Chapter 1: Introduction

(OS Structure, Modes and Services)

What is an Operating System?

- A program that acts as an intermediate/ interface between a user of a computer and the computer hardware.
- Resource allocator
- Control Program

- 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

Computer System Structure

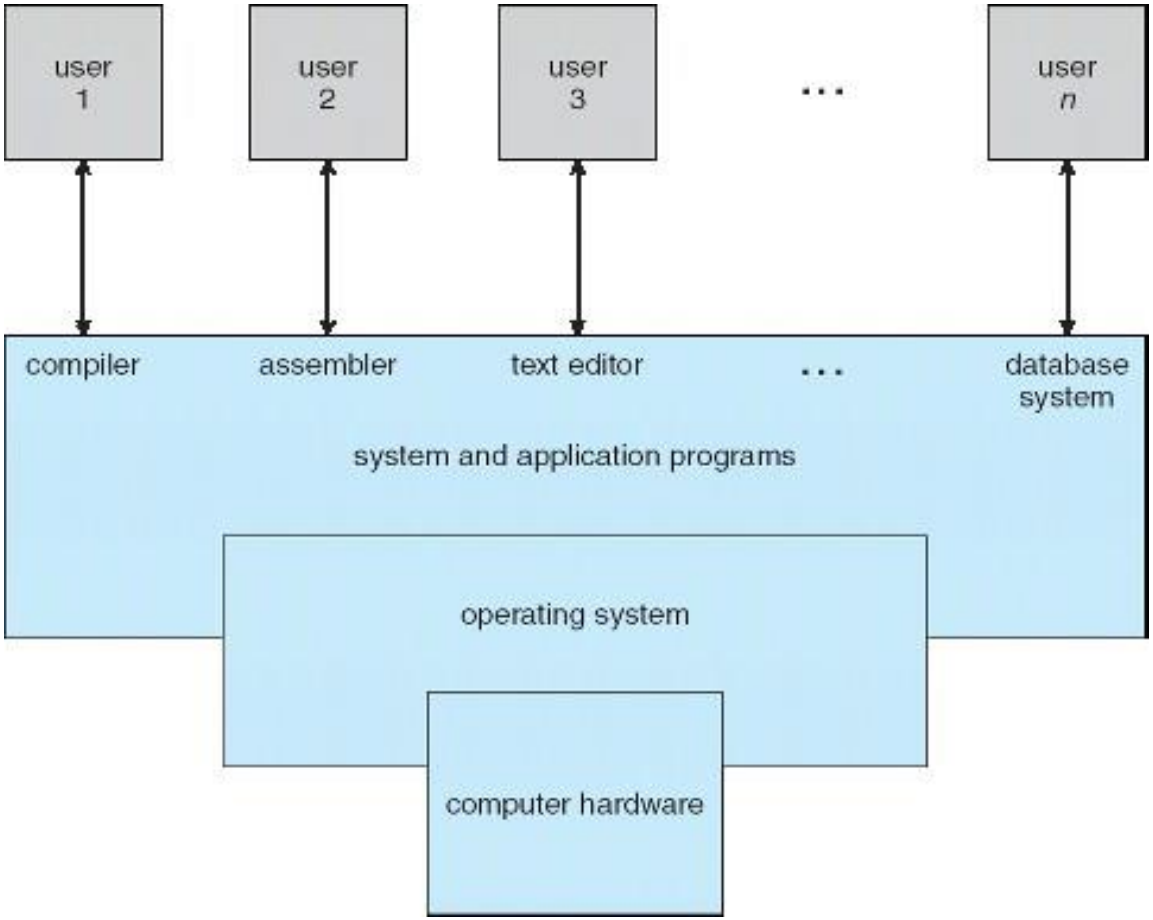
- Computer system can be divided into four components
  - **1. Hardware** – provides basic computing resources CPU, memory, I/O devices
  - **2. Operating system**

    Controls and coordinates use of hardware among various applications and users
  - **3. 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
  - **4. Users**

    People, machines, other computers

Four Components of a Computer System



Operating System Definition

To understand more fully the OS role, we explore OS from 2 view points.:

1. User view: In single user, it should be easy to use.

In other cases, where users access the same user through different terminals, More emphasize is on resource allocation and utilization.

2. System View:

- OS is a **resource allocator**
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- OS is a **control program**
  - Controls execution of programs to prevent errors and improper use of the computer

Who controls the execution of programs to prevent errors and improper use of computer?

a) Resource allocator

b) Control Program

c) Hardware

d)None of the above

(b)

Operating-System Operations

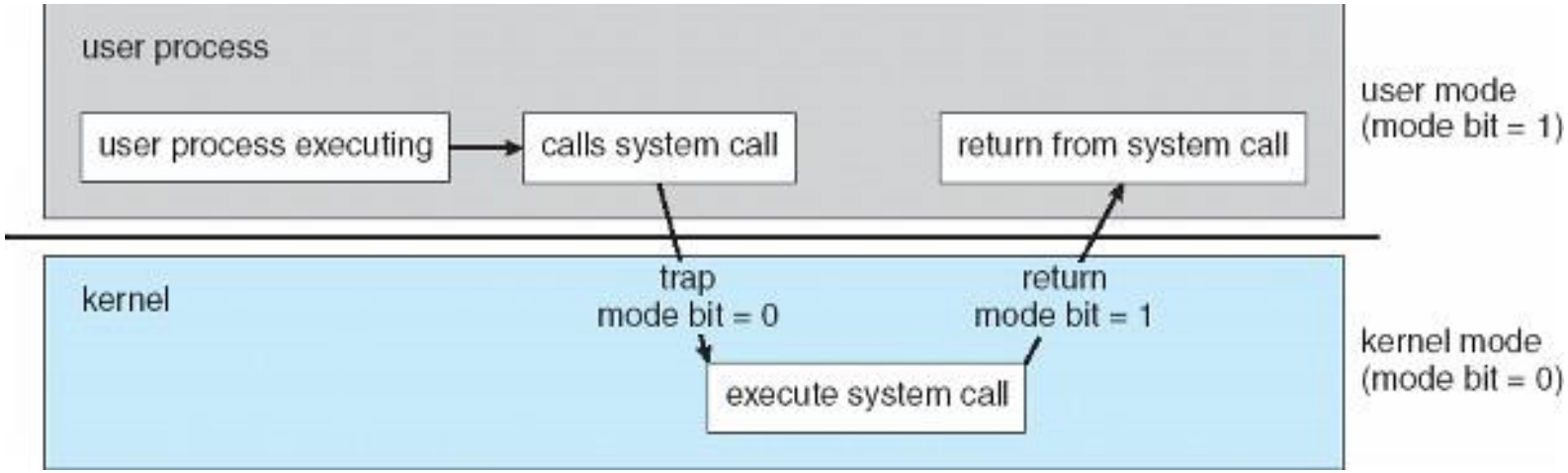
- Modern OS's are Interrupt driven.
- Program or software send generate events by using system calls. Error or request by a software creates **exception** or **trap**
  - Division by zero, request for operating system service

- **Dual-mode operation** allows OS to protect itself and other system components
  - **User mode** and **kernel mode**

  - **Mode bit** provided by hardware

    Provides ability to distinguish when system is running user code or kernel code

Transition from User to Kernel Mode



The operating system switches from user mode to kernel mode so the mode bit will change from?

a) 0 to 1

b) 1 to 0

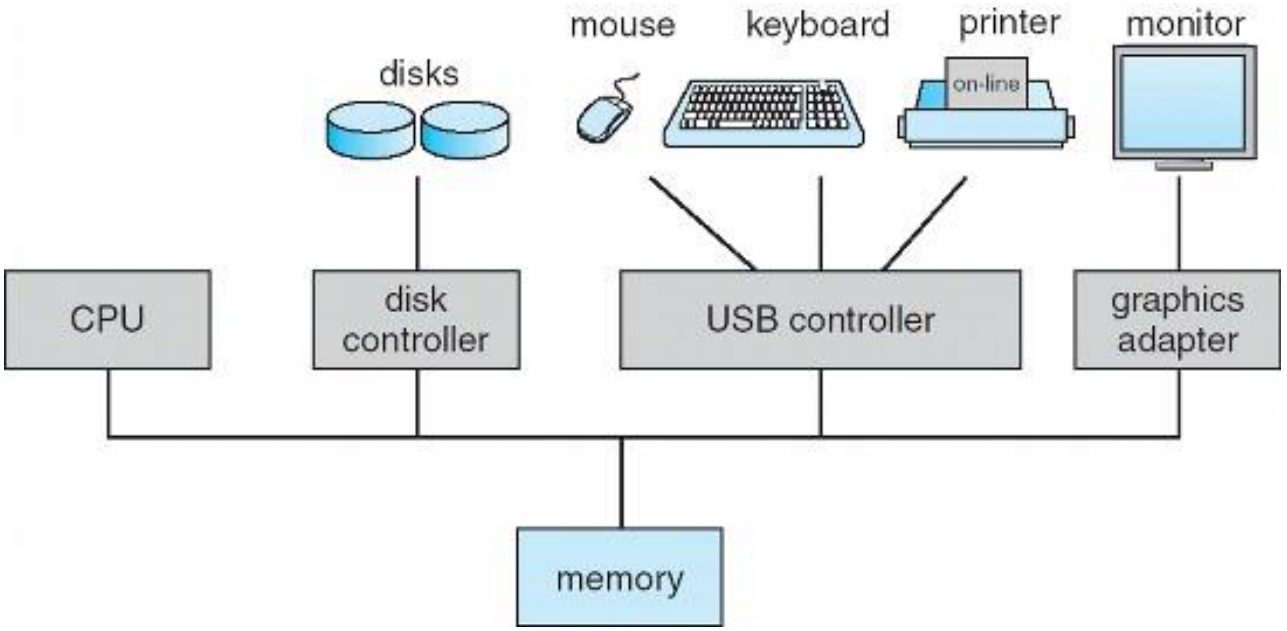
c) Remain constant

d)None

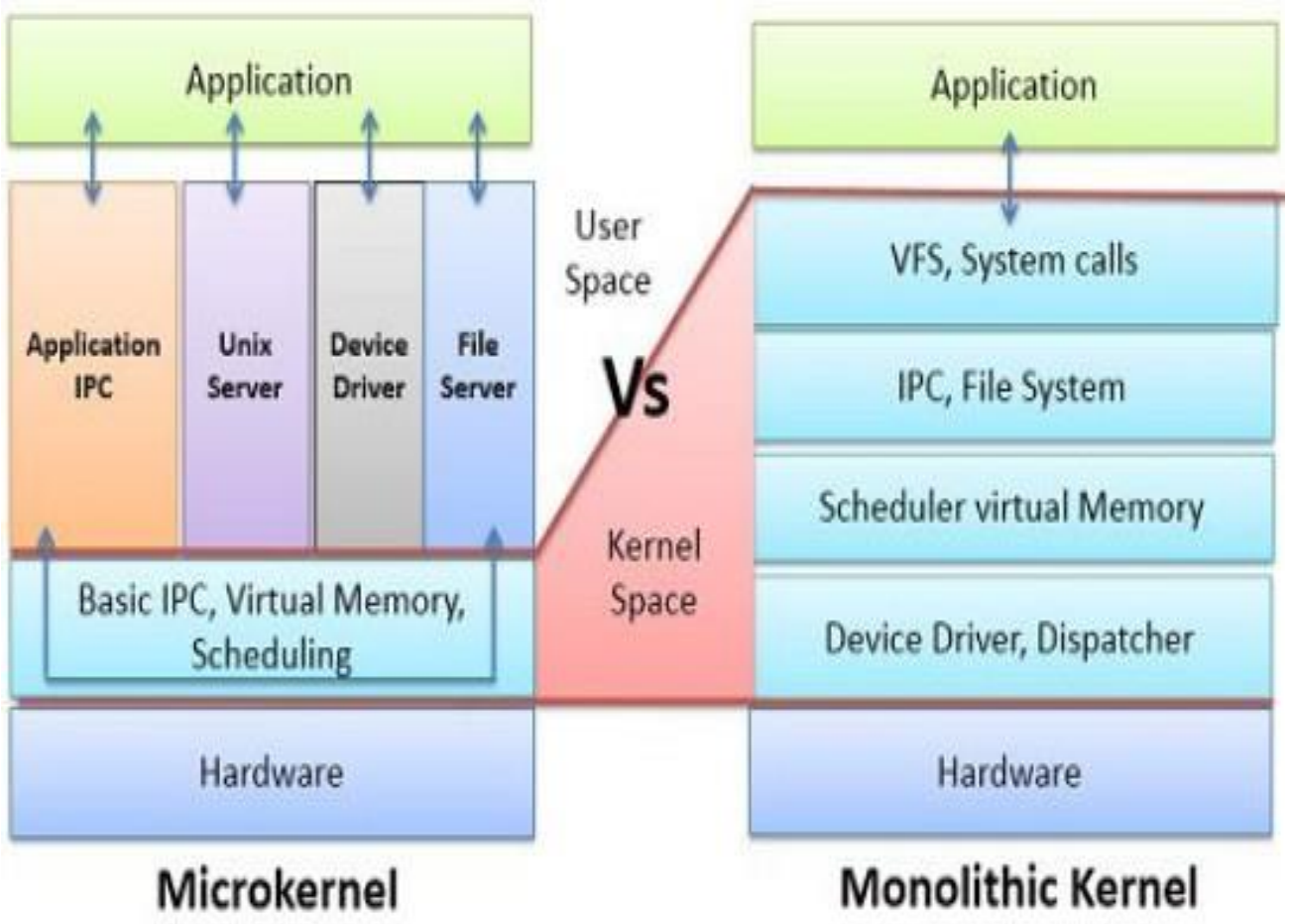
(b)

Computer System Organization

- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles



Types of kernel



Key Differences Between Microkernel and Monolithic Kernel

The basic point on which microkernel and monolithic kernel is distinguished is that **microkernel** implement user services and kernel services in **different address spaces** and **monolithic kernel** implement both user services and kernel services under **same address space**

The size of microkernel is **small** as only kernel services reside in the kernel address space. However, the size of monolithic kernel is comparatively **larger** than microkernel because both kernel services and user services reside in the same address space.

The execution of monolithic kernel is **faster** as the communication between application and hardware is established using the **system call**. On the other hands, the execution of microkernel is **slow** as the communication between application and hardware of the system is established through **message passing**

It is easy to extend microkernel because new service is to be added in user address space that is isolated from kernel space, so the kernel does not require to be modified. Opposite is the case with monolithic kernel if a new service is to be added in monolithic kernel then entire kernel needs to be modified.

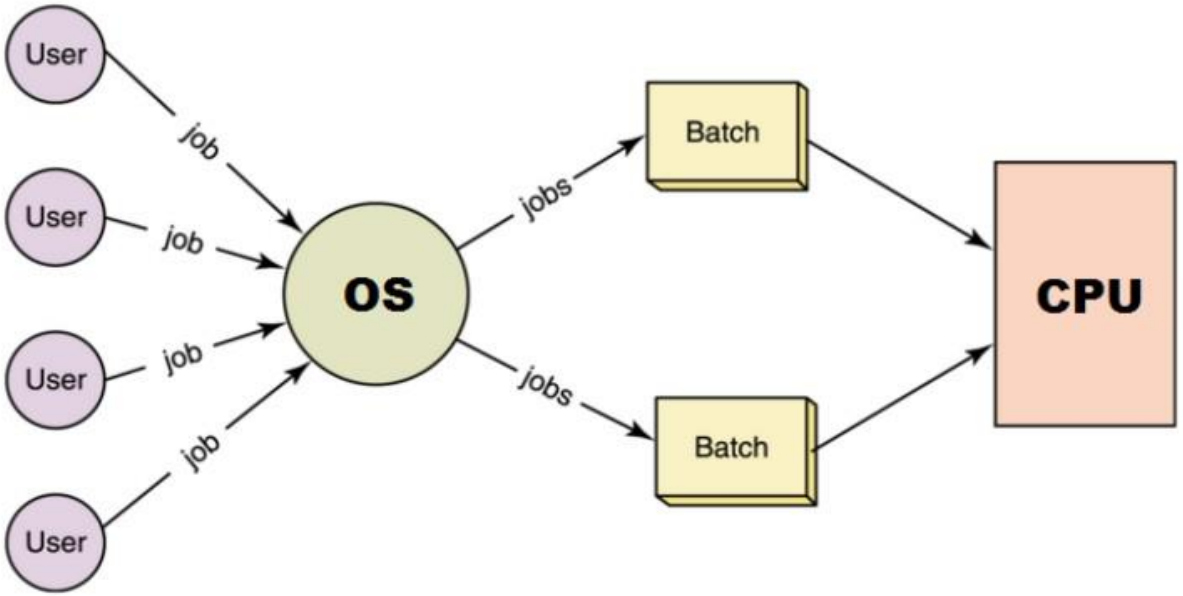
Microkernel is more **secure** than monolithic kernel as if a service fails in microkernel the operating system remain unaffected. On the other hands, if a service fails in monolithic kernel entire system fails.

Monolithic kernel designing requires **less code**, which further leads to fewer bugs. On the other hands, microkernel designing needs more code which further leads to more bugs.

TYPES OF OS

Batch Systems

“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” *



*[https://www.tutorialspoint.com/operating_system/os_types.htm]

TYPES OF OS

Batch Systems

- Early computers were Physically enormous machines run from a console
- The common input devices were card readers and tape drives.
- The common output devices were line printers, tape drives, and card punches.
- The user did not interact directly with the computer systems.
- User prepare a job -which consisted of the program, and submitted it to the computer operator.
- after minutes, hours, or days, the output appeared.
- To speed up processing, operators batched jobs with similar needs together and ran them through the computer as a group.

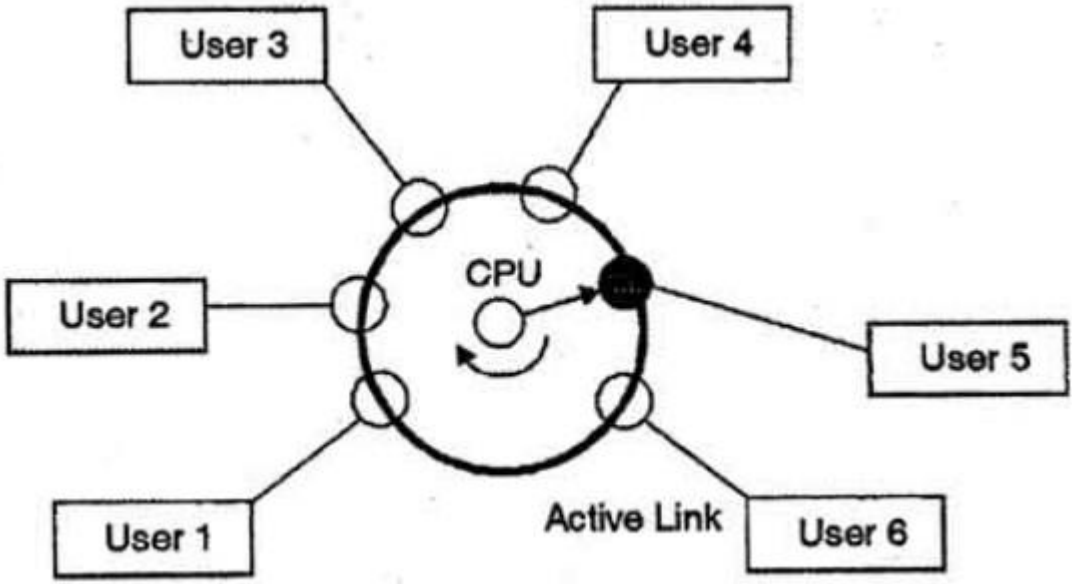
Multiprogrammed OS

- Needed for efficiency
- Single user cannot keep CPU and I/O devices busy at all times
- Multiprogramming organizes jobs (code and data) so CPU always has one to execute
- A subset of total jobs in system is kept in memory
- One job selected and run via **job scheduling**
- When it has to wait (for I/O for example), OS switches to another job

Timesharing OS

- **Timesharing (multitasking)** is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating **interactive** computing
  - Each user has at least one program executing in memory □**process**
  - If several jobs ready to run at the same time □ **CPU scheduling**
  - If processes don't fit in memory, **swapping** moves them in and out to run
  - **Virtual memory** allows execution of processes not completely in memory

Timesharing OS

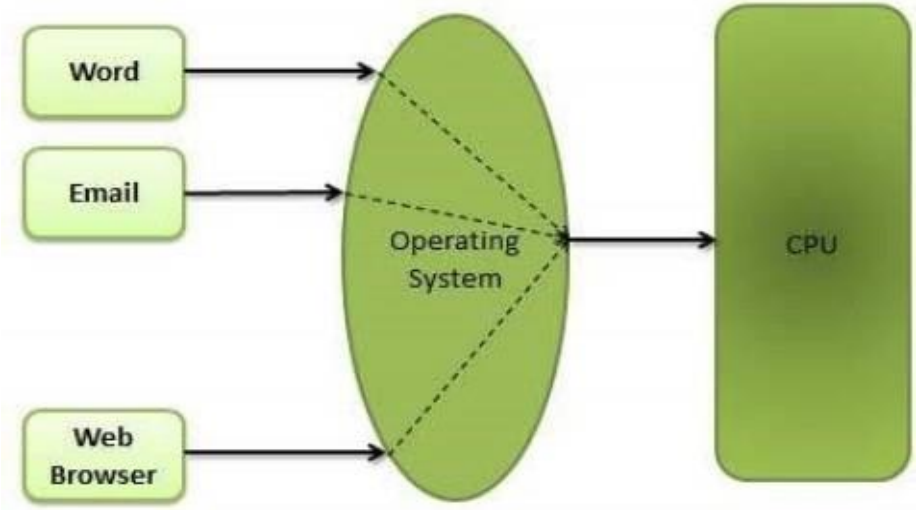


Multitasking Systems

- Types of Multitasking:

1. **Preemptive:** the operating system parcels CPU *time slices* to each program.
2. **Cooperative:** each program can control the CPU for as long as it needs it.

   If a program is not using the CPU, however, it can allow another program to use it temporarily.

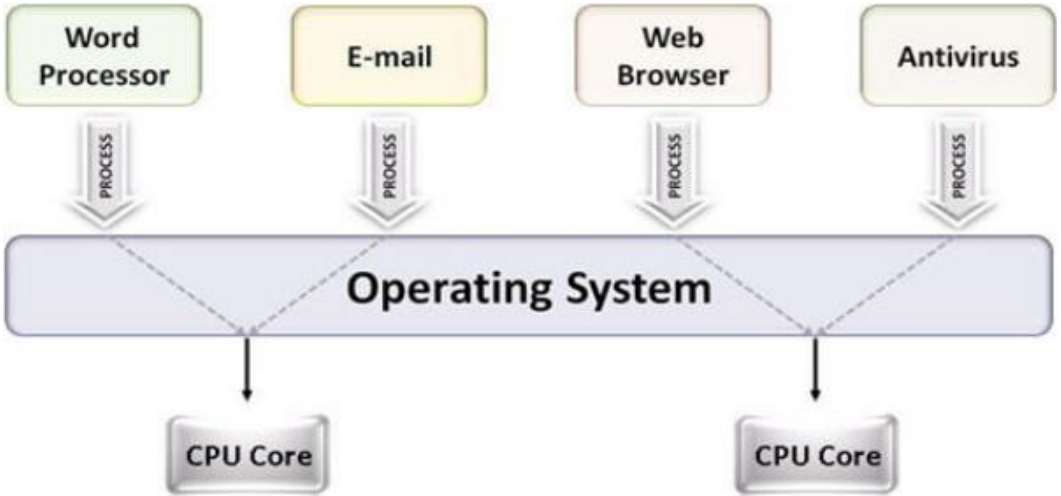


Multiprocessing OS

- Multi-processor systems; that is, they **have multiple CPU.**

Exp: dual core processor has 2 process cycles

- **Also known as parallel systems** or **tightly coupled systems**
- **Such systems have more than one processor** in close communication, sharing the computer bus, the clock, and sometimes memory and peripheral devices.



Distributed Systems

- A network is a communication path between two or more systems.
- Each system over the network keeps copy of the data, and this leads to Reliability (Because if one system crashes , data is not lost).

- CLIENT SERVER SYSTEMS
- PEER TO PEER SYSTEMS

Real Time Systems

- **Time bound systems**
- Real time systems are of 2 types:

- **1. Soft Real time Systems:** Process should complete in specific time but May have some delay (Positive delay) and will nor harm the system.
- **Exp: Session expires but can be re-logged in.**

- **2. Hard Real Time Systems:** Each process is assigned a specific time instance, and Process must complete in that time otherwise system will crash.

Real Time Embedded Systems

- is a computing environment that **reacts to input within a specific time period.**
- Time Driven
- Task specific
- Exp: Microwave, Washing Machine…

Q. In which type of operating system users do not interact directly with the computer system?

a) Multiprogramming operating systems

b) Multiprocessing operating systems

c) Batch operating systems

d)Distributed operating systems

(c)

Q. What is the objective of multiprogramming operating systems?

a) Maximize CPU utilization

b) Switch the CPU among processes

c) Achieve multitasking

d)None of the above

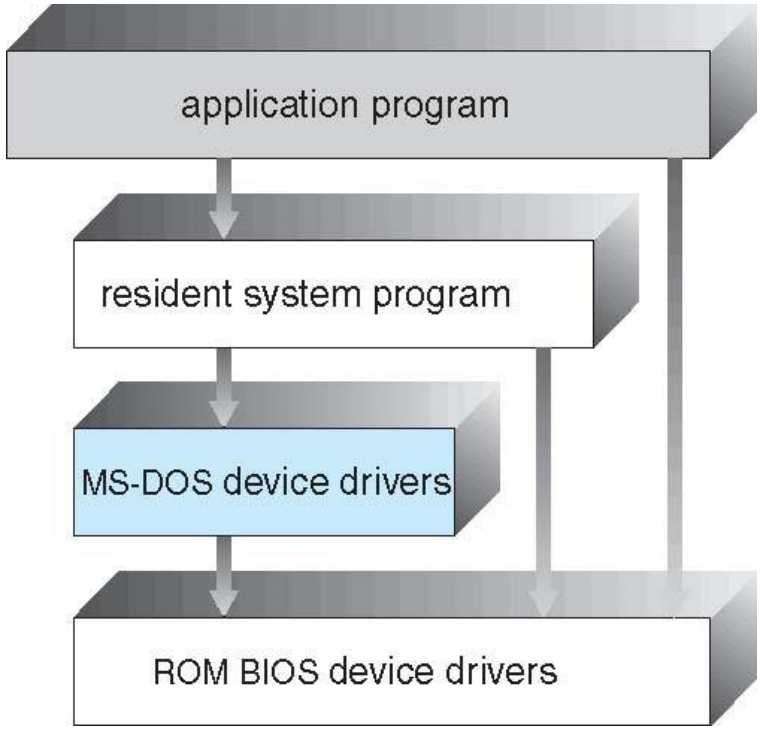
(a)

Operating System Structure

- General-purpose OS is very large program
- Various ways to structure ones
  - Simple structure – MS-DOS
  - More complex -- UNIX
  - Layered – an abstraction
  - Microkernel -Mac

Simple Structure -- MS-DOS

- MS-DOS – written to provide the most functionality in the least space
  - Not divided into modules



Non Simple Structure -- UNIX

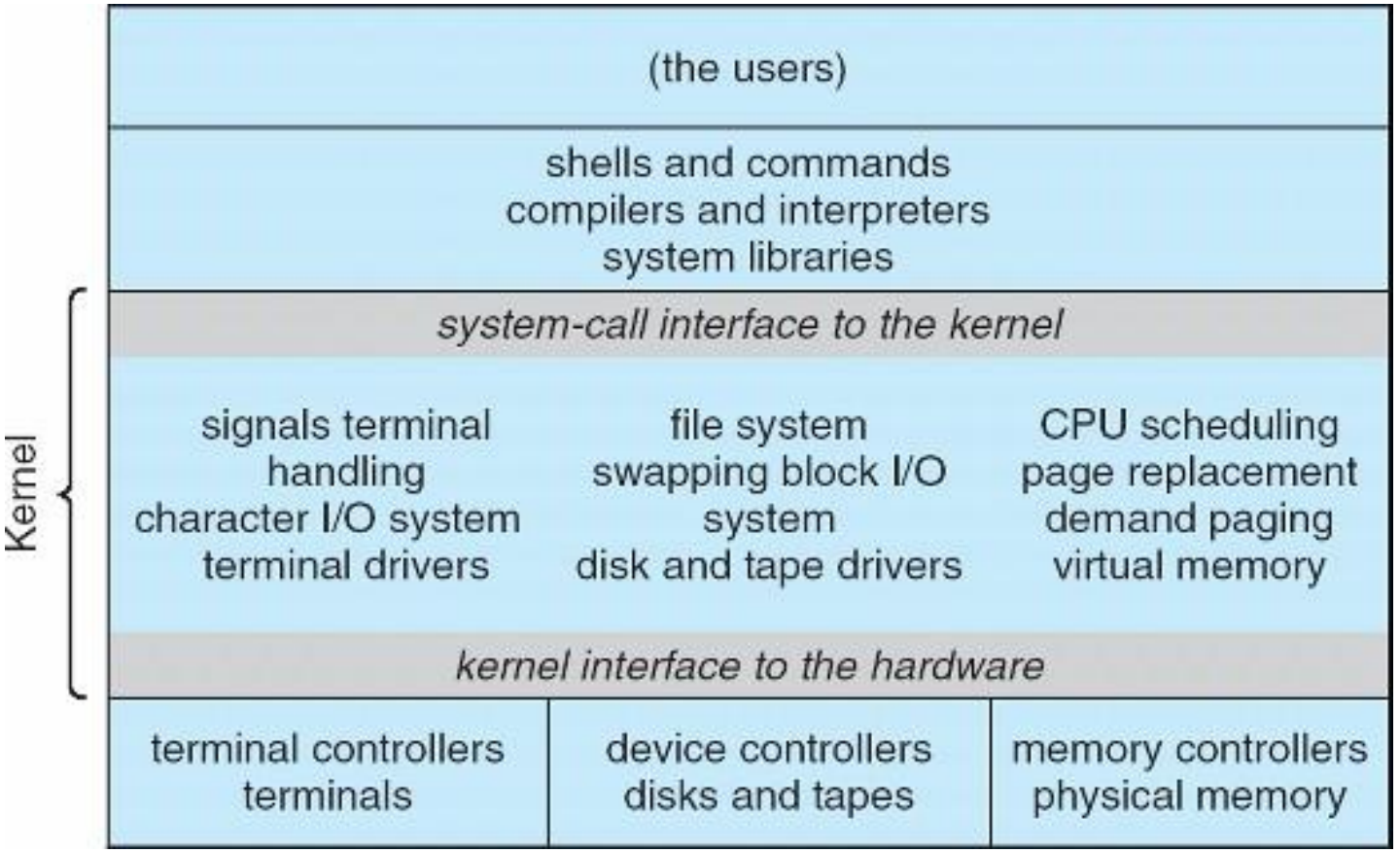
UNIX – limited by hardware functionality

The UNIX OS consists of two separable parts:

- Systems programs
- The kernel

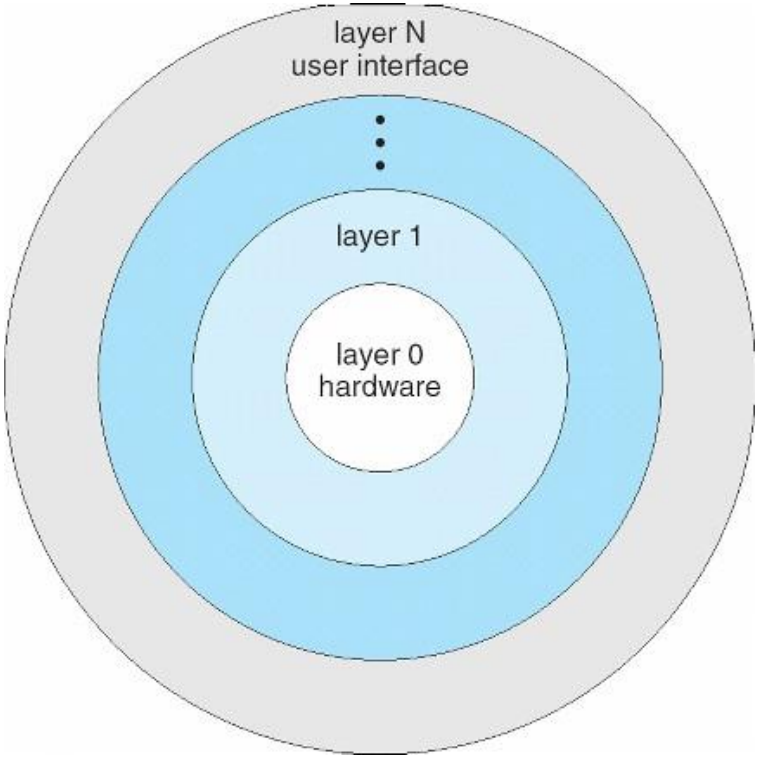
Traditional UNIX System Structure

Beyond simple but not fully layered



Layered Approach

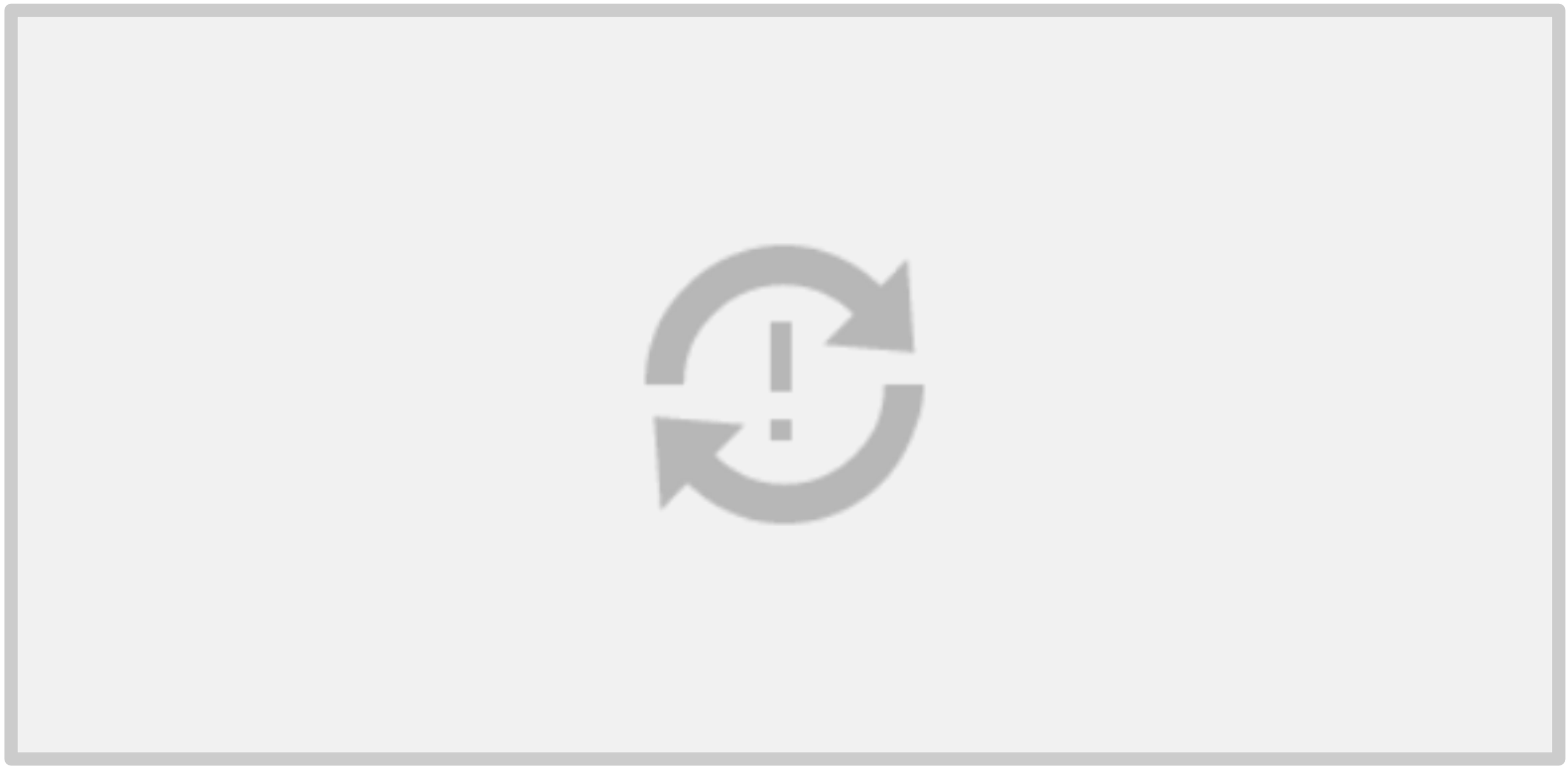
- The operating system is divided into a number of layers (levels)
- Each layer is built on top of lower layers.
- The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.
- With modularity, layers are selected, each layer uses functions (operations) and services of only lower-level layers



Microkernel System Structure

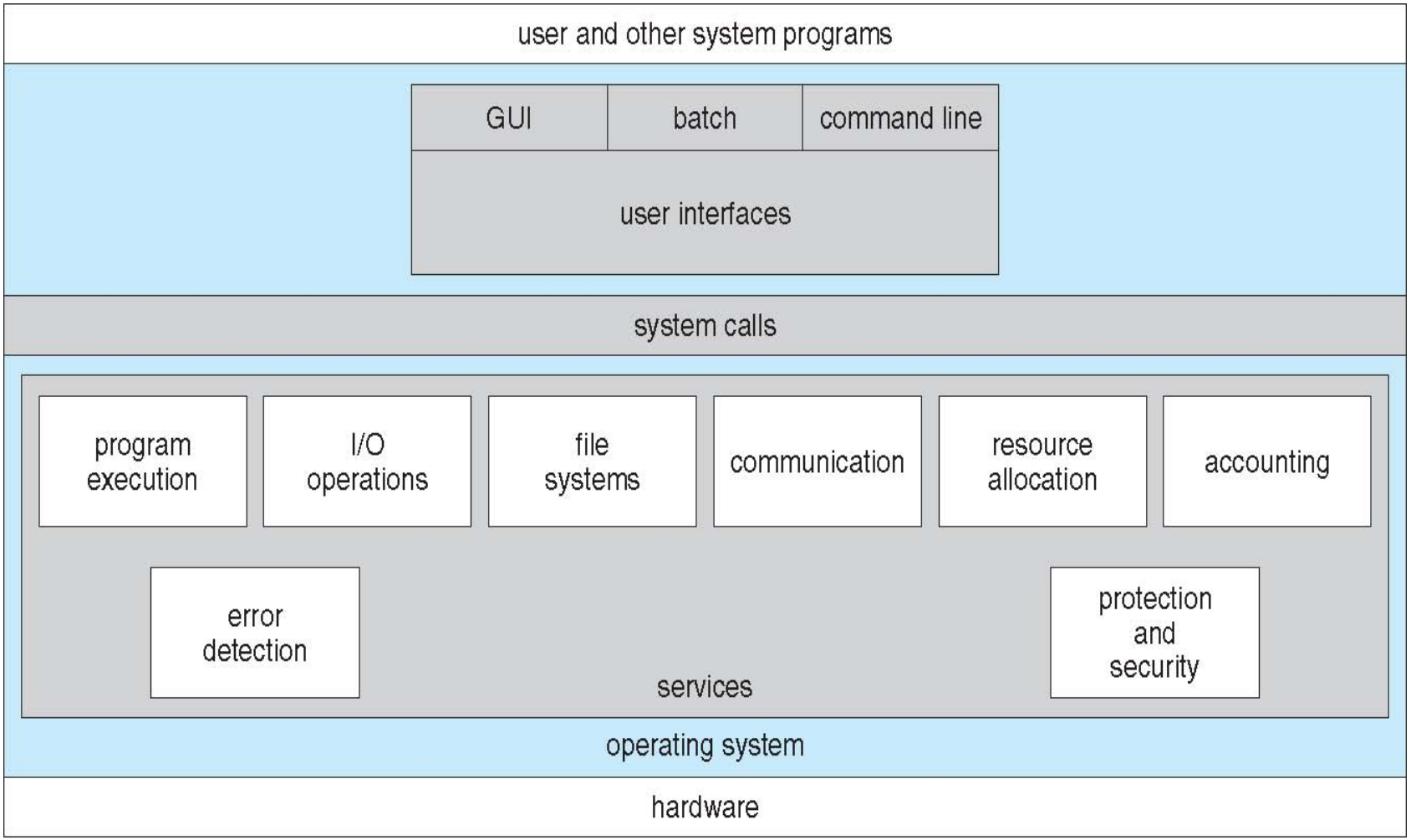
- Communication takes place between user modules using **message passing**
- Example of **microkernel: Mach**
  - Mac OS X kernel (**Darwin**) partly based on Mach
- Benefits:
  - Easier to extend a microkernel
  - Easier to port the operating system to new architectures
  - More reliable (less code is running in kernel mode)
  - More secure
- Disadvantage:
  - Performance overhead of user space to kernel space communication

Microkernel System Structure



Operating System Services

- An operating system provides an **environment for the programs to run.**
- It provides certain services to programs



Operating System Services

- Operating-system services provides functions that are helpful to the user:

  - **User interface** - Almost all operating systems have a user interface (UI)

    Varies between Command-Line (CLI), Graphics User Interface (GUI).

  - **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)

Operating System Services

- **I/O operations** - A running program may require I/O, which may involve a file or an I/O device.

- **File-system manipulation** - read and write files and directories, create and delete them, search them, list file Information, permission management.

Operating System Services

- **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 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)

- **Resource allocation –** OS must ensure allocation of resources to all programs running.

  Many types of resources - such as **CPU cycle time, main memory, and file storage**, **I/O devices**

Operating System Services

- **Accounting -** To keep track of which users use how much and what kinds of

computer resources.

- **Protection and Security -**

Protection involves **ensuring that all access to system resources is controlled**

Security of the system from outsiders requires **user authentication**, extends to defending external I/O devices from invalid access attempts

Any Query ???