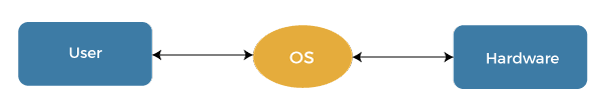
**Unit 1: Introduction of Operating System**

* **What is meant by Operating System?**



An operating system (OS) refers to a software program that plays a vital role as an intermediary between the physical hardware components of a computer and the software applications that users interact with. It serves as a crucial layer of software that facilitates communication and coordination between different parts of a computer system.

The primary purpose of an operating system is to manage and oversee the hardware resources of the computer, such as the central processing unit (CPU), memory, storage devices, input/output devices (like monitors and printers), and networking components.

By providing a set of essential functions and services, the operating system enables users and software applications to effectively utilize these hardware resources. It handles tasks such as managing multiple programs running simultaneously (process management), organizing memory allocation and deallocation (memory management), maintaining a structured system for storing and retrieving data (file system management), and facilitating communication with peripheral devices (device management).

Additionally, the operating system often offers user interfaces, ranging from text-based command lines to graphical interfaces, allowing users to interact with the computer system and its functionalities.

Furthermore, the operating system is responsible for ensuring the security and integrity of the system by implementing user authentication, access controls, and data encryption. It also handles error detection and recovery, preventing system crashes and maintaining stability.

In summary, an operating system is a fundamental software component that acts as a bridge between computer hardware and user-level software, providing the necessary services and functions for efficient and secure utilization of a computer system's resources.

* **Features of Operating System**

1. **Process Management**: Operating systems manage and allocate system resources to various processes (programs in execution). They ensure fair utilization of CPU time and memory, enabling multitasking and efficient process scheduling.
2. **Memory Management**: They oversee system memory, allocating and deallocating memory space for processes and ensuring efficient use of available memory. This feature helps prevent memory conflicts and enhances overall system performance.
3. **File System Management**: Operating systems manage files and directories, providing a structured way to store, organize, and access data. They handle tasks like file creation, deletion, reading, and writing, as well as enforcing access permissions.
4. **Device Management:** They manage hardware devices such as printers, disk drives, and input/output devices. This involves providing device drivers and a standardized interface for software to communicate with these devices.
5. **Security and Access Control**: They implement security measures to protect system resources and data. This involves user authentication, authorization mechanisms, and encryption to prevent unauthorized access.
6. **Multithreading and Multiprocessing:** Operating systems support concurrent execution of multiple threads or processes on multi-core processors, enhancing system performance.
7. **Time and Date Management:** They maintain system time and date, synchronize with external time sources, and provide scheduling mechanisms for tasks and processes.
8. **System Utilities**: Operating systems come with a set of built-in utilities for tasks like disk cleanup, data backup, and system diagnostics.
9. **Compatibility and Application Support:** They provide compatibility layers to support older software and ensure that applications can run on newer versions of the operating system.
10. **Kernel and System Services:** Operating systems provide essential system services through the kernel, which is the core component of the OS. These services include hardware abstraction, inter-process communication, and system calls for application interaction with the OS.

* **History of operating system**

1. ***First Generation (1940s - Early 1950s):***

* Vacuum Tube Era: Early computers used vacuum tubes for processing.
* No operating systems in the modern sense.
* Programmers had to manually set up and switch hardware components.
* Batch processing systems, like the UNIVAC I, were used for specific tasks.

1. ***Second Generation (Early 1950s - Mid-1960s):***

* Transition to Transistors: Transistors replaced vacuum tubes.
* Batch Processing: Operating systems started to emerge to manage batch processing jobs.
* IBM's OS/360 is a notable example from this era.

1. ***Third Generation (Mid-1960s - Late 1970s):***

* Multiprogramming: Systems could run multiple programs concurrently.
* Time-sharing systems: Allowed multiple users to interact with the computer simultaneously.
* IBM's OS/360 continued to evolve.
* Unix, developed at Bell Labs, emerged as a popular time-sharing system.

1. ***Fourth Generation (Late 1970s - Mid-1990s):***

* Microprocessors: The development of microprocessors revolutionized computing.
* Personal Computers: Operating systems like MS-DOS and Apple's Macintosh System Software catered to personal computers.
* Networked Systems: The emergence of local area networks (LANs) and wide area networks (WANs).
* Graphical User Interfaces (GUIs) became more prevalent with systems like Windows and Mac OS.

1. ***Fifth Generation (Mid-1990s - Early 2000s):***

* The Internet: The rise of the internet led to network-centric computing.
* Windows 95 and Windows NT brought significant improvements in user interfaces and system stability.
* Linux gained popularity as a powerful and open-source operating system.
* Real-time and embedded systems also flourished in this era.

1. ***Sixth Generation (Early 2000s - Present):***

* Mobile Devices: The advent of smartphones and tablets brought new operating systems like iOS and Android.
* Cloud Computing: Cloud-based operating systems and services became more prevalent.
* Virtualization and Containerization: Technologies like VMware and Docker revolutionized resource allocation and application deployment.
* Open-source development continued to grow, with Linux playing a key role.
* Security and privacy concerns became more prominent.
* Seventh Generation (Emerging):
* Quantum Computing: As quantum computing advances, it may necessitate entirely new types of operating systems to manage quantum hardware.AI Integration: Operating systems are increasingly incorporating AI and machine learning capabilities for automation and optimization
* **Importance of operating system**

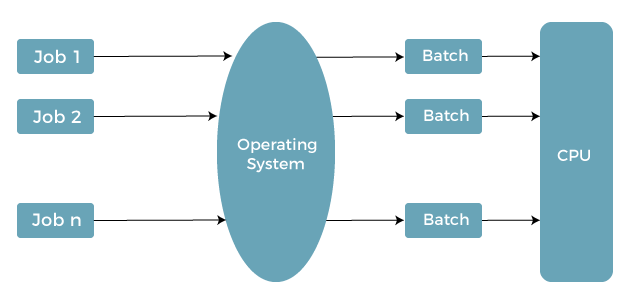
1. **Resource Management**: Efficiently allocates and manages hardware resources like CPU, memory, and storage for optimal performance.
2. **Process Scheduling:** Enables multitasking, ensuring smooth execution of multiple programs simultaneously.
3. **Memory Allocation**: Manages memory usage to prevent conflicts and ensure smooth application performance.
4. **File Organization:** Provides a structured file system for easy data storage, retrieval, and management.
5. **User Interface:** Offers intuitive interfaces, making computers accessible to users with varying levels of technical expertise.
6. **Security**: Implements measures like access control and encryption to safeguard data from unauthorized access.
7. **Networking:** Enables connection to networks for communication, data sharing, and internet access.
8. **Application Compatibility:** Provides a platform for software developers to create applications that can run on the system.
9. **Error Handling**: Detects and manages errors to prevent system crashes and data loss.
10. **Backup and Recovery:** Offers data backup and recovery options to prevent data loss.
11. **Virtualization**: Allows running multiple virtual machines on a single physical machine for efficient resource use.
12. **System Updates:** Facilitates updates and patches to improve functionality and security.
13. **Automation**: Enables automated tasks and scripts for increased efficiency.
14. **User Management**: Manages user accounts, ensuring privacy and access control.
15. **Ease of Maintenance:** Simplifies system maintenance and software installations.

* **What is Batch Operating System?**

Batch processing was very popular in the 1970s. The jobs were executed in batches. People used to have a single computer known as a mainframe. Users using batch operating systems do not interact directly with the computer.

Each user prepares their job using an offline device like a punch card and submitting it to the computer operator. Jobs with similar requirements are grouped and executed as a group to speed up processing.

Once the programmers have left their programs with the operator, they sort the programs with similar needs into batches.



* ***Types of OS-Batch System***

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

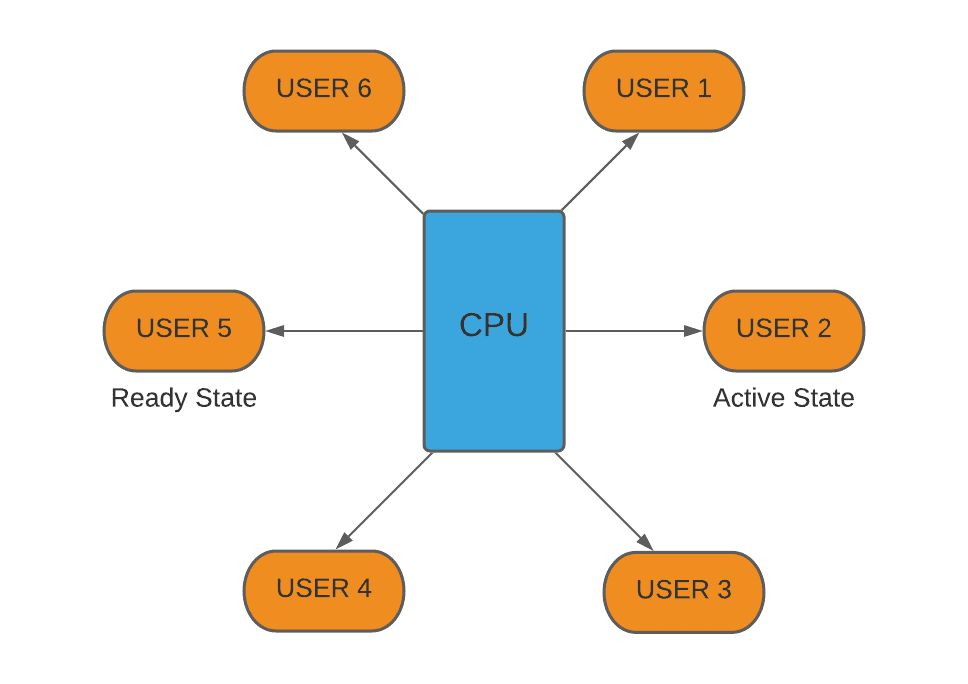
**1. Simple Batched System:**

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.

**2. 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.

* **Timesharing**



An operating system (OS) is basically a collection of software that manages computer hardware resources and provides common services for computer programs. Operating system is a crucial component of the system software in a computer system.

Time-Sharing Operating Systems is one of the important type of operating system.

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

The main difference between Time-Sharing Systems and Multiprogrammed Batch 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 implemented by the CPU by switching between them, but the switches occur so frequently. So, the user can receive an immediate response. For an example, in a transaction processing, the processor executes each user program in a short burst or quantum of computation, i.e.; if n users are present, then each user can get a time quantum. Whenever the user submits the command, the response time is in few seconds at most.

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

***Advantages of Timesharing operating systems are −***

* It provides the advantage of quick response.
* This type of operating system avoids duplication of software.
* It reduces CPU idle time.

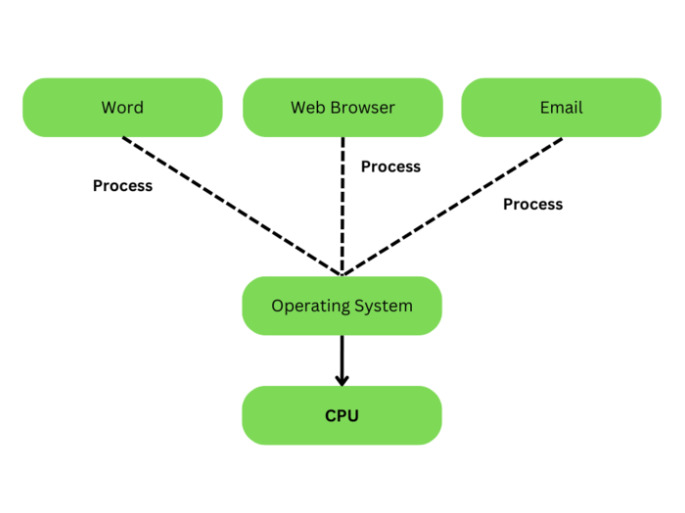
***Disadvantages of Time-sharing operating systems are −***

* Time sharing has problem of reliability.
* Question of security and integrity of user programs and data can be raised.
* Problem of data communication occurs.
* **Multitasking**

An OS that can manage numerous duties or procedures at once is known as a multitasking operating system. In simple terms, it enables the simultaneous operation of numerous programs or procedures while allocating a specific amount of memory and central processing time to each process.

Every task's distribution of resources is controlled by the operating system's kernel, which also ensures that no tasks conflict with one another. Additionally, it offers a way to move among duties rapidly, giving the impression that every process is active at once.

Contemporary systems for computing, which include personal computers, laptops, computer systems, and cellphones, frequently utilize multitasking operating systems. OS like **Windows, Linux, macOS, Android, and iOS** are a few prominent instances of multitasking systems.



Operating systems that support multiple tasks have a number of benefits over those that only support one task. They make multitasking simpler and increase productivity by enabling individuals to operate several programs simultaneously

* **Multiprogramming**

A multiprogramming operating system may run many programs on a single processor computer. If one program must wait for an input/output transfer in a multiprogramming operating system, the other programs are ready to use the CPU. As a result, various jobs may share CPU time. However, the execution of their jobs is not defined to be at the same time period.

When a program is being performed, it is known as a **"Task", "Process"**, and **"Job"**. Concurrent program executions improve system resource consumption and throughput as compared to serial and batch processing systems.

The primary goal of multiprogramming is to manage the entire system's resources. The key components of a multiprogramming system are the file system, command processor, transient area, and I/O control system. As a result, multiprogramming operating systems are designed to store different programs based on sub-segmenting parts of the transient area. The resource management routines are linked with the operating system core functions.

## *Types of the Multiprogramming Operating System*

1. **Multitasking Operating System**
2. **Multiuser Operating System**

### **1. Multitasking Operating System**

A multitasking [operating system](https://www.javatpoint.com/os-tutorial) enables the execution of two or more programs at the same time. The operating system accomplishes this by shifting each program into and out of memory one at a time. When a program is switched out of memory, it is temporarily saved on disk until it is required again.

### **2. Multiuser Operating System**

A multiuser operating system allows many users to share processing time on a powerful central computer from different terminals. The operating system accomplishes this by rapidly switching between terminals, each of which receives a limited amount of processor time on the central computer.

The operating system changes among terminals so quickly that each user seems to have continuous access to the central computer. If there are many users on a system like this, the time it takes the central computer to reply can become more obvious.

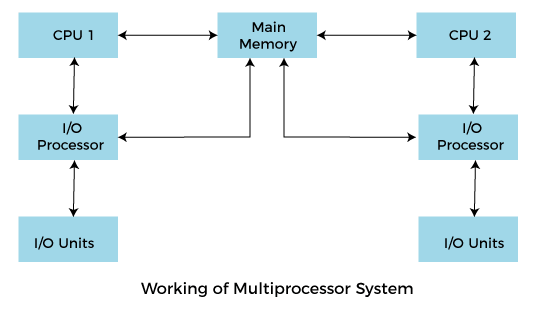
* **Multiprocessing**

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](https://www.javatpoint.com/linux-tutorial) Operating system is one of the most widely used multiprocessing systems.

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



* **Online/Web Operating System**

An Operating System is a system software that acts as an interface between computer hardware and programs requesting I/O. It manages computer hardware, software resources and allows other programs to run.

A Web Operating System is an internet based user interface that allows people to access applications not stored on their computers but completely or partly on Internet. It is a dummy operating system that does not directly interact with computer hardware and depends on traditional operating system to work. In other words, it is an interface for distributed computing system such as cloud. 

* Approach used to make a web/online operating system work
* Web operating system are majorly created using AJAX and Flash.

Flash enables users to create interactive web pages. It is a vector animation software, originally designed to create animations on web pages using Vector graphics. Over the time, it has become more and more controllable via programming. As flash files stream over Internet hassle of downloading whole file to computer before accessing parts of it is reduced.

[AJAX (Asynchronous JavaScript And XML)](https://www.geeksforgeeks.org/ajax-introduction/) is a set of technologies used to create asynchronous web applications. These web applications can send and retrieve data from a server in background without disturbing any other web page, in form of small bits of information.

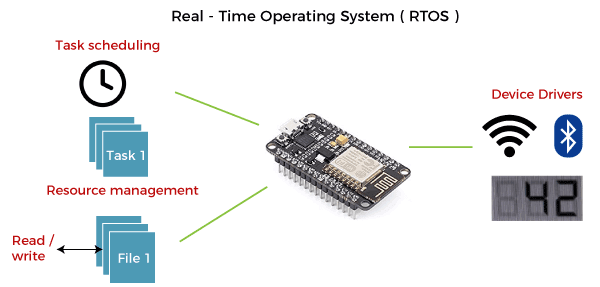
**Example –**   
There are many **WebOS** available on internet. In this example we are using SilveOs.

**Website –**[***https://www.silveos.com/***](https://www.silveos.com/)

All you need to do is login to website and there it is, on your browser a dummy operating system.

* **Real Time**

***What do you mean by 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.**
* **Distributed Operating System.**

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

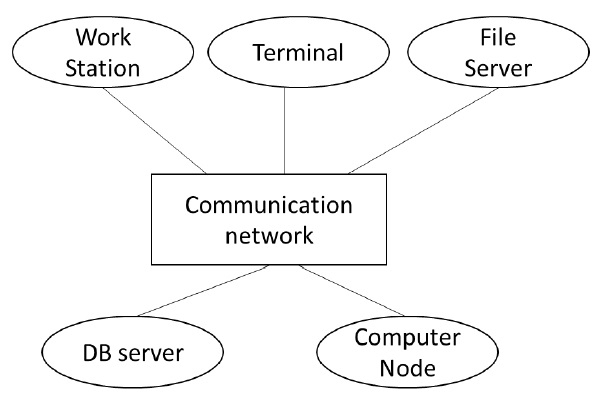
Distributed OS runs on multiple CPUs but for an end-user, it is just an ordinary centralized operating system. It can share all resources like CPU, disk, network interface, nodes, computers, etc. from one site to another site, and it increases the data available on the entire system.

All processors are connected by valid communication media such as high-speed buses and telephone lines, and in which every processor contains its own local memory along with other local processors.

According to this nature, a distributed operating system is known as a loosely coupled system. This operating system involves multiple computers, nodes, and sites, and these components are linked to each other with LAN/WAN lines.

Distributed OS is capable of sharing their computational capacity and I/O files while allowing virtual machine abstraction to users.

The distributed operating system is depicted below −



### **Characteristics of Distributed System:**

* **Resource Sharing:**It is the ability to use any Hardware, Software, or Data anywhere in the System.
* **Openness:** It is concerned with Extensions and improvements in the system (i.e., How openly the software is developed and shared with                                others)
* **Concurrency:**It is naturally present in Distributed Systems, that deal with the same activity or functionality that can be performed by separate users who are in remote locations. Every local system has its independent Operating Systems and Resources.
* **Scalability:** It increases the scale of the system as a number of processors communicate with more users by accommodating to improve the responsiveness of the system.
* **Fault tolerance:** It cares about the reliability of the system if there is a failure in Hardware or Software, the system continues to operate properly without degrading the performance the system.
* **Transparency:**It hides the complexity of the Distributed Systems to the Users and Application programs as there should be privacy in every system.
* **Heterogeneity:**Networks, computer hardware, operating systems, programming languages, and developer implementations can all vary and differ among dispersed system components.
* **Views of operating system**

The operating system can be observed from the point of view of the user or the system. This is known as the user view and system view respectively. An operating system is a framework that enables user application programs to interact with system hardware. The operating system does not perform any functions on its own, but it provides an atmosphere in which various programs and apps can do useful work. The operating system may be observed from the point of view of the user or the system, and it is known as the user view and the system view.

1. **Command language users view:**

As of my last knowledge (chat gpt) update in September 2021, there is no standard or widely recognized command called "language users view" in common operating systems or programming environments. The term "language users view" does not correspond to a specific command or functionality that is part of standard system utilities or programming languages.

If "language users view" is related to a specific software application or system that was introduced after my last update, I would recommend referring to the documentation or help resources of that particular software or system to understand its usage and functionality.

If you have more context or specific details about what you are trying to accomplish or the environment you are working in, please provide additional information, and I'll do my best to assist you further.

1. **System call users view:**

As of my last knowledge (chat got) update in September 2021, there is no standard or widely recognized system call called "users view" in common operating systems. System calls are low-level interfaces provided by the operating system to interact with its services and resources. These system calls are typically used by programs and applications to perform various operations such as file I/O, process management, and memory allocation.

If you are referring to a "users view" system call, it might be a specialized or custom system call implemented in a specific software or environment that was introduced after my last update. To understand how this system call works and its purpose, you would need to refer to the documentation or resources specific to the software or system in question.

If you have more context or specific details about the environment or software you are working with, please provide additional information, and I'll do my best to assist you further.

* **Structure of OS**

The operating system can be implemented with the help of various structures. The structure of the OS depends mainly on how the various standard components of the operating system are interconnected and melded into the kernel.

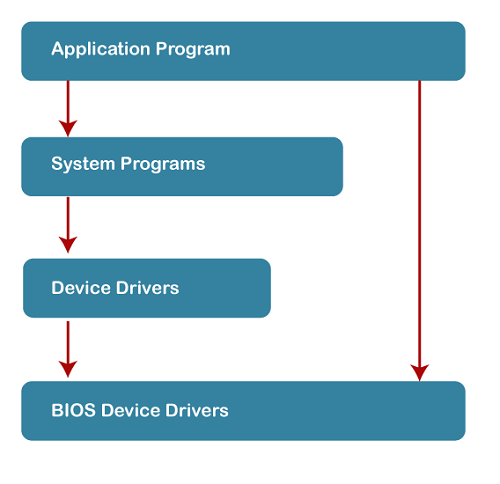
A design known as an operating system enables user application programs to communicate with the machine’s hardware. Given its complex design and need to be easy to use and modify, the operating system should be constructed with the utmost care. A straightforward way to do this is to supernaturally develop the operating system. These parts must each have unique inputs, outputs, and functionalities.

This article discusses a variety of operating system implementation structures, including those listed below, as well as how and why they function. Additionally, the operating system structure is defined.

Depending on this, we have the following structures in the operating system.

1. **Simple Structure**
2. **Monolithic Structure**
3. **Layered Structure**
   1. **Simple Structure**

Simple Structure is a type of operating system design where the whole system is managed by a single program that controls all hardware resources and services without any separation. This design is straightforward and efficient but may have issues with security and stability. However, it is not commonly used in modern operating systems.



#### **Advantages of Simple Structure:**

* Simple Structure operating systems are easy to create, and the design is simple and efficient.
* They have low overhead and are relatively fast compared to more complex operating system designs.

#### **Disadvantages of Simple Structure:**

* Because the entire operating system runs in kernel mode, it lacks protection between applications and the operating system, making it more vulnerable to crashes and security breaches.
* Simple Structure operating systems are not very scalable and may not be able to handle large or complex systems.

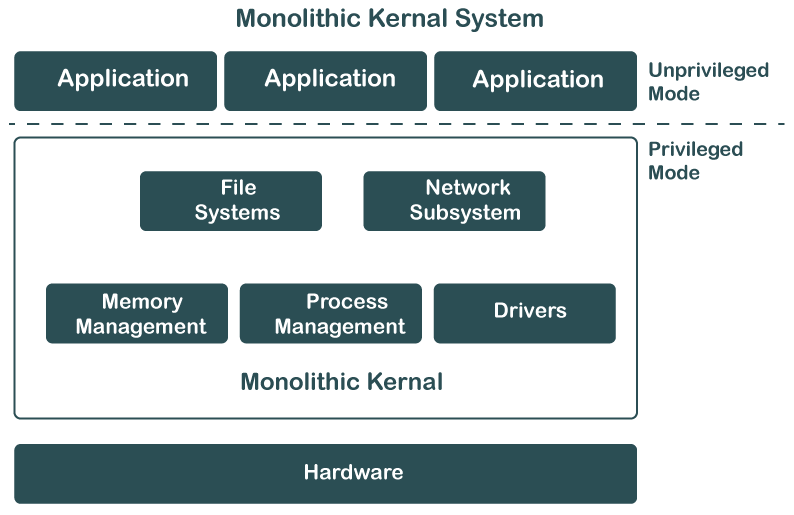
**2. Monolithic Structure**

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:**

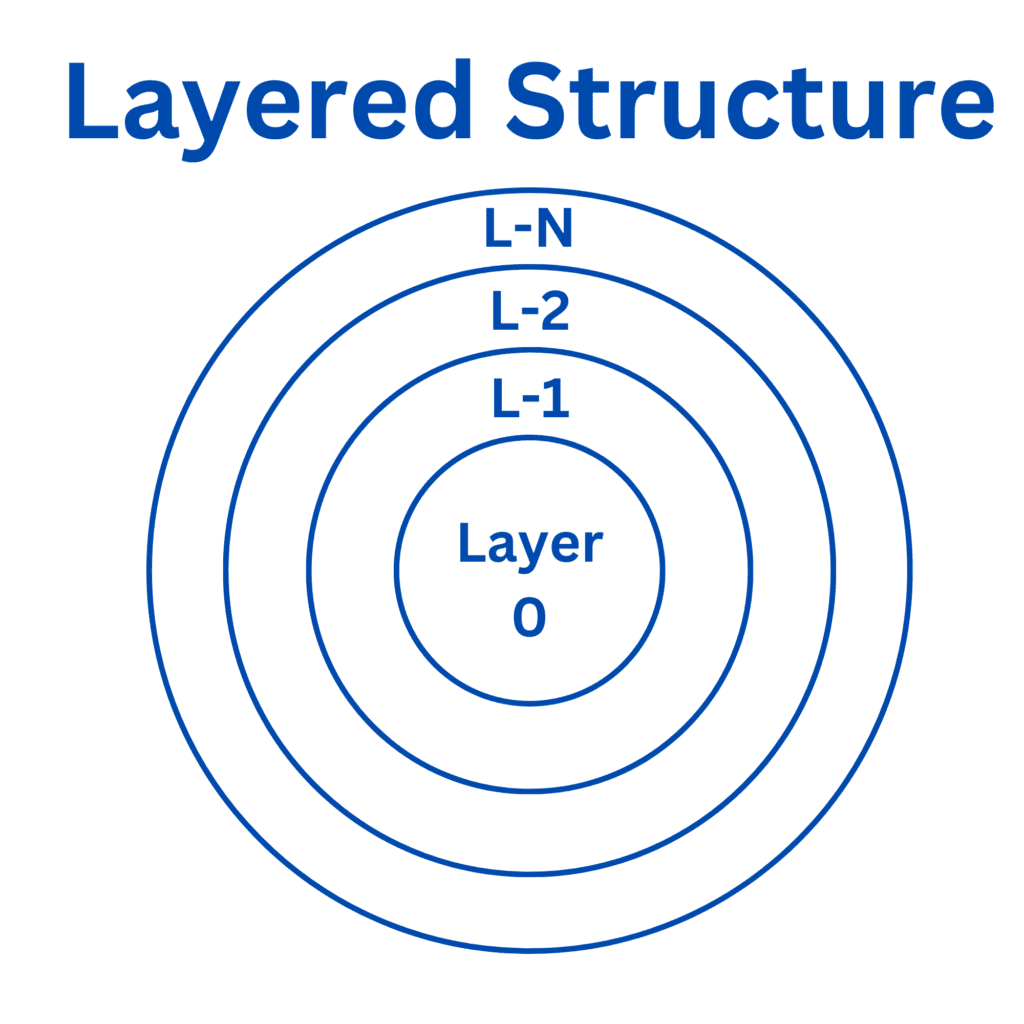
* Because layering is unnecessary and the kernel alone is responsible for managing all operations, it is easy to design and execute.
* Due to the fact that functions like memory management, file management, process scheduling, etc., are implemented in the same address area, the monolithic kernel runs 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.

**3. Layered Structure**

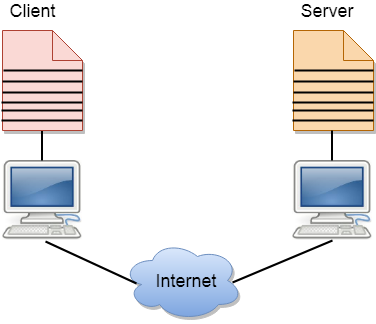
A Layered Structure is a type of operating system design that divides the operating system into different layers, each providing a specific set of services to the layer above it. This design makes the operating system easier to maintain and more flexible, as each layer can be developed and updated independently.

It also makes the operating system more secure and less likely to crash, by providing better protection between layers. The lowest layer interacts directly with the computer hardware, while the highest layer provides services to the user.

#### **Advantages of Layered Structure:**

* The layered design makes the operating system easier to manage and update, as each layer can be changed independently.
* The layered structure makes the operating system more secure and less likely to crash, as it provides better protection between layers.

#### **Disadvantages of Layered Structure:**

* The layered structure can slow down the operating system, as each layer needs to communicate with the layer above it.
* It can be hard to define the boundaries between layers, which can make it harder to find and fix problems in the operating system.
* **Client and Server model**
* A client and server networking model is a model in which computers such as servers provide the network services to the other computers such as clients to perform a user based tasks. This model is known as client-server networking model.
* The application programs using the client-server model should follow the given below strategies:
* An application program is known as a client program, running on the local machine that requests for a service from an application program known as a server program, running on the remote machine.
* A client program runs only when it requests for a service from the server while the server program runs all time as it does not know when its service is required.
* A server provides a service for many clients not just for a single client. Therefore, we can say that client-server follows the many-to-one relationship. Many clients can use the service of one server.
* Services are required frequently, and many users have a specific client-server application program. For example, the client-server application program allows the user to access the files, send e-mail, and so on. If the services are more customized, then we should have one generic application program that allows the user to access the services available on the remote computer.

## Client :

A client is a program that runs on the local machine requesting service from the server. A client program is a finite program means that the service started by the user and terminates when the service is completed.

## Server :

A server is a program that runs on the remote machine providing services to the clients. When the client requests for a service, then the server opens the door for the incoming requests, but it never initiates the service.

A server program is an infinite program means that when it starts, it runs infinitely unless the problem arises. The server waits for the incoming requests from the clients. When the request arrives at the server, then it responds to the request.

***Advantages of Client-server networks:***

1. **Centralized:** Centralized back-up is possible in client-server networks, i.e., all the data is stored in a server.
2. **Security:** These networks are more secure as all the shared resources are centrally administered.
3. **Performance:** The use of the dedicated server increases the speed of sharing resources. This increases the performance of the overall system.
4. **Scalability:** We can increase the number of clients and servers separately, i.e., the new element can be added, or we can add a new node in a network at any time.

***Disadvantages of Client-Server network:***

1. **Traffic Congestion** is a big problem in Client/Server networks. When a large number of clients send requests to the same server may cause the problem of Traffic congestion.
2. It does not have a robustness of a network, i.e., when the server is down, then the client requests cannot be met.
3. A client/server network is very decisive. Sometimes, regular computer hardware does not serve a certain number of clients. In such situations, specific hardware is required at the server side to complete the work.
4. Sometimes the resources exist in the server but may not exist in the client. For example, If the application is web, then we cannot take the print out directly on printers without taking out the print view window on the web.

* **User Operating System Interface:**

The user and operating system are connected with each other with the help of interface, so interface is used to connect the user and OS.

In computers there are different types of interface that can be used for connection with computers to users and their connection is responsible for data transfer.

Also, in computers there are different interfaces. These interfaces are not necessarily used but can be used in computers whenever it is needed. So, different types of tasks can be performed by the help of different interfaces.

* 1. **Command line interface :**

The command-line interface is an interface whenever the user needs to have different commands regarding the input and output and then a task is performed so this is called the command-line argument and it is used to execute the output and create, delete, print, copy, paste, etc.

All these operations are performed with the help of the command-line interface.

The interface is always connected to the OS so that the command given by the user directly works by the OS and a number of operations can be performed with the help of the command line interface because multiple commands can be interrupted at same time and execute only one.

The command line interface is necessary because all the basic operations in the computer are performed with the help of the OS and it is responsible for memory management. By using this we can divide the memory and we can use the memory.

### ***Advantages of Command Line Interface***

* Controls OS or application
* faster management
* ability to store scripts which helps in automating regular tasks.
* Troubleshoot network connection issues.

### ***Disadvantages of Command Line Interface***

* The steeper learning curve is associated with memorizing commands and a complex syntax.
* Different commands are used in different shells.
  1. **Graphical user interface(GUI)**

The graphical user interface is used for playing games, watching videos, etc. these are done with the help of GUI because all these applications require graphics.

The GUI is one of the necessary interfaces because only by using the user can clearly see the picture, play videos.

So we need GUI for computers and this can be done only with the help of an operating system.

When a task is performed in the computer then the OS checks the task and defines the interface which is necessary for the task. So, we need GUI in the OS.

**The basic components of GUIs are −**

* Start menu with program groups
* Taskbar which showing running programs
* Desktop screen
* Different icons and shortcuts.

***Advantages of GUIs:***

1. **User-Friendly:** GUIs are generally more user-friendly, making them accessible to a wider range of users, including those who may not be familiar with command-line syntax.
2. **Intuitive**: GUIs often use icons, buttons, and menus, which can make it easier for users to understand and perform tasks without having to memorize commands.
3. **Multitasking**: GUIs are typically better suited for multitasking, as they allow users to open multiple windows or applications simultaneously and switch between them with ease.
4. **Visual Feedback:** GUIs provide immediate visual feedback, such as progress bars or error messages, which can help users understand the status of a task.
5. **Accessibility**: GUIs can be more accessible to users with disabilities, as they often support screen readers and other assistive technologies.

***Disadvantages of GUIs:***

1. **Resource Intensive:** GUIs tend to be more resource-intensive than CLIs, requiring additional memory and processing power. This can be a disadvantage on systems with limited resources.
2. **Slower for Experts:** Experienced users who are proficient in command-line operations may find GUIs slower for performing tasks, as they often involve more clicks and mouse movements.
3. **Limited Scripting and Automation:** While GUIs are great for interactive tasks, they are not as conducive to scripting and automation as CLIs. Automation through GUIs can be complex and limited.
4. **Less Control:** GUIs abstract many low-level details, which can be an advantage for novice users but a disadvantage for those who need precise control over system functions.
5. **Not Ideal for Remote or Headless Systems**: GUIs are not practical for remote management or for systems that operate without a display (headless servers), as they rely on graphical elements.

* **What Is a System Call?**

A system call is an interface between a [program](https://phoenixnap.com/glossary/what-is-a-program) running in user space and the [operating system](https://phoenixnap.com/glossary/operating-system) (OS). Application programs use system calls to request services and functionalities from the OS's kernel. This mechanism allows the program to call for a service, like reading from [a file](https://phoenixnap.com/glossary/what-is-a-file), without accessing system resources directly.

When a program invokes a system call, the execution context switches from user to kernel mode, allowing the system to access hardware and perform the required operations safely. After the operation is completed, the control returns to user mode, and the program continues its execution.

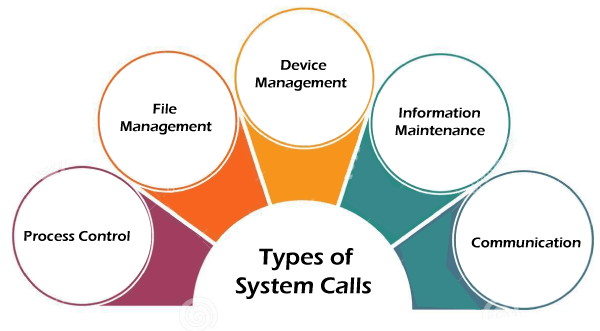
This layered approach facilitated by system calls:

* Ensures that hardware resources are isolated from user space processes.
* Prevents direct access to the kernel or hardware memory.
* Allows application code to run across different [hardware](https://phoenixnap.com/glossary/what-is-hardware) architectures.

## Types of System Calls

There are commonly five types of system calls. These are as follows:

1. **Process Control**
2. **File Management**
3. **Device Management**
4. **Information Maintenance**
5. **Communication**



Now, you will learn about all the different types of system calls one-by-one.

* 1. **Process Control :**

Process control is the system call that is used to direct the processes. Some process control examples include creating, load, abort, end, execute, process, terminate the process, etc.

* 1. **File Management :**

File management is a system call that is used to handle the files. Some file management examples include creating files, delete files, open, close, read, write, etc.

* 1. **Device Management :**

Device management is a system call that is used to deal with devices. Some examples of device management include read, device, write, get device attributes, release device, etc.

* 1. **Information Maintenance :**

Information maintenance is a system call that is used to maintain information. There are some examples of information maintenance, including getting system data, set time or date, get time or date, set system data, etc.

* 1. **Communication :**

Communication is a system call that is used for communication. There are some examples of communication, including create, delete communication connections, send, receive messages, etc.