



JAIN
DEEMED-TO-BE UNIVERSITY

SCHOOL OF
COMPUTER
SCIENCE AND IT

OPERATING SYSTEMS (22BCA2C03)

Module 1: Introduction to Operating System

Syllabus

Introduction to Operating System:

Introduction, Objectives and Functions of OS, Evolution of OS, OS Structures, OS Components, OS Services, System calls, Types of System calls, System programs, Virtual Machines.

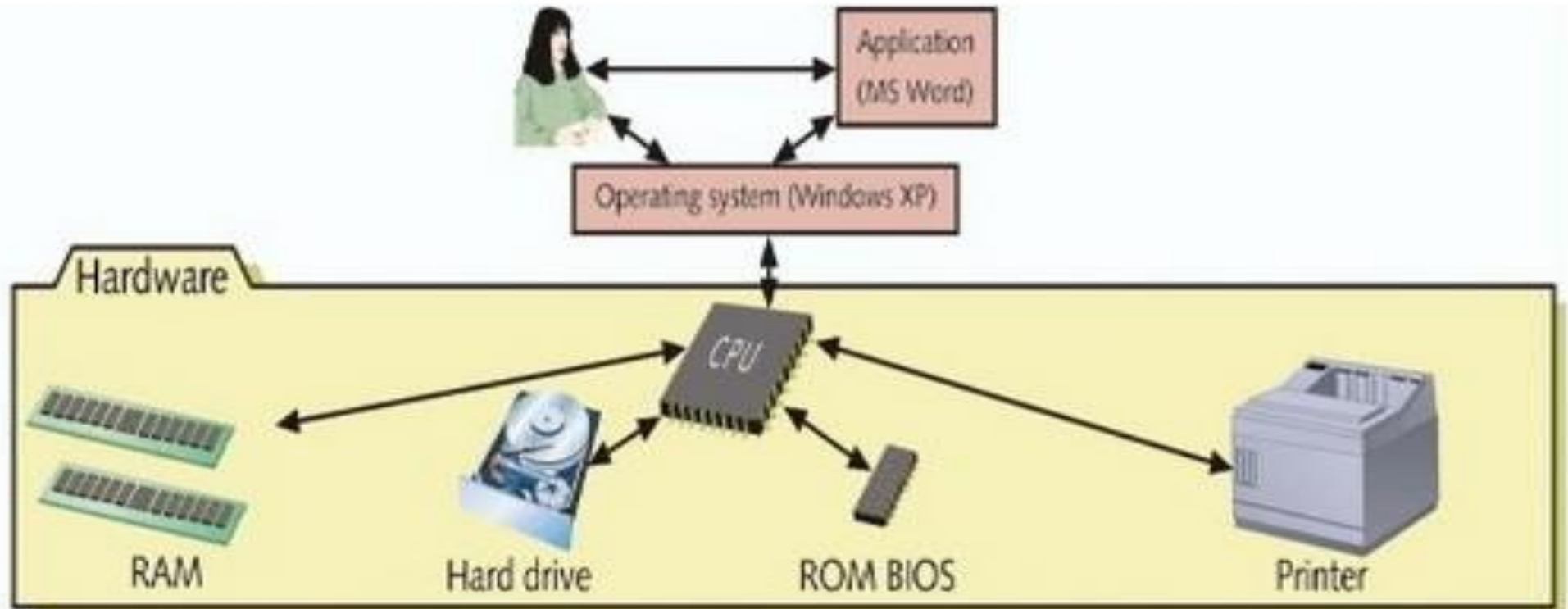
What is an Operating System?

- A computer uses hardware and software to complete any of its tasks.
- These resources include input-output (I/O) devices, memory storage, and other software which promote the functioning of the system.
- Thus, an operating system acts as the resource manager as it manages internally, all the resources and assigns them to particular programs as per the requirement of the task.

What is an Operating System?

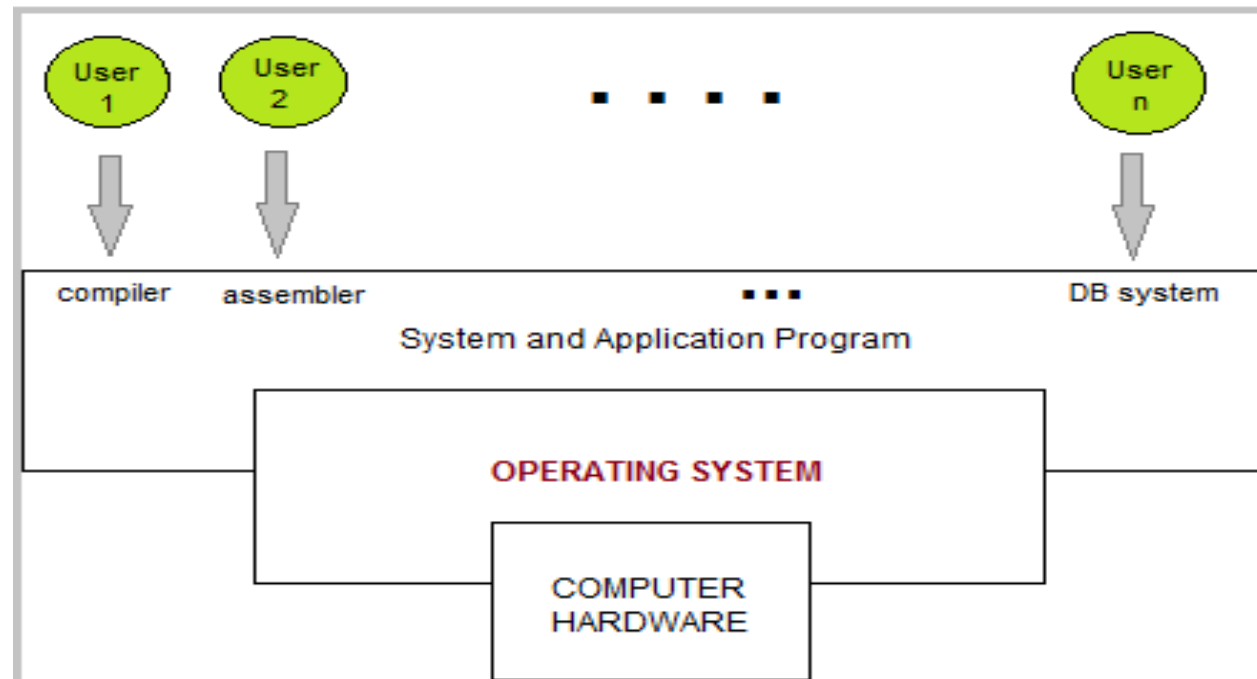
- An operating system is a computer program that handles the computer hardware.
- It equips a base for application programs and plays as an emissary between the user and the computer hardware.

What is an operating system (OS)?



- An operating system is the interface between the user and the machine.

Four Components of a Computer System



OBJECTIVES

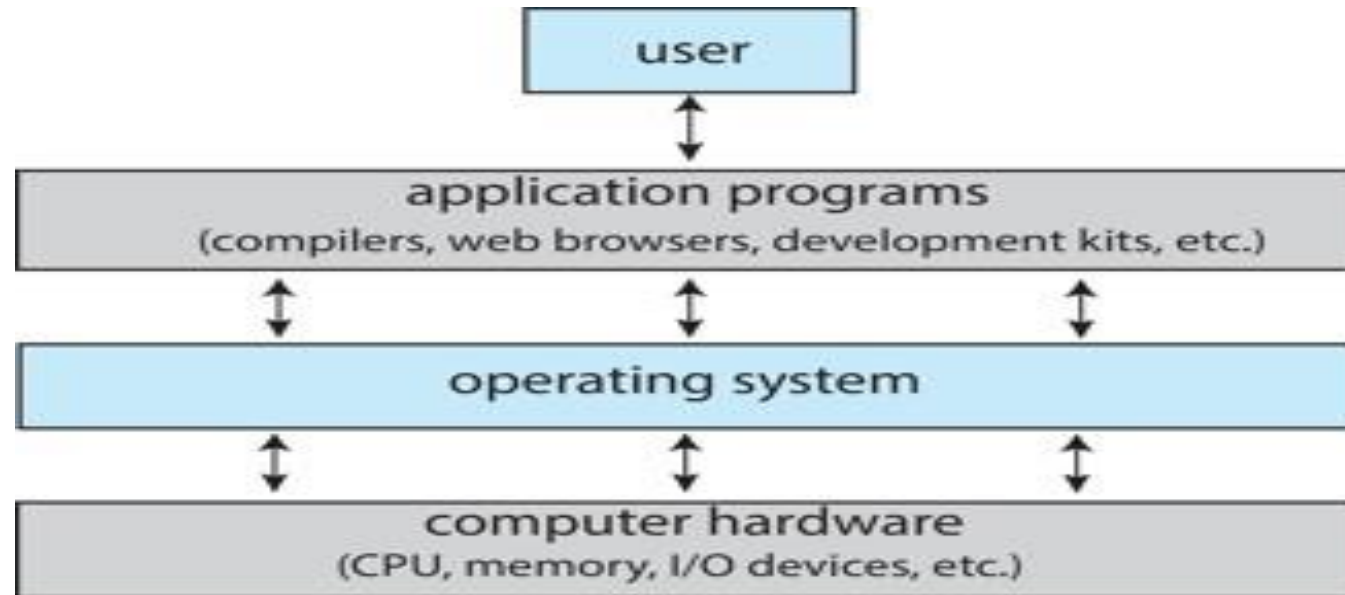
The objectives of the operating system are –

- To make the computer system convenient to use in an efficient manner.
- To hide the details of the hardware resources from the users.
- To provide users a convenient interface to use the computer system.
- To act as an intermediary between the hardware and its users, making it easier for the users to access and use other resources.

OBJECTIVES

- To manage the resources of a computer system.
- To keep track of who is using which resource, granting resource requests, and mediating conflicting requests from different programs and users.
- To provide efficient and fair sharing of resources among users and programs.

Abstract View of Components of Computer



Various Operating Systems Today



Loading of OS

- A boot program loads an Operating system when the computer is switched on. An operating system controls all the programs on the computer.
- The application programs send a request to the operating system for services over a specified Application Program Interface (API).
- An operating system does not perform any necessary function by itself. It provides an environment in which other programs can do useful work.

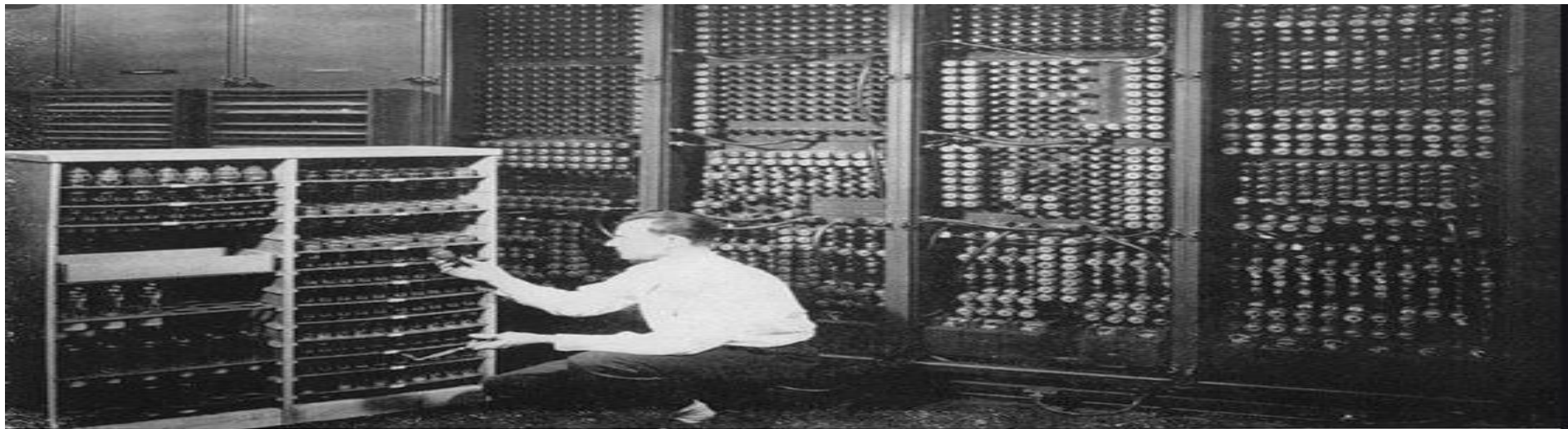
Evolution of OS

Early Evolution:

- 1945- ENIAC
- 1949- EDSAC and EDVAC
- 1949- BINAC- A successor to the ENIAC
- 1951- UNIVAC
- 1952- IBM 701
- 1956- The interrupt
- 1954-1957- The development of FORTRAN

ENIAC-Electronic Numerical Integrator and Computer.

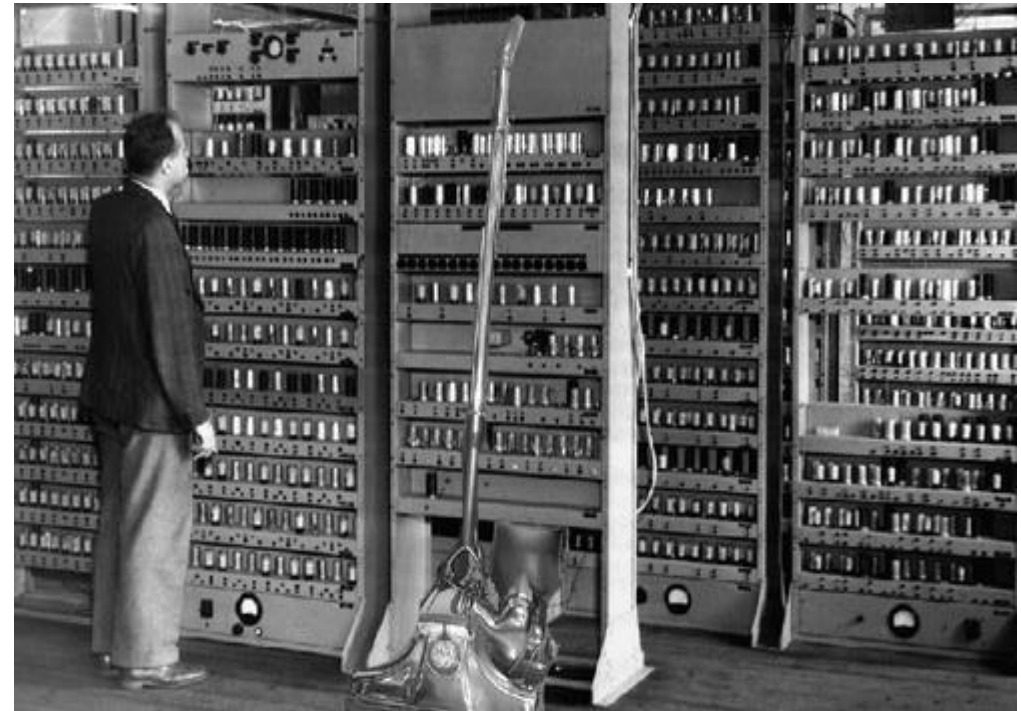
- **Electronic Numerical Integrator and Computer** was the first programmable, electronic, general-purpose **digital computer**.
- ENIAC contained 18,000 vacuum tubes; 7,200 crystal diodes; 1,500 relays; 70,000 resistors; 10,000 capacitors; and approximately 5,000,000 hand-soldered joints. It weighed more than 30 short tons.



EDSAC and EDVAC-The Electronic delay storage automatic calculator (EDSAC) , EDVAC (Electronic Discrete Variable Automatic Computer)



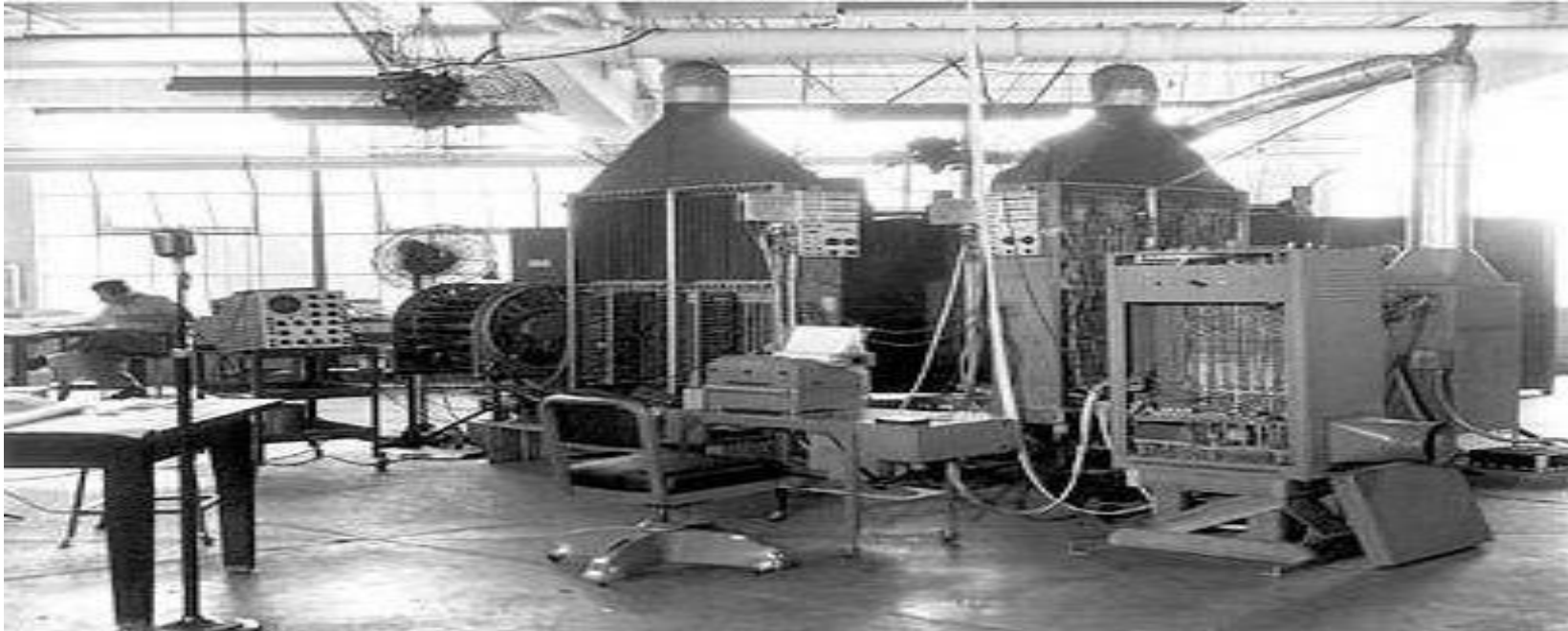
EDSAC



EDVAC

BINAC:

- The BINAC was an advanced bit-serial binary computer with two independent CPUs, each with its own 512-word acoustic mercury delay line memory. The CPUs continuously compared results to check for errors caused by hardware failures.



UNIVAC-Universal Automatic Computer

- UNIVAC I used about 5,000 vacuum tubes, weighed 16,686 pounds (8.3 short tons; 7.6 t), consumed 125 kW, and could perform about 1,905 operations per second running on a 2.25 MHz clock.

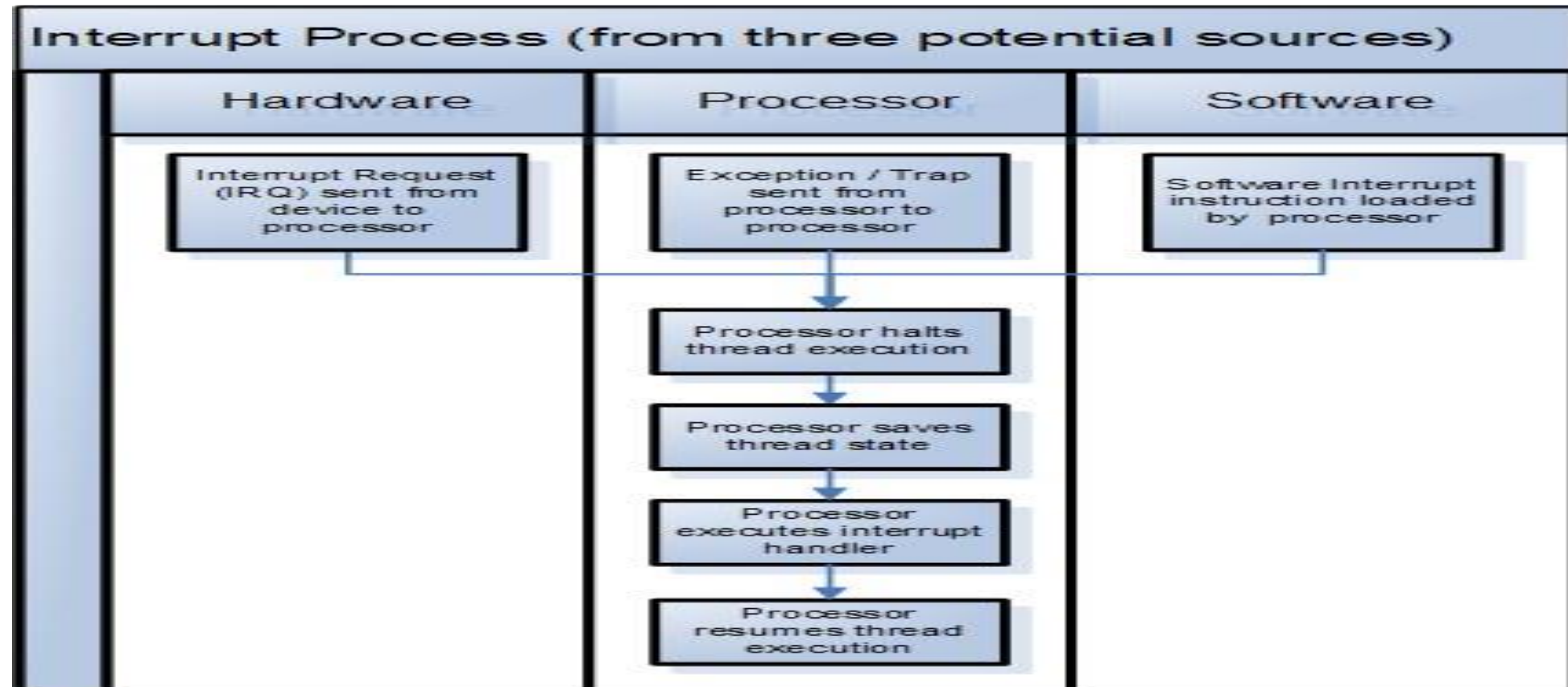


IBM 701

- The **IBM 701 Electronic Data Processing Machine**, known as the **Defense Calculator** while in development, was IBM's first commercial scientific computer and its first series production mainframe computer.
- IBM 701 - Analytical Control Unit (CPU)
- IBM 706 - Electrostatic Storage Unit (2048 words of Williams tube Memory)
- IBM 711 - Punched Card Reader (150 Cards/min.)
- IBM 716 - Printer (150 Lines/min.)
- IBM 721 - Punched Card Recorder (100 Cards/min.)
- IBM 726 - Magnetic Tape Reader/Recorder (100 Bits/inch)
- IBM 727 - Magnetic Tape Reader/Recorder (200 Bits/inch)



The interrupt



FORTRAN

Fortran is a general-purpose, compiled imperative programming language that is especially suited to numeric computation and scientific computing

```
C
    PIN=0.02
    IF (DDT.NE.0.0) THEN
    DT=DDT
    ELSE
    DT=PIN
    ENDIF
    WRITE(*, '(A)') '    PLEASE ENTER NAME OF OUTPUT FILE (FOR EXAMPLE
*   B:ZZ.DAT)'
    READ(*, '(A)') FNAMEO
    OPEN(6, FILE=FNAMEO, STATUS='UNKNOWN')
    PV=WFLX/TH
    RS=NEQ*ROU*KD/TH
    CO=CS
C
    TIME=0.0D0
    EF=0.0D0
5   CONTINUE
    GAMMA=DT/(2.0D0*DX*DX)
    BETA=DT/DX
    IF ((BETA*PV) .GT.0.5D0) GO TO 7
    IF ((GAMMA*D/(BETA*PV)) .LT.0.5D0) GO TO 6
    GO TO 8
6   DX=DX/2
    GO TO 5
7   DT=DT/2
    GO TO 5
8   CONTINUE
    N=COL/DX
    NM1=N-1
    NM2=N-2
    NP1=N+1
```

Evolution of OS

Operating systems by the late 1950s:

- By the end of the 1950s the improvised operating systems started supporting the following usages:
- It could single stream batch processing

Evolution of OS

Operating systems in the 1960s:

- 1961- The Dawn of minicomputers
- 1966- Minicomputers get cheaper
- 1967-1968 – The mouse

Supported OS Features by 1970s:

- The introduction of Multi-User and Multitasking.
- Virtual machines and dynamic address translation hardware came into limelight.
- The introduction of Personal, Interactive Systems and Modular architectures.
- 2008 – Android OS was introduced
- 2009-2022 - Multiple versions of Android were added.



Objectives of OS

- **Convenience:** An operating system allows a computer to be more convenient to use.
- **Efficiency:** An operating system supports the computer system resources to work effectively.
- **Ability to evolve:** Design an OS in a way that it allows the effective development, testing, and introduction of functions of the new system without intervening with the service.
- **Fixes:** No software is perfect. A program invariably has a few defects or bugs. Thus the need to fix those bugs from time to time.
- For example, Microsoft Windows releases various patches after fixing the bugs as per the request of the users.

Functions of OS

OS as a Resource Manager:

- An operating system is a resource manager because it preserves all the system resources like memory, processor, I/O devices etc.

Booting:

- Booting is a technique of starting the computer operating system that opens the computer to run.

Functions of OS

Memory Management:

- Regulating memory is not possible without an operating system. At a single time, various data and programs execute. If there is no operating system, the programs might blend with each other.

Loading and Execution:

- Before executing a program, the OS loads it into the memory. The Operating system provides the amenity for loading programs in memory and completing it.

Functions of OS

Data Security:

- Data is an important part of a computer system. The operating system protects the data stored on the computer from deletion, modification, or illegal use

Disk Management:

- Operating system maintains the disk space. It keeps the stored files and folders in a proper way.

Functions of OS

- **Process Management:** CPU can function one task at one time. If there are many tasks, the operating system arranges the tasks which will be done by CPU.
- **Device Controlling:** An operating system also manages all the devices of the computer. The device drivers regulate the hardware devices.
- **Printing Controlling:** Operating system also manages printing function. If a user issues two print commands at a time. It does not combine data of the files but prints them separately.

Functions of OS

- **Providing Interface:** It is used when the user interface operates with a computer mutually. User interface manages how someone can input the data or instruction and how someone can flash the information on a screen.
- **Extended Machine:** Operating System also acts like an enduring machine, which allows the users sharing of files between multiple users and provides graphical environment and various languages for communications.

Functions of OS

- **Error detection and response** :These consist of internal and external hardware errors, such as a memory error, or a device failure or malfunction.
- In each case, the OS must give a response that eliminates the error condition with less effect on running

Self-assessment Questions

1) Booting tests the system and lets the OS work. State True or False?

a) True b) False

2) Operating System provides the amenity to enhance the logical memory of the computer by adopting the physical memory of the computer system. State True or False?

a) True b) False

3) Which component creates a platform for application programs and plays as an emissary between the user and the computer hardware?

a) OS b) Virtual Machine c) Memory d) User Interface

4) Division by Zero is a hardware error. State True or False?

a) True b) False

Components of Operating System

Kernel:

- The kernel gives one of the most fundamental level of control over all the computer's hardware devices.
- The kernel is the central component of an operating system (OS). It is the component of the operating system that loads initially, and it lingers in main memory.
- It manages memory accessibility for programs in the RAM, it establishes which programs get access to which hardware resources.
- It establishes or resets the CPU's operating states for optimum operation in all times.

Components of Operating System

Process Management and Execution:

The role that the operating systems plays here are:

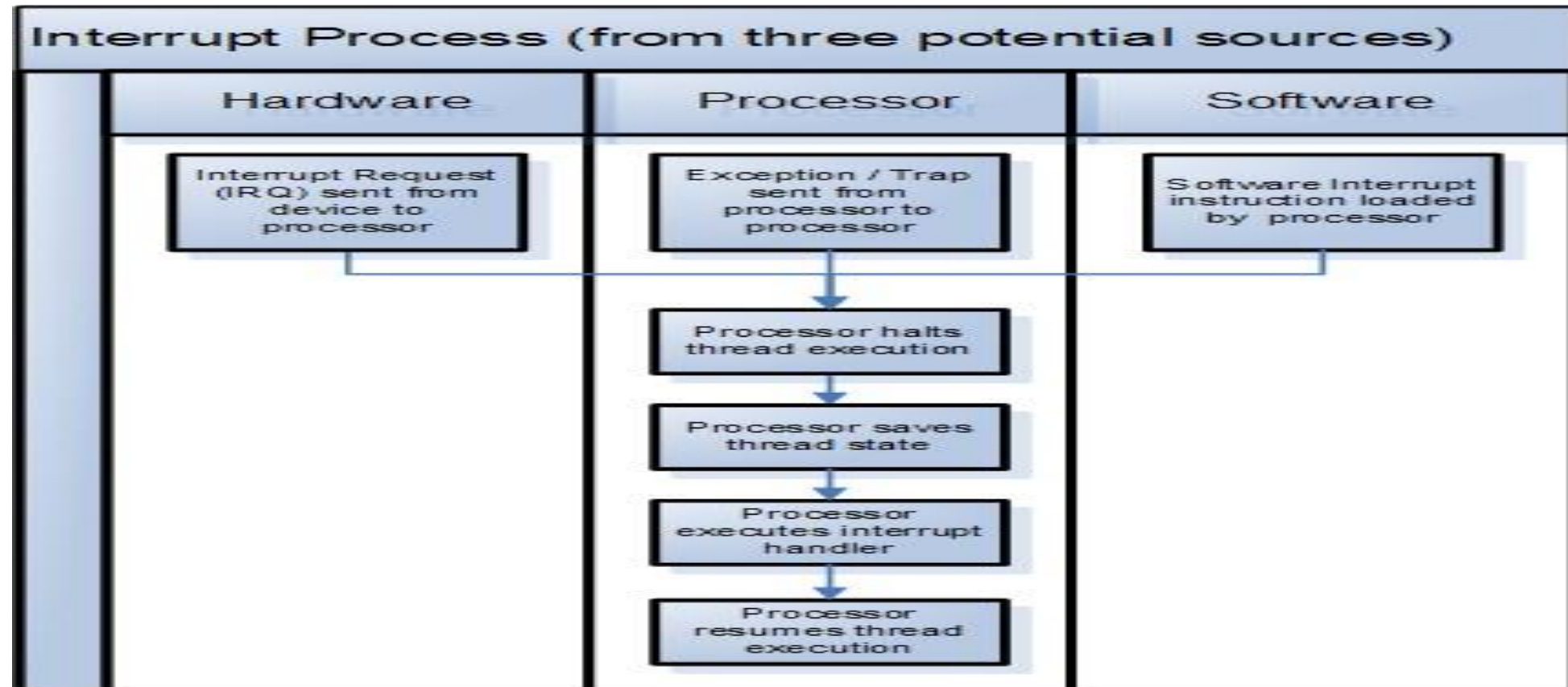
- Creation of process, keeps a track of the processor and status of the process.
- The program responsible for this task is known as the traffic controller.
- It assigns the processor (CPU) to a process.
- It deallocates the processor when a process is no longer needed.

Components of Operating System

Interrupt

- Interrupts are primary to operating systems, as they offer a reliable method for the operating system to interact with and respond to its environment.
- An interrupt is a signal from a device connected to a computer system or from a program within the computer system that needs the operating system to quit and determine exactly what to do next.
- When an interrupt is received, the computer's hardware automatically puts on hold whatever program is presently running, saves its status, and runs computer system code formerly associated with the interrupt.

The interrupt



Components of Operating System

Memory Management

- Operating System allocates and deallocates the memory for the processes. Main memory provides a fast storage that the CPU can directly access.
- It keeps a track of primary memory i.e. what part of it is in use and by whom, what part is not in use.
- In multiprogramming, the operating system decides which process will get memory, when will it get, and how much will it get.
- It allocates the memory when a process requests it to do so.
- It deallocates the memory when a process no longer needs it or has been terminated.

Components of Operating System

Files and Disk Management

- All the operating systems incorporate support for numerous file systems. The modern file systems include a hierarchy of directories.

Networks

- Most of the present operating systems are proficient enough in using the TCP/IP networking protocols, which means that one system can appear on another network and share the resources such as files, scanners, printers, etc. through wireless or wired connections

Components of Operating System

Multitasking:

- Multitasking describes the operating of multiple independent computer programs on the same computer system.
- The operating system has the ability to keep an eye on where you are in these jobs and go from one to the various other without losing information.
- Since a lot of computers can do at most one or two things at once, this is usually done using time-sharing, which means that each program utilizes a share of the computer's time to perform.

Components of Operating System

Security :

- The operating system grants access to few resources, directly or indirectly, such as privileged system calls, files on a local disk, personal information about users, and the services that are offered by the programs operating on the system.

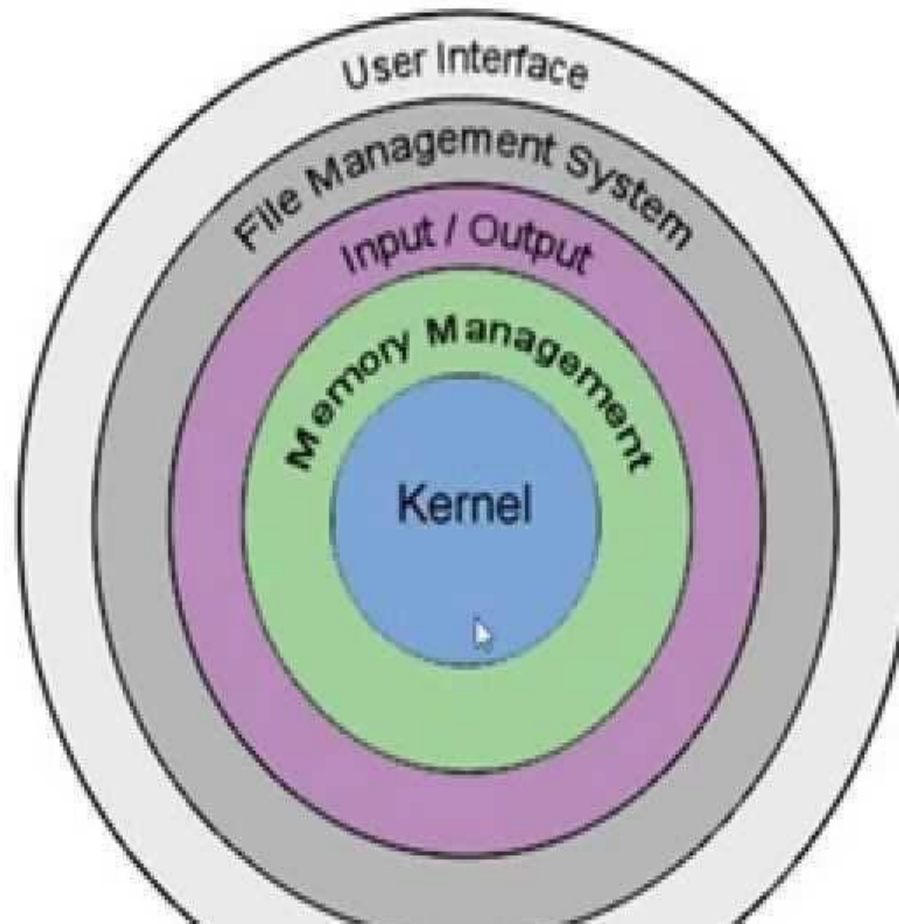
User Interface

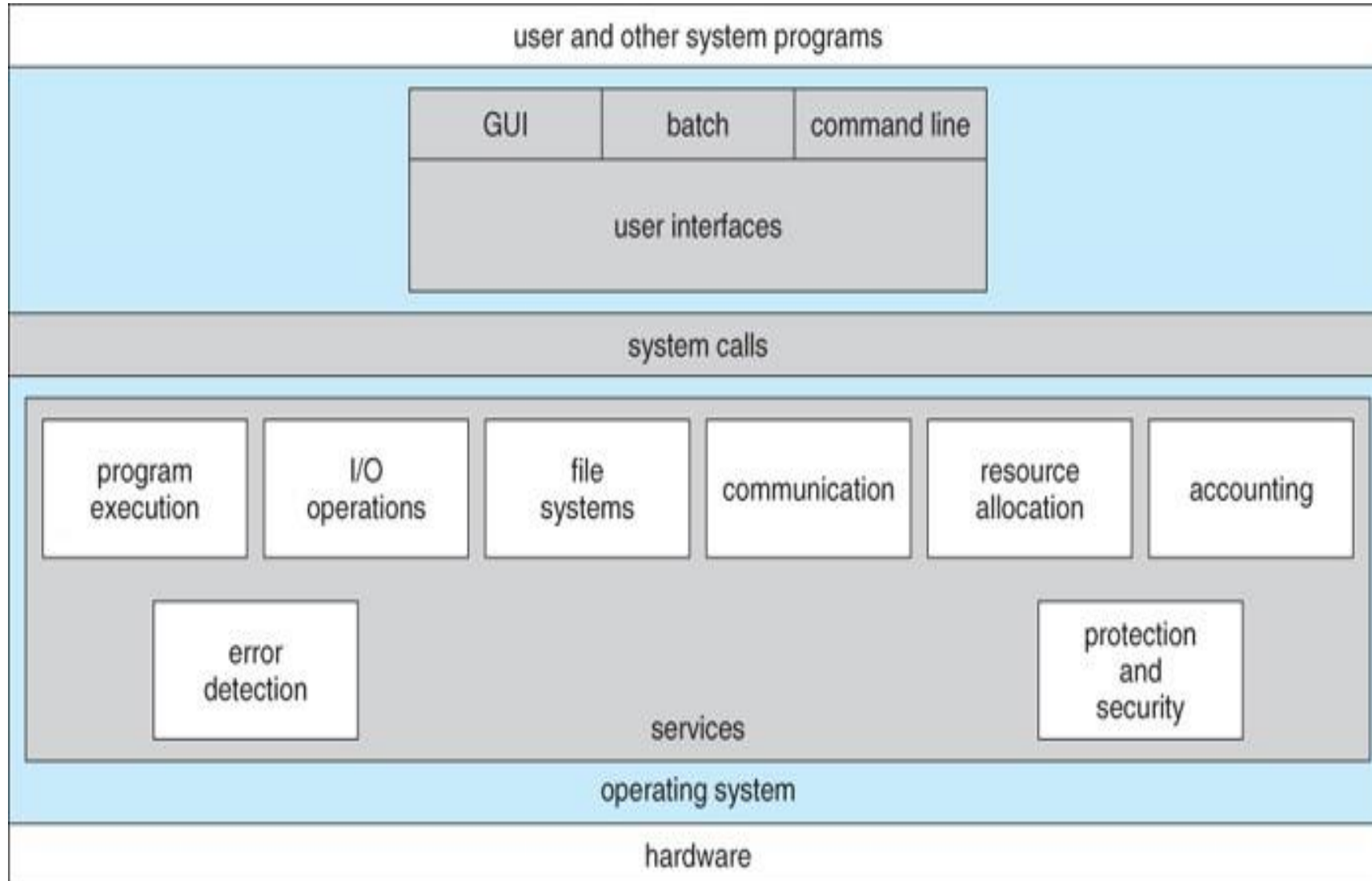


Self-assessment Questions

- 5) The role of the operating system is
- a) Holds the process
 - b) Starts the process
 - c) Share the process
 - d) None of the above
- 6) Memory management is used to
- a) Allocate memory
 - b) De-allocates memory
 - c) Both a & b
 - d) None of the above

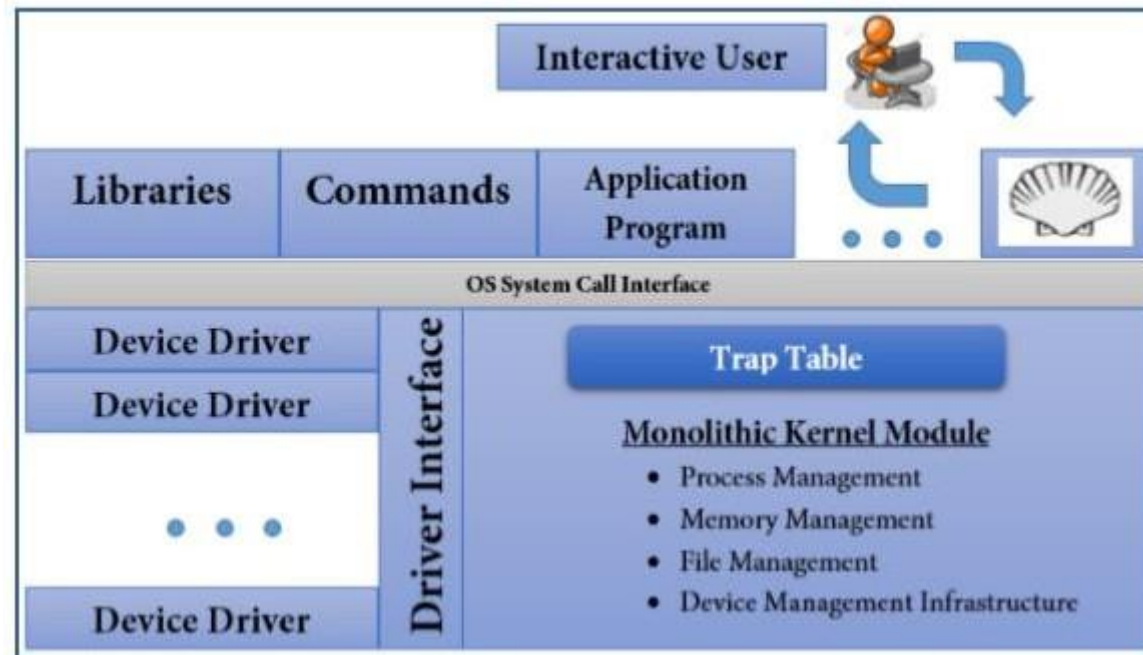
Layered Structure





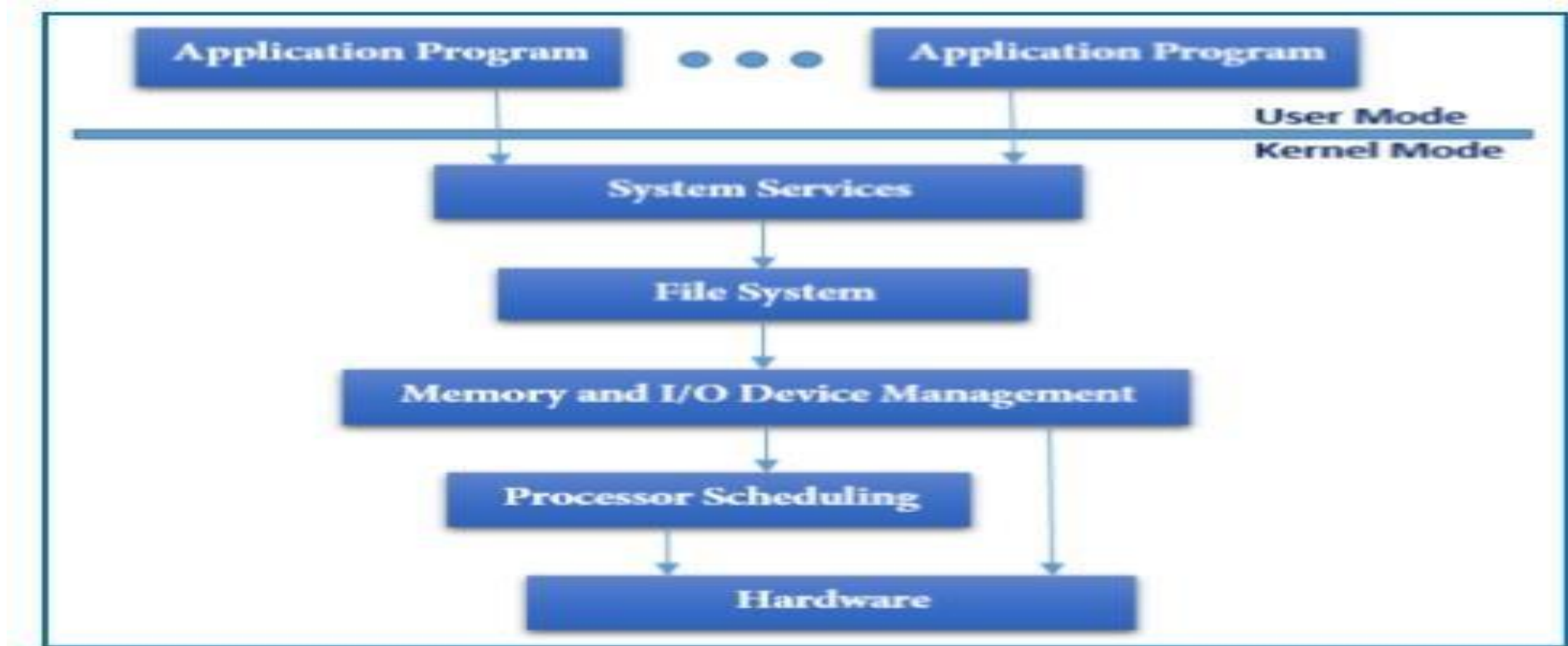
Types of Operating System Structure

- Monolithic Approach



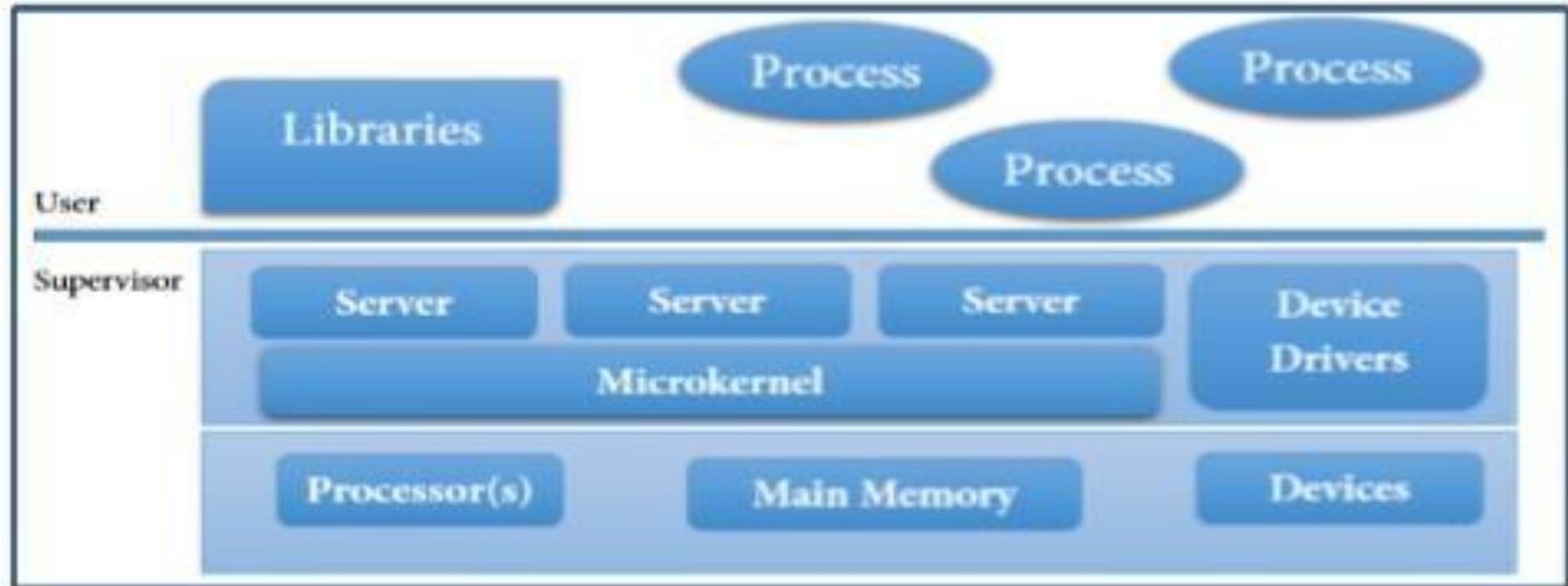
Types of Operating System Structure

- Layered Approach



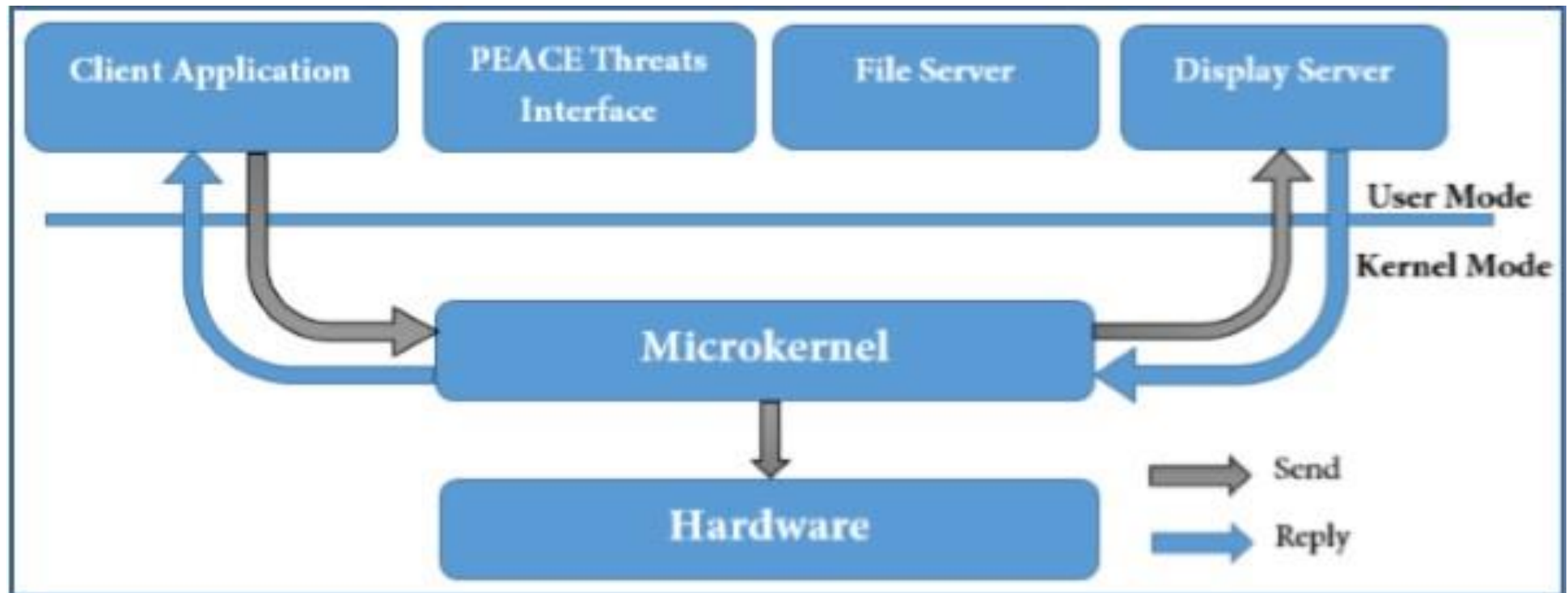
Types of Operating System Structure

- Microkernels



Types of Operating System Structure

- Client-Server Model

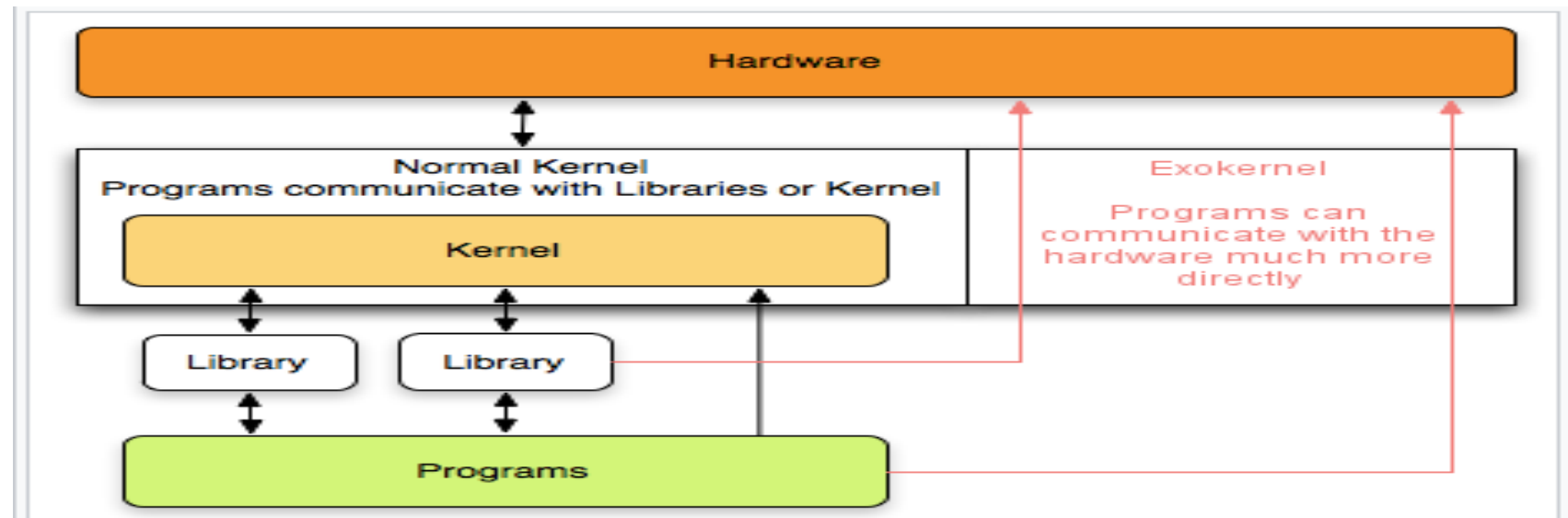


Types of Operating System Structure

- Virtual Machines

A virtual machine is well known as software computers, which like a physical computer, operates an operating system and various applications in a system

- Exokernels





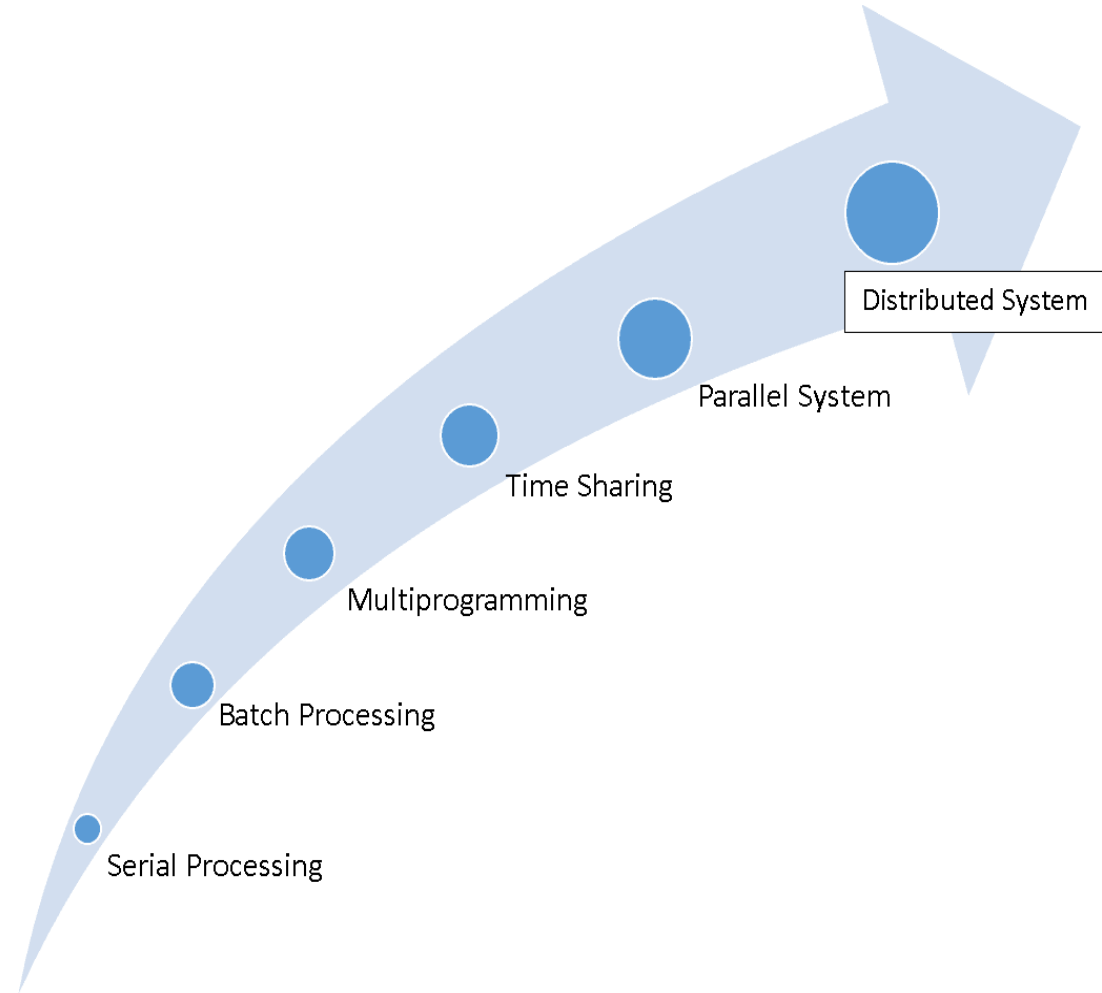
Self-assessment Questions

- 9) If various jobs are ready to be brought into the memory and if there is no space to accommodate all of them, then the system needs to choose among them. This decision-making is known as _____
- a) Job Scheduling
 - b.) CPU Scheduling
 - c) System Process
 - d.) Job Pool
- 10) For any program to execute, it should be in the main memory. State True or False?
- a.) True
 - b.) False
- 11) Which of the following is a piece of software or even code that contains minimum functions and aspects needed to run an operating system?
- a.) Virtual Machine
 - b.) Microkernel
 - c.) Client-Server
 - d.) Exokernel



JAIN
DEEMED-TO-BE UNIVERSITY

SCHOOL OF
COMPUTER
SCIENCE AND IT



1. Serial Processing

History of the operating system started in 1950. Before 1950, the programmers directly interact with the hardware there was no operating system at that time. If a programmer wishes to execute a program on those days, the serial steps are necessary (next slide).

Serial Processing

- Type the program or punched card.
- Convert the punched card to a card reader.
- Submit to the computing machine, if there are any errors, the error was indicated by the lights.
- The programmer examined the register and main memory to identify the cause of an error.
- Take outputs on the printers.
- Then the programmer is ready for the next program.

Drawback:

This type of processing is difficult for users, it takes much time and the next program should wait for the completion of the previous one. The programs are submitted to the machine one after one, therefore the method is said to be **serial processing**.

2. Batch operating system

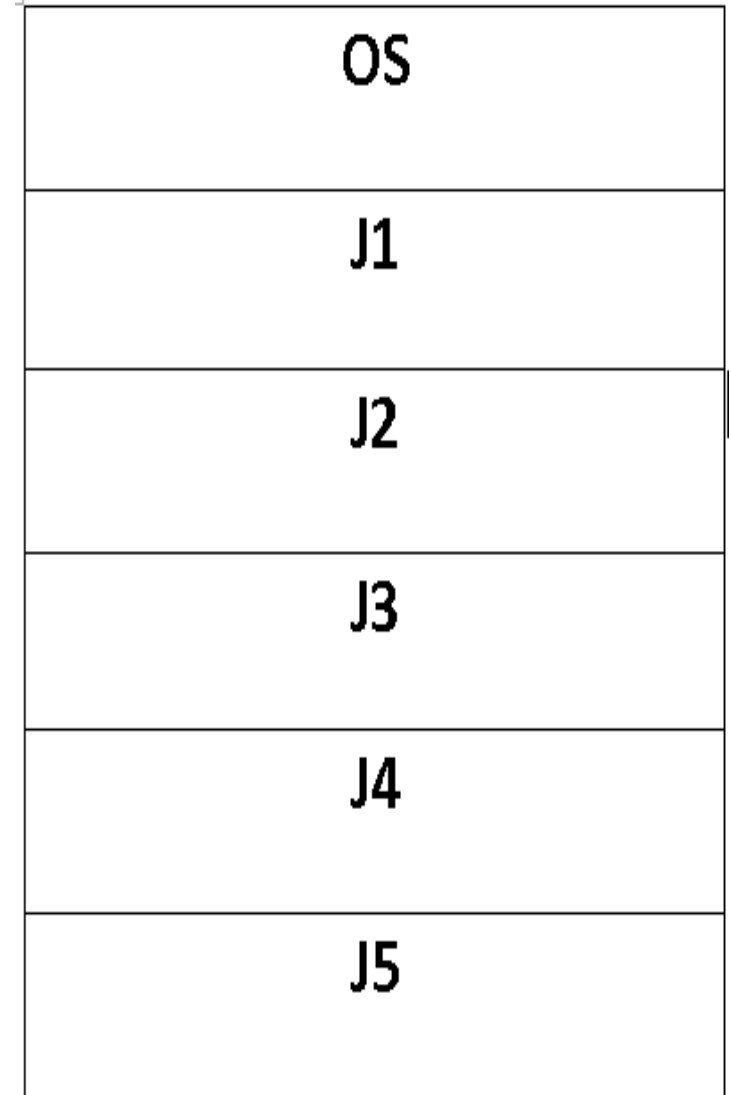
The users of a batch operating system do not interact with the computer directly. Each user prepares his job on an off-line device like punch cards and submits it to the computer operator. To speed up processing, jobs with similar needs are batched together and run as a group. The programmers leave their programs with the operator and the operator then sorts the programs with similar requirements into batches.

The problems with Batch Systems are as follows –

- Lack of interaction between the user and the job.
- CPU is often idle, because the speed of the mechanical I/O devices is slower than the CPU.
- Difficult to provide the desired priority.

3. Multiprogramming

Multiprogramming is a technique to execute the number of programs simultaneously by a single processor. In multiprogramming, a number of processes reside in main memory at a time. The OS(Operating System) picks and begins to execute one of the jobs in main memory. Consider the following figure, it depicts the layout of the multiprogramming system. The main memory consisting of 5 jobs at a time, the CPU executes one by one.



Multiprogramming

In the non-multiprogramming system, the CPU can execute only one program at a time, if the running program is waiting for any I/O device, the CPU becomes idle so it will effect on the performance of the CPU.

But in a multiprogramming environment, if any I/O wait happened in a process, then the CPU switches from that job to another job in the job pool. So, the CPU is not idle at any time.

Advantages:

- Can get efficient memory utilization.
- CPU is never idle so the performance of CPU will increase.
- The throughput of CPU may also increase.
- In the non-multiprogramming environment, the user/program has to wait for CPU much time. But waiting time is limited in multiprogramming.

Multiprogramming



Multiprogramming Operating System

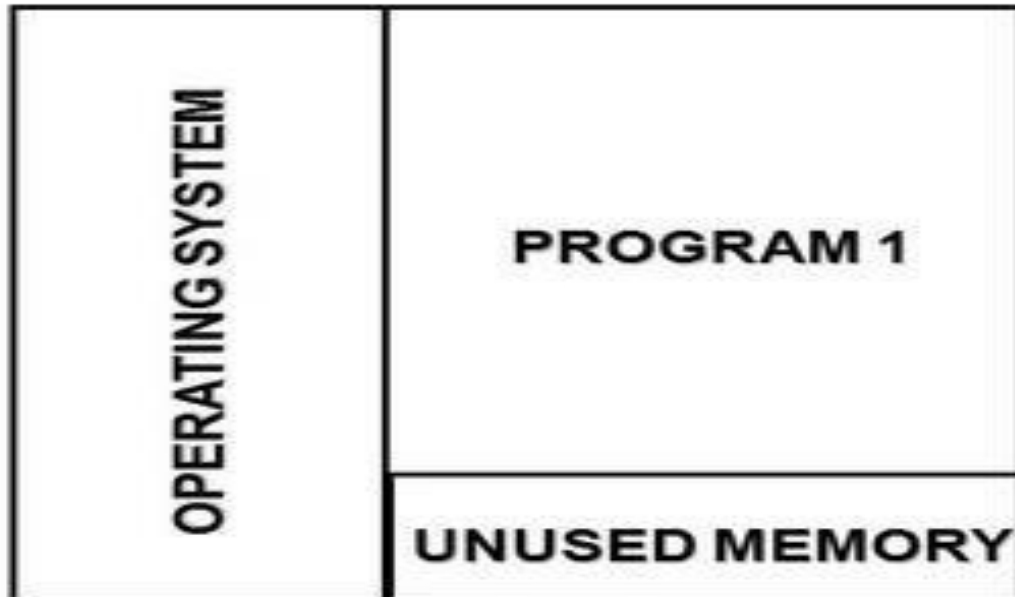
Multiprogramming



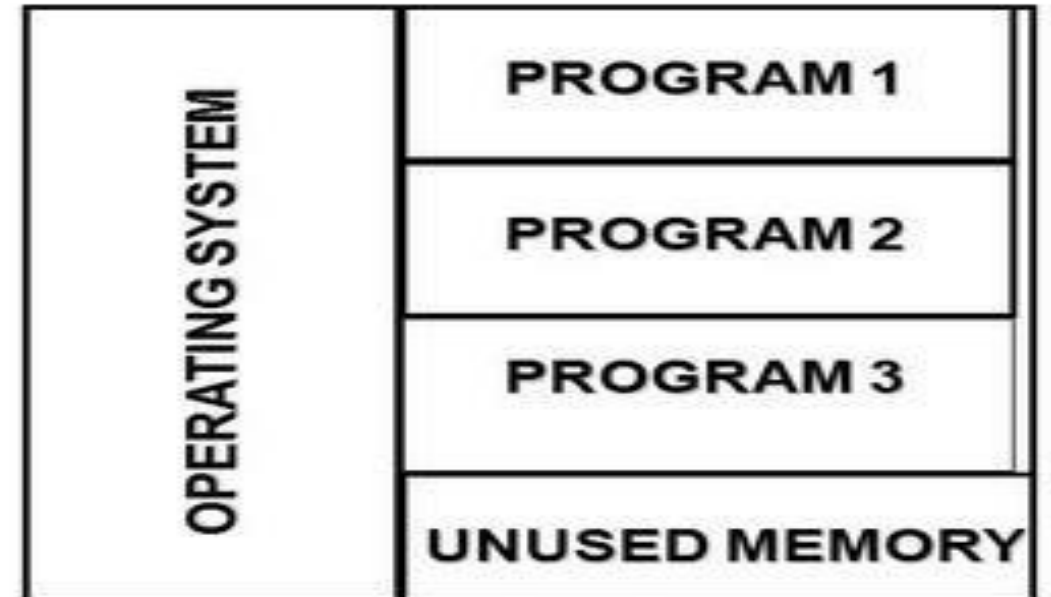
Single Programming Vs Multiprogramming

MULTIPROGRAMMING

TRADITIONAL SINGLE-PROGRAM SYSTEM



MULTIPROGRAMMING ENVIRONMENT





BATCH PROCESSING VERSUS MULTIPROGRAMMING

BATCH PROCESSING

Grouping of several processing jobs to be executed one after another by a computer without any user interaction

Batch processing is slower

CPU might stay without performing a task

Helps to minimize human interactions and cost

MULTIPROGRAMMING

Ability of an OS to execute multiple programs at the same time on a single processor

Multiprogramming is faster

CPU always perform a task

Provides maximum CPU utilization

Difference B/W

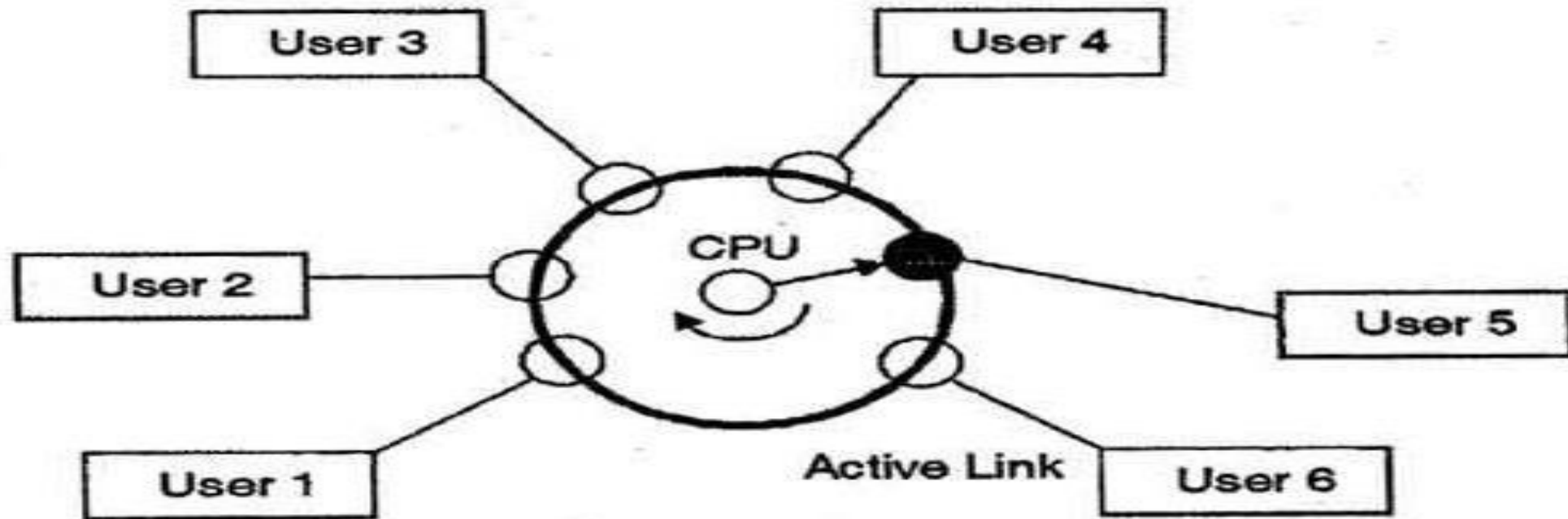
Multiprogramming & Multiprocessing

Multiprogramming	Multiprocessing
1) Multiprogramming concept is implemented in systems with a single processor.	1) Multiprocessing concept is implemented in systems with a multiple processors.
2) In such a system, a portion of one program is executed first, then second and so on.	2) In multiprocessing, several program of one or more program are executed at a same time.

4. Time-sharing operating systems

Time-sharing or **multitasking** is a logical extension of multiprogramming. Multiple jobs are executed by the CPU switching between them. The CPU scheduler selects a job from the ready queue and switches the CPU to that job. When the time slot expires, the CPU switches from this job to another. In this method, the CPU time is shared by different processes. So, it is said to be "**Time-Sharing System**". Generally, time slots are defined by the operating system.

Time Sharing Systems



Time-sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time.

Advantages:

The main **advantage of the time-sharing system** is efficient CPU utilization. It was developed to provide interactive use of a computer system at a reasonable cost. A time shared OS uses CPU scheduling and multiprogramming to provide each user with a small portion of a time-shared computer.

Another advantage of the time-sharing system over the batch processing system is, the user can interact with the job when it is executing, but it is not possible in batch 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.

5. Parallel System

There is a trend multiprocessor system, such system have more than one processor in close communication, sharing the computer bus, the clock, and sometimes memory and peripheral devices. These systems are referred to as "Tightly Coupled" system. Then the system is called **a parallel system**. In the parallel system, a number of processors are executing there job in parallel.

Advantages:

- It increases the throughput.
- By increasing the number of processors(CPU), to get more work done in a shorter period of time.

6. Distributed System

In a **distributed operating system**, the processors cannot share a memory or a clock, each processor has its own local memory. The processor communicates with one another through various communication lines, such as high-speed buses. These systems are referred to as "Loosely Coupled" systems.

Advantages:

- If a number of sites connected by high-speed communication lines, it is possible to share the resources from one site to another site, for example, s1 and s2 are two sites. These are connected by some communication lines. The site s1 having a printer, but the site does not have any print. Then the system can be altered without moving from s2 to s1. Therefore, resource sharing is possible in the distributed operating system.
- A big computer that is partitioned into a number of partitions, these sub-partitions are run concurrently in distributed systems.
- If a resource or a system fails in one site due to technical problems, we can use other systems/resources in some other sites. So, the reliability will increase in the distributed system.

7. Network operating System

A Network Operating System runs on a server and provides the server the capability to manage data, users, groups, security, applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), a private network or to other networks.

Examples of network operating systems include Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD.

The advantages of network operating systems are as follows –

- Centralized servers are highly stable.
- Security is server managed.
- Upgrades to new technologies and hardware can be easily integrated into the system.
- Remote access to servers is possible from different locations and types of systems.

The disadvantages of network operating systems are as follows –

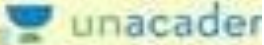
- High cost of buying and running a server.
- Dependency on a central location for most operations.
- Regular maintenance and updates are required.

Multitasking

In computing, multitasking is the concurrent execution of multiple tasks over a certain period of time. New tasks can interrupt already started ones before they finish, instead of waiting for them to end



Multiprogramming Vs Multitasking

✕ Multiprogramming	Multitasking 
Multiprogramming means more than one process in main memory which are ready to execute	Multitasking is when more than one task is executed at a single time utilizing multiple CPUs.
It is based on the concept of context switching.	It is based on the concept of time sharing.
It takes more time to execute the process.	It takes less time to execute the task or process.
The idea is to reduce the CPU idle time for as long as possible.	The idea is allow multiple processes to run simultaneously via time sharing.

Multitasking Vs Time Sharing

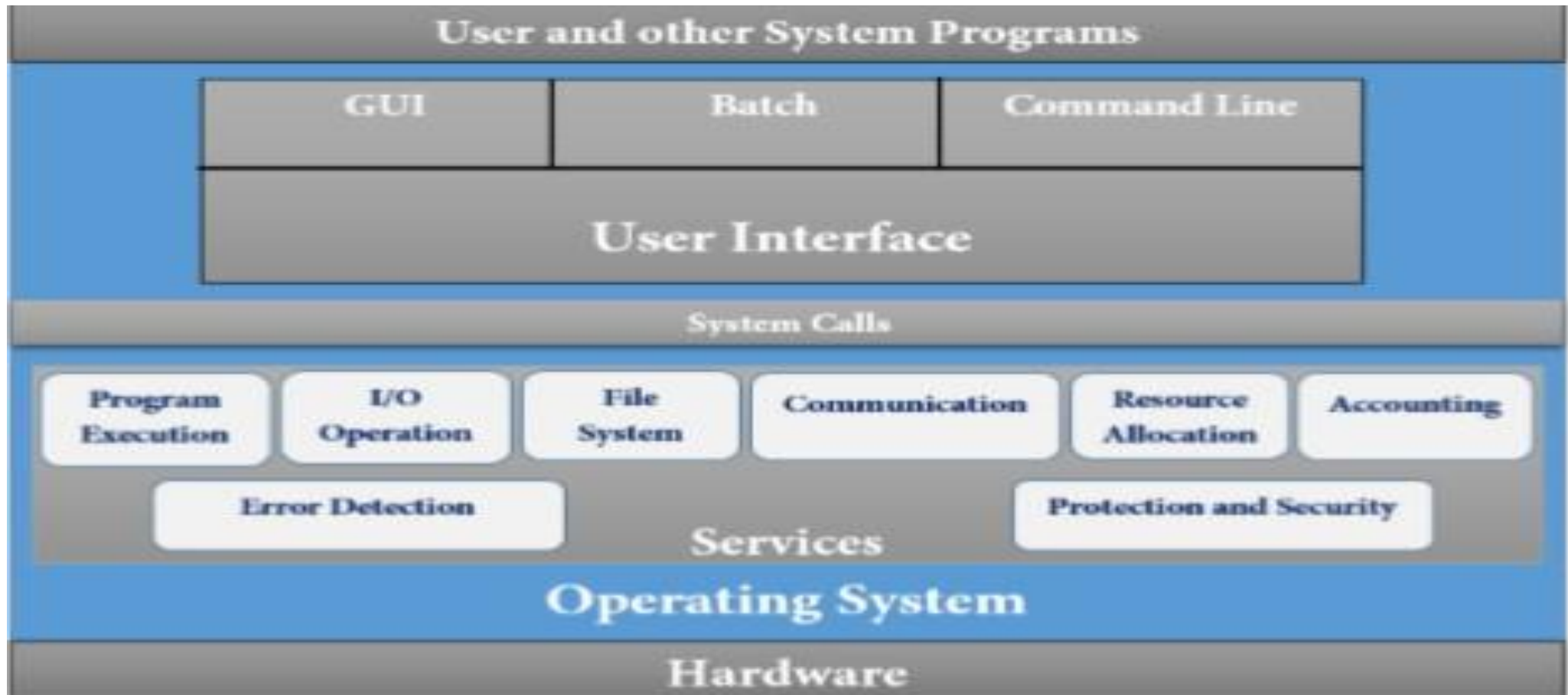
- Multitasking is the method where **multiple tasks(processes)** are performed during **same period of time** using **multiple CPUs**.
- Time Sharing is described as the sharing of computing resource among many users by means of multiprogramming and multi tasking.



Self-assessment Questions

- 7) The CPU or the I/O devices can be kept busy by a single user all the time in
- a.) Multiprogramming
 - b.) Single programming
 - c.) Multiprocessing
 - d.) Single processing
- 8) An extension of multiprogramming is time-sharing
- a.) Resource sharing
 - b.) Task sharing
 - c.) Time sharing
 - d.) None of the above

Operating System Services



Operating System Services

User-Interface: All the operating systems have a user interface, which can takes various forms

- **Command Line Interface (CLI)** is one of the forms of user interface. This form uses the text commands and various processes.
- **Batch Interface** is another form of user interface. This form makes use of the commands and the directives to manage those commands that are put into the files, and executes those commands.

Operating System Services

- **Program Execution:**

The purpose of the computer system is to allow the users to execute a program in an efficient manner.

Following are the important activities of an operating system concerning program management:

- Installs a program into the memory
- Carries the program
- Manages the execution of the program
- It proposes a mechanism for the process synchronization
- It provides a technique for process communication
- It provides a technique to handle the deadlock

Operating System Services

I/O Operation: Every program needs input, and after processing the input that is given by the user, it generates output.

- An Operating System handles the communication between a user and the device drivers.
- I/O operation intends to read or write an operation with any specific I/O device or any other file.
- Operating system gives the access to the I/O device when it is required

Operating System Services

File System Manipulation:

Following are the important activities of an operating system related to file management:

- The program needs to read a file or write a file
- The operating system allows certain operations on the file
- Permission differs from read-only, read-write, denied and so on
- An Operating System provides an interface to the user to create/delete files
- An Operating System provides an interface to the user to create/delete directories
- An Operating System provides an interface to create a backup of the file system

Operating System Services

Error Detection:

- User programs cannot manage the error detection and error correction because it involves observing the entire computing process.
- A user program can associate with the corresponding operation of the operating system when given authorization.

The important activities of an operating system concerning error handling are:

- The OS continuously explores for possible errors.
- The operating system chooses a right action for accurate and consistent computing.

Operating System Services

Resource Allocation:

- During multi-tasking, an operating system plays a vital role in the allocation of the required resources to each and every process for its better usage.
- The resources are CPU, main memory, tape drive or secondary storage, etc.
- The important activities of an operating system concerning resource management are:
 - The OS maintains various resources using schedulers.
 - For the better usage of CPU, it uses the algorithms called CPU scheduling algorithm.

Operating System Services

Accounting: In a multitasking process, accounting increases the efficiency of the system with the distribution of resources to each process.

Operating System Services

- **Protection System :**

The prominent activities of an operating system concerning protection are:

- The OS assures to control all approaches to the system resources.
- The OS assures that the external I/O devices are kept secure from invalid access attempts.
- The OS grants authentication features for each user, using passwords.

Self Assessment Questions

- 1) The OS assures that the external I/O devices are kept secure from invalid access attempts. State True or False?
 - a) True
 - b) False
- 2) In a multitasking process, accounting decreases the efficiency of the system with the distribution of resources to each process. State True or False?
 - a) True
 - b) False
- 3) What is the full form of CLI?
 - a) Command Line Interface
 - b) Common Line Interface
 - c) Common Language Interface
 - d) Command Language Interface
- 4) In which form, does the Operating system interact with various types of processes?
 - a) Virtual Memory
 - b) GUI
 - c) Shared Memory
 - d) Secondary Memory

Self Assessment Questions

- 5) Where is the program loaded to make it run?
- a) CPU
 - b) RAM
 - c) Niche Memory
 - d) ROM



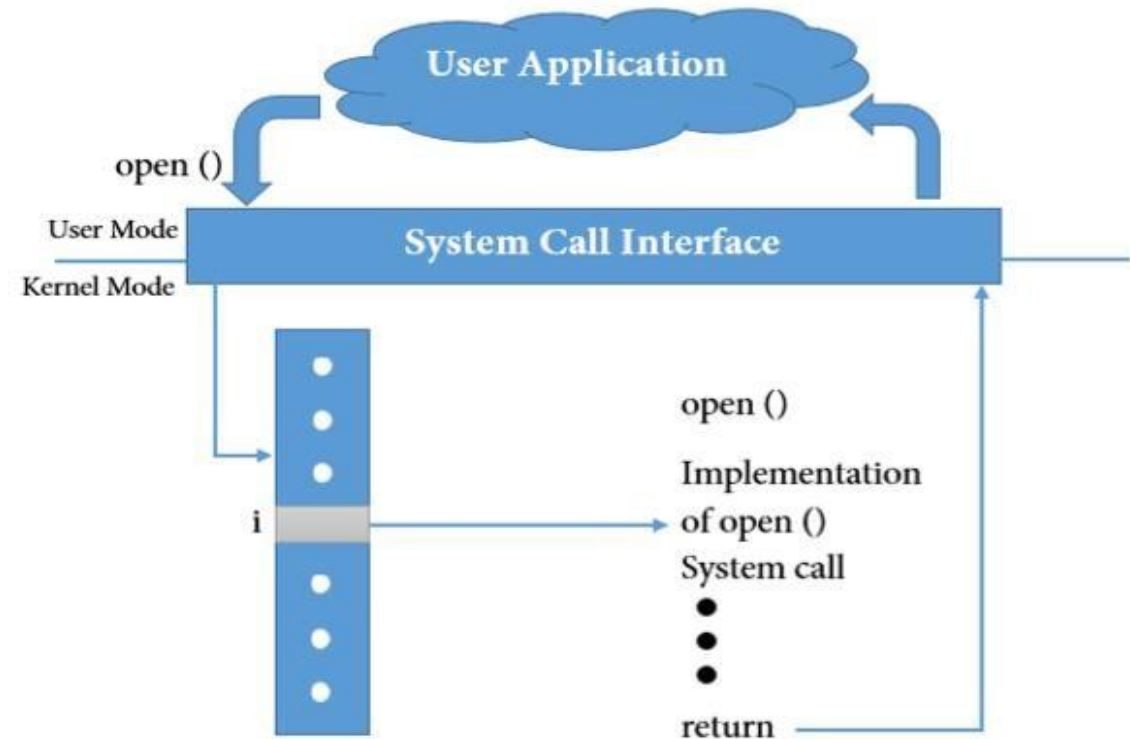
```
[root@linux-server root]#  
[root@linux-server root]# objdump -D a.out | grep -A20 main.:  
08048460 <main>:  
  8048460:      55                push    %ebp  
  8048461:      89 e5            mov     %esp,%ebp  
  8048463:      83 ec 08         sub     $0x8,%esp  
  8048466:      90                nop  
  8048467:      c7 45 fc 00 00 00 00 movl    $0x0,0xffffffffc(%ebp)  
  804846e:      89 f6            mov     %esi,%esi  
  8048470:      83 7d fc 09      cmpl    $0x9,0xffffffffc(%ebp)  
  8048474:      7e 02            jle     8048478 <main+0x18>  
  8048476:      eb 18            jmp     8048490 <main+0x30>  
  8048478:      83 ec 0c         sub     $0xc,%esp  
  804847b:      68 08 85 04 08   push    $0x8048508  
  8048480:      e8 93 fe ff ff   call    8048318 <_init+0x38>  
  8048485:      83 c4 10         add     $0x10,%esp  
  8048488:      8d 45 fc         lea     0xffffffffc(%ebp),%eax  
  804848b:      ff 00            incl    (%eax)  
  804848d:      eb e1            jmp     8048470 <main+0x10>  
  804848f:      90                nop  
  8048490:      b8 00 00 00 00   mov     $0x0,%eax  
  8048495:      c9                leave  
  8048496:      c3                ret  
[root@linux-server root]# _
```

System Calls

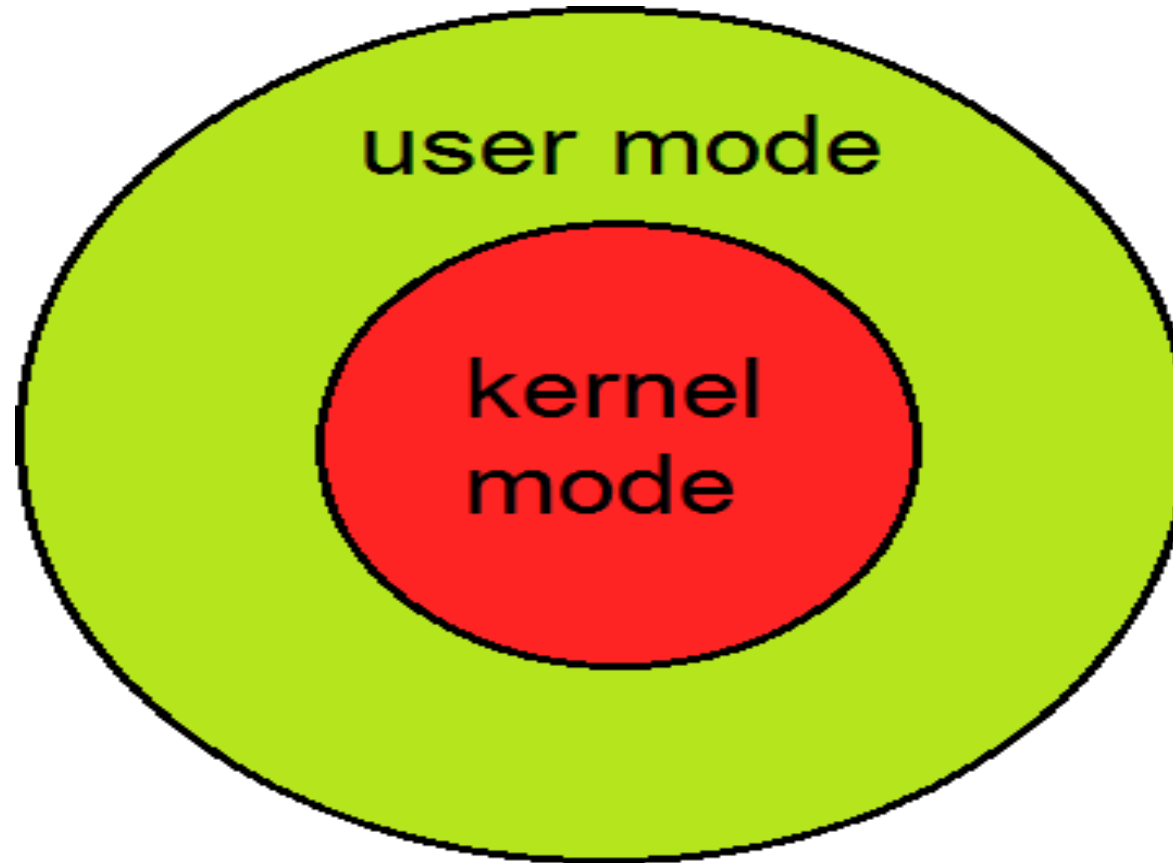
- To perform an operation, the user needs to get a request for a service from the System. Before any request, the user will make a special call which is also known as the System Call.
- A System Call is a process in which a computer program asks for a service from the kernel of the operating system on which the execution takes place.

System Calls

- When a user opens the system for the first time, then the system is in the **user mode**.
- When the user requests for a service, then the user mode turns to the **Kernel Mode**, which just listens to the request of the user, processes that request, and displays the results.



OPERATING SYSTEM MODES



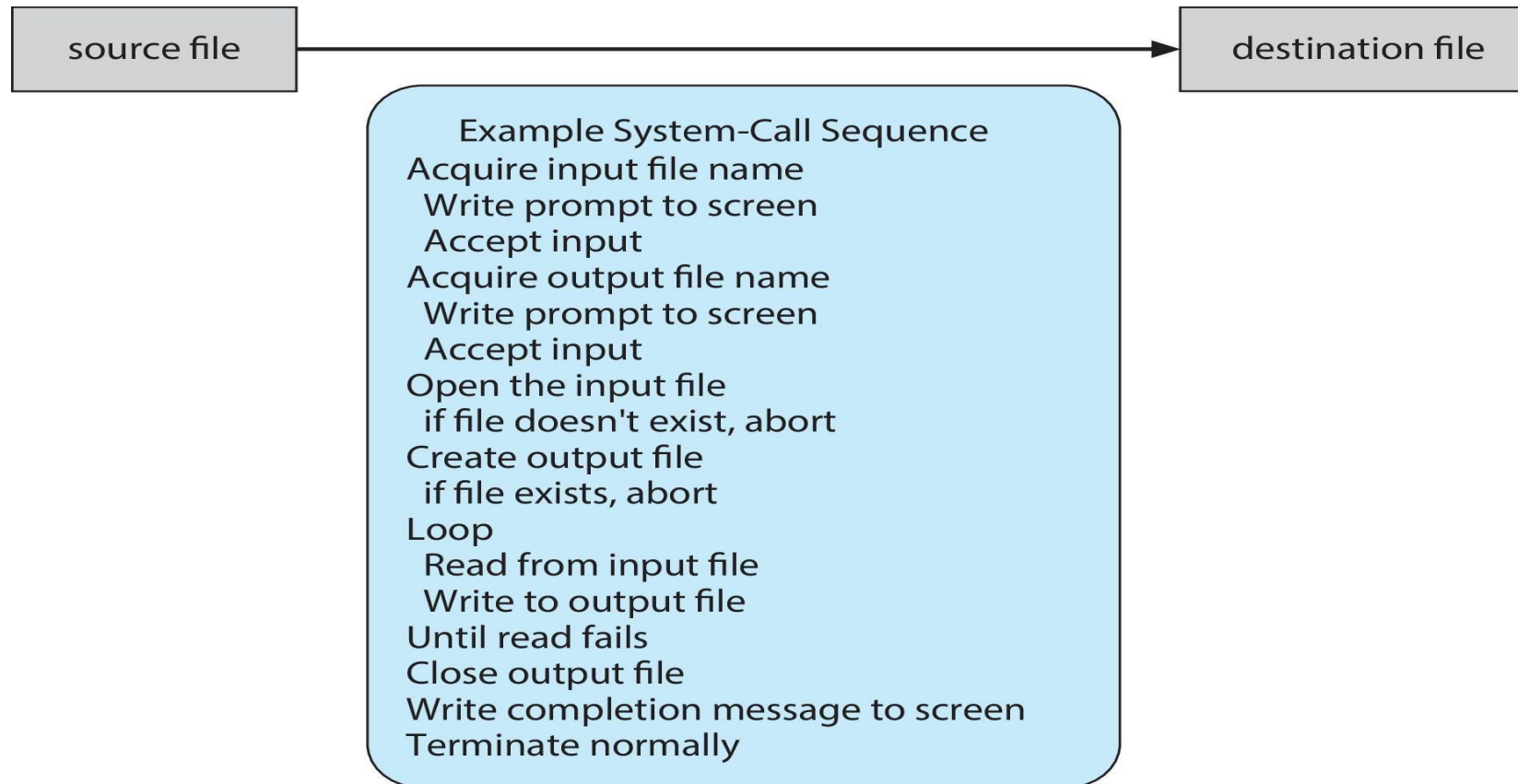
Kernel Mode

- When CPU is in **kernel mode**, the code being executed can access any memory address and any hardware resource.
- Hence kernel mode is a very privileged and powerful mode.
- If a program crashes in kernel mode, the entire system will be halted.

User Mode

- When CPU is in **user mode**, the programs don't have direct access to memory and hardware resources.
- In user mode, if any program crashes, only that particular program is halted.
- That means the system will be in a safe state even if a program in user mode crashes.
- Hence, most programs in an OS run in user mode.

Example

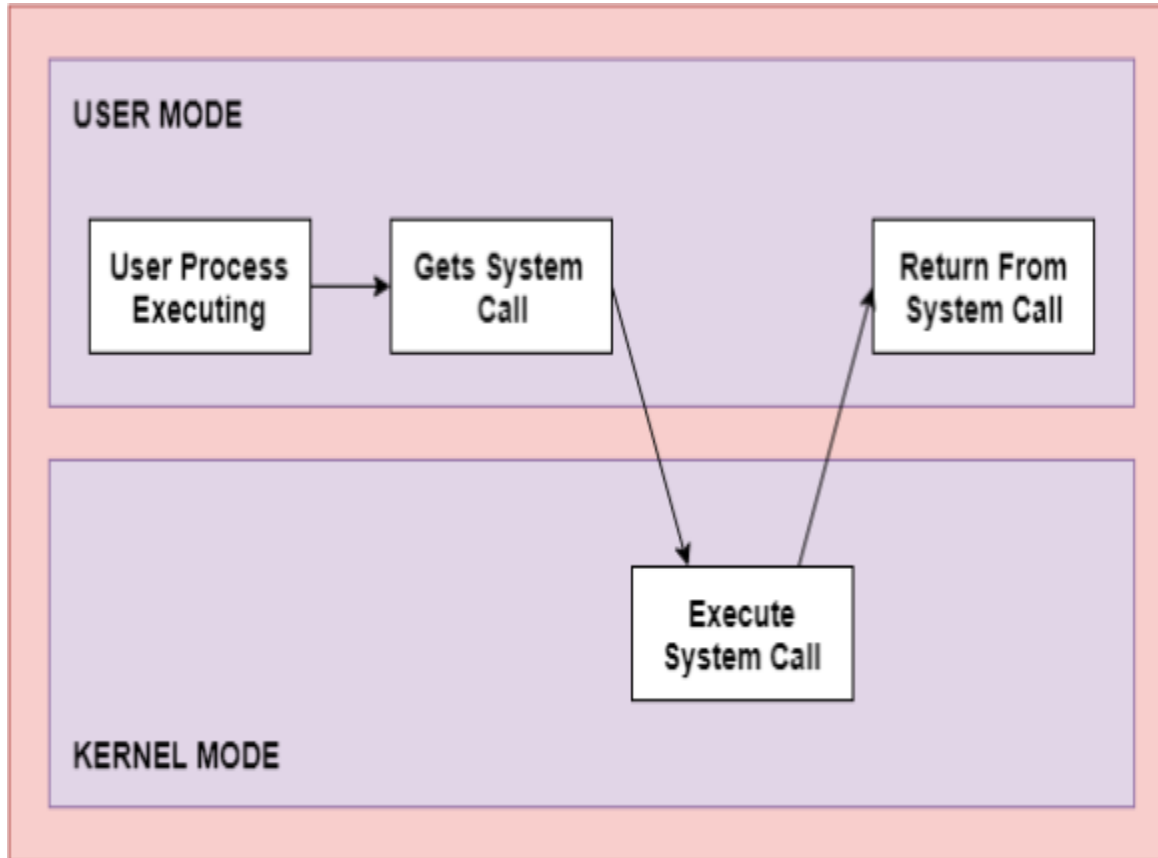


HOW DOES A SYSTEM CALL WORK?

- When a program in user mode requires access to RAM or a hardware resource, it must ask the kernel to provide access to that resource. This is done via something called a **system call**.
- When a program makes a system call, the mode is switched from user mode to kernel mode. This is called a **context switch**.
- Then the kernel provides the resource which the program requested. After that, another context switch happens which results in change of mode from kernel mode back to user mode.

Generally, system calls are made by the user level programs in the following situations:

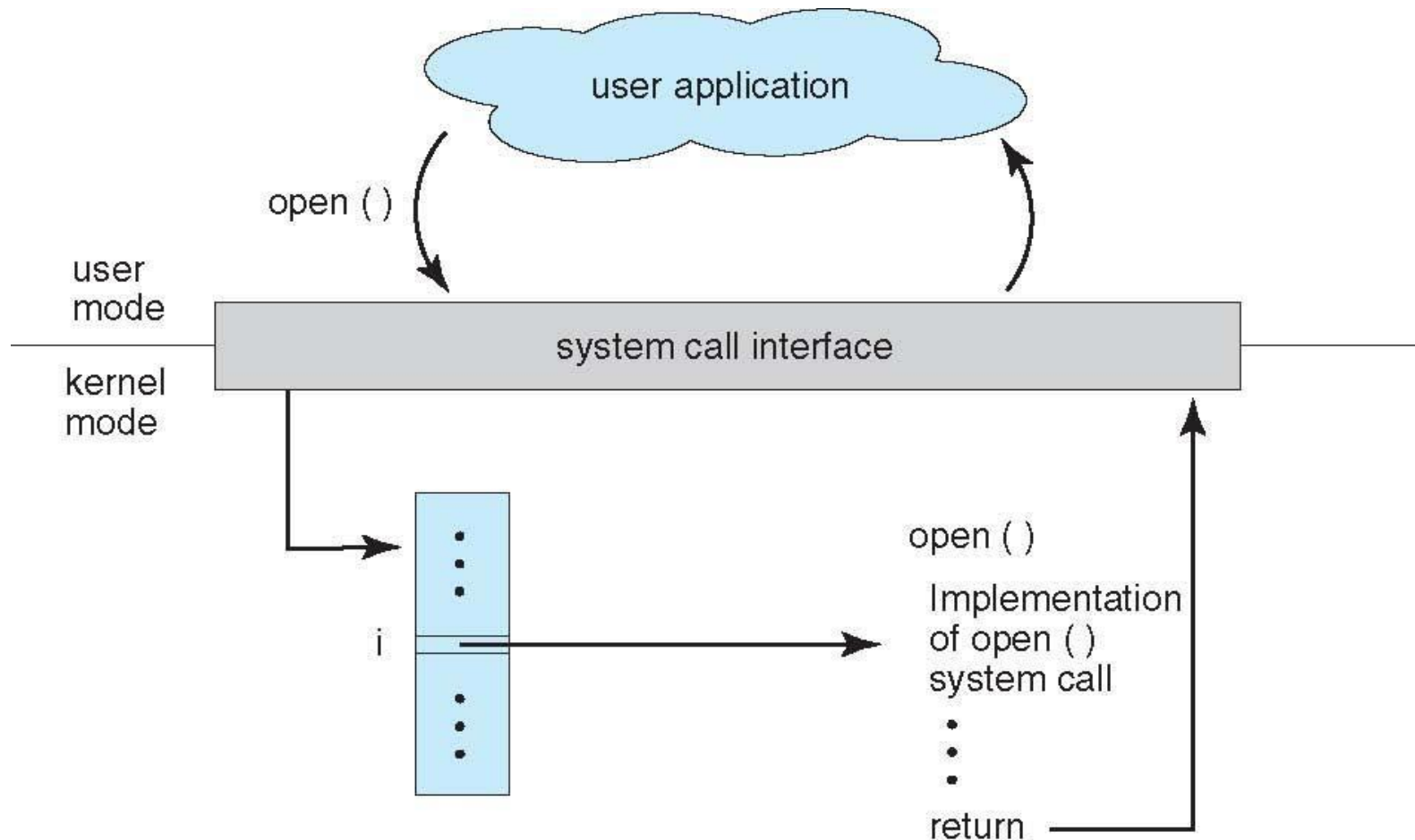
- Creating, opening, closing and deleting files in the file system.
- Creating and managing new processes.
- Creating a connection in the network, sending and receiving packets.
- Requesting access to a hardware device, like a mouse or a printer.



SYSTEM CALLS

- Programming interface to the services provided by the OS
- Typically written in a high-level language (C or C++)
- Mostly accessed by programs via a high-level **Application Program Interface (API)** rather than direct system call use
- Three most common APIs are Win32 API for Windows, POSIX API for POSIX-based systems (including virtually all versions of UNIX, Linux, and Mac OS X), and Java API for the Java virtual machine (JVM)

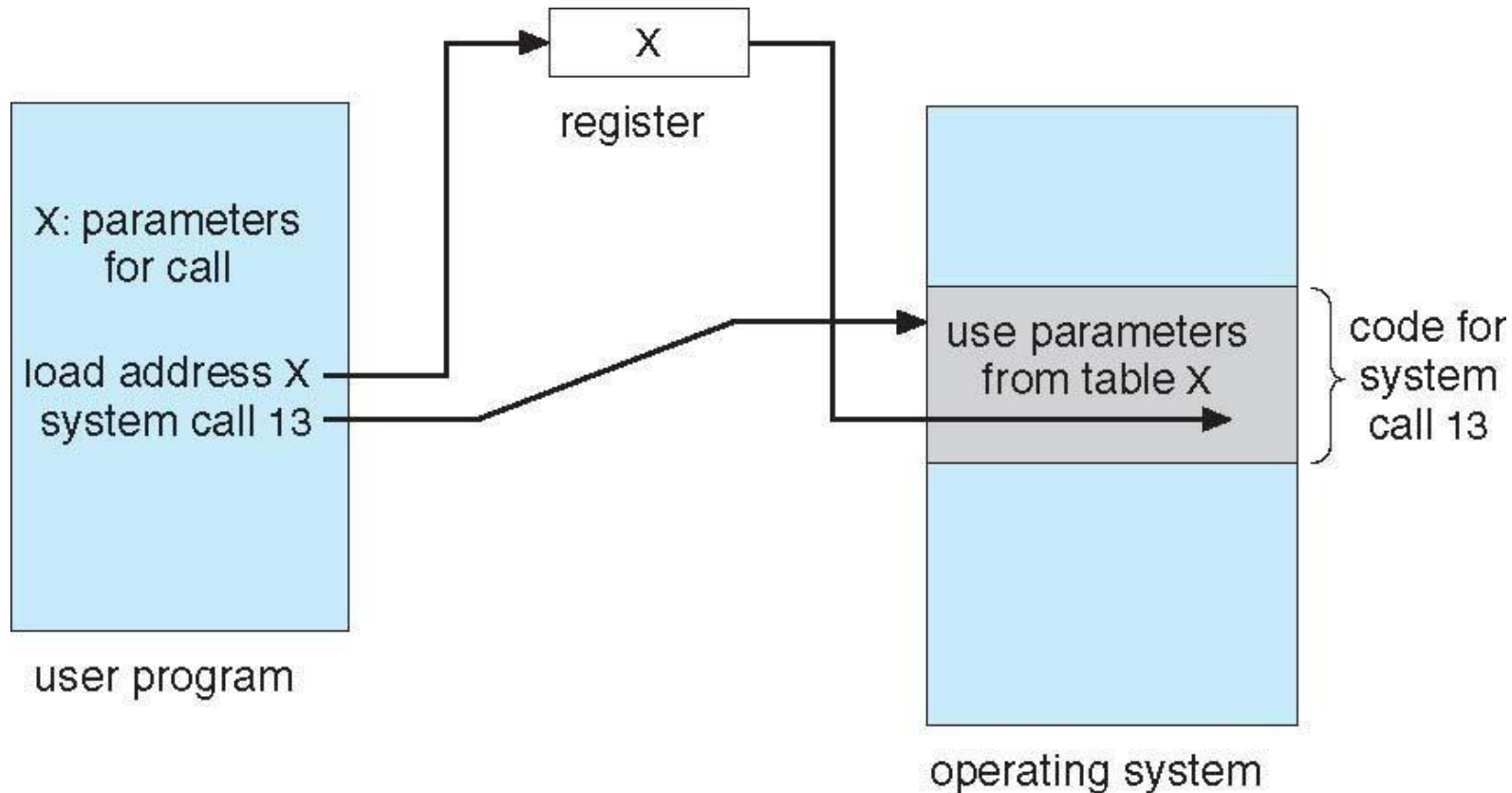
API – System Call – OS Relationship



System Call Parameter Passing

- Often, more information is required than simply identity of desired system call
 - Exact type and amount of information vary according to OS and call
- Three general methods used to pass parameters to the OS
 - Simplest: pass the parameters in *registers*
 - In some cases, may be more parameters than registers
 - Parameters stored in a *block*, or table, in memory, and address of block passed as a parameter in a register
 - This approach taken by Linux and Solaris
 - Parameters placed, or *pushed*, onto the *stack* by the program and *popped* off the stack by the operating system
 - Block and stack methods do not limit the number or length of parameters being passed

Parameter Passing via Table



Services Provided by System Calls

- Process creation and management
- Main memory management
- File Access, Directory and File system management
- Device handling(I/O)
- Protection
- Networking, etc.

TYPES OF SYSTEM CALLS

- **Process control:** end, abort, create, terminate, allocate and free memory.
- **File management:** create, open, close, delete, read file etc.
- **Device management**
- **Information maintenance**
- **Communication**

1. Process control

- A process or the job which is currently run in the system always want to load and execute another program.
- The command interpreter executes a program for example when the user clicks on the mouse button.
- Now the question arises that when the loaded program terminates when the control return. When any new program terminates in the system the memory the currently running program must be saved. So in this way we can efficiently create a program which calls another program and both are running concurrently.
- For the multiprogramming, we have created jobs or process. So the system calls for this purpose is called create a process or submit the job. When a new job, process or the group of the process is created then we have to manage the execution of these process.

System Calls for Process Management

- `fork ()`: To create a new process or spawn a child process
- `exec ()`: To run a new program or replace a current process with a new one
- `wait ()`: To make the process to wait or force the parent to suspend the execution
- `exit ()`: To terminate the process or the execution
- `getpid()`: To find the unique process id
- `getppid()`: To find the parent process id
- `nice ()`: To bias the currently running process

2. File management

- In the file management system, we should be able to create or delete the files. For creating a file the system call requires the names of the file and some attribute of the files.
- When the file is created we should be able to open and use these files. We can also read-write or reposition the files. After all the work is done we need to close the file which indicates that the file is no longer needed. We can also use these operations for directories also.
- In the file management system, we should be able to determine the values of various attributes for the file and directories.
- There are various file attributes like name, types, size, location, accounting information and so on.
- The two important system calls get attribute and file set attribute is required for this function.

System Calls for File Management

- `CreateFile()`: Creates a file
- `ReadFile()`: Reads a file
- `WriteFile()`: Writes a file
- `CloseHandle()`: Closes and invalidates the specified object handle

For Unix, few examples are:

- `open()`: Opens a file and possibly create a file or device
- `read()`: Reads from a file
- `write()`: Writes to a file
- `close()`: Closes a file descriptor, so that it no longer refers to any file and may be reused

System Calls for Directory Management

- **mkdir(name, mode):** Creates a new directory. Modes are of three digits. The first digit represents the owner, the second represents the group and the third represents other users.
- **rmdir(name):** Removes an empty directory
- **link(name1, name2):** Creates a new entry, name2, that is pointing to name1
- **unlink(name):** Removes a directory entry
- **mount(special, name, flag):** Mounts a file system
- **umount(special):** Unmounts a file system

3. Device management

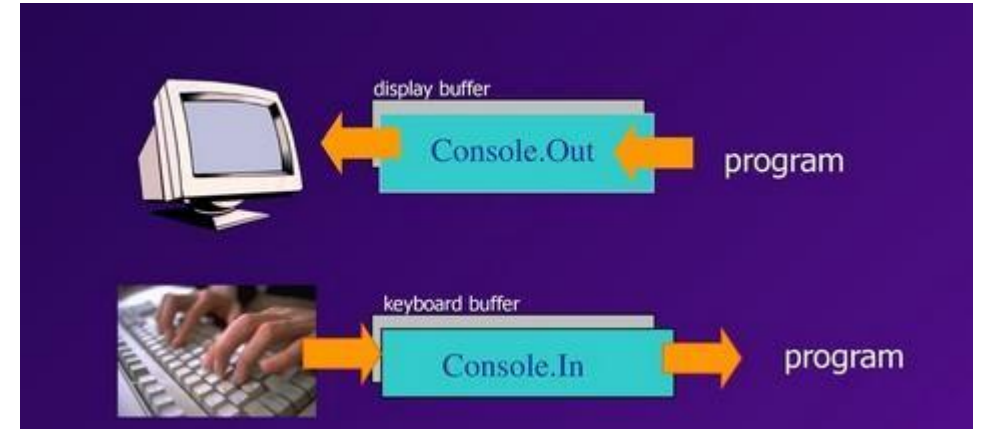
- When the program is running in the system it needs some resources such as memory, input, output, and so on.
- The resources are only granted when they are available and not held by another process. If the resources are not free the program has to wait for the resources.
- Files are the abstract or virtual devices. So there are many system calls which are needed for the files and also for the devices.
- If there are multiple users then first it requests for the devices and after finishing the work with these devices we should release it. When the devices are allocated to the user then we can perform different operations like reading, write and execute.

System Calls for Device Management

- The user programs request for the device, and they release the device after using it.
- **SetConsoleMode():** Sets the input mode of a console's input buffer or the output mode of a console screen buffer.
- **ReadConsoleMode():** Reads keyboard input from a console's input buffer.
- **WriteConsole():** Writes a character string to a console screen buffer beginning at the current cursor location .

For Unix, few examples are:

- **ioctl():** Manipulates the underlying device parameters of special files
- **read():** Reads the data in bytes
- **write():** Writes the data from the buffer



4. Information management

- In information management, various system calls are used to communicate the information between the operating system and user program.
- For example, some system has a system call forget the current time or date.
- There are many another system call like for the number of the user, different versions of operating system., free space in memory etc.

System Calls for Information Management

Few System Calls help to transfer the information between the Operating system and the user program. For example, date or time.

For Windows, few examples are:

- **GetCurrentProcessID()**: Retrieves the process identifier of the calling process
- **SetTimer()**: Creates a timer with the specified time-out value
- **Sleep()**: Suspends the execution of the current thread until the time-out interval elapses .

For Unix, few examples are:

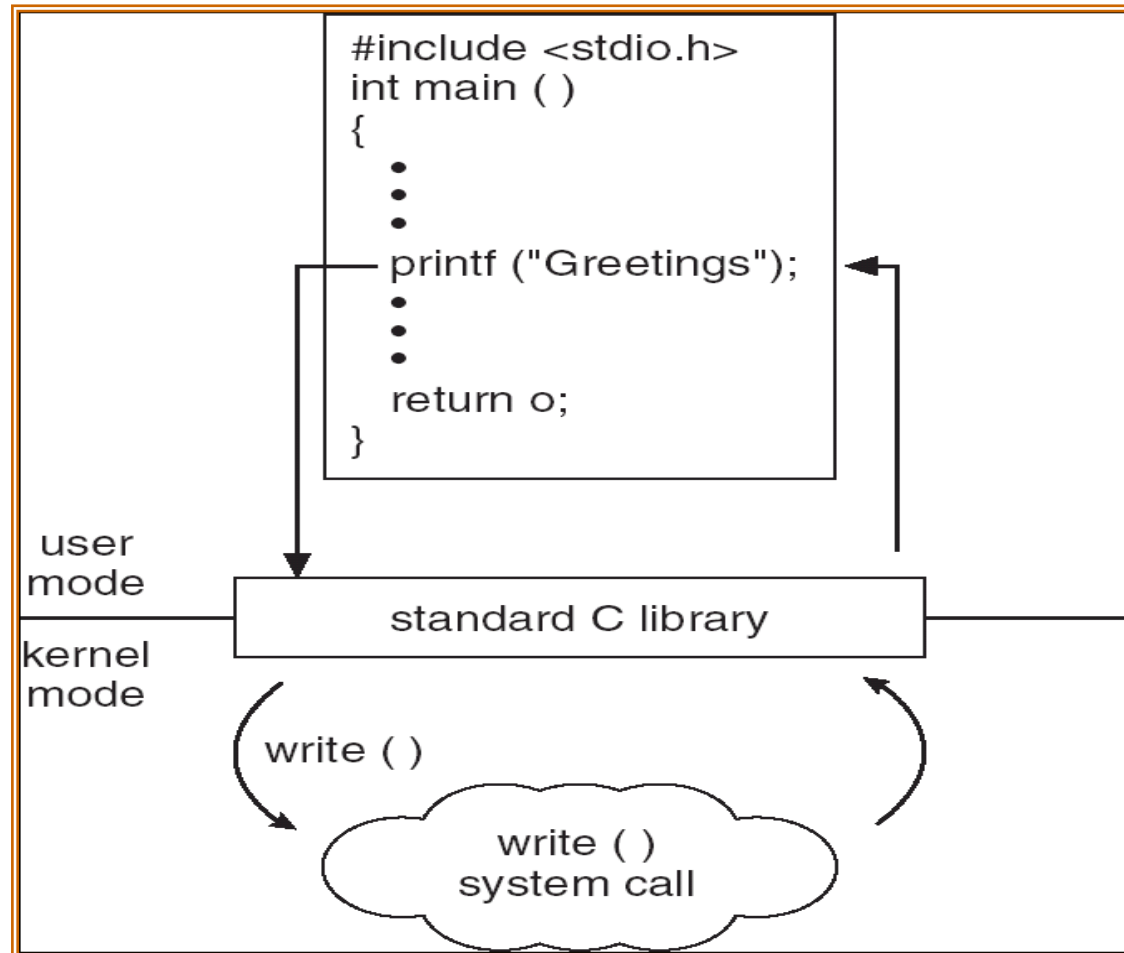
- **getpid()**: Gets process identification
- **alarm()**: Sets an alarm clock for delivery of a signal
- **sleep()**: Suspends the program for a specific time

5. Communication

- There are mainly two models for communication. First is the message passing model in this the information is exchanged between the process and this communication facility is provided by the operating system.
- When the process wants to communicate first the other communicator should be know so that the communication is established and they can communicate.
- Second is the shared memory model in this model different process shared memory and communicate with each other.

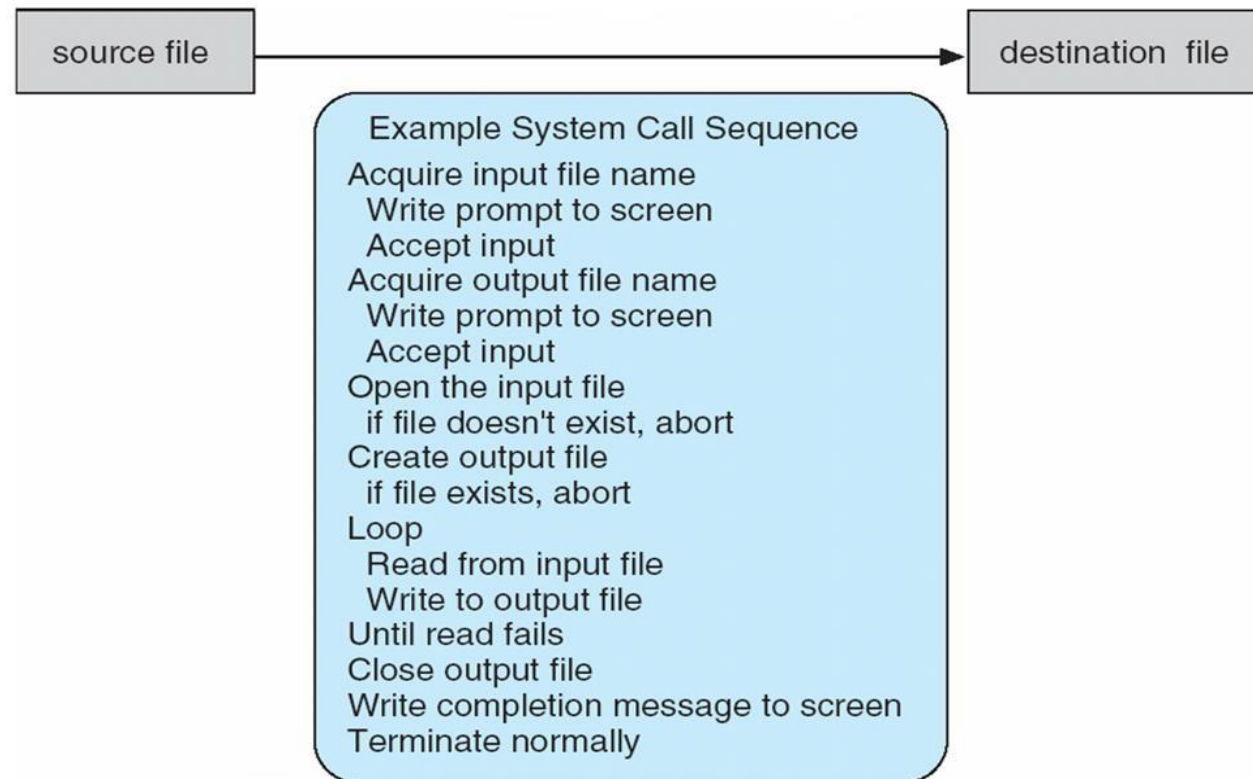
Standard C Library Example

C program invoking printf() library call, which calls write() system call



Example of System Calls

System call sequence to copy the contents of one file to another file.



Miscellaneous System Calls

- **chdir(dirname):** Changes the working directory
- **chmod(name, mode):** Changes a file's protection bits. On Linux and other Unix like operating systems, there is a set of rules for each file which defines who can access that file, and how they can access it.
- **kill(pid, signal):** Send a signal to a process
- **seconds = time(&seconds):** Gets the elapsed time since Jan 1, 1970

SYSTEM PROGRAM

- A system program is nothing but a special utility program that creates a user-friendly environment where the user can perform his desired work efficiently.
- Operating system, interpreter, compiler, editor etc. are all system programs. They provide a built-in environment to the user where several necessary functions like the ones for compiling, editing, interrupt handling, memory management etc. are already defined and hence the user does not need to write any code for such functions.

- They can be divided into:
 - File manipulation/management
 - Status information
 - File modification
 - Programming language support
 - Program loading and execution
 - Communications
 - Background services
 - Application programs
- Most users' view of the operation system is defined by system programs, not the actual system calls

- **File management** - Create, delete, copy, rename, print, dump, list, and generally manipulate files and directories
- **Status information**
 - Some ask the system for info - date, time, amount of available memory, disk space, number of users
 - Others provide detailed performance, logging, and debugging information
 - Typically, these programs format and print the output to the terminal or other output devices
 - Some systems implement a **registry** - used to store and retrieve configuration information

- **File modification**

- Text editors to create and modify files
- Special commands to search contents of files or perform transformations of the text

- **Programming-language support** - Compilers, assemblers, debuggers and interpreters sometimes provided

- **Program loading and execution**- Absolute loaders, relocatable loaders, linkage editors, and overlay-loaders, debugging systems for higher-level and machine language

- **Communications** - Provide the mechanism for creating virtual connections among processes, users, and computer systems

- Allow users to send messages to one another's screens, browse web pages, send electronic-mail messages, log in remotely, transfer files from one machine to another

- **Background Services**

- Launch at boot time
 - Some for system startup, then terminate
 - Some from system boot to shutdown
- Provide facilities like disk checking, process scheduling, error logging, printing
- Run in user context not kernel context
- Known as **services, subsystems, daemons**

- **Application programs**

- Don't pertain to system
- Run by users
- Not typically considered part of OS
- Launched by command line, mouse click, finger poke

VIRTUAL MACHINES

- A **virtual machine** takes the layered approach to its logical conclusion. It treats hardware and the operating system kernel as though they were all hardware.
- A virtual machine provides an interface *identical* to the underlying bare hardware.
- The operating system **host** creates the illusion that a process has its own processor and (virtual memory).
- Each **guest** provided with a (virtual) copy of underlying computer.

Virtual Machines - History and Benefits

- First appeared commercially in IBM mainframes in 1972
- Fundamentally, multiple execution environments (different operating systems) can share the same hardware
- Protect from each other
- Some sharing of file can be permitted, controlled
- Communicate with each other, other physical systems via networking
- Useful for development, testing
- **Consolidation** of many low-resource use systems onto fewer busier systems
- “Open Virtual Machine Format”, standard format of virtual machines, allows a VM to run within many different virtual machine (host) platforms



Self-assessment Questions

- 6) What is kill (pid, signal) System Call used for?
- a) Change the working directory
 - b) Send a signal to a process
 - c) Unmount a file system
 - d) Changes a file's protection bits
- 7) getppid() helps to find the unique process id. State True or False?
- a) True
 - b) False
- 8) Which of the following is a system call for Directory management?
- a) getpid()
 - b) getppid()
 - c) kill(pid, signal)
 - d) mount(special, name, flag)
- 9) A normal user can execute calls relating to process management. State True or False?
- a) True
 - b) False

Self Assessment Questions

10) When a user opens the system for the first time, then the system is in the user mode.

State True or False?

a) True

b) False

11) What is API?

a) Application Processing ID

b) Application Program ID

c) Application Prime ID

d) Application Program Interface

Operating System Design and Implementation

Design goals:

- At the highest level, the choice of hardware and the various types of systems i.e., batch system, time-shared system, single user system, multiuser system, distributed system, real-time system, or general purpose, affects the design of the system.
- User goals: system should be convenient to use, easy to learn and to use, reliable, safe and fast
- People who create, design, maintain and operate the system: system should be easy to design, carry out, and manage as well as be flexible, reliable, error-free, and productive

Operating System Design and Implementation

Implementation:

- At first, operating systems were created in assembly, but now C/C++ is the common language used for this. The first program that was not in assembly language is MCP (master control program).
- Linux and Windows OS are written mostly in C, although there are some small sections of assembly code for device drivers and for saving and restoring the state of registers
- The assembly code with small blocks needs to relate to some low-level I/O functions in device drivers, turning the interrupts on and off and the Test and apply Instructions for integrating the facilities.
- Using higher level languages provides and addresses the code faster. It makes the OS easier to shift to different hardware platforms.

Operating System Design and Implementation

- It can also load all facets of the system, from CPU and records to device controllers and the texts of main memory. Later it starts the operating system.
- A problem in this system is that the bootstrap code can be changed only when the change occurs in the ROM hardware chips.
- Some systems solve this problem through Erasable Programmable Read-Only Memory (EPROM), which can be read-only except when clearly given a command that can be written.

System Boots

- The process of starting a computer by loading the kernel is known as booting the system.
- In most of the computer systems, a small piece of code is present and is known as the **bootstrap program or bootstrap loader**. It detects the kernel, loads into the main memory and starts its execution.
- A boot partition of the disk is called a boot disk or system disk.

Assignments and Video Links

- Collect the features of various OS features and compare it with Windows OS.
- Operating system fundamental:
https://www.tutorialspoint.com/operating_system/index.htm
- Video Links:
- Introduction to operating system:
 - https://www.youtube.com/watch?v=2i2N_Qo_FyM&list=PLEbnTDJUr_IIf_BnzJkkN_J0Tl3iXtL8vq
 - <https://www.coursera.org/learn/os-power-user>
 - <https://www.linkedin.com/learning/comptia-it-fundamentals-fc0-u61-cert-prep-1-computer-basics-hardware-and-operating-systems>