Introduction

- Computer System = Hardware + Software
- Software = Application Software + System Software(OS)
- An Operating System is a system Software that acts as an intermediary/interface between a user of a computer and the computer hardware.
- Operating system goals:
  - Execute user programs and make solving user problems easier
  - Make the computer system convenient to use
  - Use the computer hardware in an efficient manner

Introduction

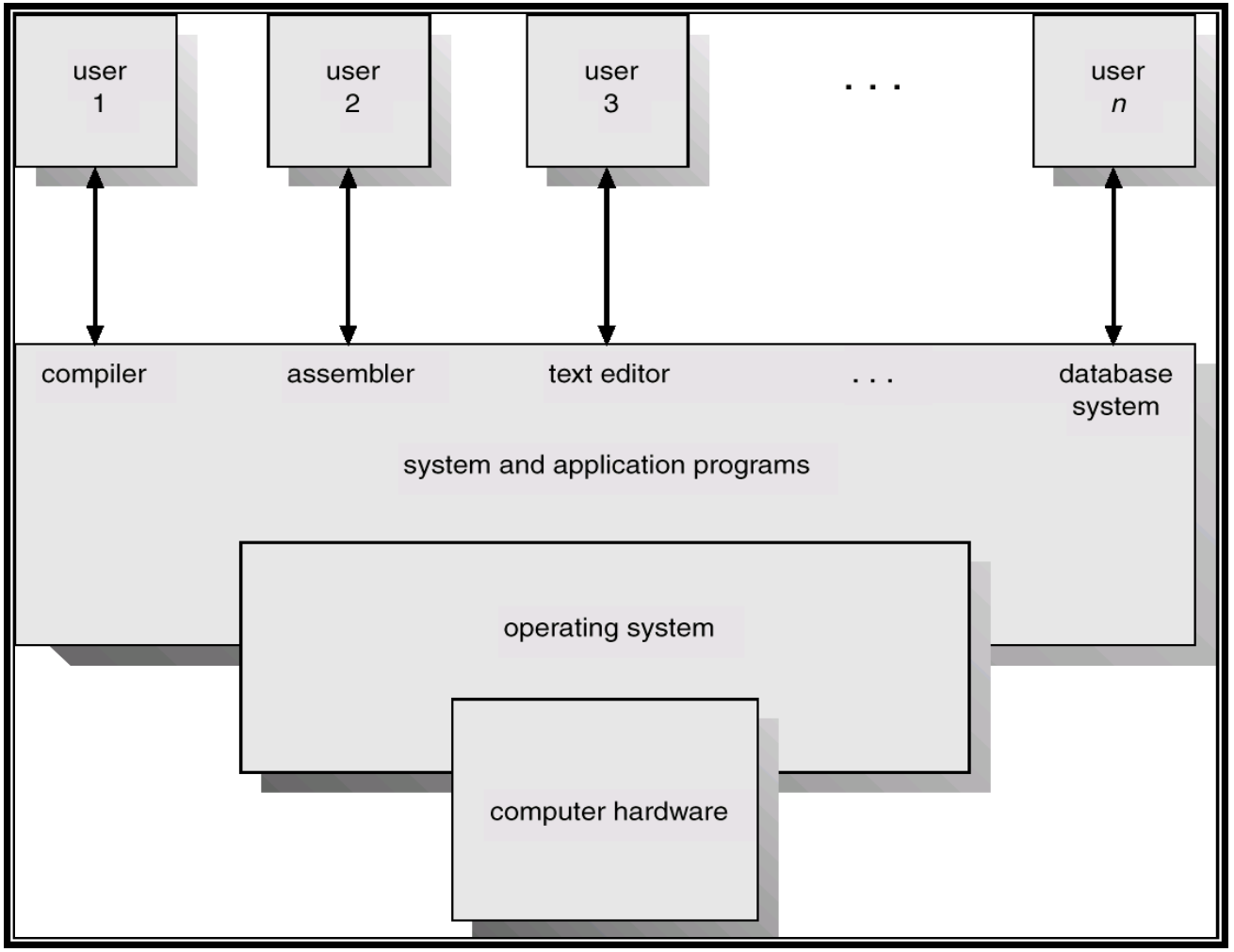
- The purpose of an operating system is to provide an environment in which a user can execute programs in a convenient and efficient manner.
- An operating system is a software that manages the computer hardware.
- The hardware must provide appropriate mechanisms to ensure the correct operation of the computer system and to prevent user programs from interfering with the proper operation of the system.
- Some examples of operating systems are UNIX, Linux, MS-DOS, MS-Windows, MacOS etc

- **Kernel** is central component of an operating system that manages operations of computer and hardware.
- It basically manages operations of memory and CPU time.
- It is core component of an operating system.
- Kernel acts as a bridge between applications and data processing performed at hardware level using inter-process communication and system calls.
- Kernel loads first into memory when an operating system is loaded and remains into memory until operating system is shut down again.
- It is responsible for various tasks such as disk management, task management, and memory management.

Computer System Components

1. Hardware – provides basic computing resources (CPU, memory, I/O devices).
2. Operating system – controls and coordinates the use of the hardware among the various application programs for the various users.
3. Applications programs – define the ways in which the system resources are used to solve the computing problems of the users (compilers, database systems, video games, business programs).
4. Users (people, machines, other computers).

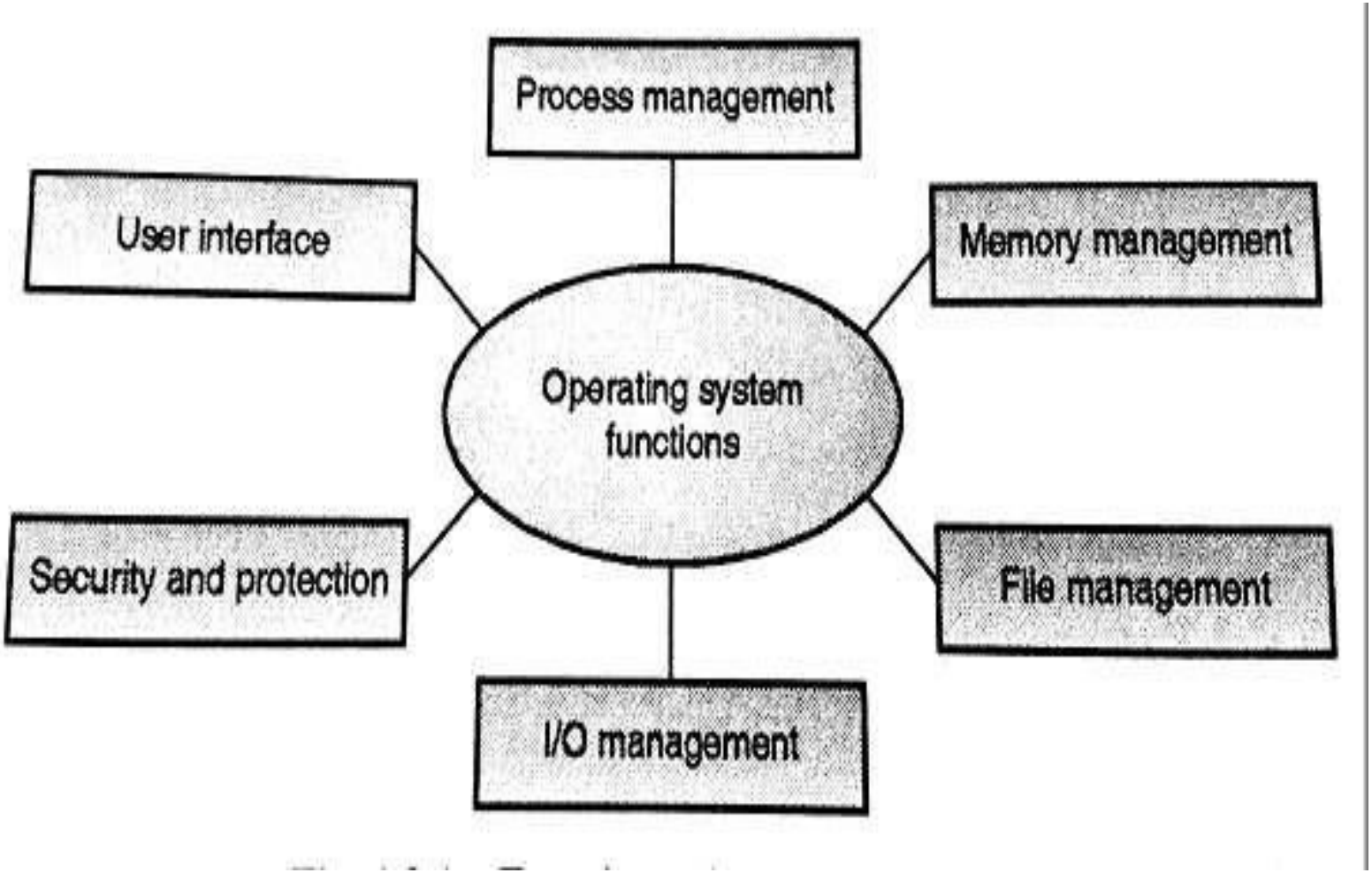
Abstract View of System Components



Operating System Definitions

- Resource allocator – manages and allocates resources.
- Control program – controls the execution of user programs and operations of I/O devices .
- User view and System view
- Kernel – the one program running at all times (all else being application programs).

Functions of Operating System



1. Process Management

- A process is a program in execution.
- A process needs certain resources, including CPU time, memory, files, and I/O devices, to accomplish its task.
- The operating system is responsible for the following activities in connection with process management.
- Process creation and deletion.
- process suspension and resumption.
- Provision of mechanisms for:
  i. process synchronization
  ii. process communication

1. Process Management

- Based on priority, it is important to allow more important processes to execute first than others.
- In a multi-programming environment, the OS decides the order in which processes have access to the processor, and how much processing time each process has.
- This function of OS is called process scheduling.

2. Memory management

- Memory is a large array of words or bytes, each with its own address.
- It is a repository of quickly accessible data shared by the CPU and I/O devices.
- Main memory is a **volatile** storage device. When the computer made turn off everything stored in RAM will be erased automatically.
- In addition to the physical RAM installed in your computer, most modern operating systems allow your computer to use a *virtual memory system*.

2. Memory management

- *Virtual memory allows your computer to use part of a permanent storage device (such as a hard disk) as extra memory.*
- The operating system is responsible for the following activities in connections with memory management:
  - Keep track of which parts of memory are currently being used and by whom.
  - Decide which processes to load when memory space becomes available.
  - Allocate and de-allocate memory space as needed.

3. File Management

- A file is a collection of related information defined by its creator.
- *File systems provide the conventions for the encoding, storage and management of data on a storage device such as a hard disk.*
- A file system is organized into directories for efficient or easy navigation and usage. These directories may contain other directories and other files.
- An Operating System carries out the following file management activities.
- It keeps track of where information is stored, user access settings and status of every file, and more… These facilities are collectively known as the file system.

3. File Management

- The operating system is responsible for the following activities in connections with file management:
  - ✦ File creation and deletion.
  - ✦ Directory creation and deletion.
  - ✦ Support of primitives for manipulating files and directories.
  - ✦ Mapping files onto secondary storage.
  - ✦ File backup on stable (nonvolatile) storage media.

4. I/O Device Management

- An OS manages device communication via their respective drivers.
- It performs the following activities for device management.

→ Keeps track of all devices connected to the system.

→ designates a program responsible for every device known as the Input/Output controller.

→ Decides which process gets access to a certain device and for how long.

→ Allocates devices in an effective and efficient way.

→ Deallocates devices when they are no longer required.

4. I/O Device Management

- *Device controllers are components on the motherboard that act as an interface between the CPU and the actual device.*
- *Device drivers, which are the operating system software components that interact with the devices controllers.*

.

5. Information Management

- The Operating System is responsible for using all the information and resources available on the machine in the most protected way.
- The operating system uses password protection to protect user data and similar other techniques.
- It also prevents unauthorized access to programs and user data by assigning access right permission to files and directories.

6. User Interface Mechanism

- A **user interface** (**UI**) controls how you enter data and instructions and how information is displayed on the screen
- There are two types of user interfaces

1. Command Line Interface

In a command-line interface, a user types commands represented by short keywords or abbreviations or presses special keys on the keyboard to enter data and instructions

2. Graphical user Interface With a graphical user interface (GUI), you interact with menus and visual images

Batch Systems

- Early computers were physically enormous machines run from a console
- The common input devices were card readers and tape drives
- The output devices were line printers, tape drives and card punches.
- The user did not directly interact with the system
- The user prepared a job which includes program, data and control information about the nature of the job (control cards) and submitted it to the computer operator.

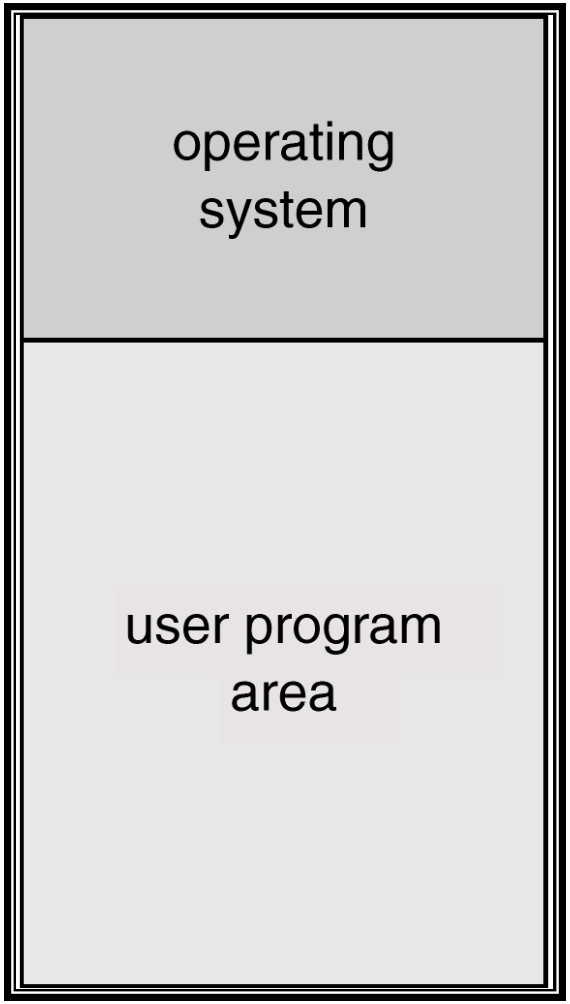
Batch Systems

- The job was usually in the form of punch cards.At some later time the output appeared
- Output consisted of the result of the program as well as a dump of the final memory and register contents for debugging
- The OS in these early computers was fairly simple
- It's major task was to transfer control automatically from one job to the next.

Batch Systems

- The OS was always resident in memory
- This type of OS accepts more than one jobs and these jobs are batched/ grouped together according to their similar requirements.
- This is done by computer operator.
- Whenever the computer becomes available, the batched jobs are sent for execution and gradually the output is sent back to the user.
- It allowed only one program at a time.
- This OS is responsible for scheduling the jobs according to priority and the resource required.

Memory Layout for a Simple Batch System



Simple Batch Systems

- Reduce setup time by batching similar jobs
- Automatic job sequencing – automatically transfers control from one job to another.
- First rudimentary operating system.

2. Multiprogramming Operating System:

- This type of OS is used to execute more than one jobs simultaneously by a single processor.
- A single user cannot ,keep the CPU or the I/O devices busy at all times.
- **Multiprogramming** increases CPU utilization by organizing jobs so that the CPU always has one job to execute.
- The concept of multiprogramming is described as follows:

* All the jobs that enter the system are stored in the job pool(in disc).

*The operating system picks and begins to execute one of the jobs in the main memory and begins to execute.

2. Multiprogramming Operating System:

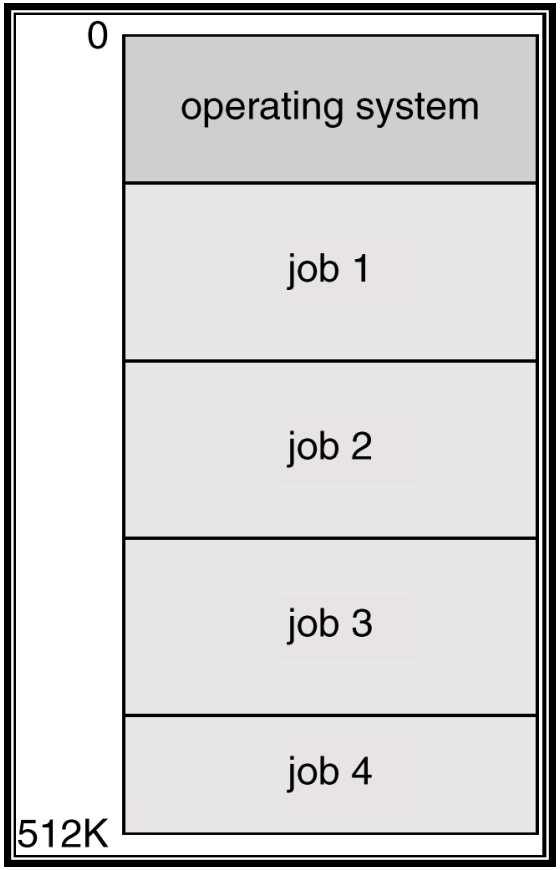
*During execution, the job may have to wait for some task, such as an I/O operation, to complete.

* In a multiprogramming system, the operating system simply switches to ,and executes another job . When that job needs to wait, the CPU is switched to another job, and so on.

- When the first job finishes waiting and it gets the CPU back.
- As long as at least one job needs to execute, the CPU is never idle.
- Multiprogramming operating systems use the mechanism of job scheduling and CPU scheduling.

Multiprogrammed Batch Systems

Several jobs are kept in main memory at the same time, and the CPU is multiplexed among them.



OS Features Needed for Multiprogramming

- I/O routine supplied by the system.
- Memory management – the system must allocate the memory to several jobs.
- CPU scheduling – the system must choose among several jobs ready to run.
- Allocation of devices.

Advantages of Batch Operating System:

- Processors of the batch systems know how long the job would be when it is in queue
- Multiple users can share the batch systems
- The idle time for the batch system is very less
- It is easy to manage large work repeatedly in batch systems

Disadvantages of Batch Operating System:

- The computer operators should be well known with batch systems
- Batch systems are hard to debug
- It is sometimes costly
- The other jobs will have to wait for an unknown time if any job fails

Examples of Batch based Operating System: IBM's MVS

Time sharing (or multitasking) OS

- ❖ Time sharing (or multitasking) OS is a logical extension of multiprogramming.
- ❖ It provides extra facilities such as:
- Faster switching between multiple jobs to make processing faster.
- Allows multiple users to share computer system simultaneously.
- The users can interact with each job while it is running.
- The user gives instructions to the OS using keyboard or mouse and waits for immediate results and the response time should be short.

Time sharing (or multitasking) OS

- These systems uses the concept of virtual memory for effective utilization of memory space.
- Hence, in this OS, no jobs are discarded.
- Each one is executed using virtual memory concept.
- It uses CPU scheduling, memory management, disc management and security management.
- Examples: CTSS, MULTICS, CAL, UNIX,windows etc.
- Allows multiple users to share computer system simultaneously.
- As the system switches from one user to the next ,each user is given the impression that the entire system is dedicated to his/her use, even though it is being shared among many users.

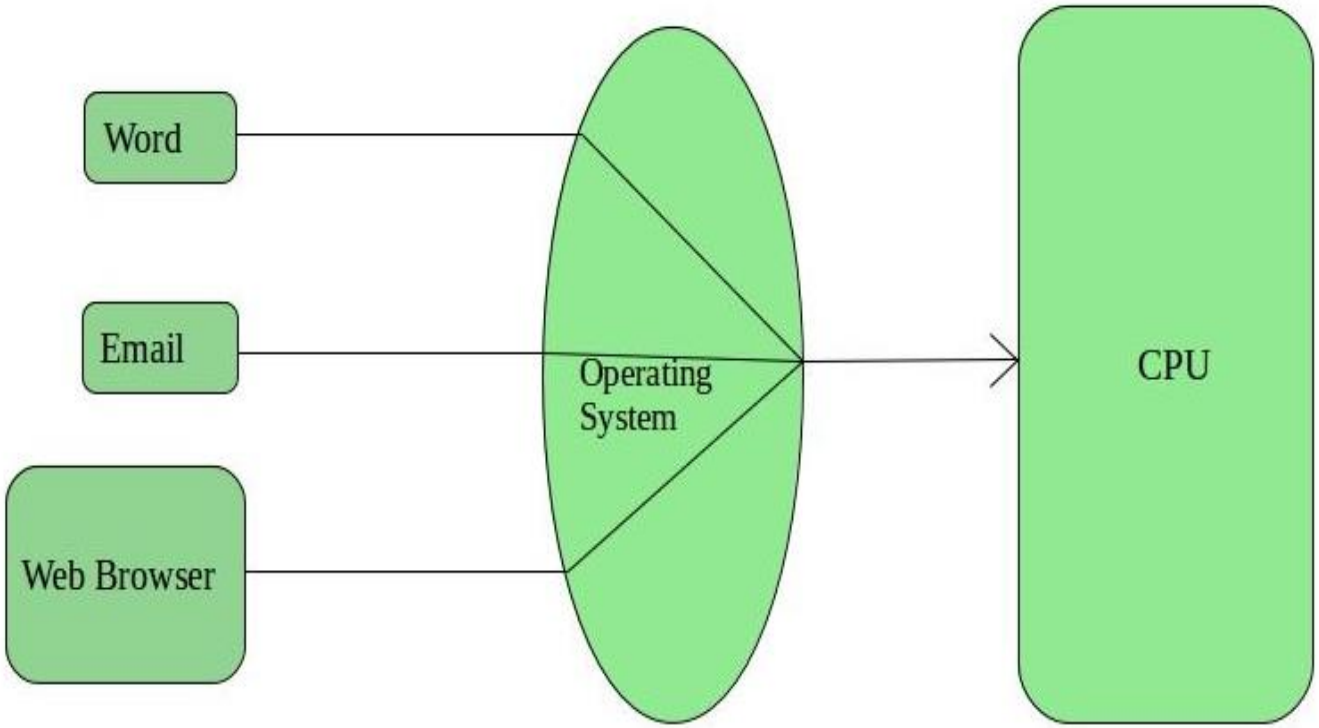
Time-Sharing Systems–Interactive Computing

- A time shared operating system uses CPU scheduling and multi-programming to provide each user with a small portion of a shared computer at once.
- Each user has at least one separate program in memory.
- A program loaded into memory and executes, it performs a short period of time either before completion or to complete I/O.
- This short period of time during which user gets attention of CPU is known as time slice, time slot or quantum.

Time sharing (or multitasking) OS

- Time shared operating systems are more complex than multiprogrammed operating systems.
- In both, multiple jobs must be kept in memory simultaneously, so the system must have memory management and security.
- To achieve a good response time, jobs may have to swap in and out of disk from main memory which now serves as a backing store for main memory.
- A common method to achieve this goal is virtual memory, a technique that allows the execution of a job that may not be completely in memory.

Time-Sharing Operating Systems



Time-Sharing Operating Systems

- 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.
- **Advantages of Time-Sharing OS:**
  - Each task gets an equal opportunity
  - CPU idle time can be reduced
- **Disadvantages of Time-Sharing OS:**
  - Reliability problem
  - One must have to take care of the security and integrity of user programs and data
- **Examples of Time-Sharing OS**

  Multics, Unix, etc.

Real-Time Systems

- A real-time operating system (RTOS) is a multitasking operating system intended for applications with fixed deadlines (real-time computing).
- Well-defined fixed-time constraints.
- Such applications include some small embedded systems, automobile engine controllers, industrial robots, spacecraft, industrial control, and some large-scale computing systems.
- Often used as a control device in a dedicated application such as controlling scientific experiments, medical imaging systems, industrial control systems, and some display systems.
- The real time operating system can be classified into two categories:
- 1. hard real time system and
- 2. soft real time system.

Real-Time Systems (Cont.)

- Hard real-time:
- A hard real-time system guarantees that critical tasks be completed on time.
- This goal requires that all delays in the system be bounded, from the retrieval of stored data to the time that it takes the operating system to finish any request made of it.
- Such time constraints dictate the facilities that are available in hard real-time systems.
  - Secondary storage is limited or absent, data stored in short term memory, or read-only memory (ROM)
  - Conflicts with time-sharing systems, not supported by general-purpose operating systems.

Real-Time Systems (Cont.)

- Soft real-time
- A soft real-time system is a less restrictive type of real-time system.
- Here, a critical real-time task gets priority over other tasks and retains that priority until it completes.
- Soft real time system can be mixed with other types of systems.
- Due to less restriction, they are risky to use for industrial control and robotics.
  - Limited utility in industrial control of robotics
  - Useful in applications (multimedia, virtual reality) requiring advanced operating-system features.

- **Advantages of RTOS:**
- **Maximum Consumption:** Maximum utilization of devices and system, thus more output from all the resources
- **Task Shifting:** The time assigned for shifting tasks in these systems are very less.
- **Focus on Application:** Focus on running applications and less importance to applications which are in the queue.
- Real-time **operating system in** the **embedded system:** Since the size of programs are small, RTOS can also be used in embedded systems like in transport and others.
- **Error Free:** These types of systems are error-free.
- **Memory Allocation:** Memory allocation is best managed in these types of systems.

- **Disadvantages of RTOS:**
- **Limited Tasks:** Very few tasks run at the same time and their concentration is very less on few applications to avoid errors.
- **Use heavy system resources:** Sometimes the system resources are not so good and they are expensive as well.
- **Complex Algorithms:** The algorithms are very complex and difficult for the designer to write on.
- **Device driver and interrupt signals:** It needs specific device drivers and interrupts signals to respond earliest to interrupts.
- **Examples of Real-Time Operating Systems are:**
- Scientific experiments,
- medical imaging systems,
- industrial control systems,
- weapon systems,
- robots,
- air traffic control systems, etc.

- **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.
- Eg:MS DOS

- 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.

Advantages of Single User Operating System

- **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:** 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.

multi-user operating system

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

- A multi-user operating system varies from a connected single-user operating system in that each user accesses the same operating system from different machines.