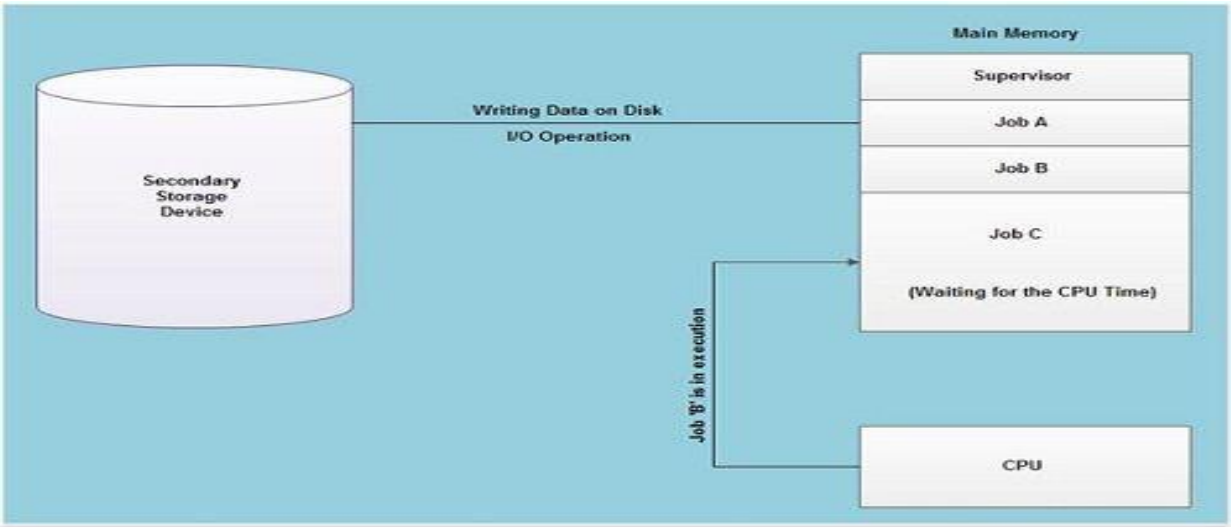
4. When a process is finished, the next job from the job spool is run without any user interaction.
5. CPU utilization gets improved.

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.

3. Multiprogramming Operating System:- In multi-programming system, when one program is waiting for I/O transfer; there is another program ready to utilize the CPU. So it is possible for several jobs to share the time of the CPU. But it is important to note that multi-programming is not defined to be the execution of jobs at the same instance of time. Rather it does mean that there are a number of jobs available to the CPU (placed in main memory) and a portion of one is executed then a segment of another and so on. A simple process of multi-programming is shown in figure



As shown in fig, at the particular situation, job' A' is not utilizing the CPU time because it is busy in I/ 0 operations. Hence the CPU becomes busy to execute the job 'B'. Another job C is waiting for the CPU for getting its execution time. So in this state the CPU will never be idle and utilizes maximum of its time.

A program in execution is called a "Process", "Job" or a "Task". The concurrent execution of programs improves the utilization of system resources and enhances the system throughput as compared to batch and serial processing. In this system, when a process requests some I/O to allocate; meanwhile the CPU time is assigned to another ready process. So, here when a process is switched to an I/O operation, the CPU is not set idle.

Multi-programming is a common approach to resource management. The essential components of a single-user operating system include a command processor, an input/ output control system, a file system, and a transient area. A multi-programming operating system builds on this base, subdividing the transient area to hold several independent programs and adding resource management routines to the operating system's basic functions.

Advantages

There are various advantages of the multiprogramming operating system. Some of the advantages are as follows:

1. It provides less response time.
2. It may help to run various jobs in a single application simultaneously.
3. It helps to optimize the total job throughput of the computer.

4. Various users may use the multiprogramming system at once.
5. Short-time jobs are done quickly in comparison to long-time jobs.
6. It may help to improve turnaround time for short-time tasks.
7. It helps in improving CPU utilization and never gets idle.
8. The resources are utilized smartly.

Disadvantages

There are various disadvantages of the multiprogramming operating system. Some of the disadvantages are as follows:

1. It is highly complicated and sophisticated.
2. The CPU scheduling is required.
3. Memory management is needed in the operating system because all types of tasks are stored in the main memory.
4. The harder task is to handle all processes and tasks.
5. If it has a large number of jobs, then long-term jobs will require a long wait.

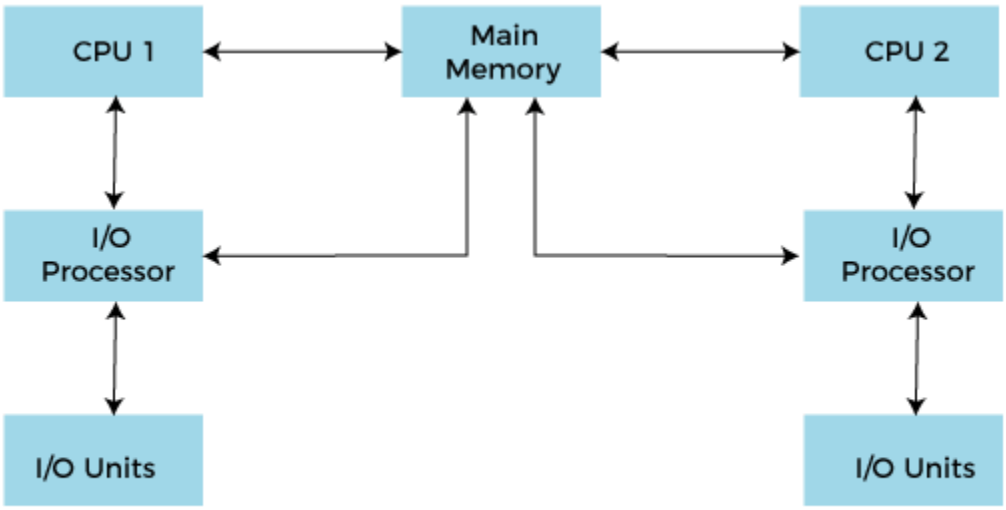
Examples of Multiprogramming Operating System

There are various examples of multiprogramming operating systems, including download apps, transfer data, MS-Excel, Google Chrome, Firefox browser, and many more apps. Other examples are Windows O/S, UNIX O/S, Microcomputers such as XENIX, MP/M, and ESQview.

Multiprocessor Operating System: - 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.



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.

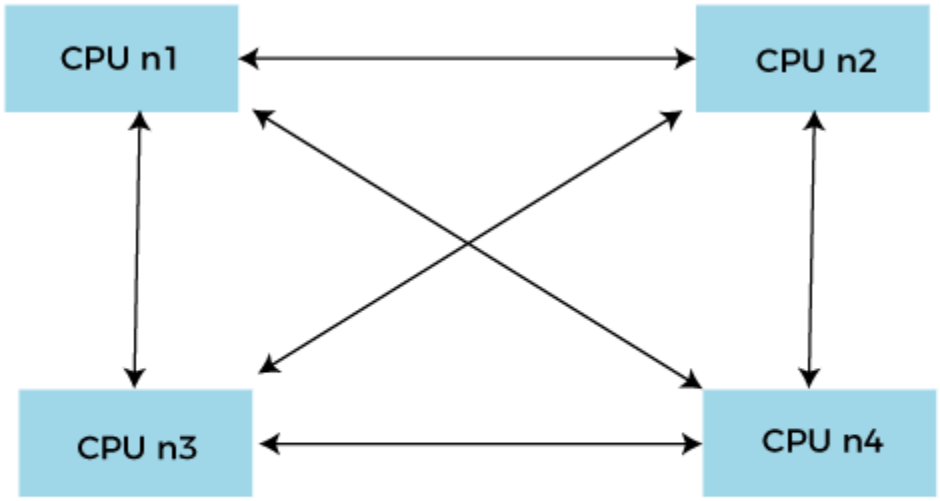
Types of multiprocessing systems

- 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.



Symmetrical Processing with 4 CPUs

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:

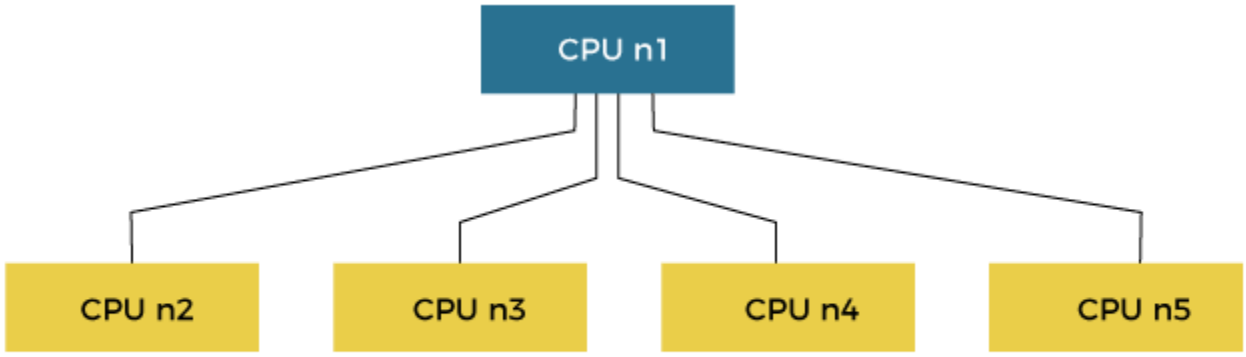
- These systems are fault-tolerant. Failure of a few processors does not bring the entire system to a halt.

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.



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:

There are multiple advantages to multiprocessor systems. Some of these are −

More reliable Systems

In a multiprocessor system, even if one processor fails, the system will not halt. This ability to continue working despite hardware failure is known as graceful degradation. For example: If there are 5 processors in a multiprocessor system and one of them fails, then also 4 processors are still working. So the system only becomes slower and does not ground to a halt.

Enhanced Throughput

If multiple processors are working in tandem, then the throughput of the system increases i.e. number of processes getting executed per unit of time increase. If there are N processors then the throughput increases by an amount just under N.

More Economic Systems

Multiprocessor systems are cheaper than single processor systems in the long run because they share the data storage, peripheral devices, power supplies etc. If there are multiple processes that share data, it is better to schedule them on multiprocessor systems with shared data than have different computer systems with multiple copies of the data.

Disadvantages of Multiprocessor Systems

There are some disadvantages as well to multiprocessor systems. Some of these are:

Increased Expense

Even though multiprocessor systems are cheaper in the long run than using multiple computer systems, still they are quite expensive. It is much cheaper to buy a simple single processor system than a multiprocessor system.

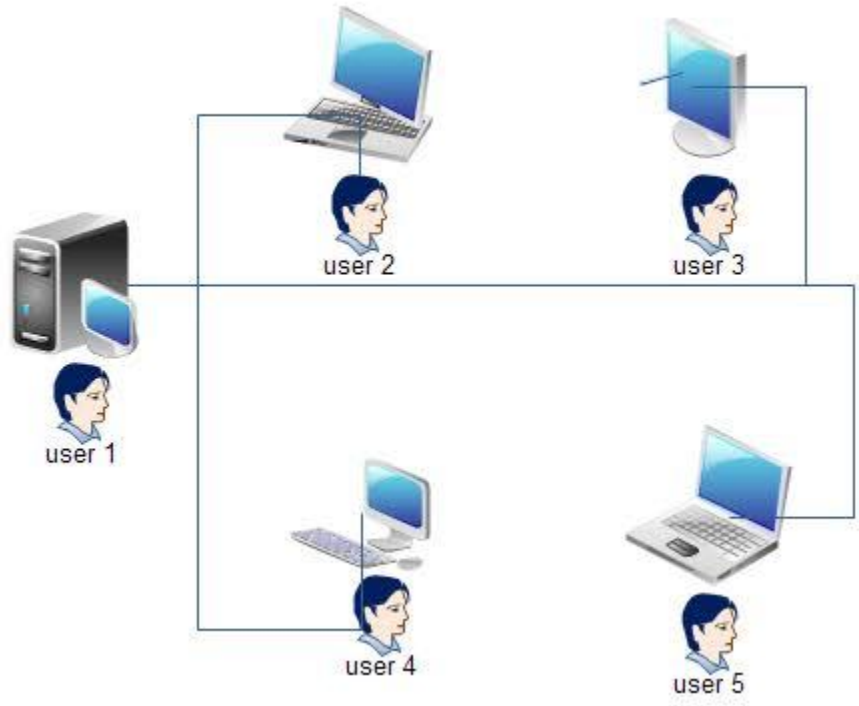
Complicated Operating System Required

There are multiple processors in a multiprocessor system that share peripherals, memory etc. So, it is much more complicated to schedule processes and impart resources to processes than in single processor systems. Hence, a more complex and complicated operating system is required in multiprocessor systems.

Large Main Memory Required

All the processors in the multiprocessor system share the memory. So a much larger pool of memory is required as compared to single processor systems.

Multiuser operating system A Multi-user operating system is a computer operating system which allows multiple users to access the single system with one operating system on it. It is generally used on large mainframe computers. **Example**: Linux, Unix, Windows 2000, Ubuntu, Mac OS etc., In the multi-user operating system, different users connected at different terminals and we can access, these users through the network as shown in the diagram.



Features of the Multi-user Operating System

• Multi-tasking- Using multi-user operating system we can perform multiple tasks at a time, i.e. we can run more than one program at a time. Example: we can edit a word document while browsing the internet.

Resource sharing- we can share different peripherals like printers, hard drives or we can share a file or data. For this, each user is given a small time slice of CPU time.
• Background processing- It means that when commands are not processed firstly, then they are executed in the background while another programs are interacting with the system in the real time.

Types of Multi-user Operating System

A multi-user operating system is of 3 types which are as follows:
1. **Distributed Systems**: in this, different computers are managed in such a way so that they can appear as a single computer. So, a sort of network is formed through which they can communicate with each other.
2. **Time-sliced Systems**: in this, a short period is assigned to each task, i.e. each user is given a time slice of the CPU time. As we know these time slices are tiny, so it appears to the users that they all are using the mainframe computer at the same time.
3. **Multiprocessor Systems**: in this, the operating system utilises more than one processor.

Example: Linux, Unix, Windows XP

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.

Characteristics of Multi-User Operating System

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

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.

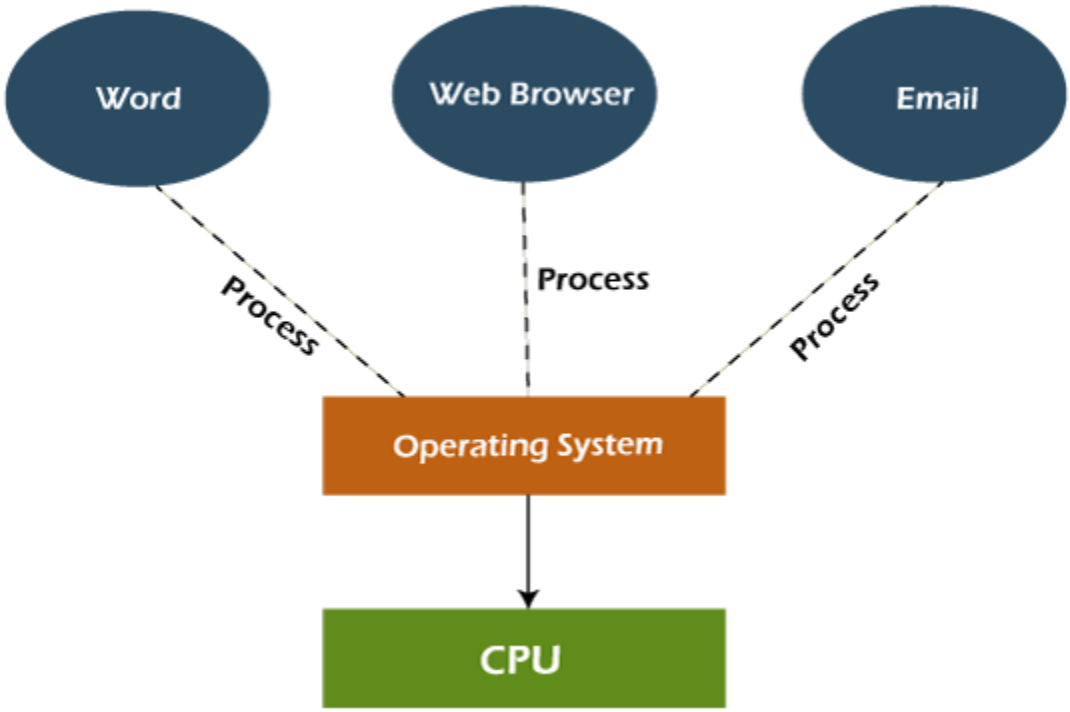
Disadvantages of Multi-User Operating System

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.

Multitasking operating system Multitasking term used in a modern computer system. Multitasking is a logical extension of a multiprogramming system that supports multiple programs to run concurrently. In a **multitasking operating system**, more than one task is executed at the same time. In this technique, multiple tasks, also known as processes, share common processing resources such as a CPU.



Types of Multitasking Operating System:

There are two types of multitasking operating system

Cooperative Multitasking:

Windows and Mac OS used cooperative multitasking. A Windows program would do some small unit of work in response to a message and then relinquish the CPU to the operating system until the program got another message. That worked well, as long as all programs were written with consideration for other programs and had no **bugs**.

Preemptive Multitasking:

Desktop operating systems use preemptive multitasking. Unix used this form of multitasking from the beginning. Windows starting using preemptive multitasking with Windows NT and Windows 95. Macintosh gained preemptive multitasking with OS X. This **type of operating system** programs to tell them it's time to give another program a turn with the CPU.

Multitasking Operating System Benefits:

In Multitasking Operating System, memory, hard drive, and virtual memory are better managed. Processor time is also well used as the wait time is less.

Suppose any user is downloading any file from the internet. Now he can do other tasks on the computer without waiting for the file to download completely.

One process cannot change data of another process in the main memory. Each process can only use his own space allocated.

If any program generates errors then other programs running on the computer are not affected by it. For example, if Google chrome crashes then your data in MS Word is not affected.

CMT can handle multiple users at a time. Every user can run their own programs independently of other users.

Advantages

Various advantages of multitasking are as follows:

Manage Several Users

This operating system is more suited to supporting multiple users simultaneously, and multiple apps can run smoothly without interfering with system performance.

Virtual Memory

The greatest virtual memory system is found in multitasking operating systems. Because of virtual memory, any program does not require a long wait time to complete its tasks; if this problem arises, those programs are moved to virtual memory.

Good Reliability

Multitasking operating systems give more flexibility to several users, and they are happier as a result. On which each user can execute single or multiple programs simultaneously.

Secured Memory

The multitasking operating systems have well-defined memory management. Due to this operating system does not allow any types of permissions for undesirable programs to waste memory.

Time Shareable

All tasks are allotted a specified amount of time so that they do not have to wait for the CPU.

Background Processing

A multitasking operating system provides a better environment for background processes to run. These background programs are not visible to most users, but they help other programs like firewalls, antivirus software, and others run well.

Optimize the computer resources

A multitasking operating system may manage various computer resources like I/O devices, RAM, hard disk, CPU, and others.

Use Several Programs

Users can run many programs simultaneously, like an internet browser, games, MS Excel, PowerPoint, and other utilities.

Disadvantages

Various disadvantages of multitasking are as follows:

Processor Boundation

The system may run programs slowly because of the poor speed of their processors, and their reaction time might rise when processing many programs. To solve this problem, more processing power is required.

Memory Boundation

The computer's performance may get slow performance due to the multiple programs run at the same time because the main memory gets overloaded while loading multiple programs. Because the CPU is unable to provide different times for each program, reaction time increases. The primary cause of this issue is that it makes use of low-capacity RAM. As a result, the RAM capacity can be raised to provide a solution.

CPU Heat Up

The multiple processors are busier at the same time to complete any task in a multitasking environment, so the CPU generates more heat.

Multithreading operating system

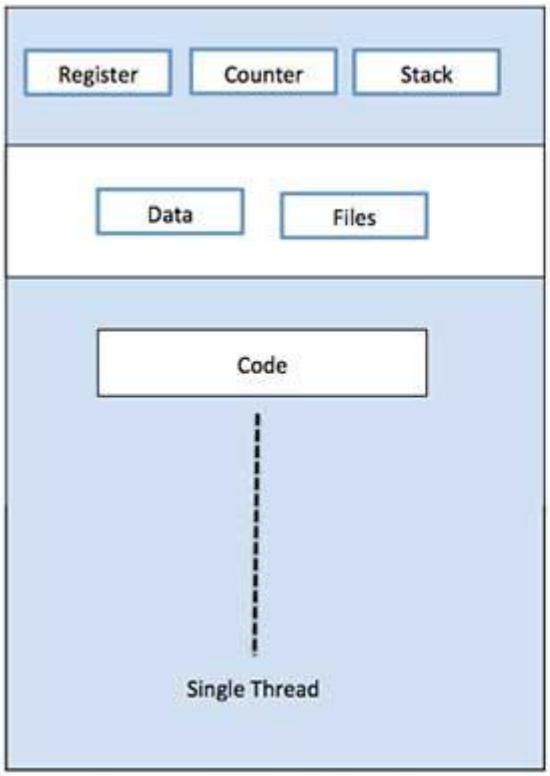
What is Thread?

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.

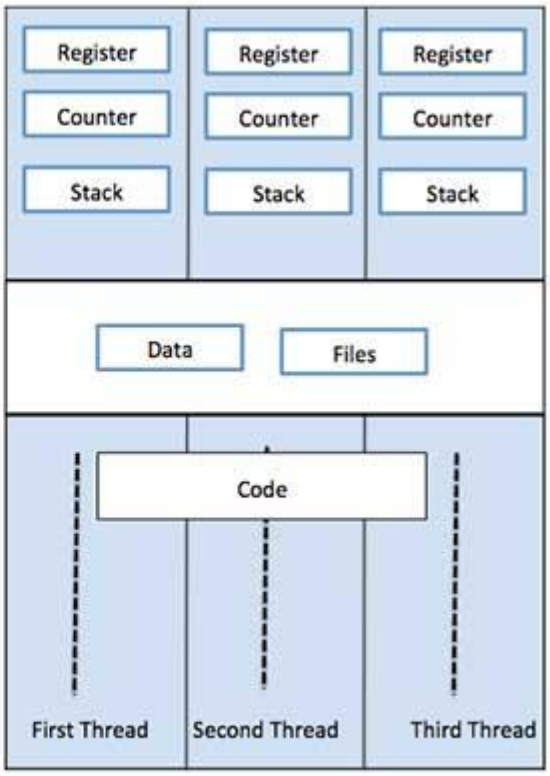
A thread shares with its peer threads few information like code segment, data segment and open files. When one thread alters a code segment memory item, all other threads see that.

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

SR.No.	Process	Thread
1	Process is heavy weight or resource	Thread is light weight, taking lesser

	intensive.	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.

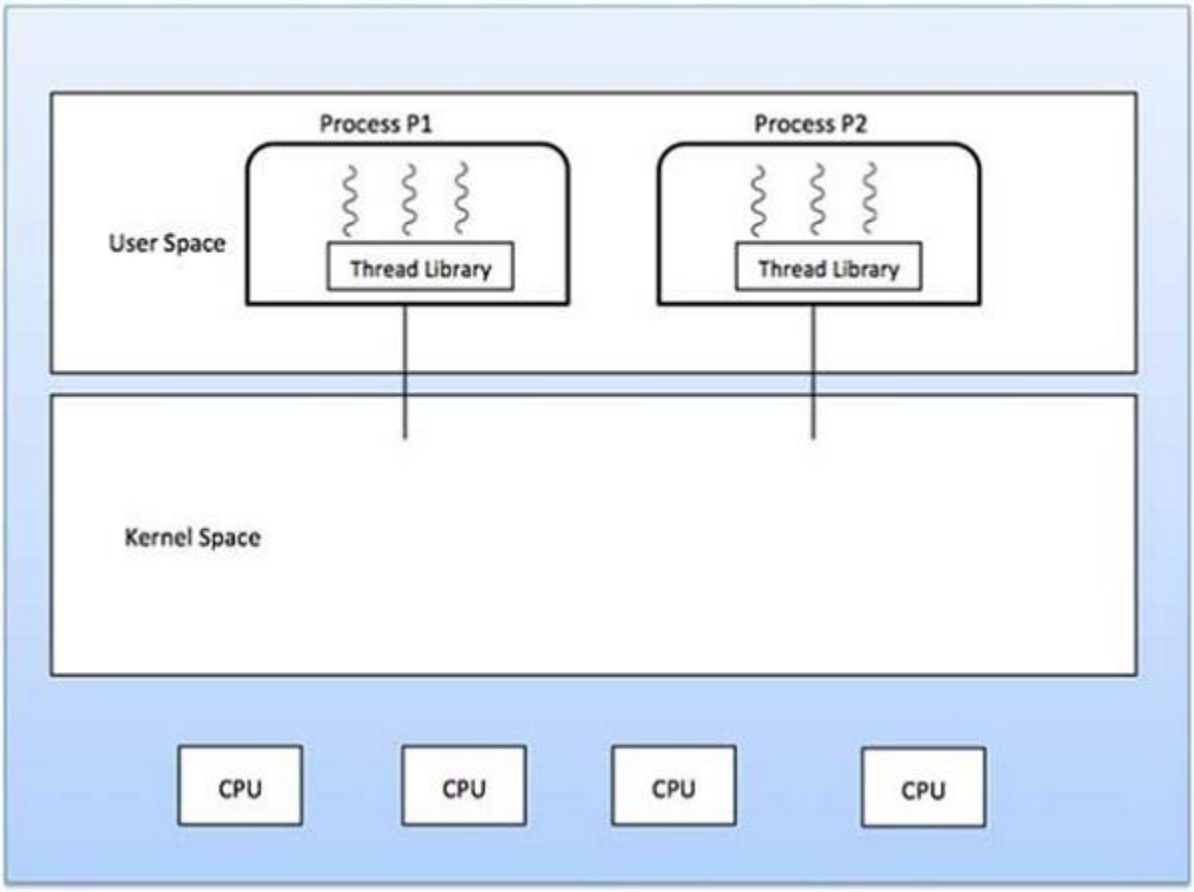
Types of Thread

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

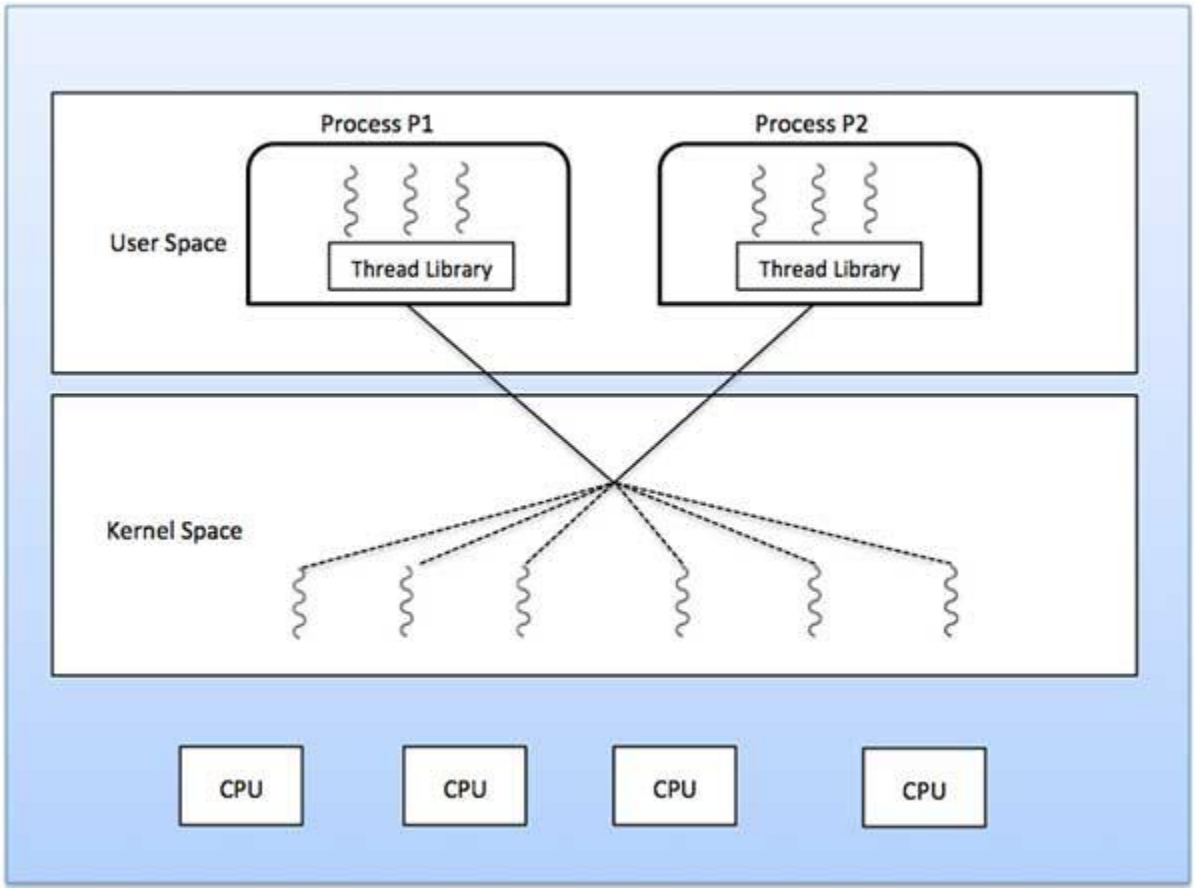
- Many to many relationship.
- Many to one relationship.
- One to one relationship.

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

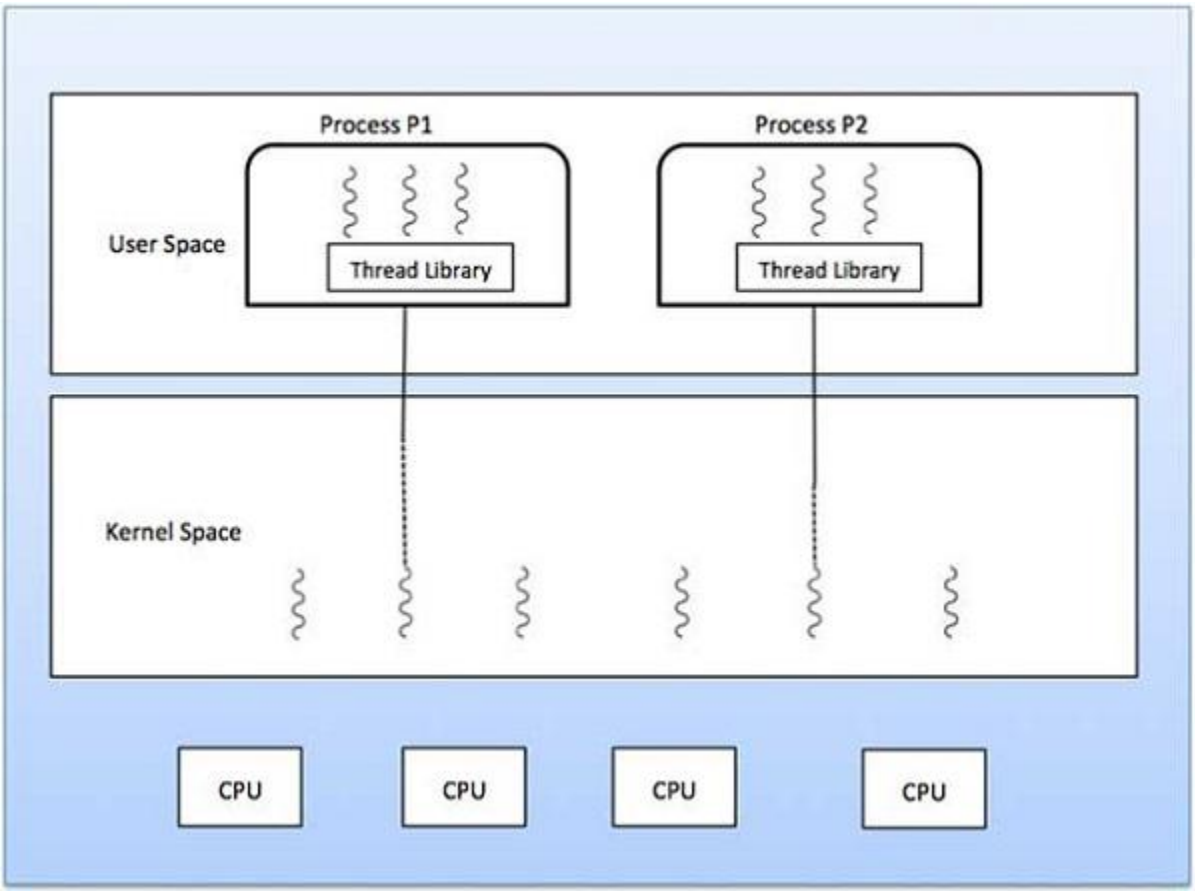
run in parallel on a multiprocessor machine. This model provides the best accuracy on concurrency and when a thread performs a blocking system call, the kernel can schedule another thread for execution.



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.

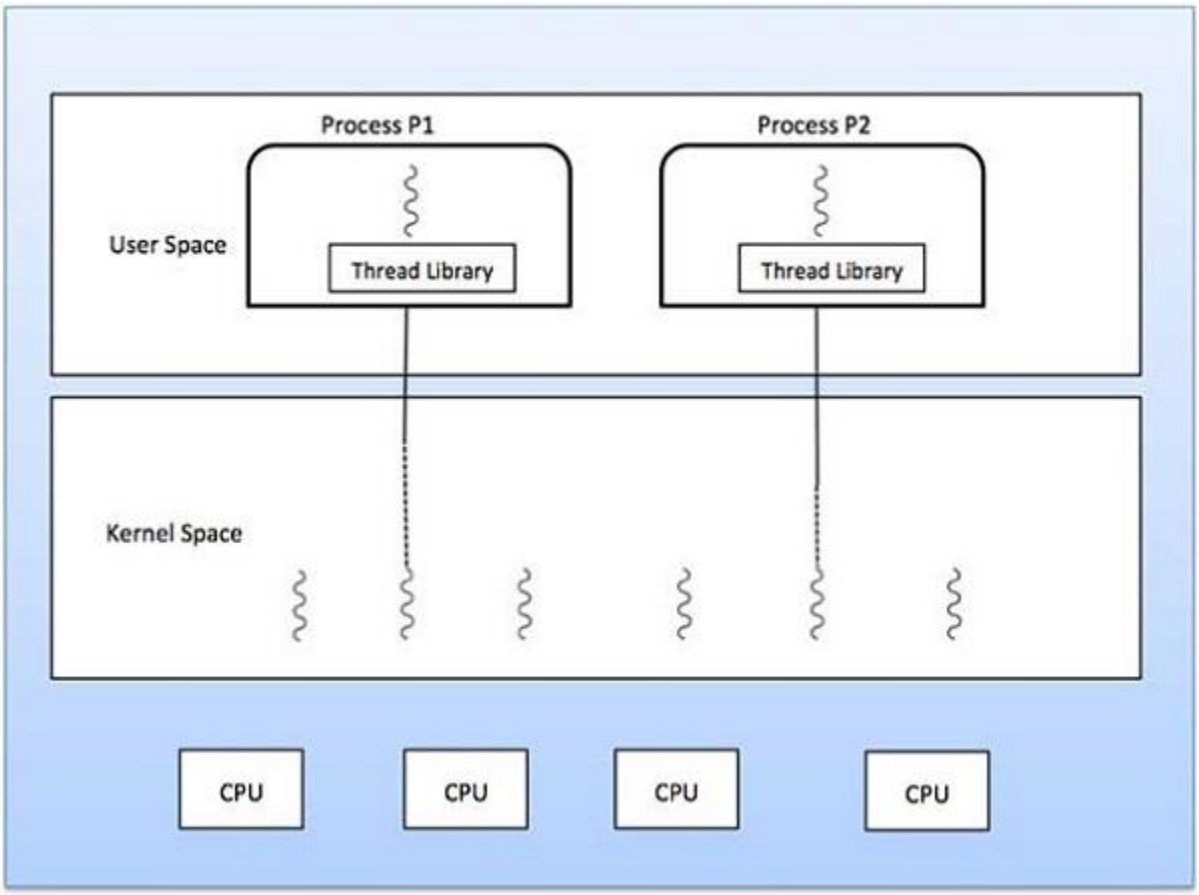
If the user-level thread libraries are implemented in the operating system in such a way that the system does not support them, then the Kernel threads use the many-to-one relationship modes.



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.



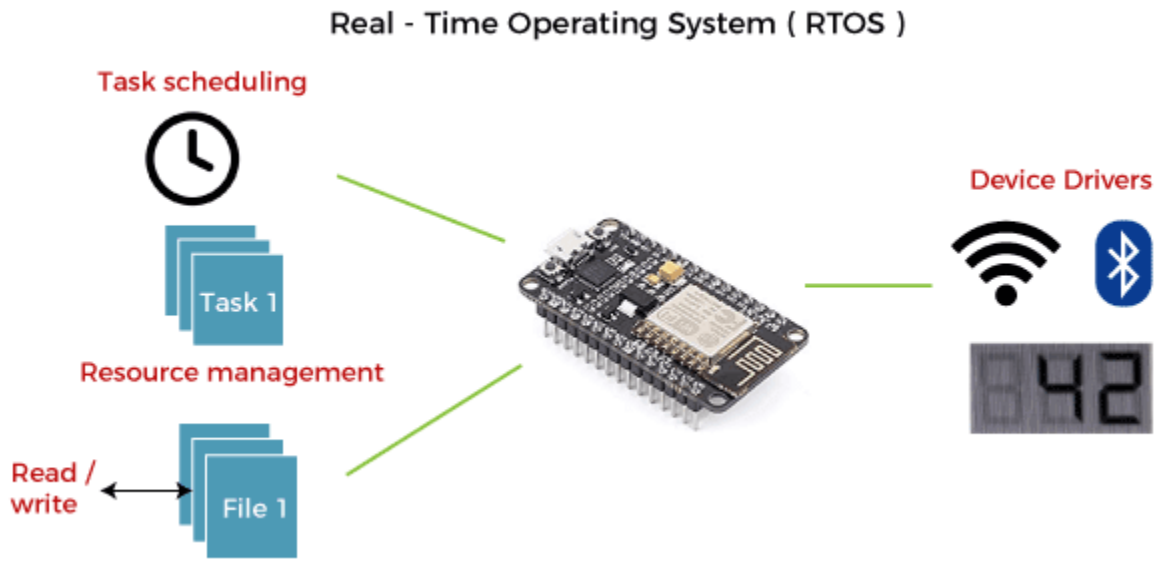
Difference between User-Level & Kernel-Level Thread

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.

Real Time operating system

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.

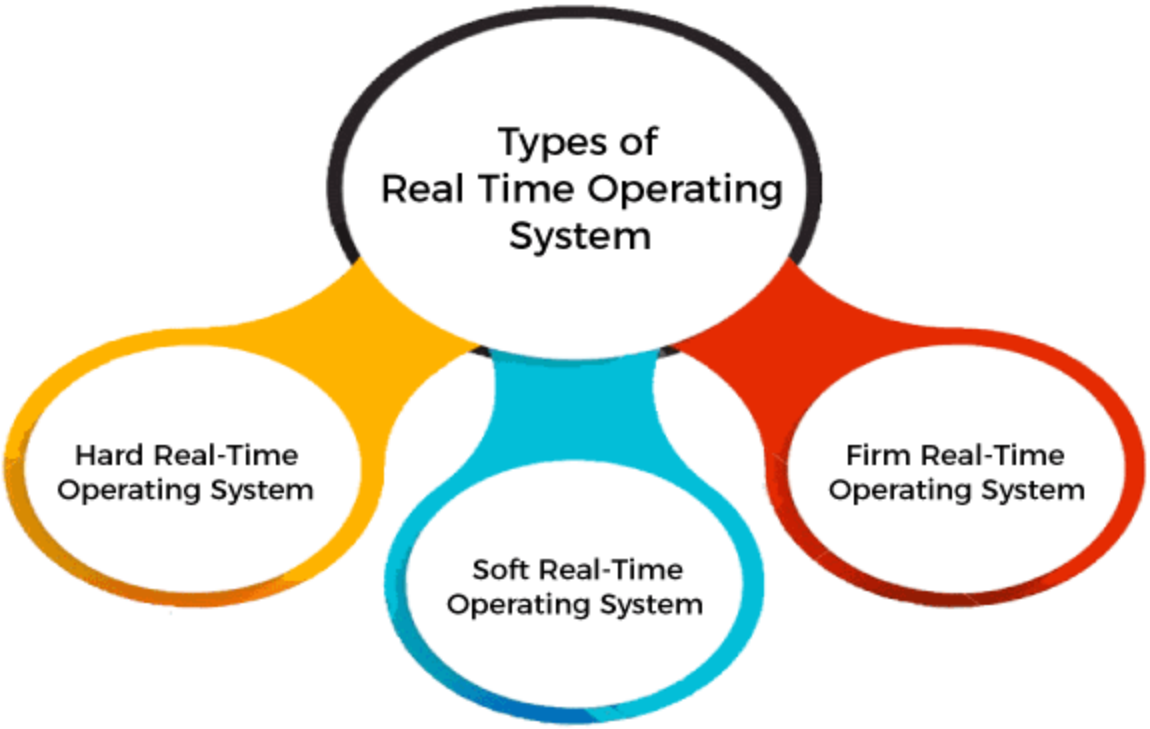
Applications of Real-time operating system (RTOS):

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. 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.

Difference between in GPOS and RTOS

Here are important differences between GPOS and RTOS:

SR. No.	General-Purpose Operating System (GPOS)	Real-Time Operating System (RTOS)
1	It used for desktop PC and laptop.	It is only applied to the embedded application.
2	Process-based Scheduling.	Time-based scheduling used like round-robin scheduling.
3	Interrupt latency is not considered as important as in RTOS.	Interrupt lag is minimal, which is measured in a few microseconds.
4	No priority inversion mechanism is present in the system.	The priority inversion mechanism is current. So it cannot modify by the system.
5	Kernel's operation may or may not be preempted.	Kernel's operation can be preempted.
6	Priority inversion remain unnoticed	No predictability guarantees

Applications of Real Time Operating System

Real-time systems are used in:

- Airlines reservation system.
- Air traffic control system.
- Systems that provide immediate updating.
- Used in any system that provides up to date and minute information on stock prices.
- Defence application systems like RADAR.
- Networked Multimedia Systems
- Command Control Systems
- Internet Telephony
- Anti-lock Brake Systems
- Heart Pacemaker

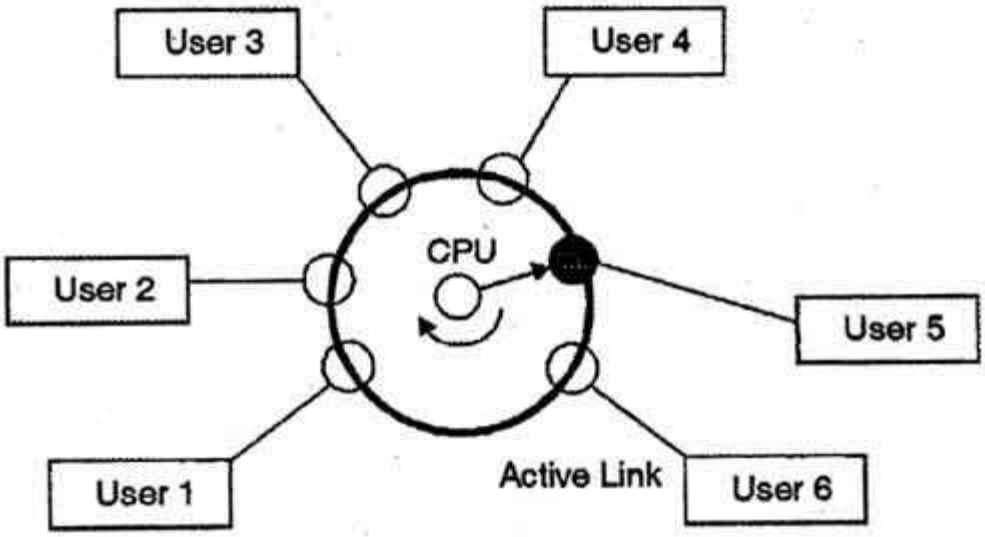
Time sharing operating system

A **time sharing system** allows many users to share the computer resources simultaneously. In other words, time sharing refers to the allocation of computer resources in time slots to several programs simultaneously. For example a mainframe computer that has many users logged on to it. Each user uses the resources of the mainframe -i.e. memory, CPU etc. The users feel that they are exclusive user of the CPU, even though this is not possible with one CPU i.e. shared among different users.

The time sharing systems were developed to provide an interactive use of the computer system. A time shared system uses CPU scheduling and multi-programming to provide each user with a small portion of a time-shared computer. It allows many users to share the computer resources simultaneously. As the system switches rapidly from one user to the other, a short time slot is given to each user for their executions.

The time sharing system provides the direct access to a large number of users where CPU time is divided among all the users on scheduled basis. The OS allocates a set of time to each user. When this time is expired, it passes control to the next user on the system. The time allowed is extremely small and the users are given the impression that they each have their own CPU and they are the sole owner of the CPU. This short period of time during that a user gets attention of the CPU; is known as a *time slice or a quantum.*

The concept of time sharing system is shown in figure.



In above figure the user 5 is active but user 1, user 2, user 3, and user 4 are in waiting state whereas user 6 is in ready status.

As soon as the time slice of user 5 is completed, the control moves on to the next ready user i.e. user 6. In this state user 2, user 3, user 4, and user 5 are in waiting state and user 1 is in ready state. The process continues in the same way and so on.

The time-shared systems are more complex than the multi-programming systems. In time-shared systems multiple processes are managed simultaneously which requires an adequate management of main memory so that the processes can be swapped in or swapped out within a short time.

Time-sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing.

The main difference between Multiprogrammed Batch Systems and Time-Sharing Systems is that in case of multiprogrammed batch systems, the objective is to maximize processor use, whereas in Time-Sharing Systems, the objective is to minimize response time.

Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, the processor executes each user program in a short burst or quantum of computation. That is, if **n** users are present, then each user can get a time quantum. When the user submits the command, the response time is in few seconds at most.

The operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems that were designed primarily as batch systems have been modified to time-sharing systems.

Advantages of Timesharing operating systems are as follows −

- Provides the advantage of quick response.
- Avoids duplication of software.
- Reduces CPU idle time.

Disadvantages of Time-sharing operating systems are as follows −

- Problem of reliability.
- Question of security and integrity of user programs and data.
- Problem of data communication.

Network operating system

Unlike operating systems, such as Windows, that are designed for single users to control one computer, **network operating systems (NOS)** coordinate the activities of multiple computers across a network. The network operating system acts as a director to keep the network running smoothly.

Network Operating System is the software that allows multiple computers to communicate, share files and hardware devices with one another. Some popular network operating systems are Novell Netware, Windows *NT/2000*, Linux, Sun Solaris, UNIX, and IBM *OS/2*.

Did you Know? The first network operating system was Novell NetWare, released in 1983. After Netware, other network operating systems were released, including Banyan VINES and Microsoft Windows NT. Some examples of other network operating systems include Windows 2000, Microsoft Windows XP, Sun Solaris, and Linux.

A common **example of network operating system** is a collection of personal computers along with a common printer, server and file server for archival storage, all tied together by a local network.

Features of Network Operating System

- It allows multiple computers to connect so that they can share data, files and hardware devices.
- Provide basic operating system features such as support for processors, protocols, automatic hardware detection and support multi-processing of applications.
- Provide security features such as authentication, login restrictions, and access control.
- Provide name and directory services.
- Provide file, print, web services and back-up services.
- Support Internetworking such as routing and WAN ports.
- User management and support for logon and logoff, remote access; system management, administration, and auditing tools with graphical interfaces.
- It has clustering capabilities.
- It also provides basic network administration utilities like access to the user.
- It also provides priority to the printing jobs which are in the queue in the network.
- It detects the new hardware whenever it is added to the system.

Types of Network Operating Systems

There are two basic types of **network operating systems**, the peer-to-peer NOS and the client/server NOS:

1. **Peer-to-peer network operating systems** allow users to share network resources saved in a common, accessible network location. In this architecture, all devices are treated equally in terms of functionality. Peer-to-peer usually works best for small to medium LANs and is cheaper to set up.
2. **Client/server network operating systems** provide users with access to resources through a server. In this architecture, all functions and applications are unified under one file server that can be used to execute individual client actions regardless of physical location. The client/server tends to be most expensive to implement and requires a large amount of technical maintenance. An advantage to the client/server model is that the network is controlled centrally, makes changes or additions to technology easier to incorporate.

Network Operating System Diagram

Advantages of Network Operating System:

- Highly stable centralized servers
- Security concerns are handled through servers
- New technologies and hardware up-gradation are easily integrated into the system
- Server access is possible remotely from different locations and types of systems

Disadvantages of Network Operating System:

- Servers are costly
- User has to depend on a central location for most operations
- Maintenance and updates are required regularly

Network Operating System Software

The following links include some of the more popular peer-to-peer and client/server network operating systems.

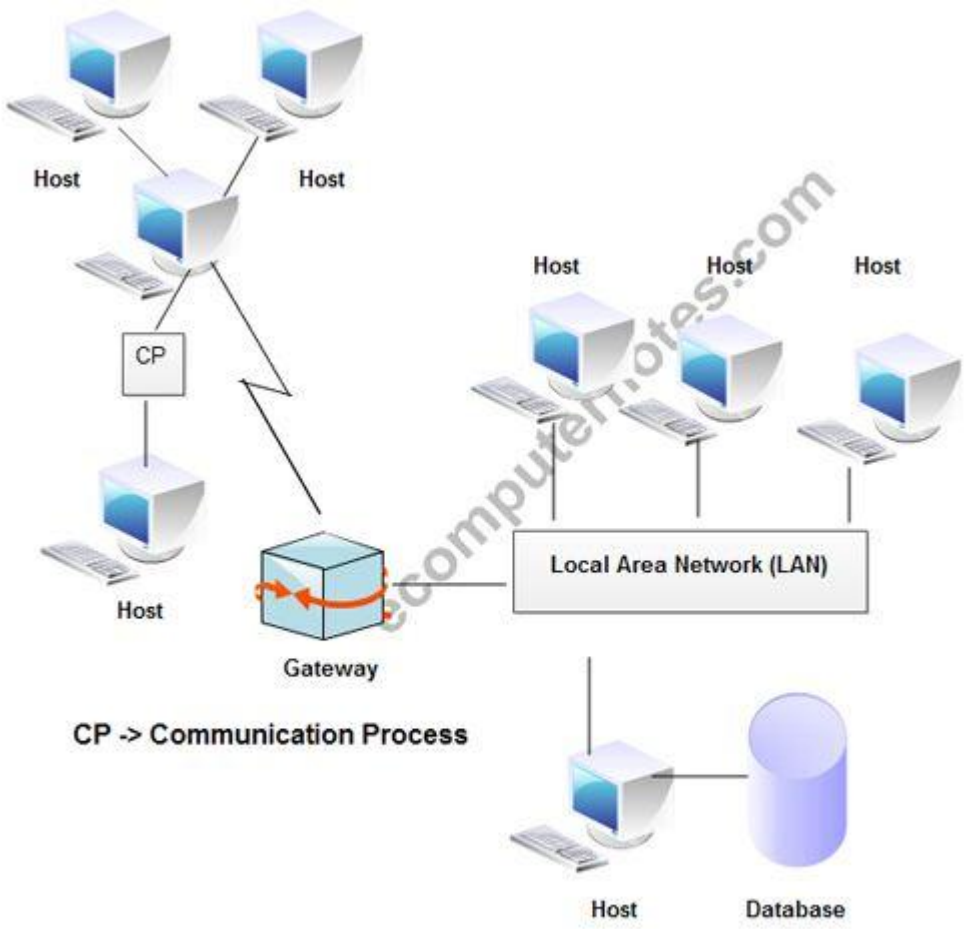
1. Macintosh OS X
2. Microsoft Windows Server
3. UNIX

4. Linux

Distributed Operating System

Distributed Operating System is a model where distributed applications are running on multiple computers linked by communications. A distributed operating system is an extension of the network operating system that supports higher levels of communication and integration of the machines on the network.

This system looks to its users like an ordinary centralized operating system but runs on multiple, independent central processing units (CPUs).

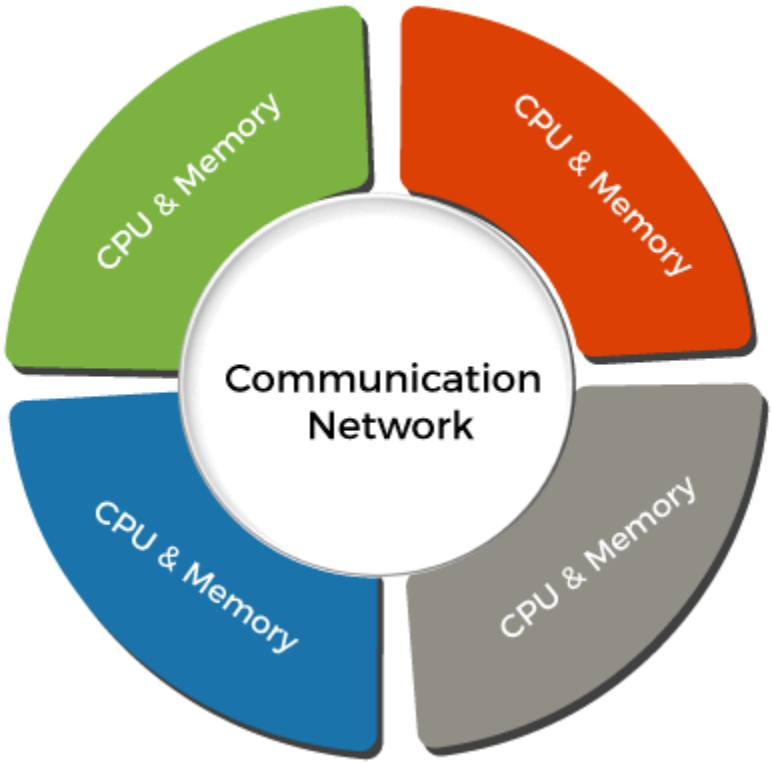


A Typical View of Distributed System

A distributed operating system **(DOS)** is an essential type of operating system. Distributed systems use many central processors to serve multiple real-time

applications and users. As a result, data processing jobs are distributed between the processors.

It connects multiple computers via a single communication channel. Furthermore, each of these systems has its own processor and memory. Additionally, these **CPUs** communicate via high-speed buses or telephone lines. Individual systems that communicate via a single channel are regarded as a single entity. They're also known as **loosely coupled systems**.



This operating system

consists of numerous computers, nodes, and sites joined together via **LAN/WAN**

lines. It enables the distribution of full systems on a couple of center processors, and it supports many real-time products and different users. Distributed operating systems can share their computing resources and I/O files while providing users with virtual machine abstraction.

These systems are referred as *loosely coupled systems* where each processor has its own local memory and processors communicate with one another through various communication lines, such as high speed buses or telephone lines. By loosely coupled systems, we mean that such computers possess no hardware connections at the CPU – memory bus level, but are connected by external interfaces that run under the control of software.

The Distributed OS involves a collection of autonomous computer systems, capable of communicating and cooperating with each other through a LAN / WAN. A Distributed Os provides a virtual machine abstraction to its users and wide sharing of resources like as computational capacity, I/O and files etc.

The structure shown in fig contains a set of individual computer systems and workstations connected via communication systems, but by this structure we cannot say it is a distributed system because it is the software, not the hardware, that determines whether a system is distributed or not.

The users of a true distributed system should not know, on which machine their programs are running and where their files are stored. LOCUS and MICROS are the best examples of distributed operating systems.

Using LOCUS operating system it was possible to access local and distant files in uniform manner. This feature enabled a user to log on any node of the network and to utilize the resources in a network without the reference of his/her location. MICROS provided sharing of resources in an automatic manner. The jobs were assigned to different nodes of the whole system to balance the load on different nodes.

Below given are some of the examples of distributed operating systems:

l. IRIX operating system; is the implementation of UNIX System V, Release 3 for Silicon Graphics multiprocessor workstations.

2.DYNIX operating system running on Sequent Symmetry multiprocessor computers.

3.AIX operating system for IBM RS/6000 computers.

4.Solaris operating system for SUN multiprocessor workstations.

5.Mach/OS is a multi-threading and multitasking UNIX compatible operating system;

6.OSF/1 operating system developed by Open Foundation Software: UNIX compatible.

Types of Distributed Operating System

There are various types of Distributed Operating systems. Some of them are as follows:

1. **Client-Server Systems**
2. **Peer-to-Peer Systems**
3. **Middleware**
4. **Three-tier**
5. **N-tier**

Client-Server System

This type of system requires the client to request a resource, after which the server gives the requested resource. When a client connects to a server, the server may serve multiple clients at the same time.

Client-Server Systems are also referred to as "Tightly Coupled Operating Systems". This system is primarily intended for multiprocessors and homogenous multicomputer. Client-Server Systems function as a centralized server since they approve all requests issued by client systems.

Server systems can be divided into two parts:

1. Computer Server System

This system allows the interface, and the client then sends its own requests to be executed as an action. After completing the activity, it sends a back response and transfers the result to the client.

2. File Server System

It provides a file system interface for clients, allowing them to execute actions like file creation, updating, deletion, and more.

Peer-to-Peer System

The nodes play an important role in this system. The task is evenly distributed among the nodes. Additionally, these nodes can share data and resources as needed. Once again, they require a network to connect.

The Peer-to-Peer System is known as a "Loosely Couple System". This concept is used in computer network applications since they contain a large number of processors that do not share memory or clocks. Each processor has its own local memory, and they interact with one another via a variety of communication methods like telephone lines or high-speed buses.

Middleware

Middleware enables the interoperability of all applications running on different operating systems. Those programs are capable of transferring all data to one other by using these services.

Three-tier

The information about the client is saved in the intermediate tier rather than in the client, which simplifies development. This type of architecture is most commonly used in online applications.

N-tier

When a server or application has to transmit requests to other enterprise services on the network, n-tier systems are used.

Features of Distributed Operating System

There are various features of the distributed operating system. Some of them are as follows:

Openness

It means that the system's services are freely displayed through interfaces. Furthermore, these interfaces only give the service syntax. For example, the type of function, its return type, parameters, and so on. Interface Definition Languages are used to create these interfaces (IDL).

Scalability

It refers to the fact that the system's efficiency should not vary as new nodes are added to the system. Furthermore, the performance of a system with 100 nodes should be the same as that of a system with 1000 nodes.

Resource Sharing

Its most essential feature is that it allows users to share resources. They can also share resources in a secure and controlled manner. Printers, files, data, storage, web pages, etc., are examples of shared resources.

Flexibility

A DOS's flexibility is enhanced by modular qualities and delivers a more advanced range of high-level services. The kernel/ microkernel's quality and completeness simplify the implementation of such services.

Transparency

It is the most important feature of the distributed operating system. The primary purpose of a distributed operating system is to hide the fact that resources are shared. Transparency also implies that the user should be unaware that the resources he is accessing are shared. Furthermore, the system should be a separate independent unit for the user.

Heterogeneity

The components of distributed systems may differ and vary in operating systems, networks, programming languages, computer hardware, and implementations by different developers.

Fault Tolerance

Fault tolerance is that process in which user may continue their work if the software or hardware fails.

Examples of Distributed Operating System

There are various examples of the distributed operating system. Some of them are as follows:

Solaris

It is designed for the SUN multiprocessor workstations

OSF/1

It's compatible with Unix and was designed by the Open Foundation Software Company.

Micros

The MICROS operating system ensures a balanced data load while allocating jobs to all nodes in the system.

DYNIX

It is developed for the Symmetry multiprocessor computers.

Locus

It may be accessed local and remote files at the same time without any location hindrance.

Mach

It allows the multithreading and multitasking features.

Applications of Distributed Operating System

There are various applications of the distributed operating system. Some of them are as follows:

Network Applications

DOS is used by many network applications, including the Web, peer-to-peer networks, multiplayer web-based games, and virtual communities.

Telecommunication Networks

DOS is useful in phones and cellular networks. A DOS can be found in networks like the Internet, wireless sensor networks, and routing algorithms.

Parallel Computation

DOS is the basis of systematic computing, which includes cluster computing and grid computing, and a variety of volunteer computing projects.

Real-Time Process Control

The real-time process control system operates with a deadline, and such examples include aircraft control systems.

Advantages and Disadvantages of Distributed Operating System

There are various advantages and disadvantages of the distributed operating system. Some of them are as follows:

Advantages

There are various advantages of the distributed operating system. Some of them are as follow:

1. It may share all resources (CPU, disk, network interface, nodes, computers, and so on) from one site to another, increasing data availability across the entire system.
2. It reduces the probability of data corruption because all data is replicated across all sites; if one site fails, the user can access data from another operational site.
3. The entire system operates independently of one another, and as a result, if one site crashes, the entire system does not halt.
4. It increases the speed of data exchange from one site to another site.
5. It is an open system since it may be accessed from both local and remote locations.
6. It helps in the reduction of data processing time.
7. Most distributed systems are made up of several nodes that interact to make them fault-tolerant. If a single machine fails, the system remains operational.

Disadvantages

There are various disadvantages of the distributed operating system. Some of them are as follows:

1. The system must decide which jobs must be executed when they must be executed, and where they must be executed. A scheduler has limitations, which can lead to underutilized hardware and unpredictable runtimes.
2. It is hard to implement adequate security in DOS since the nodes and connections must be secured.
3. The database connected to a DOS is relatively complicated and hard to manage in contrast to a single-user system.
4. The underlying software is extremely complex and is not understood very well compared to other systems.
5. The more widely distributed a system is, the more communication latency can be expected. As a result, teams and developers must choose between availability, consistency, and latency.
6. These systems aren't widely available because they're thought to be too expensive.
7. Gathering, processing, presenting, and monitoring hardware use metrics for big clusters can be a real issue.

A mobile operating system (OS) is software that allows smartphones, tablet PCs (personal computers) and other devices to run applications and programs.

A mobile OS typically starts up when a device powers on, presenting a screen with icons or tiles that present information and provide application access. Mobile operating systems also manage cellular and wireless network connectivity, as well as phone access.

Types of mobile operating systems

There are numerous mobile device operating systems available today, and two of the most widely adopted are the iPhone's OS, Apple iOS, and Google's open source OS, Google Android. These two mobile OSes take different approaches to the mobile operating system.

Apple distributes the only devices that natively support iOS, and it takes a "walled garden" approach, in which Apple regulates all mobile apps and services that can run on the iOS devices. Apple developed iOS to run on its own XNU kernel. Apple has

also released several device-specific mobile operating systems such as watchOS for the Apple Watch and iPadOS for the iPad tablets.

Google takes a different approach with Android, which is open source. This means that mobile device OEMs (original equipment manufacturers) can customize the Android source code and customize it to fit their devices. Android runs on the Linux kernel.

There are other mobile operating systems available, but their adoption rates are well below those of iOS and Android. These other operating systems include KaiOS, Sailfish OS and Huawei's Harmony OS.

KaiOS, based on the discontinued Mozilla Firefox OS, runs mostly on dumb phones -- also known as feature phones. These mobile devices have very limited computing power, but the latest version of KaiOS can bring more comprehensive feature sets to these devices such as an app store and the Google Assistant.

Sailfish OS is based on multiple open source projects and runs mostly on smartphones and tablet computers from Jolla, which developed the operating system, and Sony mobile devices. Sailfish OS and KaiOS both run on the Linux kernel.

Huawei's Harmony OS, initially released in August 2019, can also run on IoT devices. Huawei's devices typically run Android operating system, but Harmony OS may replace Android on these devices in the future. Harmony OS runs on a microkernel that Huawei developed.

There are numerous mobile operating systems that are no longer commonly supported. The BlackBerry 10 operating system, which succeeded BlackBerry OS in 2013, has extended support available but for a small install base. Nokia's Symbian operating system received its last update in 2013 and is no longer supported.

LG webOS -- formerly Palm webOS and HP webOS -- transitioned from a smartphone and tablet operating system to a smart TV operating system after LG

acquired the OS from Hewlett-Packard in 2013. The Microsoft's Windows Phone operating system was discontinued by Microsoft in 2017. Windows 8 and Windows 8.1 function as both traditional desktop operating systems and mobile operating systems, but mainstream support for these OSes is no longer available for Windows mobile devices or desktops.

Features of mobile operating systems

Mobile operating systems deliver various features to users, and the distinguishing feature that mobile operating systems offer is the ability to connect to the internet via the smartphone's built-in modem and a wireless service provider such as Verizon or AT&T. This is a major difference between mobile operating systems and most desktop operating systems, which rely on a Wi-Fi network or Ethernet connection to access the internet.

Many mobile OSes offer a native web browser application, which allows users to search the internet and visit webpages. Mobile operating systems also offer application stores, which allow users to download and interface with mobile applications. Several mobile operating systems also have native GPS (global positioning system) applications that allow users to search for locations, follow step-by-step directions and, in some cases, share location with different devices. The GPS feature, of course, relies on the mobile device's hardware and can't run without that support.

While some native applications come with mobile operating systems such as iOS and Android, the mobile app stores open up new possibilities for users. In some cases, the applications add new features and improved user interface (UI) for websites that are accessible via a web browser, but other applications bring new functionality to the mobile devices.

Other common mobile operating system features include native email applications that allow users to link their work and personal email accounts, a calendar application

that users can keep track of tasks, meetings and events, and a contacts library to organize and search for contact information.

Most mobile operating systems -- other than Android -- are tied to specific hardware with little flexibility. Users can jailbreak or root devices, however, which allows them to install another mobile OS or unlock restricted applications.

Android Operating System: Pros and Cons

Advantages of the Android operating system

- 1. Open source
- 2. Greater freedom
- 3. Diversity and versatility
- 4. Multitasking system
- 5. Greater interaction
- 6. Community

Disadvantages of the Android operating system

- 1. Battery life
- 2. Complexity
- 3. Poor quality of some applications
- 4. Vulnerability
- 5. Lack of update support

Operating System - Services

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.

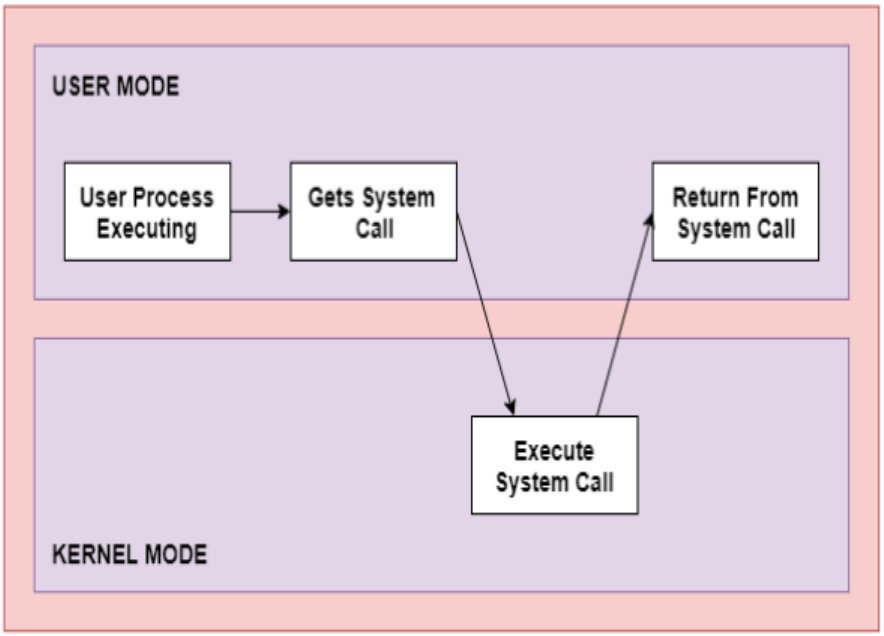
Protection refers to a mechanism or a way to control the access of programs, processes, or users to the resources defined by a computer system. Following are the major activities of an operating system with respect to protection –

- 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.

SYSTEM CALLS IN OPERATING SYSTEM

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

A figure representing the execution of the system call is given as follows –



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

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

- If a file system requires the creation or deletion of files. Reading and writing from files also require a system call.
- Creation and management of new processes.
- Network connections also require system calls. This includes sending and receiving packets.
- Access to a hardware devices such as a printer, scanner etc. requires a system call.

Types of System Calls

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

Process Control

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

File Management

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

Device Management

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

Information Maintenance

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

Communication

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

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

Types of System Calls	Windows	Linux
Process Control	CreateProcess() ExitProcess()	fork() exit()

Types of System Calls	Windows	Linux
	WaitForSingleObject()	wait()
File Management	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Device Management	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shmget() mmap()

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

open()

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

read()

The read() system call is used to access data from a file that is stored in the file system. The file to read can be identified by its file descriptor and it should be

opened using open() before it can be read. In general, the read() system calls takes three arguments i.e. the file descriptor, buffer which stores read data and number of bytes to be read from the file.

write()

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

close()

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

OPERATING SYSTEM STRUCTURE

operating systems have complex structures. It is easier to create an operating system in pieces, much as we break down larger issues into smaller, more manageable subproblems. Every segment is also a part of the operating system. Operating system structure can be thought of as the strategy for connecting and incorporating various operating system components within the kernel. Operating systems are implemented using many types of structures

SIMPLE STRUCTURE

It is the most straightforward operating system structure, but it lacks definition and is only appropriate for usage with tiny and restricted systems. Since the interfaces and degrees of functionality in this structure are clearly defined, programs are able to access I/O routines, which may result in unauthorized access to I/O procedures.

This organizational structure is used by the MS-DOS operating system:

- There are four layers that make up the MS-DOS operating system, and each has its own set of features.
- These layers include ROM BIOS device drivers, MS-DOS device drivers, application programs, and system programs.
- The MS-DOS operating system benefits from layering because each level can be defined independently and, when necessary, can interact with one another.
- If the system is built in layers, it will be simpler to design, manage, and update. Because of this, simple structures can be used to build constrained systems that are less complex.
- When a user program fails, the operating system as whole crashes.
- Because MS-DOS systems have a low level of abstraction, programs and I/O procedures are visible to end users, giving them the potential for unwanted access.

The following figure illustrates layering in simple structure:

Advantages of Simple Structure:

- Because there are only a few interfaces and levels, it is simple to develop.
- Because there are fewer layers between the hardware and the applications, it offers superior performance.

Disadvantages of Simple Structure:

- The entire operating system breaks if just one user program malfunctions.
- Since the layers are interconnected, and in communication with one another, there is no abstraction or data hiding.
- The operating system's operations are accessible to layers, which can result in data tampering and system failure.

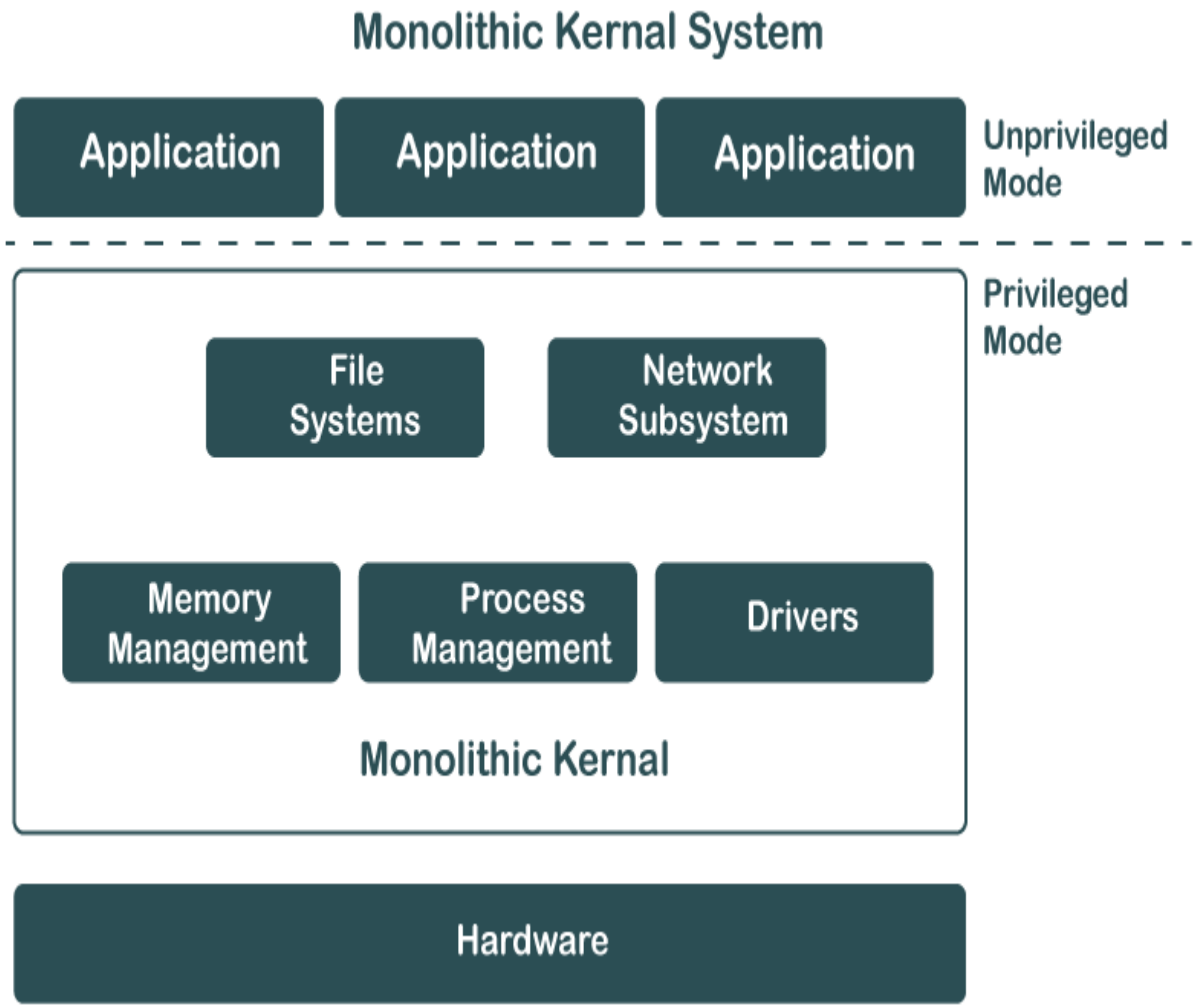
MONOLITHIC STRUCTURE

The monolithic operating system controls all aspects of the operating system's operation, including file management, memory management, device management, and operational operations.

The core of an operating system for computers is called the kernel (OS). All other System components are provided with fundamental services by the kernel. The operating system and the hardware use it as their main interface. When an operating system is built into a single piece of hardware, such as a keyboard or mouse, the kernel can directly access all of its resources.

The monolithic operating system is often referred to as the monolithic kernel. Multiple programming techniques such as batch processing and time-sharing increase a processor's usability. Working on top of the operating system and under complete command of all hardware, the monolithic kernel performs the role of a virtual computer. This is an old operating system that was used in banks to carry out simple tasks like batch processing and time-sharing, which allows numerous users at different terminals to access the Operating System.

The following diagram represents the monolithic structure:



Advantages of Monolithic Structure:

- Because layering is unnecessary and the kernel alone is responsible for managing all operations, it is easy to design and execute.
- Due to the fact that functions like memory management, file management, process scheduling, etc., are implemented in the same address area, the monolithic kernel runs rather quickly when compared to other systems. Utilizing the same address speeds up and reduces the time required for address allocation for new processes.

Disadvantages of Monolithic Structure:

- The monolithic kernel's services are interconnected in address space and have an impact on one another, so if any of them malfunctions, the entire system does as well.

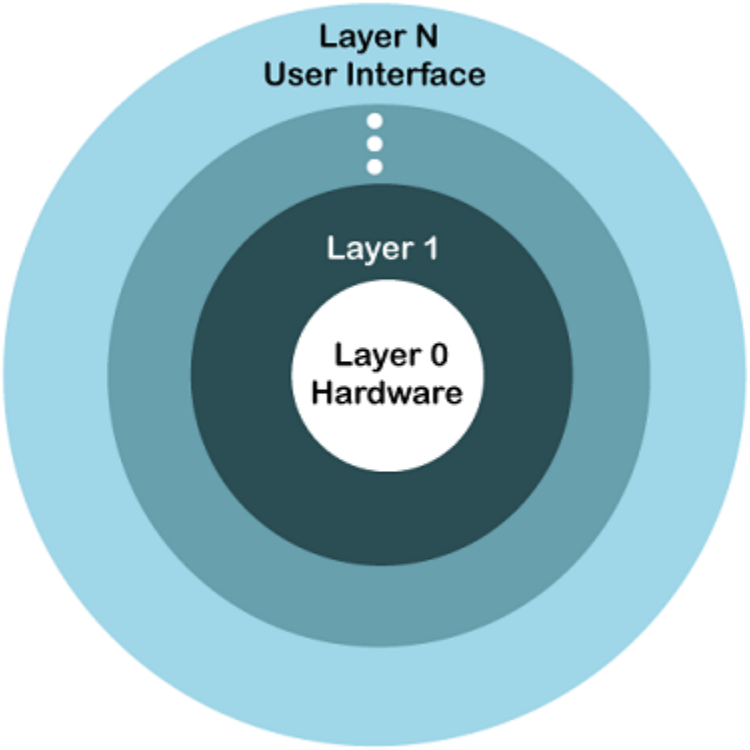
- It is not adaptable. Therefore, launching a new service is difficult.

LAYERED STRUCTURE

The OS is separated into layers or levels in this kind of arrangement. Layer 0 (the lowest layer) contains the hardware, and layer 1 (the highest layer) contains the user interface (layer N). These layers are organized hierarchically, with the top-level layers making use of the capabilities of the lower-level ones.

The functionalities of each layer are separated in this method, and abstraction is also an option. Because layered structures are hierarchical, debugging is simpler, therefore all lower-level layers are debugged before the upper layer is examined. As a result, the present layer alone has to be reviewed since all the lower layers have already been examined.

The image below shows how OS is organized into layers:



Advantages of Layered Structure:

- Work duties are separated since each layer has its own functionality, and there is some amount of abstraction.
- Debugging is simpler because the lower layers are examined first, followed by the top layers.

Disadvantages of Layered Structure:

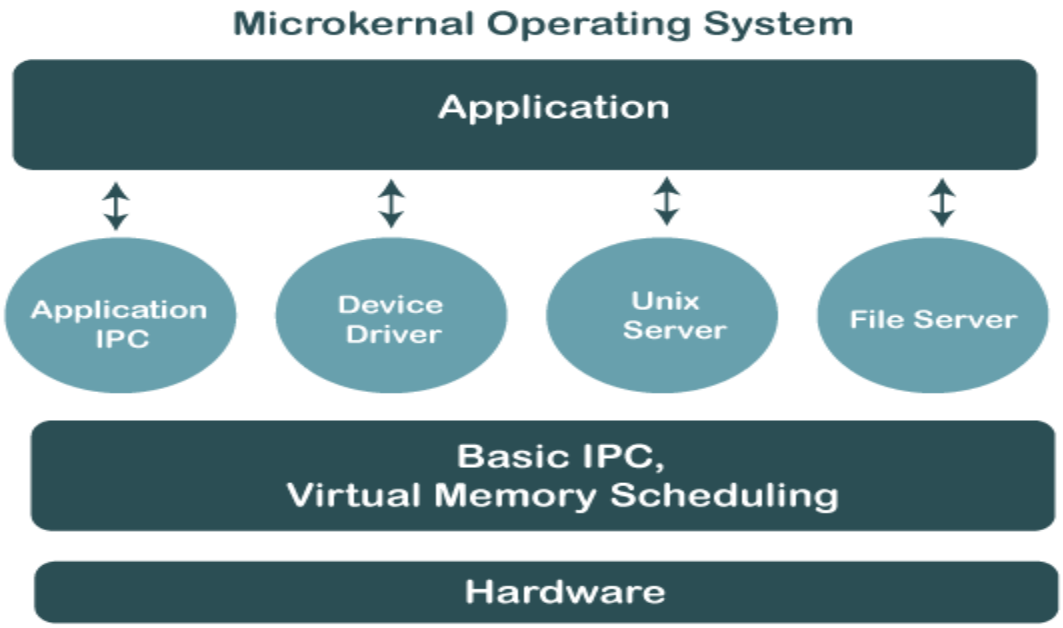
- Performance is compromised in layered structures due to layering.
- Construction of the layers requires careful design because upper layers only make use of lower layers' capabilities.

MICRO-KERNEL STRUCTURE

The operating system is created using a micro-kernel framework that strips the kernel of any unnecessary parts. Systems and user applications are used to implement these optional kernel components. So, Micro-Kernels is the name given to these systems that have been developed.

Each Micro-Kernel is created separately and is kept apart from the others. As a result, the system is now more trustworthy and secure. If one Micro-Kernel malfunctions, the remaining operating system is unaffected and continues to function normally.

The image below shows Micro-Kernel Operating System Structure:



Advantages of Micro-Kernel Structure:

- It enables portability of the operating system across platforms.
- Due to the isolation of each Micro-Kernel, it is reliable and secure.
- The reduced size of Micro-Kernels allows for successful testing.
- The remaining operating system remains unaffected and keeps running properly even if a component or Micro-Kernel fails.

Disadvantages of Micro-Kernel Structure:

- The performance of the system is decreased by increased inter-module communication.
- The construction of a system is complicated.

VIRTUAL MACHINES (VMs)

The hardware of our personal computer, including the CPU, disc drives, RAM, and NIC (Network Interface Card), is abstracted by a virtual machine into a variety of various execution contexts based on our needs, giving us the impression that each execution environment is a separate computer. A virtual box is an example of it.

Using CPU scheduling and virtual memory techniques, an operating system allows us to execute multiple processes simultaneously while giving the impression that each one is using a separate processor and virtual memory. System calls and a file system are examples of extra functionalities that a process can have that the hardware is unable to give. Instead of offering these extra features, the virtual machine method just offers an interface that is similar to that of the most fundamental hardware. A virtual duplicate of the computer system underneath is made available to each process.

We can develop a virtual machine for a variety of reasons, all of which are fundamentally connected to the capacity to share the same underlying hardware while concurrently supporting various execution environments, i.e., various operating systems.

Disk systems are the fundamental problem with the virtual machine technique. If the actual machine only has three-disc drives but needs to host seven virtual machines, let's imagine that. It is obvious that it is impossible to assign a disc drive to every virtual machine because the program that creates virtual machines would require a sizable amount of disc space in order to offer virtual memory and spooling. The provision of virtual discs is the solution.

The result is that users get their own virtual machines. They can then use any of the operating systems or software programs that are installed on the machine below. Virtual machine software is concerned with programming numerous virtual machines simultaneously into a physical machine; it is not required to take into account any user-support software. With this configuration, it may be possible to break the challenge of building an interactive system for several users into two manageable chunks.

Advantages of Virtual Machines:

- Due to total isolation between each virtual machine and every other virtual machine, there are no issues with security.
- A virtual machine may offer an architecture for the instruction set that is different from that of actual computers.
- Simple availability, accessibility, and recovery convenience.

Disadvantages of Virtual Machines:

- Depending on the workload, operating numerous virtual machines simultaneously on a host computer may have an adverse effect on one of them.
- When it comes to hardware access, virtual computers are less effective than physical ones.