MODULE 1: Fundamentals of Operating System

By
Prof Rohini Sawant

Introduction to OPERATING SYSTEM

Windows

Linux

Ubuntu

Mac OS X
iOS

Android

→ An Operating System (OS) is a program that manages the computer hardware.

→ It also provides a basis for Application Programs and acts as an intermediary between computer User and compter Hardware.

What is an OS?

- An operating system is software that manages a computer's hardware.
- It also provides a basis for application programs and acts as an intermediary between the computer user and the computer hardware.
- Operating systems are everywhere, from cars and home appliances that include "Internet of Things" devices, to smart phones, personal computers, enterprise computers, and cloud computing environments.

What is an OS?

- A program that acts as an intermediary between a user of a computer and the computer hardware
- Operating system goals:
  a. Execute user programs and make solving user problems **easier**
  b. Make the computer system **convenient** to use
  c. Use the computer hardware in an **efficient** manner

1.1 General Definition

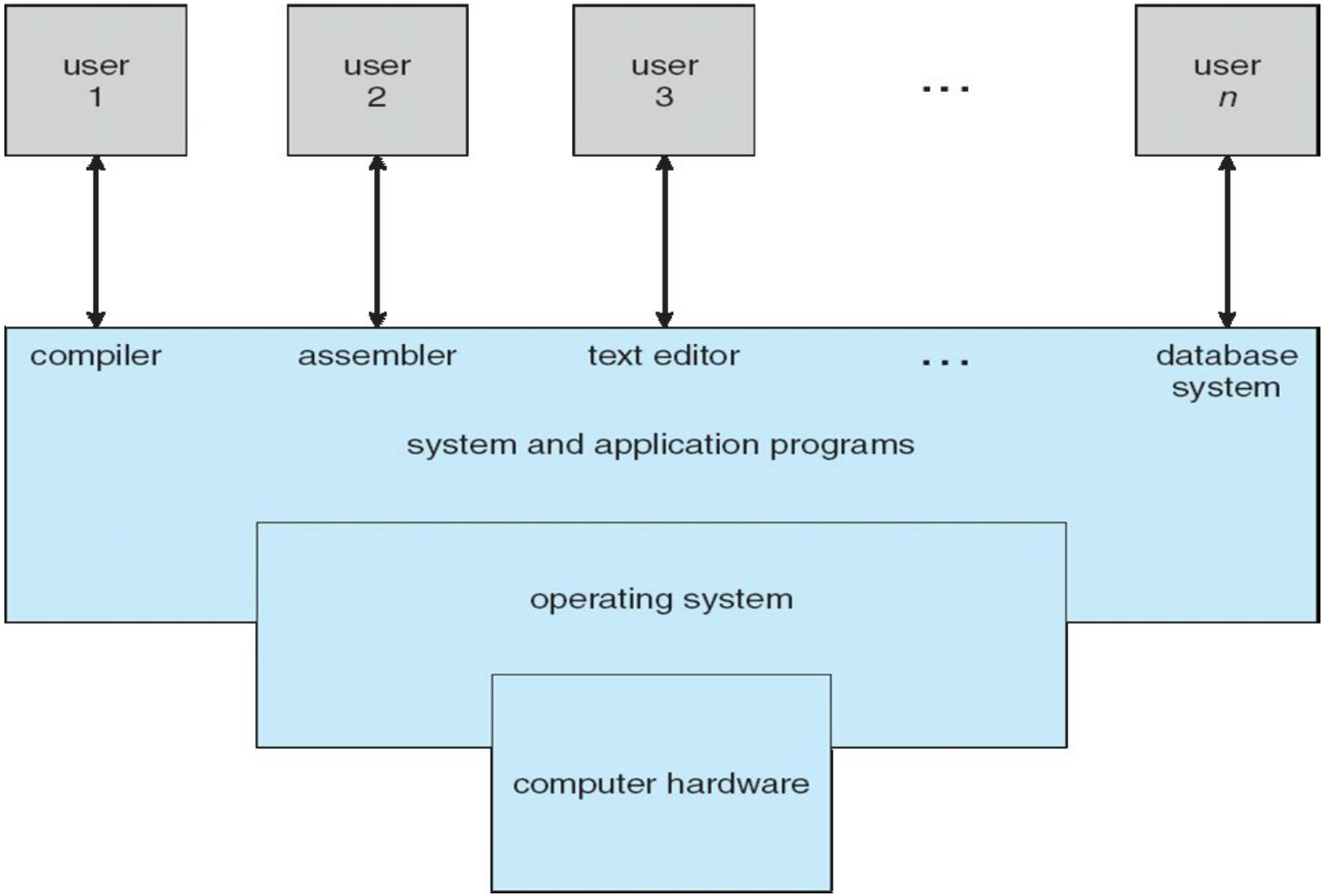
- An OS is a program which acts as an *interface* between computer system users and the computer hardware.
- It provides a user-friendly environment in which a user may easily develop and execute programs.
- Otherwise, hardware knowledge would be mandatory for computer programming.
- So, it can be said that an OS hides the complexity of hardware from uninterested users.

COMPUTER SYSTEM STRUCTURE

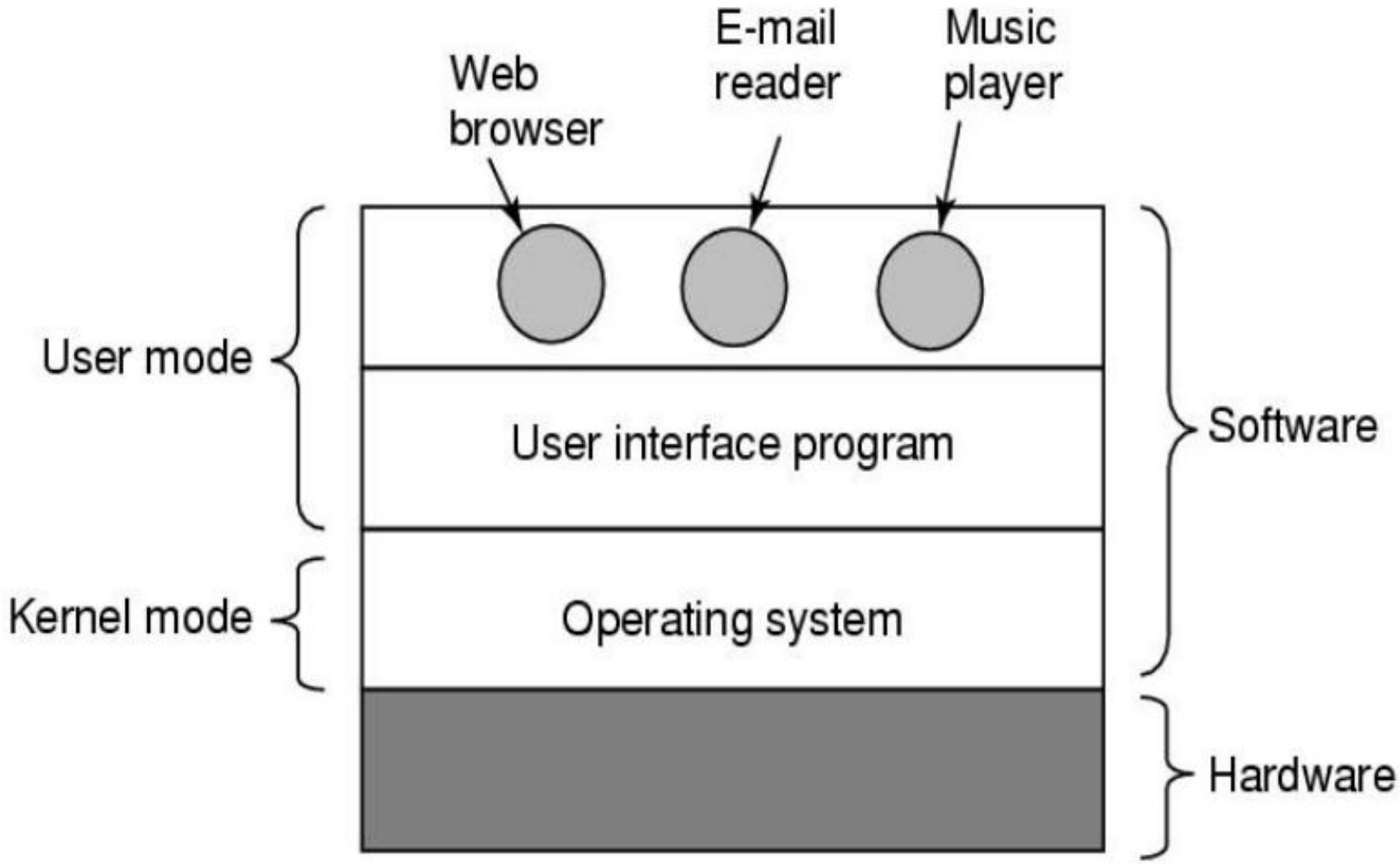
Computer system can be divided into four components:

- Hardware – provides basic computing resources
- CPU, memory, I/O devices
- Operating system: Controls and coordinates use of hardware among various applications and users
- Application programs: define the ways in which the system resources are used to solve the computing problems of the users Word processors, compilers, web browsers, database systems, video games

FOUR COMPONENTS OF A COMPUTER SYSTEM



?Where does the OS fit in



OBJECTIVES OF OPERATING SYSTEM

- **CONVENIENCE-** Computer System can be conveniently used due to Operating System.
- **EFFICIENCY-** all the resources utilized by user applications in efficient manner due to Operating System.
- **EVOLVE-** Operating system permits Efficient development, testing and flexibility.

FUNCTIONS OF OPERATING SYSTEM

PROCESS MANAGEMENT

- In multiprogramming environment, the OS decides which process gets the processor when and for how much time. This function is called process scheduling. An Operating System does the following activities for processor management −
- Keeps tracks of processor and status of process. The program responsible for this task is known as traffic controller.
- Allocates the processor (CPU) to a process.
- De-allocates processor when a process is no longer required.

FUNCTIONS OF OPERATING SYSTEM

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

- Main memory provides a fast storage that can be accessed directly by the CPU. For a program to be executed, it must in the main memory.
- An Operating System does the following activities for memory management
  - Keeps tracks of primary memory, i.e., what part of it are in use by whom, what part are not in use.
  - In multiprogramming, the OS decides which process will get memory when and how much.
  - Allocates the memory when a process requests it to do so.
  - De-allocates the memory when a process no longer needs it or has been terminated.

FUNCTIONS OF OPERATING SYSTEM

FILE MANAGEMENT-

- A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions.
- An Operating System does the following activities for file management −
- Keeps track of information, location, uses, status etc. The collective facilities are often known as **file system**.
- Decides who gets the resources.
- Allocates the resources.
- De-allocates the resources.

FUNCTIONS OF OPERATING SYSTEM

DEVICE MANAGEMENT

- An Operating System manages device communication via their respective drivers. It does the following activities for device management −
- Keeps tracks of all devices. Program responsible for this task is known as the I/O controller.
- Decides which process gets the device when and for how much time.
- Allocates the device in the efficient way.
- De-allocates devices.

SECURITY & PROTECTION

- Security module protects the data and information of a computer system against malware threat and authorized access.
- The OS keeps the system and programs safe and secure through authentication. A user id and password decide the authenticity of the user.

OPERATING SYSTEM INTERFACES

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.


C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.
C:\Documents and Settings>

OPERATING SYSTEM INTERFACES

GRAPHICAL USER INTERFACE

- 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.

WHAT IS A GUI?


Recycle Bin
Lenovo MapLife
OneKey Recovery
Acrobat.com
Lenovo Ready.com...
Study.com

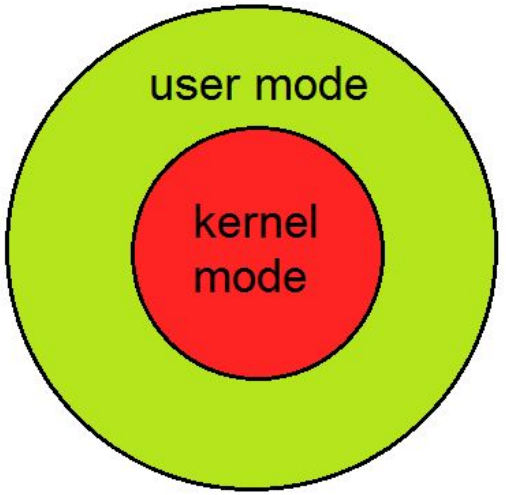
OPERATING SYSTEM OPERATIONS

DUAL MODE & MULTI MODE OPERATIONS

- We must be able to distinguish between the execution of operating-system code and user defined code.
- The approach is to separate the two modes of operation:
- user mode and kernel mode.
- A bit, called the mode bit is added to the which indicates the current mode: kernel (0) or user (1).
- The dual mode of operation provides us with the means for protecting the operating system from errant users.
- Operating System needs to function in the dual mode because the Kernel Level programs perform all the bottom level functions of the OS like process management, Memory management, etc.
- If the user alters these, then this can cause an entire system failure.

MODES OF OPERATING SYSTEM

- **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.



OPERATING SYSTEM SERVICES

- User Interface
- Program Execution
- File System Manipulation
- Communications
- Error Detection
- Resource Allocation
- Logging
- Security

OPERATING SYSTEM SERVICES

- Some services provide functions helpful to the user:
- User interface - Almost all operating systems have a user interface (UI) viz Command-Line (CLI), Graphics User Interface (GUI), Batch
- Program execution - The system must be able to load a program into memory and to run that program, end execution, either normally or abnormally (indicating error)
- I/O operations - A running program may require I/O, which may involve a file or an I/O device.
- File-system manipulation - The file system is of particular interest. Obviously, programs need to read and write files and directories, create and delete them, change them, search them, list file information, perform permission management.

OPERATING SYSTEM SERVICES

- Communications – Processes may exchange information, on the same computer or between computers over a network. Communications may be via shared memory or through message passing (packets moved by the OS)
- Error detection – OS needs to be constantly aware of possible errors. May occur in the CPU and memory hardware, in I/O devices, in user program For each type of error, OS should take the appropriate action to ensure correct and consistent computing. Debugging facilities can greatly enhance the user's and programmer's abilities to efficiently use the system

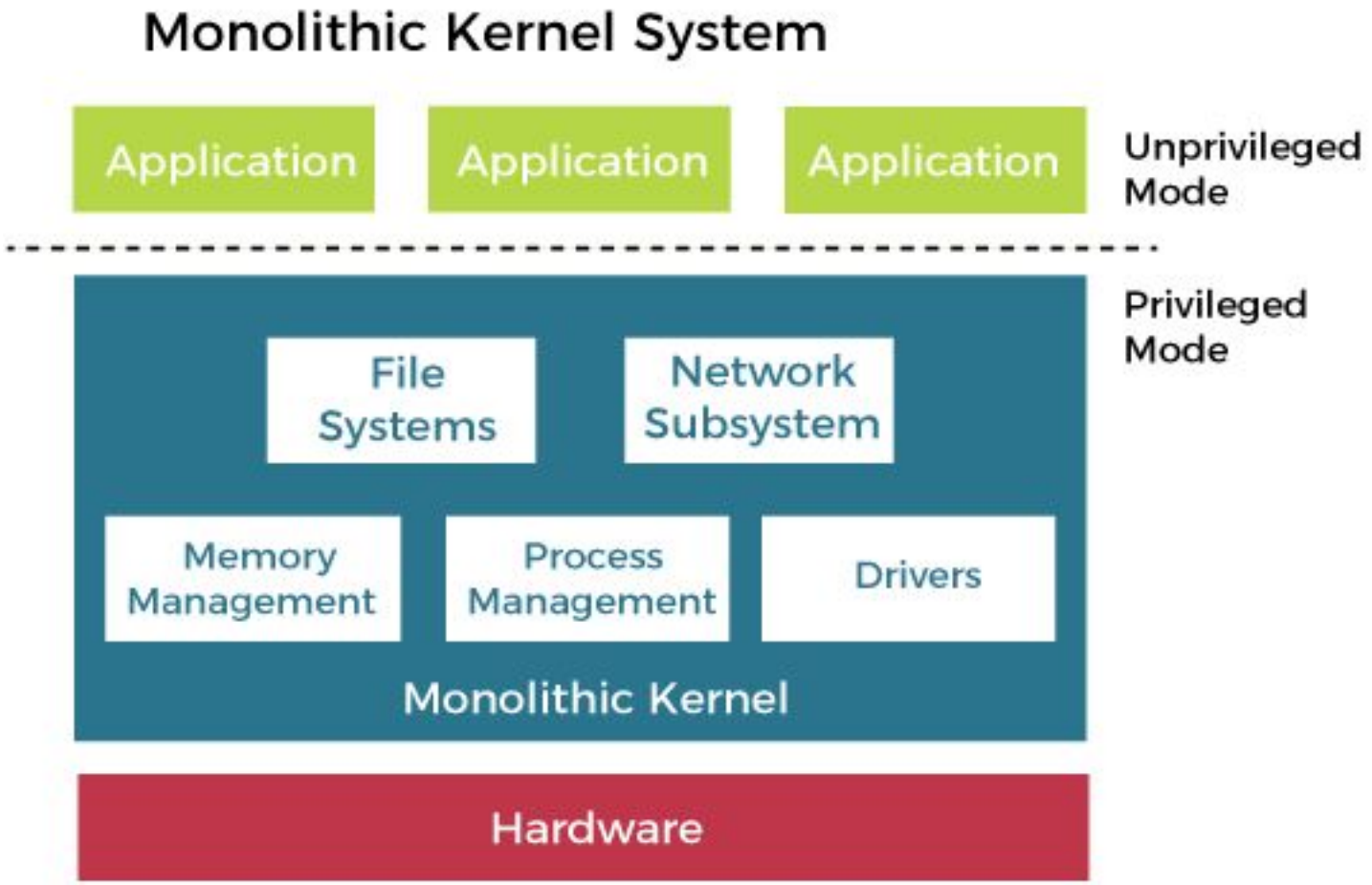
OPERATING SYSTEM STRUCTURE

- Monolithic
- Layered
- Microkernel
- Virtual machines
- Client server model
- Exokernels

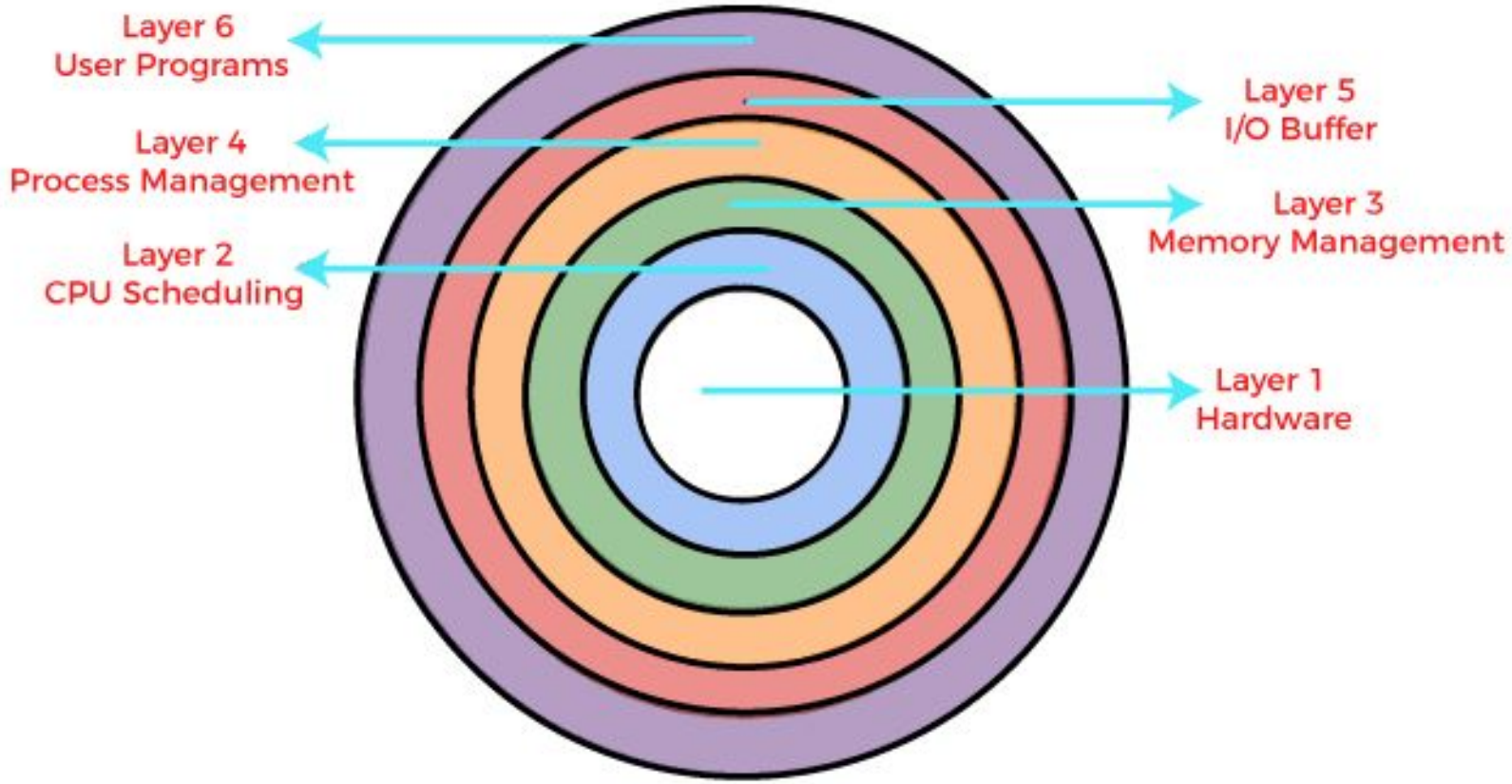
MONOLITHIC OS STRUCTURE

- The monolithic operating system is a very basic operating system in which file management, memory management, device management, and process management are directly controlled within the kernel.
- The kernel can access all the resources present in the system. In monolithic systems, each component of the operating system is contained within the kernel.
- The monolithic operating system is also known as the monolithic kernel. A monolithic kernel is an operating system architecture where the entire operating system is working in kernel space..
- Monolithic architecture has the following advantages, such as:
- Simple and easy to implement structure.
- Faster execution due to direct access to all the services
- Some disadvantages of monolithic architecture:
- If any service fails in the monolithic kernel, it leads to the failure of the entire system.
- The addition of new features or removal of obsolete features is very difficult.
- Security issues are always there because there is no isolation among various servers present in the kernel.

MONOLITHIC OS STRUCTURE



LAYERED OS STRUCTURE



LAYERED OS STRUCTURE

- This approach breaks up the operating system into different layers.
- This allows implementers to change the inner workings, and increases modularity.
- As long as the external interface of the routines don't change, developers have more freedom to change the inner workings of the routines.
- With the layered approach, the bottom layer is the hardware, while the highest layer is the user interface.
- The main advantage is simplicity of construction and debugging.
- The main difficulty is defining the various layers.
- The main disadvantage is that the OS tends to be less efficient than other implementations.

MICROKERNEL OS STRUCTURE

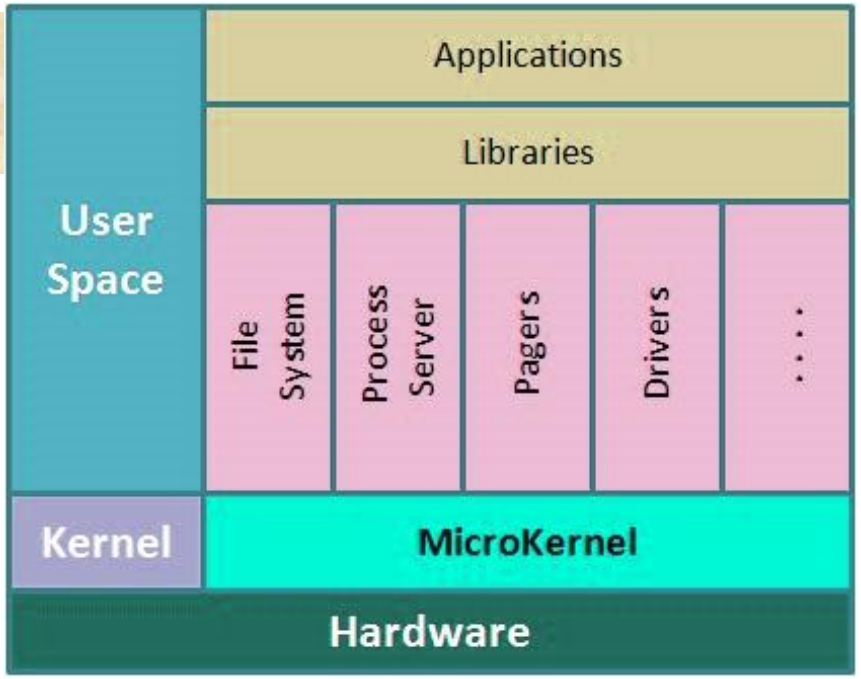
- From Kernel Os structure non essential components are removed and implemented as System and user-level programs.
- They interact with local and remote server processes and well suited to distributed env.
- communication takes place through message passing.

Adv: **Expanding new services** in user space have no modification for kernel.

- Security, Portability and reliability.
- Uniform interfaces,

Disadv: Performance decrease due to increased system function overhead.

Kernel minimum



DIFFERENCE

Monolithic

- Kernel size large
- OS is complex to design
- Request may be serviced faster
- No msg passing & context switching while kernel performs the job
- Entire process works in kernel mode
- No replaceable modules of Kernel

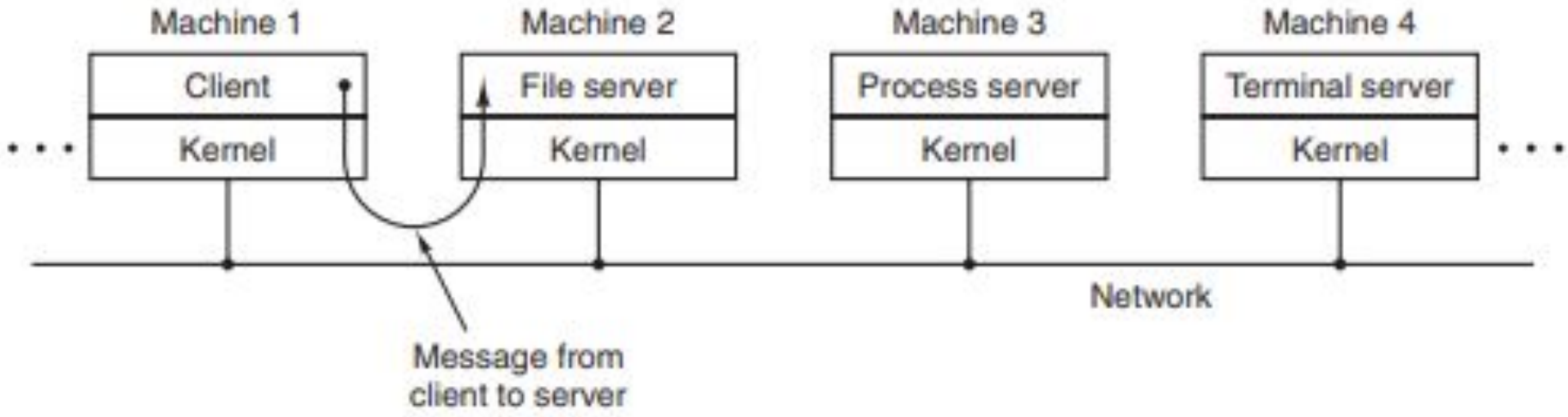
MicroKernel

- Kernel size small
- OS is Easy to design,install
- Slower than monolithic
- Requires msg passing & context switch
- Process runs in user as well as kernel mode
- Can have any no of replaceable modules to kernel

CLIENT SERVER MODEL

- Client-Server Model of a computer is the one in which many clients (remote processors) request and receive service from a centralized server (host computer).
- Client computers provide an interface to allow a computer user to request services of the server and to display the results the server returns. Servers wait for requests to arrive from clients and then respond to them.
- Ideally, a server provides a standardized transparent interface to clients so that clients need not be aware of the specifics of the system (i.e., the hardware and software) that is providing the service.
- Clients are often situated at workstations or on personal computers, while servers are located elsewhere on the network, usually on more powerful machines

CLIENT SERVER MODEL



VIRTUAL MACHINE OS STRUCTURE

- Fundamental idea – abstract hardware of a single computer into several different execution environments

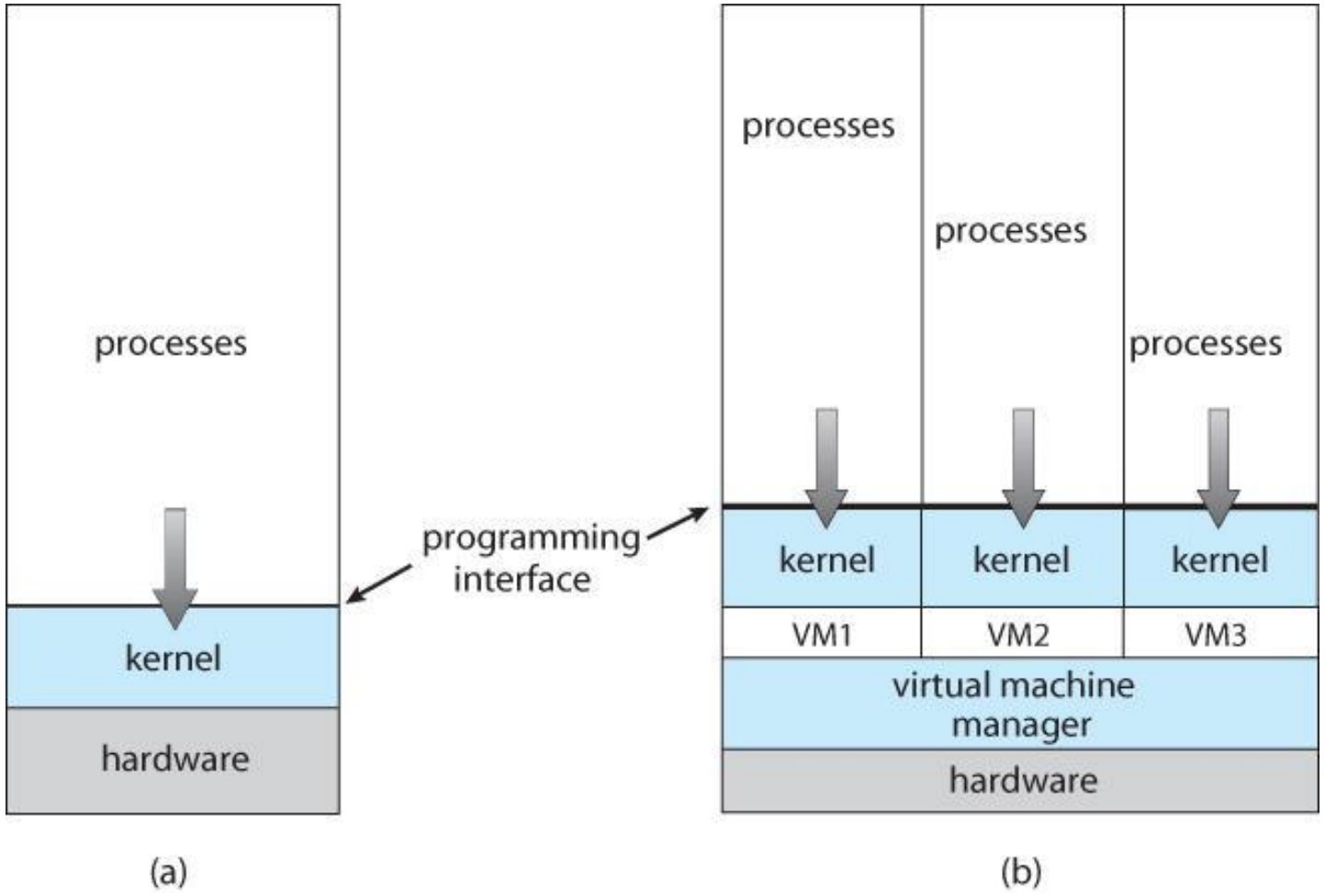
->Similar to layered approach

->But layer creates virtual system (virtual machine, or VM) on which operation systems or applications can run.

Several components are:

- Host – underlying hardware system
- Virtual machine manager (VMM) or hypervisor – creates and runs virtual machines by providing interface that is identical to the host
- Guest – process provided with virtual copy of the host
- Usually an operating system a Single physical machine can run multiple operating systems concurrently, each in its own virtual machine

VIRTUAL MACHINE OS STRUCTURE



EXOKERNEL OS STRUCTURE

- The exokernel architecture is designed to separate resource protection from management to facilitate application-specific customization.
- The exokernel operating system attempts to address this problem by eliminating the notion that an operating system must provide abstractions upon which to build applications.
- The idea is to impose as few abstractions as possible on the developers and to provide them with the liberty to use abstractions as and when needed.
- The exokernel architecture is built such that a small kernel moves all hardware abstractions into untrusted libraries known as library operating systems.

SYSTEM CALLS

- 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.
- System calls provide an interface to the services made available by an operating system.
- These calls are generally available as functions written in C and C++

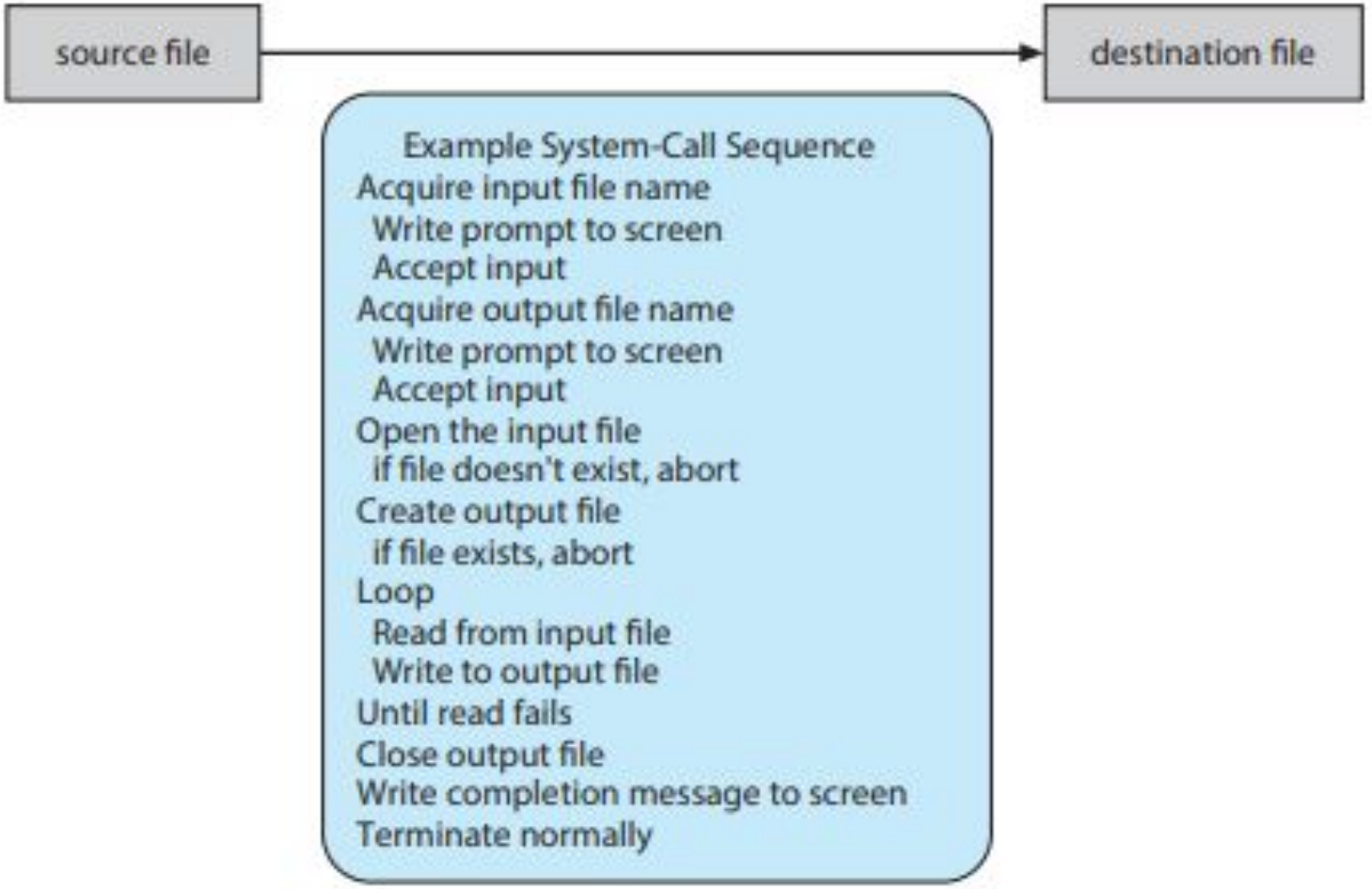


Figure 2.5 Example of how system calls are used.

TYPES OF SYSTEM CALLS

PROCESS CONTROL

- Create Process
- Terminate Process
- Load Execute
- Get Process attributes, set process attributes
- Wait Event, Signal Event
- Allocate and Free Memory

FILE MANAGEMENT

- create file, delete file
- open, close
- read, write, reposition
- get file attributes, set file attributes

TYPES OF SYSTEM CALLS

DEVICE MANAGEMENT

- request device, release device
- read, write, reposition
- get device attributes, set device attributes
- logically attach or detach devices

INFORMATION MAINTENANCE

- get time or date, set time or date
- get system data, set system data
- get process, file, or device attributes
- set process, file, or device attributes

TYPES OF SYSTEM CALLS

COMMUNICATIONS

- create, delete communication connection
- send, receive messages
- transfer status information
- attach or detach remote devices

PROTECTION

- get file permissions
- set file permissions

SYSTEM PROGRAMS

- System programs provide a convenient environment for program development and execution. The can be divided into:

- File manipulation
- Status information
- File modification
- Programming language support
- Program loading and execution
- Communications

SYSTEM PROGRAMS

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

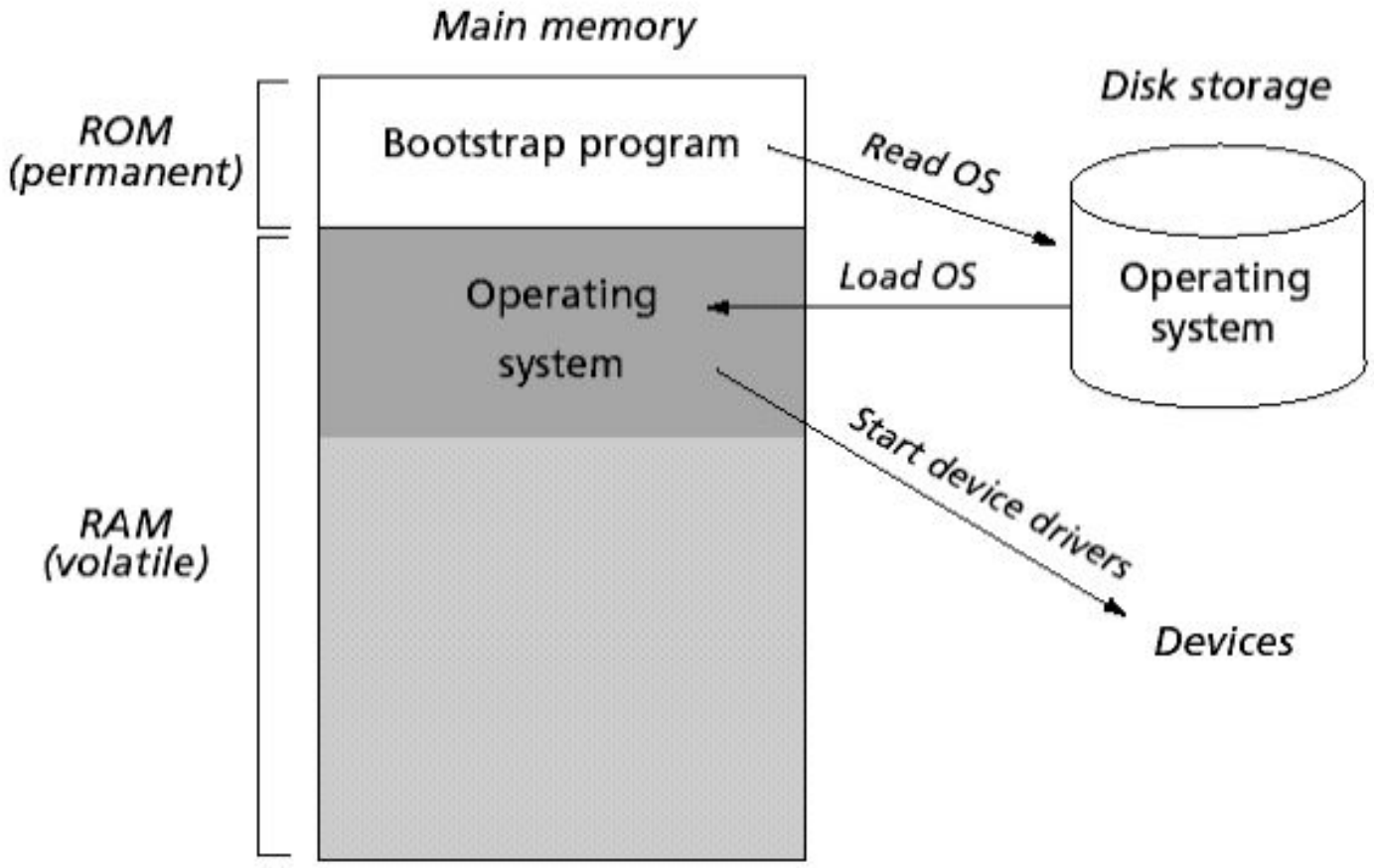
SYSTEM PROGRAMS

- **PROGRAMMING-LANGUAGE SUPPORT -** Compilers, assemblers, debuggers and interpreters sometimes provided for most common languages.
- **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.

SYSTEM BOOT

- The procedure of starting a computer by loading the kernel is known as Booting the system.
- Hence it needs a special program, stored in ROM to do this job known as the Bootstrap loader. Example: BIOS
- The initial bootstrap program is found in the BIOS read-only memory.
- This program is in the form of Read Only Memory(ROM) because the RAM is in unknown state at system startup. ROM is convenient because it needs no initialization & cannot be affected by a computer virus.
- This program can run diagnostics, initialize all components of the system, loads and starts the Operating System loader.
- The loader program loads and starts the operating system.
- When the Operating system starts, it sets up needed data structures in memory, sets several registers in the CPU, and then creates and starts the first user level program. From this point, the operating system only runs in response to interrupts.

SYSTEM BOOT


Main memory
ROM
(permanent)
Bootstrap program
Operating
system
RAM
(volatile)
Disk storage
Read OS
Load OS
Operating
system
Start device drivers
Devices