Introduction to Operating System

Unit #1

Marwadi University

Department of Computer Engineering

Operating System
Sem 4
3140702
5 Credits

Prof. Chetan Chudasama

Course Outcomes

After completion of this course, student will be able to

- Analyze the structure of OS and basic architectural components involved in OS design
- Compare and contrast various CPU scheduling algorithms.
- Evaluate the requirements for the process synchronization and co-ordination in contemporary operating system.
- Analyze various algorithms for memory management, I/O management and security aspects of operating system.
- Write shell scripts in Unix/Linux O.S and write simple programs using kernel system calls. Also understand virtualization concept.

Table of Content

Table of Content

- Definition
  What is an OS?
  Where does the OS fit in?
  Flow of Communication
  Functions of Operating system
  Different views of OS
  Goals/objective of OS

Definition: What is an Operating System ?

A modern computer consists of

- One or more processors
- Main memory
- Disks
- Printers
- Various input/output devices.
- who manages(control) these hardwares?????

Definition: What is an Operating System ?

A modern computer consists of

- Managing all these varied components requires a layer of software – the **Operating System (OS).**

Definition: What is an Operating System ?

Definition

- An Operating System is a program that acts as an intermediary/interface between a **user** of a computer and the computer **hardware**.

Definition

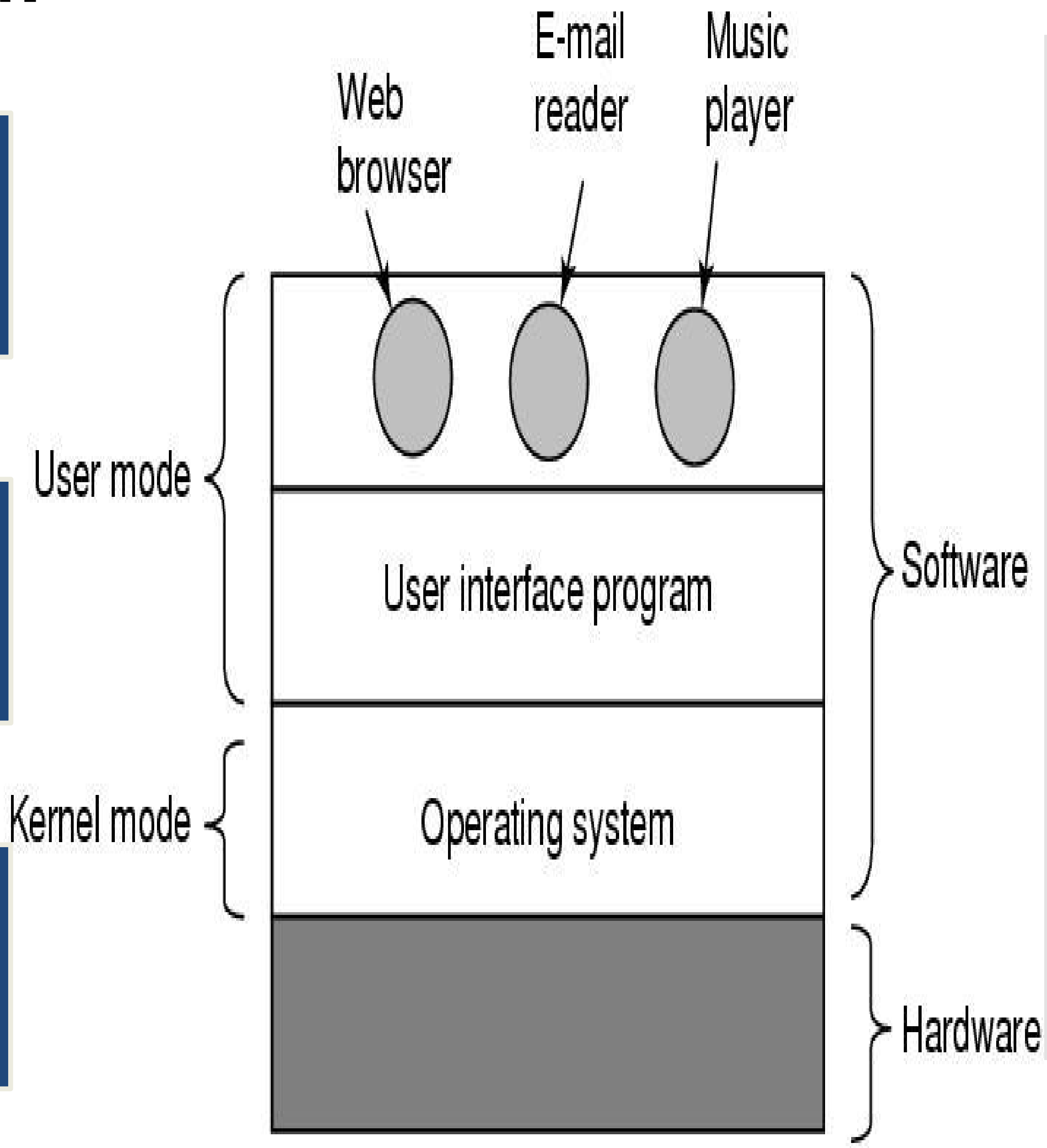
- It manages hardware resources and provides various service to users
- It provides a user-friendly environment in which a user may *easily* develop and execute programs.
- Otherwise, hardware knowledge would be mandatory for computer programming.
- So, it can be said that an OS *hides the complexity* of hardware from uninterested users.

Definition: Where does the OS fit in?

As show in the figure hardware is at bottom, consist of chips, boards, disks.

On the top of hardware is the software.

Computer have two modes of operations: user mode and kernel mode.

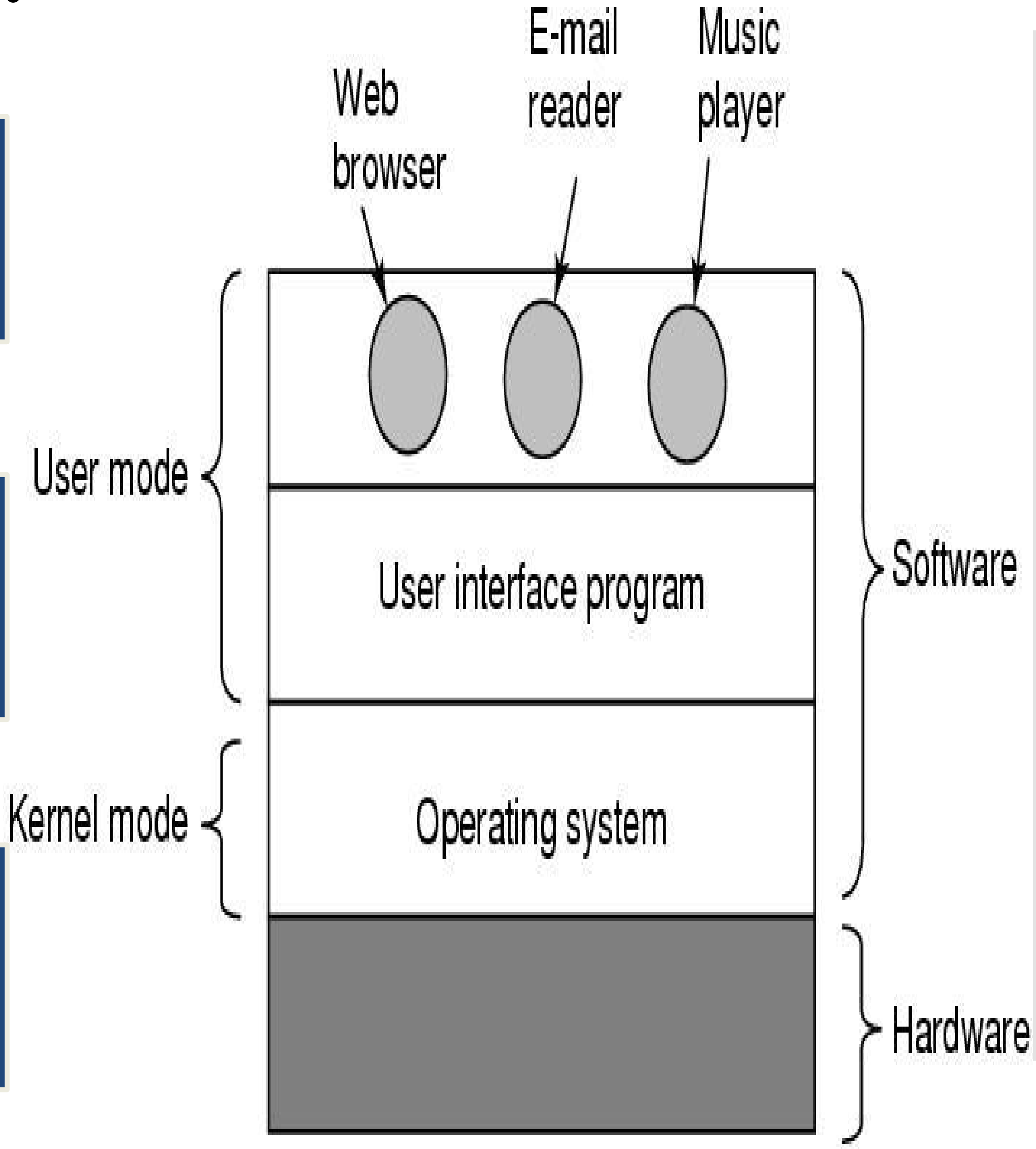


Definition: Where does the OS fit in?

OS is most fundamental software and runs in kernel mode.

Here OS have complete access to all the hardware and can *execute any instruction* on the machine.

Rest of the software runs in user mode.

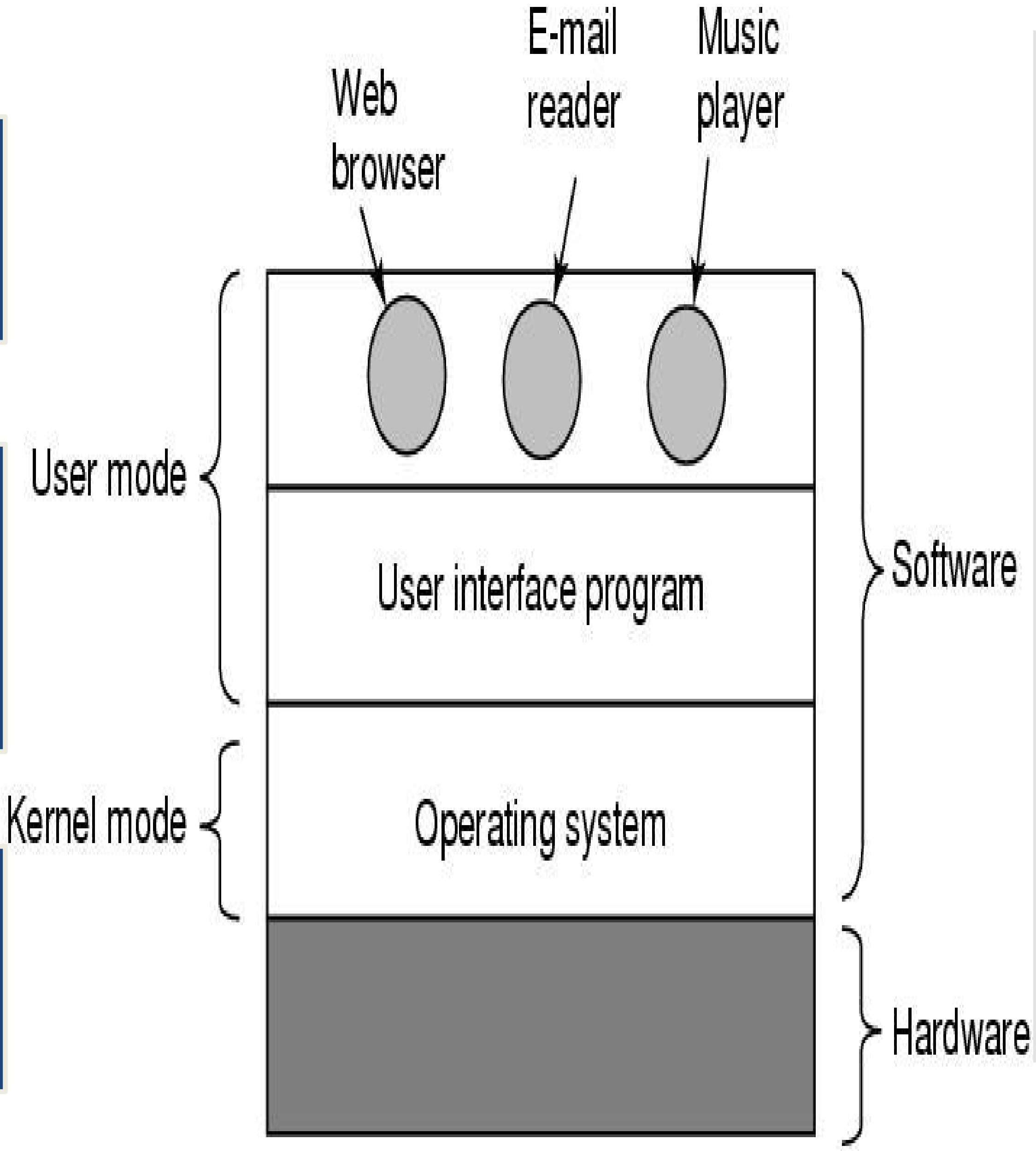


Definition: Where does the OS fit in?

Kernal mode

Here OS have complete access to all the hardware and can *execute any instruction* on that a machine is capable of executing.

has high privileged(rights)

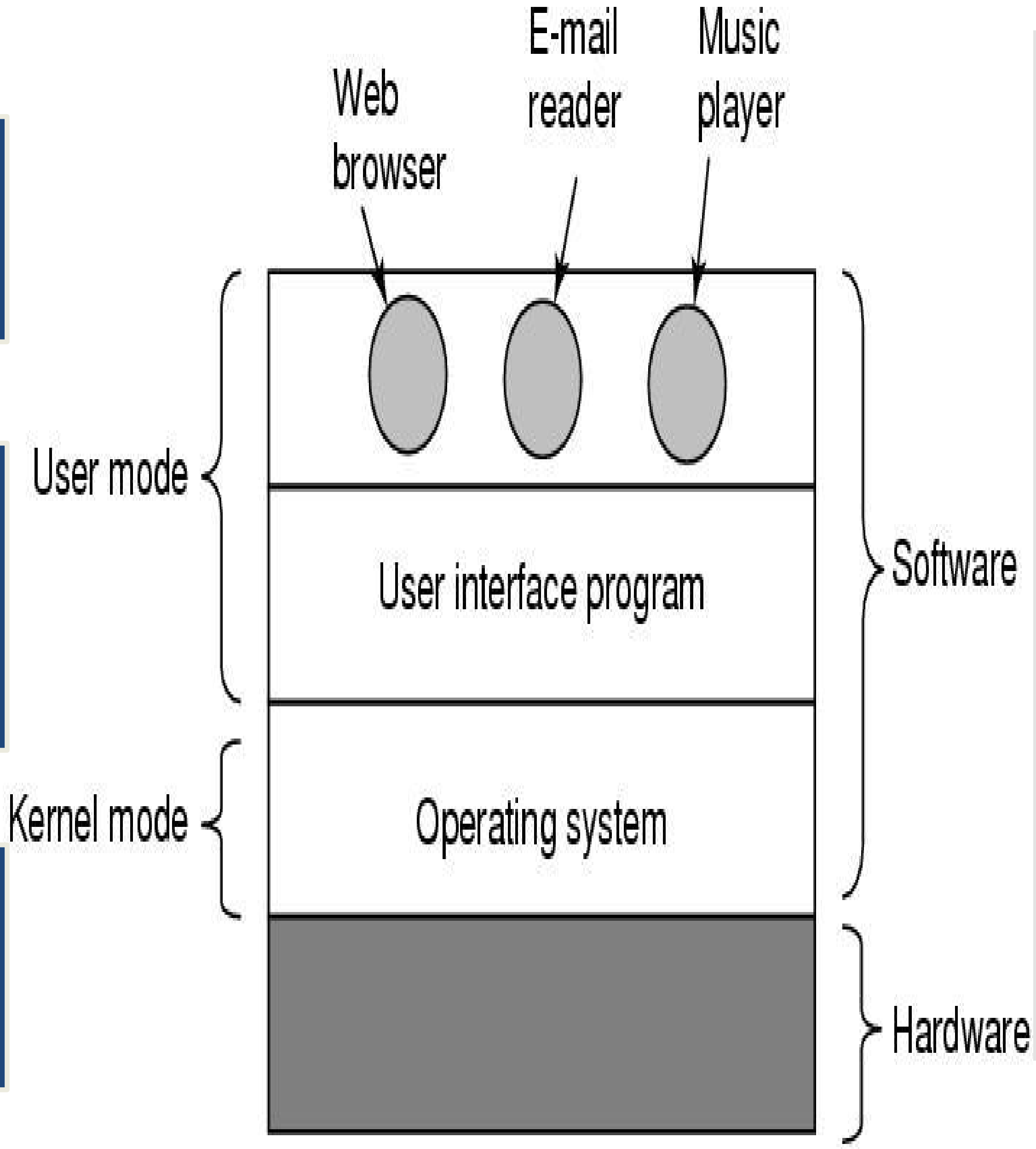


Definition: Where does the OS fit in?

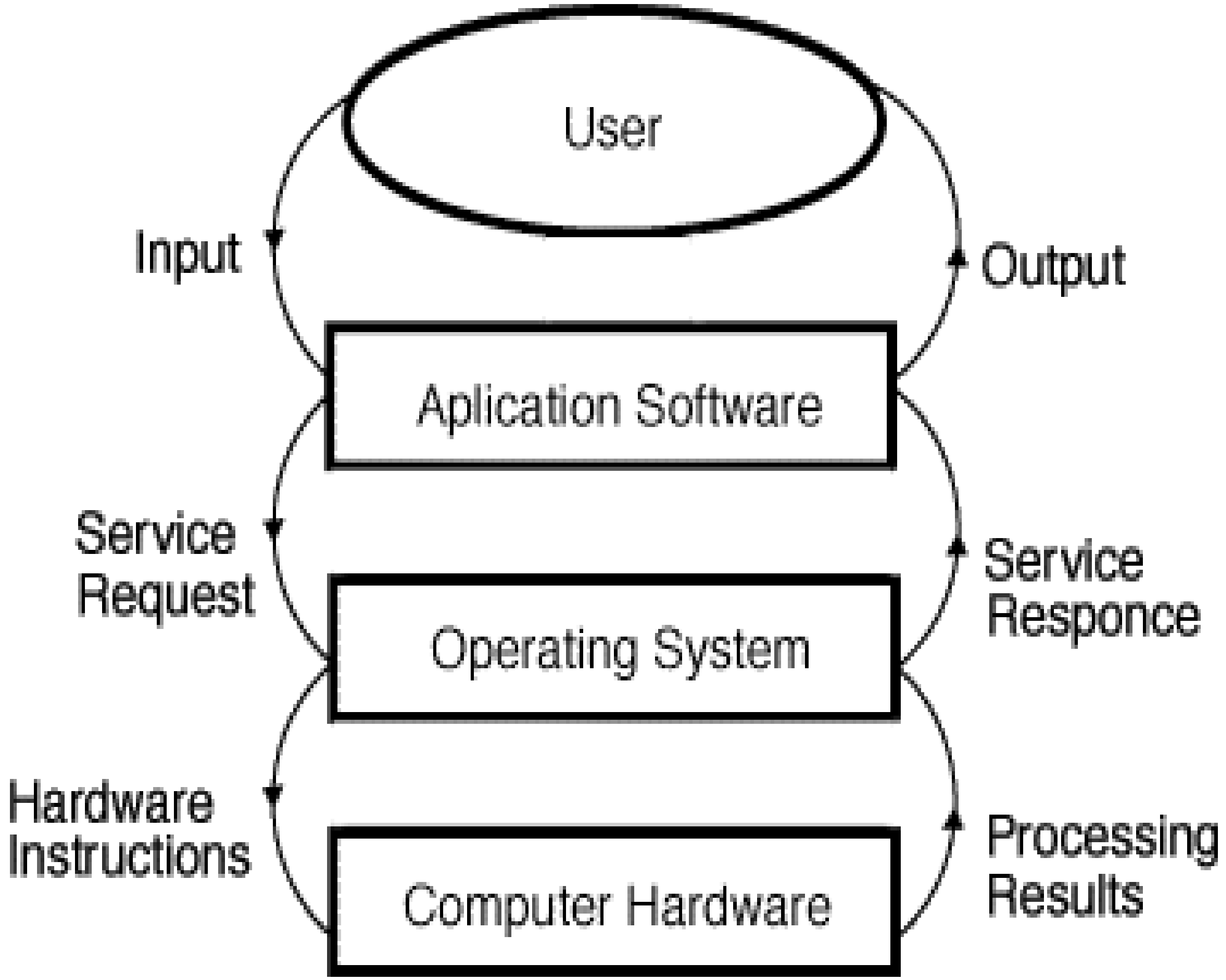
User mode

can *execute only subset(few) of the machine instruction.*

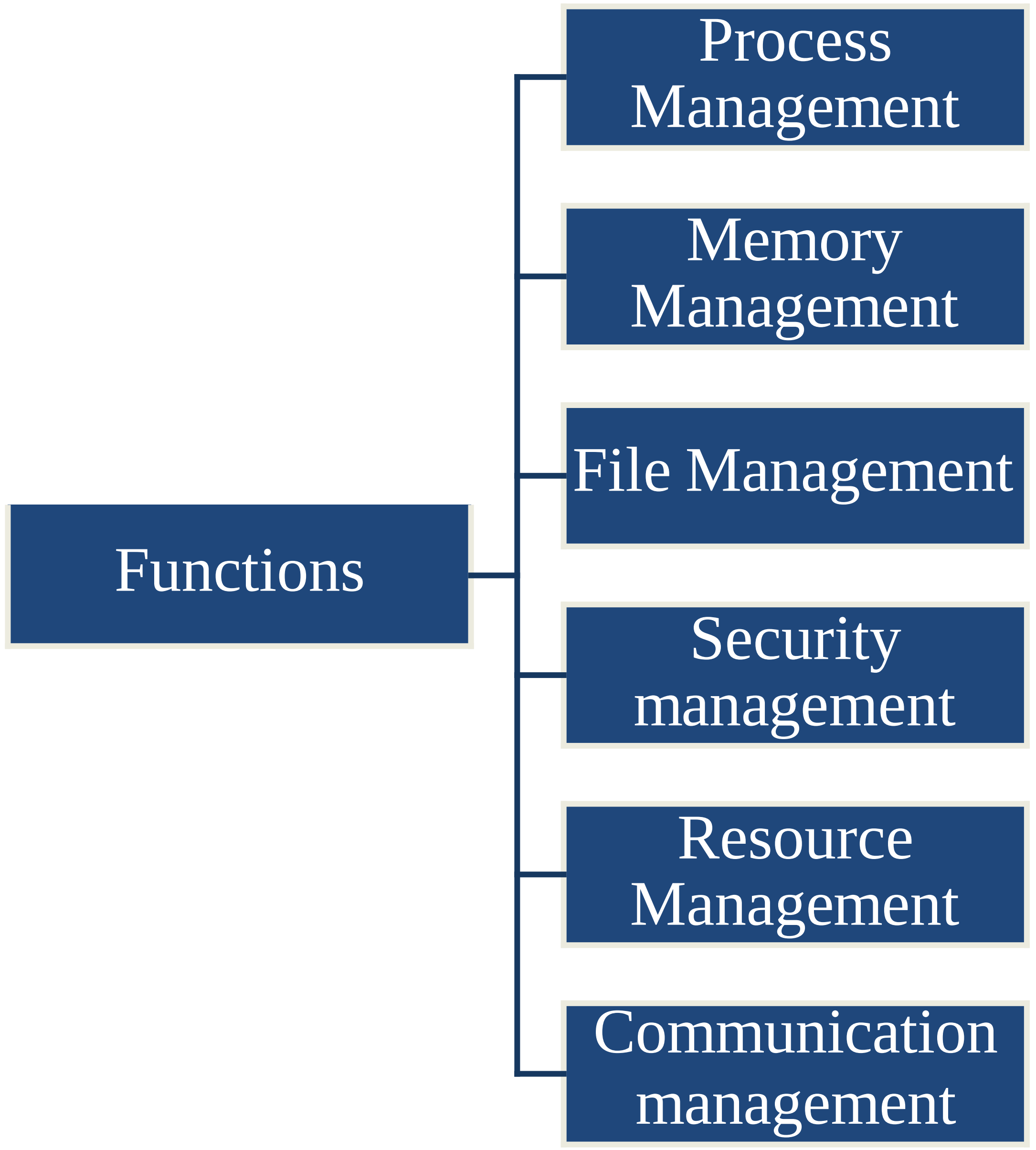
less privileged(rights)



Definition: Flow of Communication



Definition: Functions of Operating system



Definition: Functions of Operating system

Process management

- *By process management OS manages many kinds of activities:*
  - All process from start to shut down i.e. open, save, copy, install, print.
  - Creation and deletion of user and system processes.

Definition: Functions of Operating system

Memory Management

- *The major activities of an operating regard to memory-management are:*
  - Decide which process are loaded into memory when memory space becomes available.
  - Allocate and deallocate memory space as needed.

Definition: Functions of Operating system

File management

- *The file management system allows the user to perform such tasks:*
  - Creating files and directories
  - Renaming files
  - Coping and moving files
  - Deleting files

Definition: Functions of Operating system

Security Management

- *By security management OS manages many tasks such as:-*
  - Alert messages
  - Virus protection
  - Dialogue boxes
  - Firewall
  - Passwords

Definition: Functions of Operating system

Resource Management

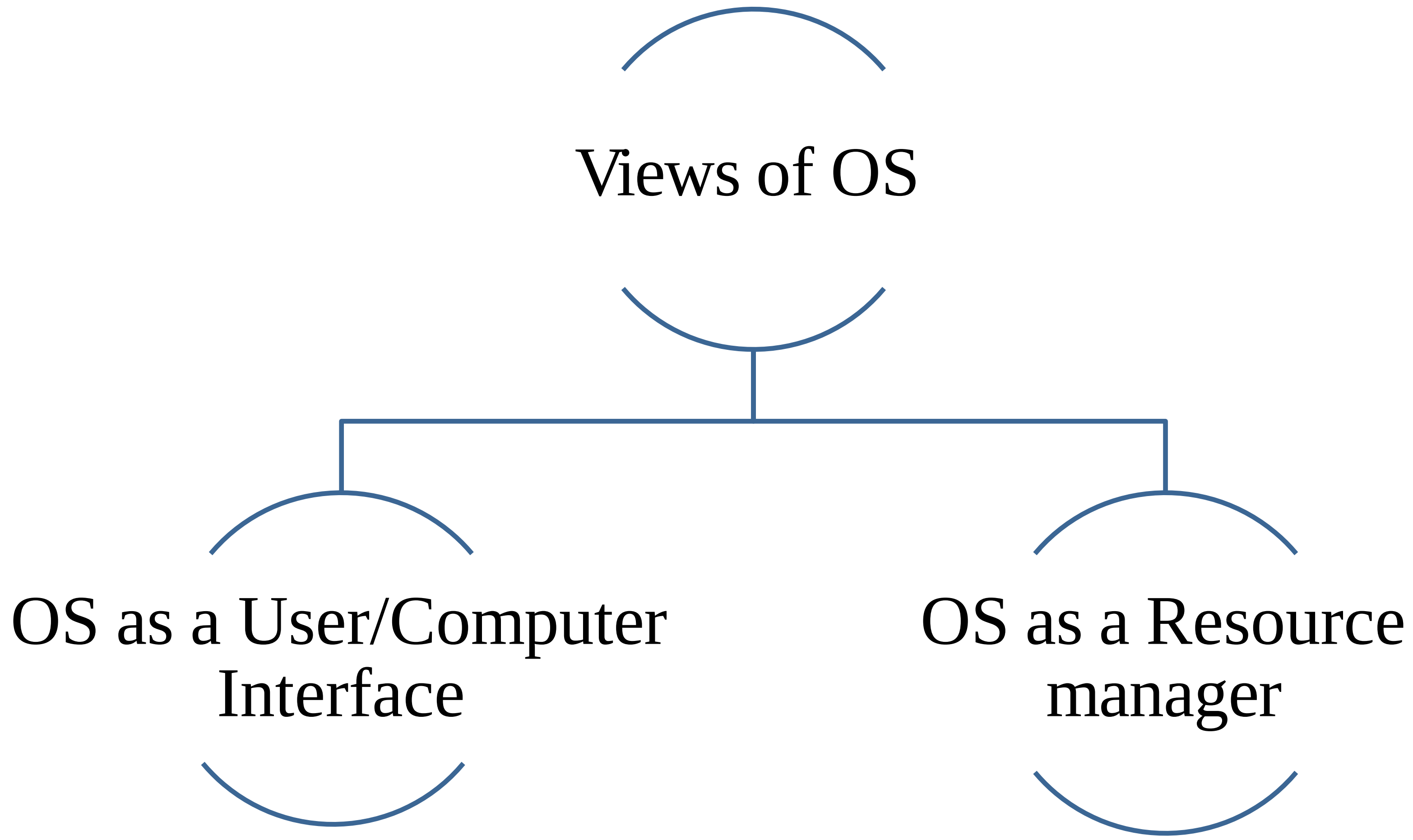
- *To manage Resources OS perform many tasks such as:*
  - Install drivers required for input and output, memory, power.
  - Coordination among peripherals.

Definition: Functions of Operating system

Communication Management

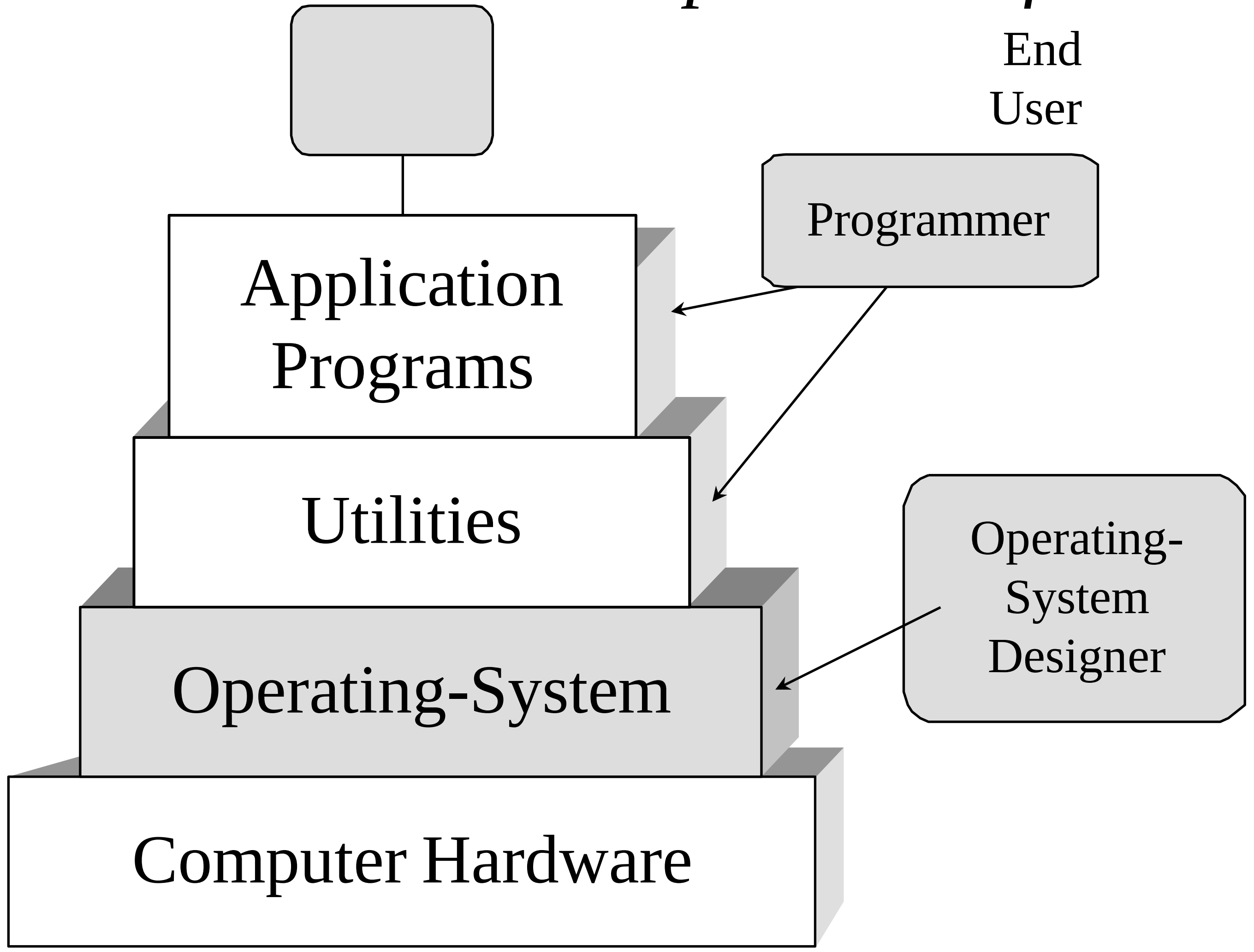
- *For proper coordination OS performs communication management:*
  - Communication between *user-application software-hardware.*
  - One computer to another via LAN or WAN.

Definition: Different views of OS



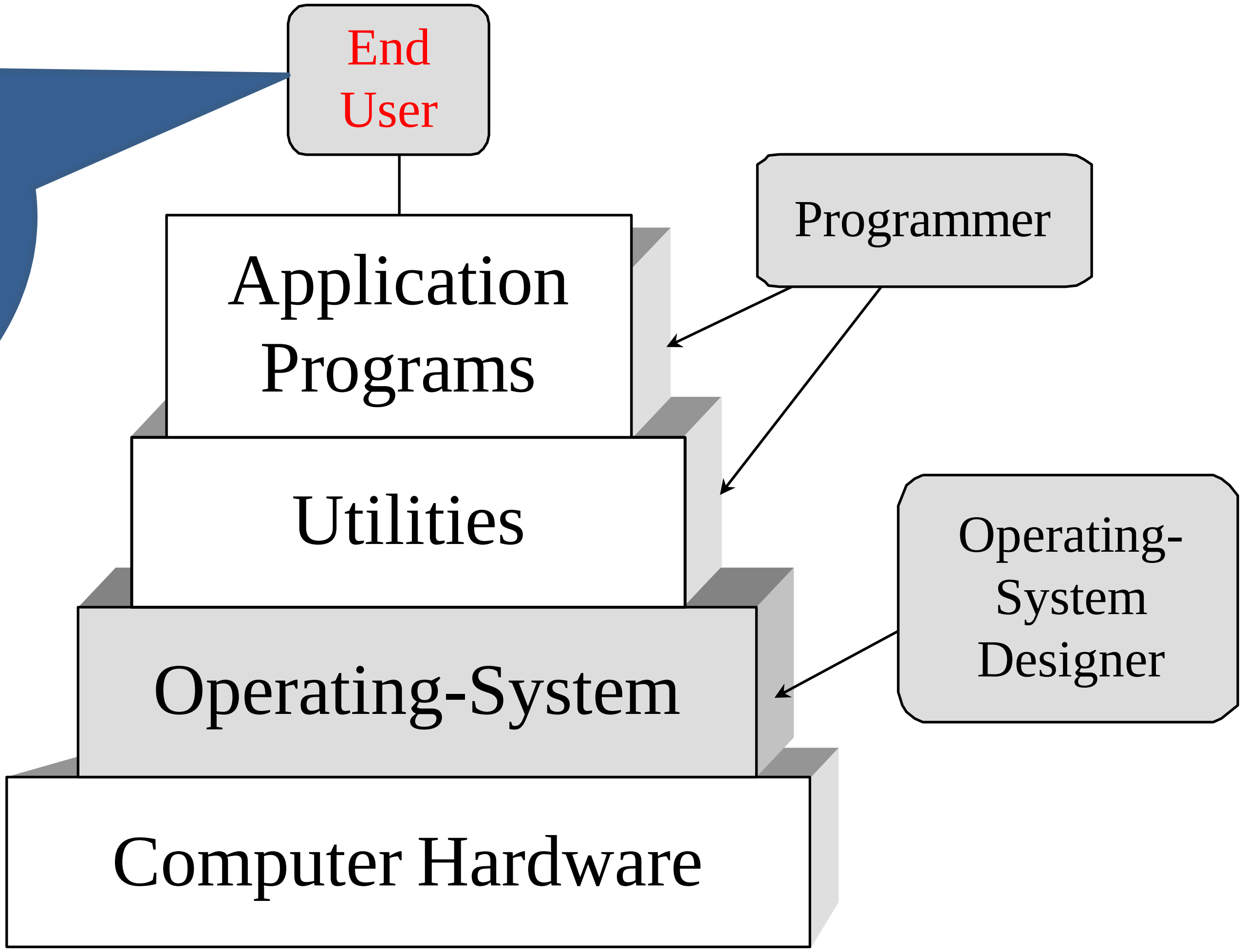
Definition: Different views of OS

- *OS as a User/Computer Interface:*



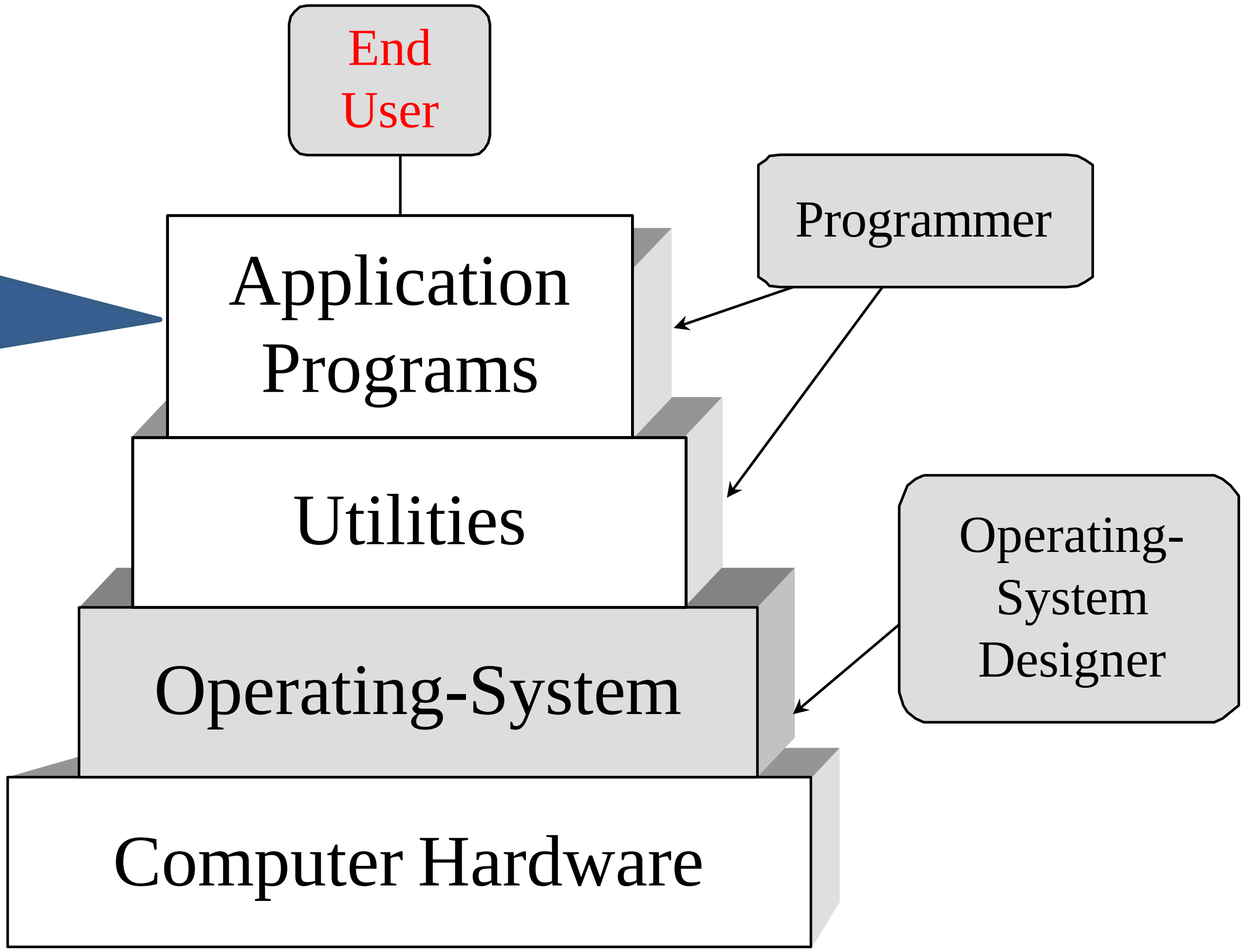
Different views of OS: OS as a User/Computer

End user only see the computer system
in terms of set of application, he/she is not
concern with the details of the computer hardware.



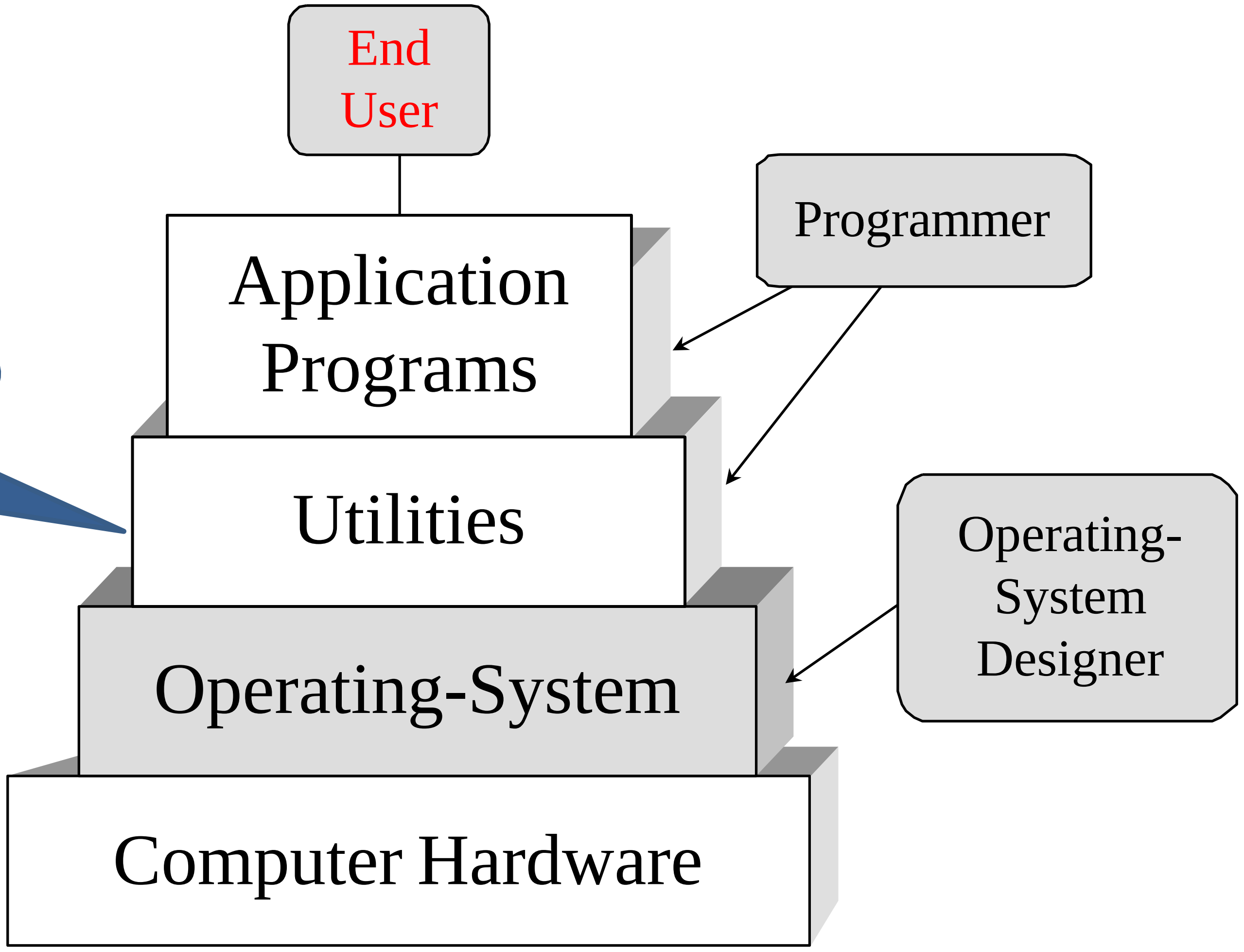
Different views of OS: OS as a User/Computer Interface

An application can be expressed in a programming language and is developed by an application programmer.

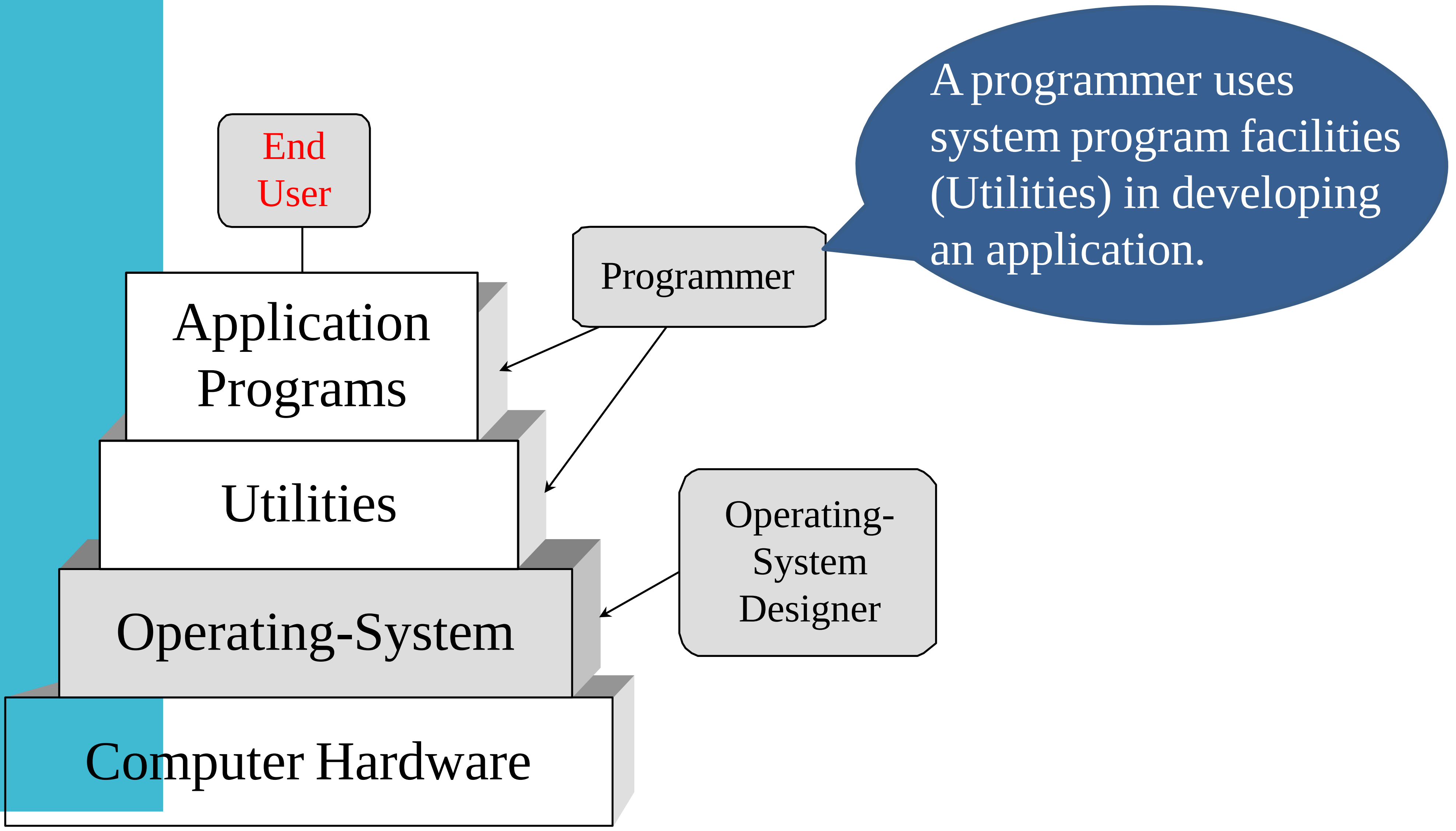


Different views of OS: OS as a User/Computer Interface

system program facilities are know as utilities

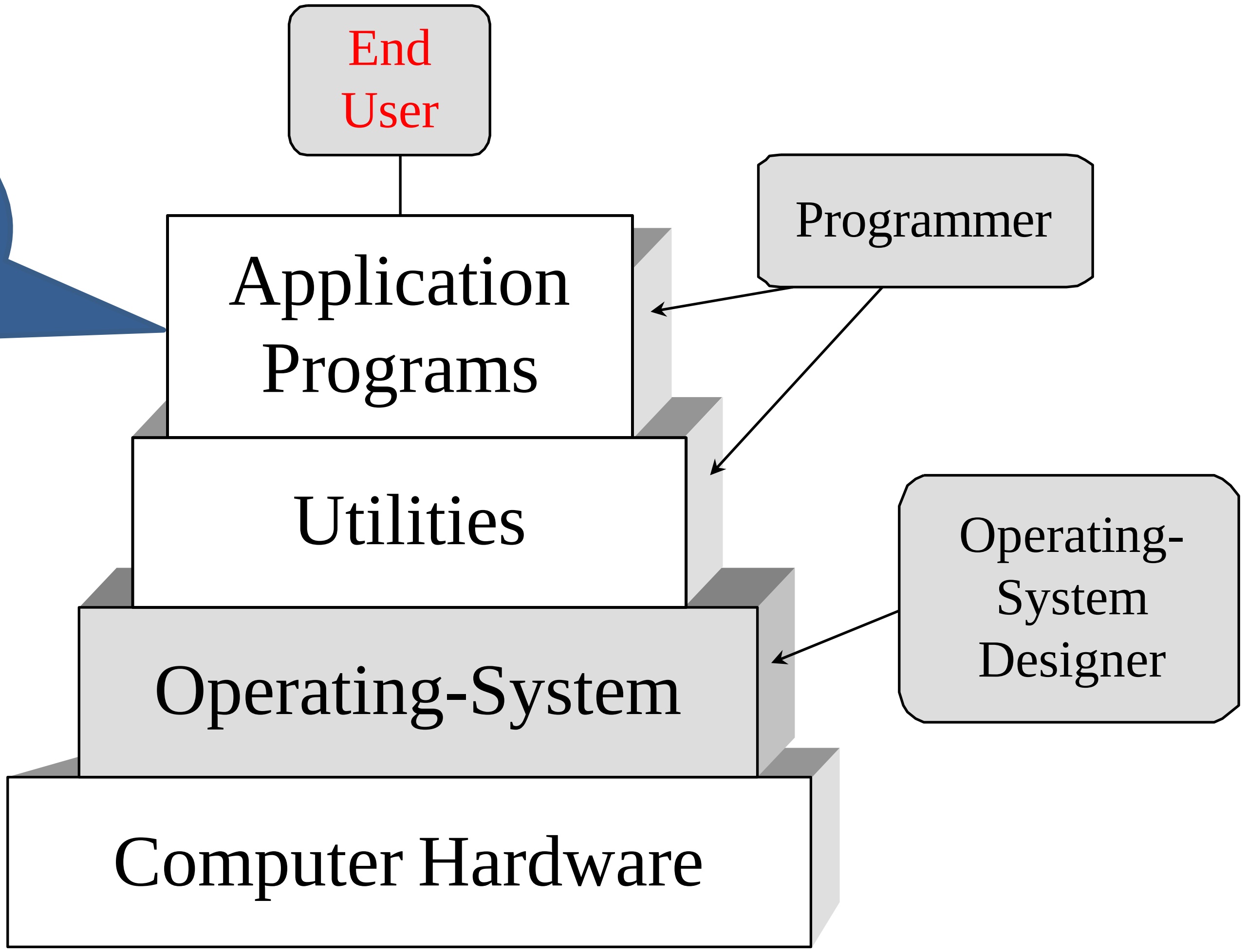


Different views of OS: OS as a User/Computer Interface


End User
Application Programs
Utilities
Operating-System
Computer Hardware
Programmer
Operating-System Designer
A programmer uses system program facilities (Utilities) in developing an application.

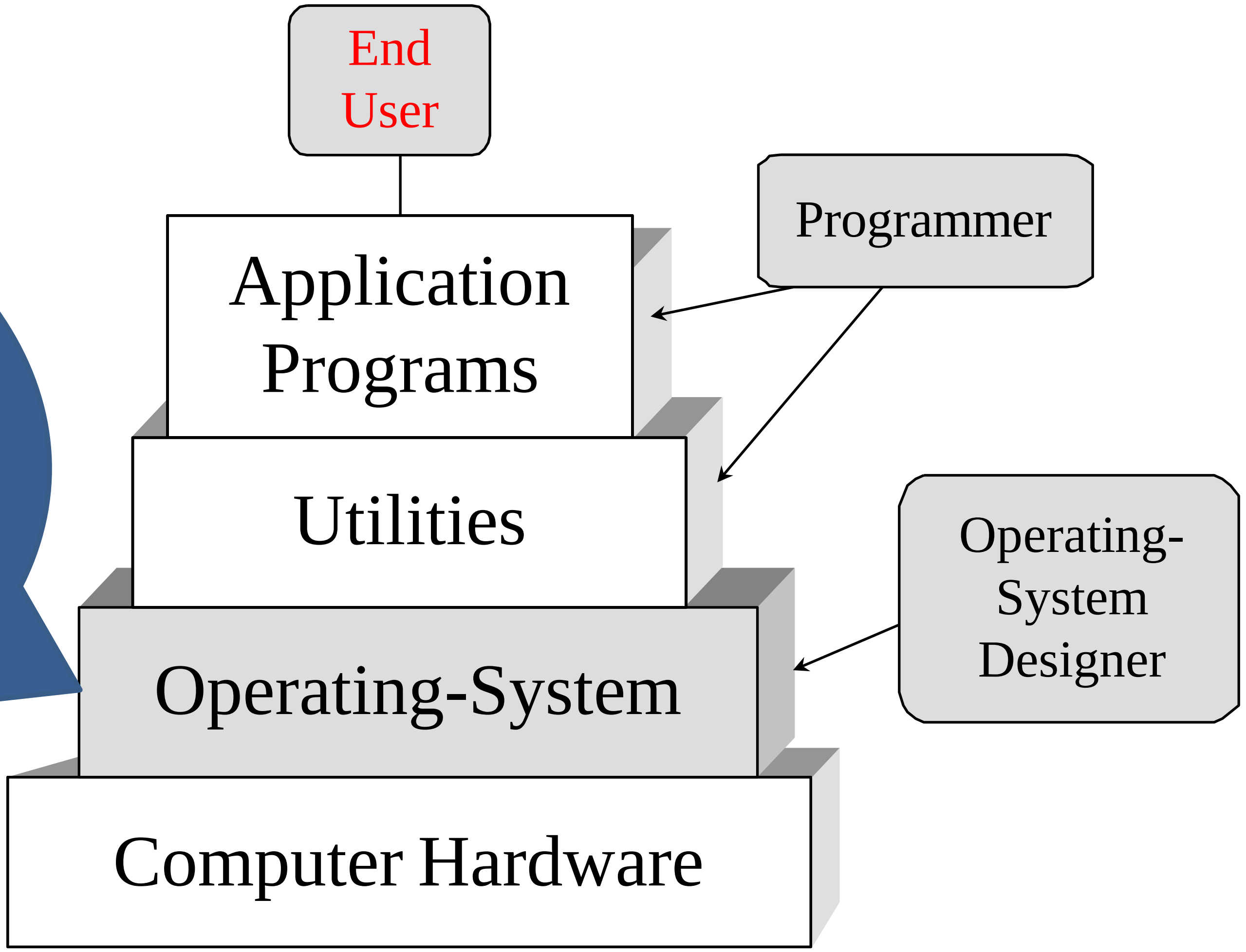
Different views of OS: OS as a User/Computer Interface

When application is running it can invoke the utilities to perform certain functions.



Different views of OS: OS as a User/Computer Interface

OS is made up of most important collection of system programs
OS act as mediator.
Make easier for the **Programmer** and **application program** to access the facilities and services of OS.



Different views of OS: OS as a Resource manager

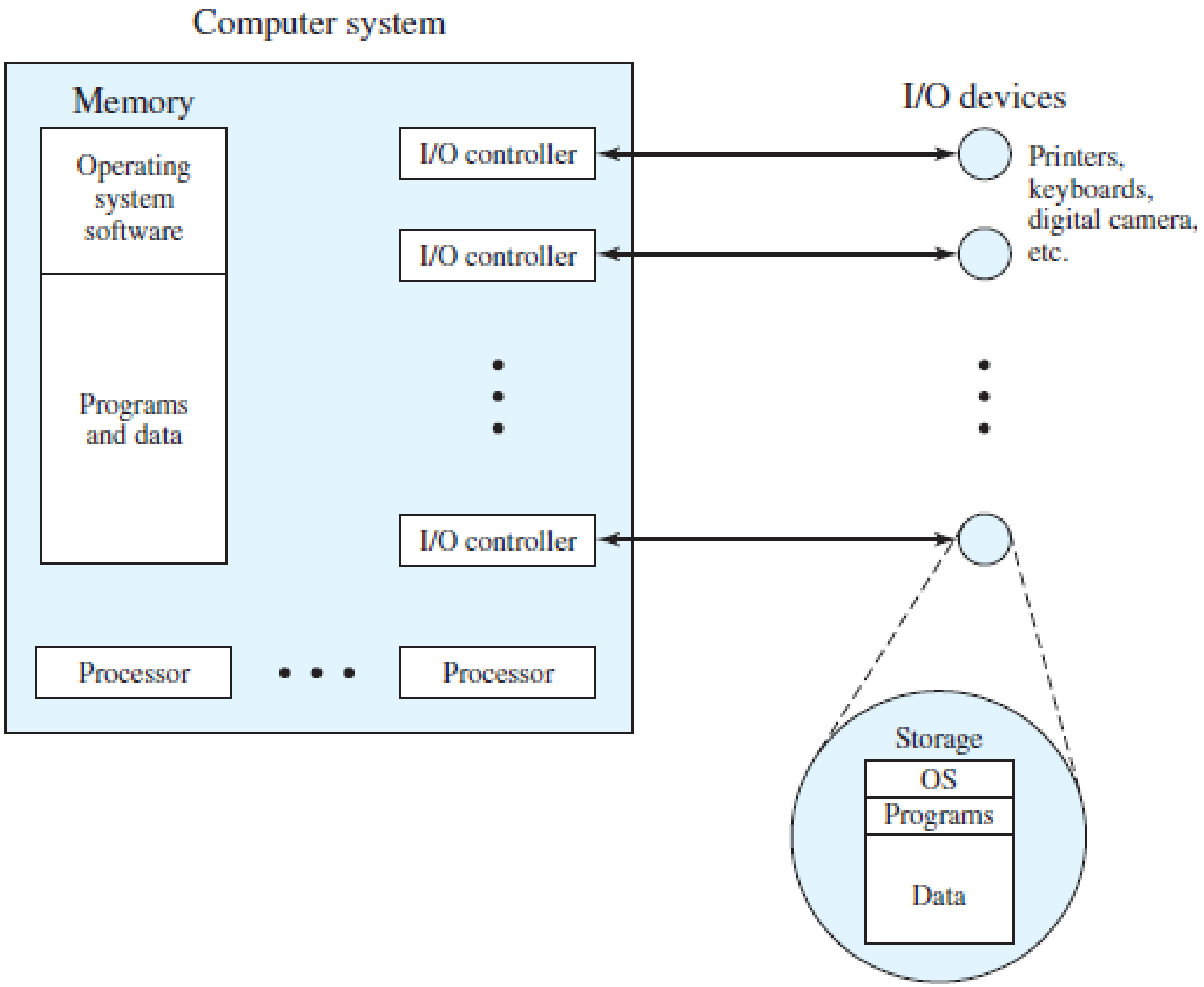
OS as a Resource manager (OS from system view)

- A computer is a set of resources which preform the function such as store, process and transfer the data.
- If a computer system is used by multiple application(or users),then they will compete for these resources.
- OS is responsible for managing the resources to control this functions.

Different views of OS: OS as a Resource manager

- Some portion of the OS is loaded in the main memory other part of the main memory may contain the user programs and data.
- Allocation of recourses is controlled jointly by OS and processor(I/O controller).
- OS decides:
  - when I/O device can be used by a program in execution
  - When program can access the files.
  - How much processor time to be devoted to particular process/ program.

Different views of OS: OS as a Resource manager



Definition: Goals/objective of OS

- Control/execute user/application programs.
- Make the computer system convenient to use.
- Ease the solving of user problems.
- Use the computer hardware in an efficient manner.

❖Generations of Operating System

Generations of Operating systems

- First Generation
- Second Generation
- Third Generation
- Fourth Generation

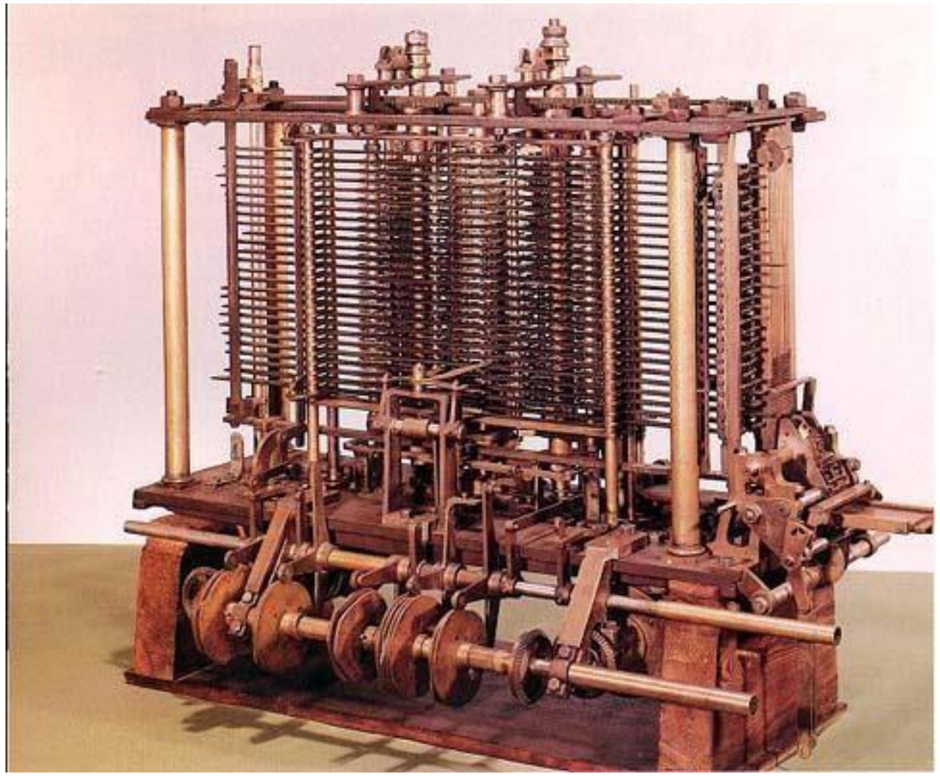
Generations of Operating systems: Zero generation

The history of operating system is linked with the history and development of various generations of computer systems.

The first true digital computer was designed by English mathematician Charles Babbage. He is known as Father of Digital Computer.

Computer had a Mechanical Design, It was slow and unreliable and they are known as “Analytical Engine”.

Generations of Operating systems: Zero generation


Babbage’s analytical engine

Generations of Operating systems: First generation

First Generation (1945-1955)

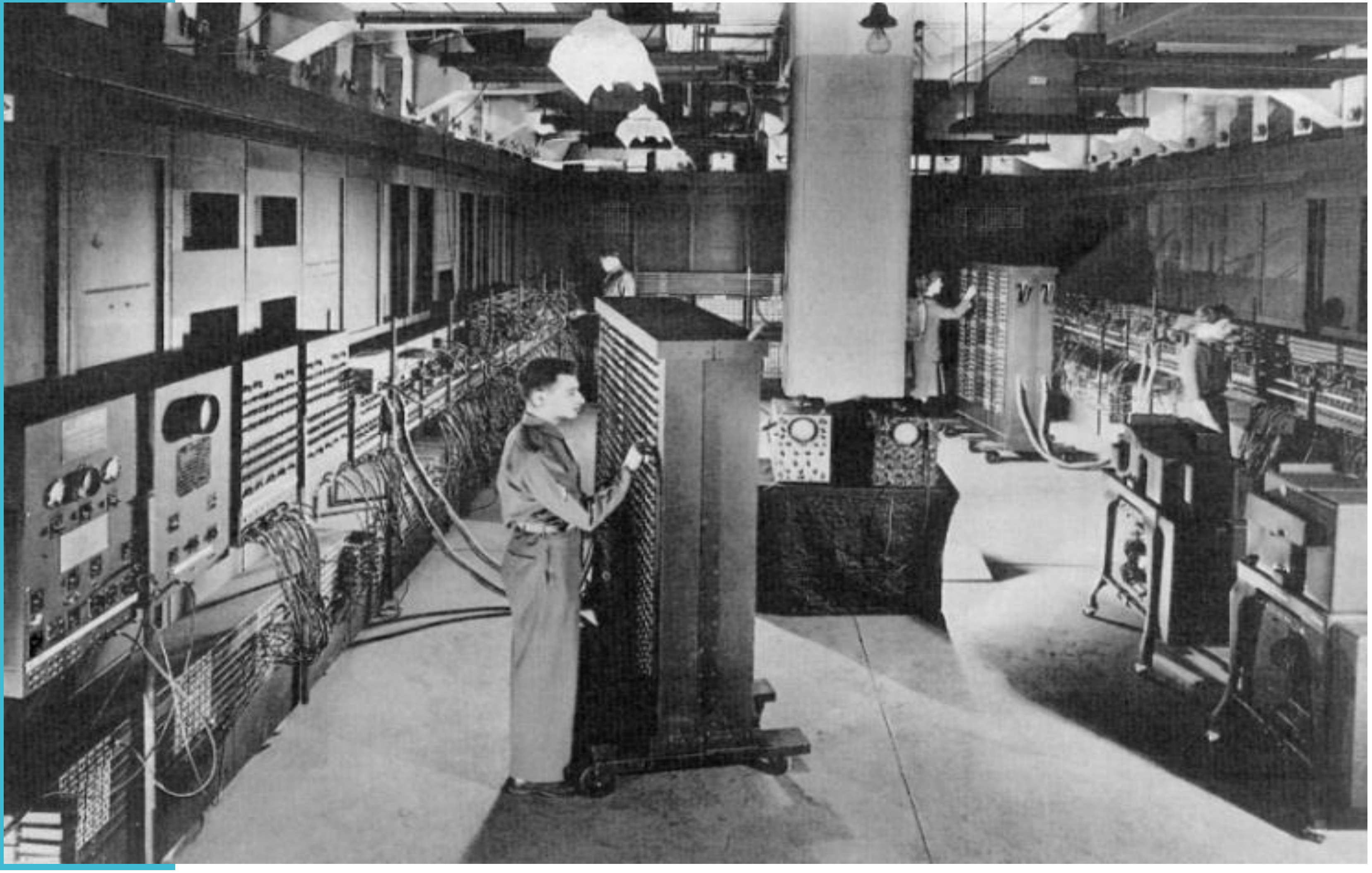
• Technology	: Vacuum Tubes
• Operating System	: Not present
• Language	: Machine language

Generations of Operating systems: First generation

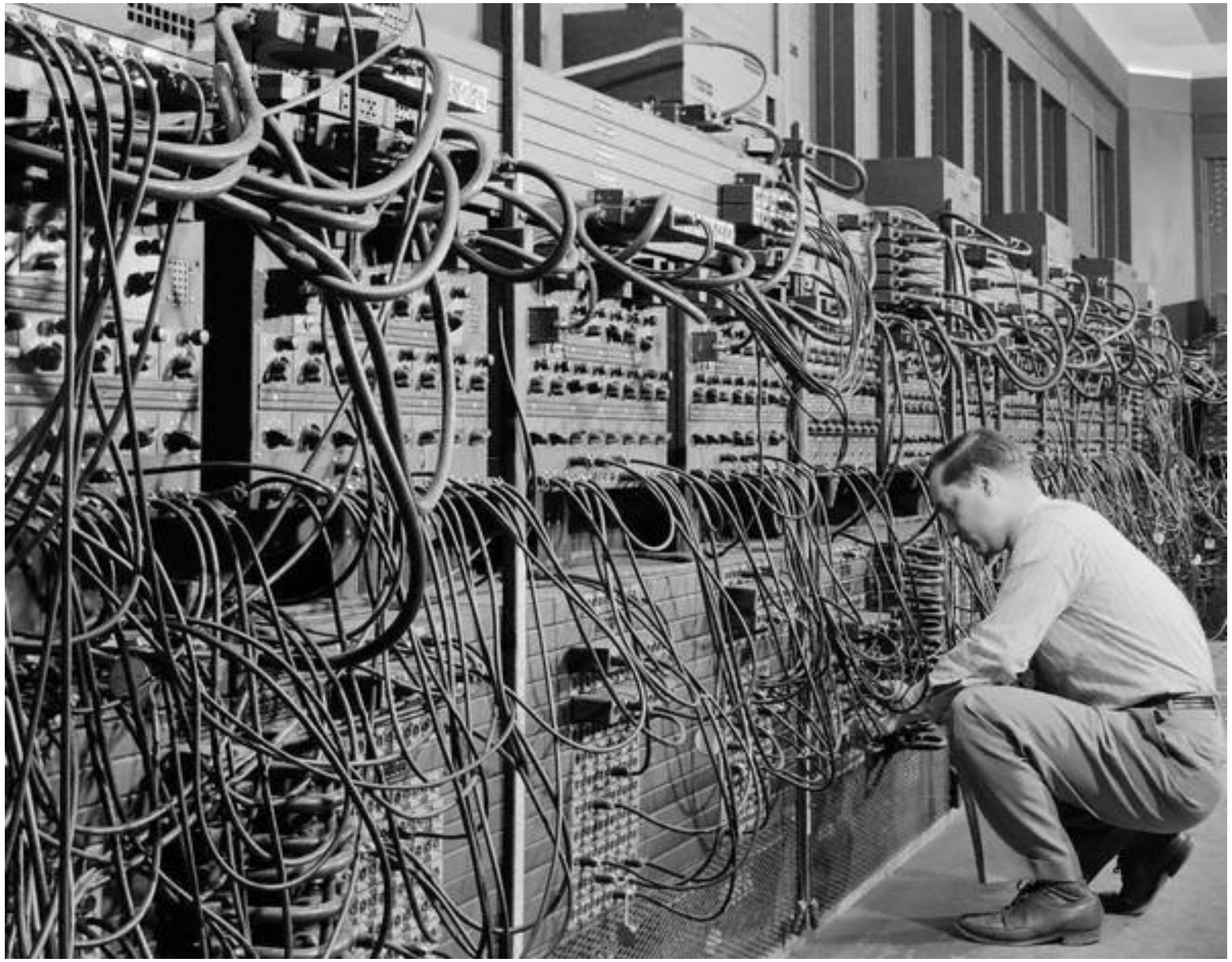
Mechanical Components are replaced by Electronic Components (Vacuum Tubes)

Vacuum Tubes

Generations of Operating systems: First generation

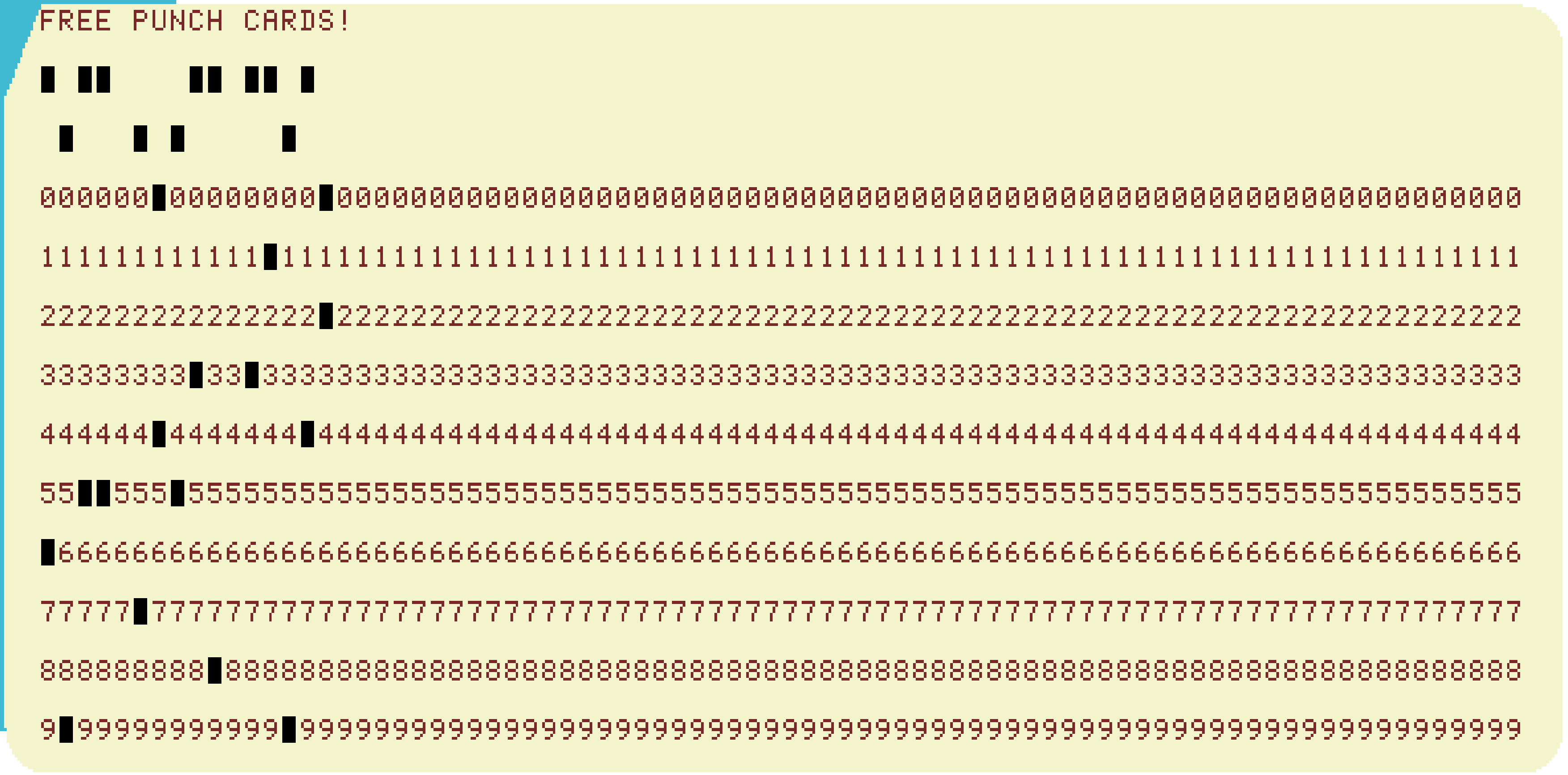


First Generation Computers



coding by cable connections (plug board)

punch card



Generations of Operating systems: First generation

Working

- Computer run one job at time. Programmers have to enter the plug board or punch card into the computer, run it, record the result. (might require rewiring!).

Problem

- lots of wasted computer time!
- CPU was idle during first and last steps
- Computers were ***very*** expensive!

Generations of Operating systems: First generation

Features of First Generation Computers

- They used vacuum tubes as main electronic component.
- They were large in size, slow in processing and had small storage capacity.
- They consumed lots of electricity and produced excessive heat.
- They were less reliable than later generation computers
- They used machine level language for programming

Generations of Operating systems: Second generation

Second Generation (1955-65)

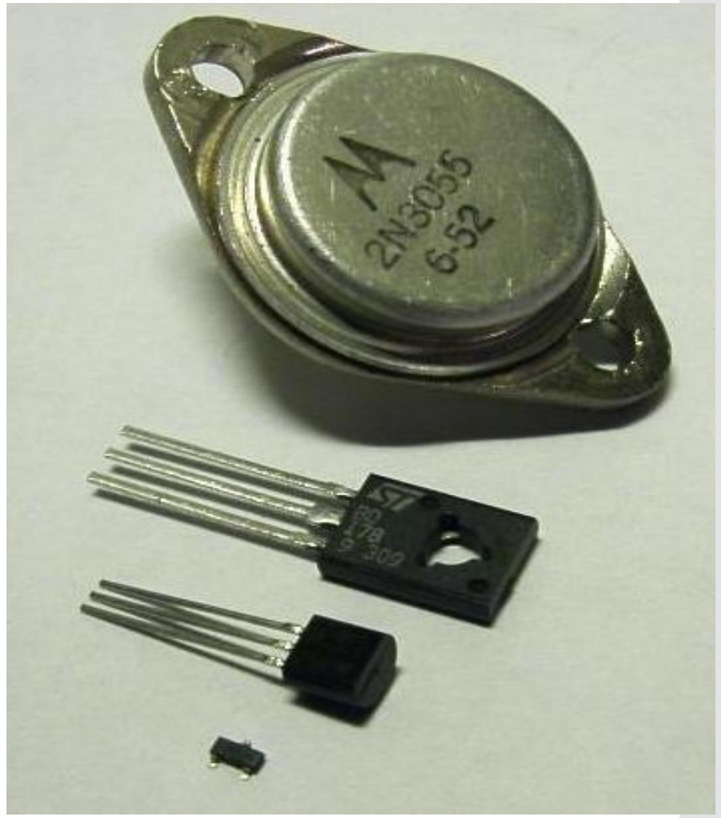
• Technology	: Transistors
• Operating system	: Present
• Language	: Assembly and High level language

Generations of Operating systems: Second generation

Around 1955, Transistors were introduced.

Operating systems were designed which is known as FMS (Fortran Monitor System) and IBMSYS.

Used assembly language. Used FORTRAN as high level language.

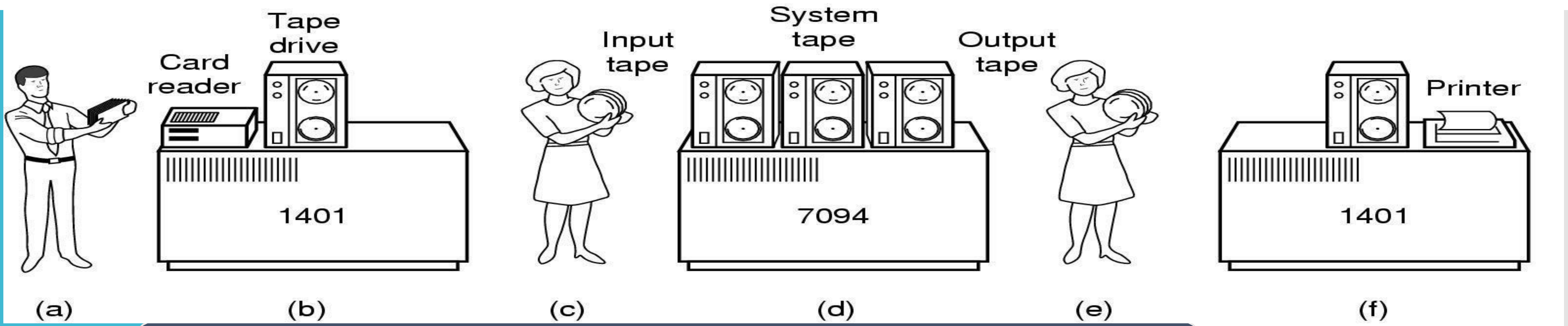

Transistors

Generations of Operating systems: Second generation

Batch system

- To reduce the time new methodology is adopted know as batch system.
- To execute the program two commuters were used IBM 1401 for reading cards, copying tapes, and printing output, and IBM 7094 for real computing (numerical calculation).

Generations of Operating systems: Second generation



Batch system: Working

- Collect a tray full of jobs and read them onto magnetic tape using small computer such as IBM 1401.
- Then magnetic tapes are mounted on a tape drive., operator load special program which read the first job from tape and run it.
- Output was written onto second tape, after each job finished, the OS automatically read next job from the tape.
- Output tape inserted into IBM 1401 for printing.

Generations of Operating systems: Second generation

Second Generation Computers

Generations of Operating systems: Second generation

Advantage:

- Able to compute scientific and engineering calculations.
- Cost and Size of a Computer is reduced.
- Programmers job was simplified.

Generations of Operating systems: Second generation

Features of Second Generation Computers

- Second generation computers used transistors as their main electronic component.
- These computers were much smaller, reliable and more powerful
- Apart from machine language, assembly language were developed and used in this generation
- Some high level languages like COBOL & FORTRAN were introduced towards the end of second generation
- Printers, tape storage, disk storage, memory were started from second generation of computers
- Processing speed improved to microseconds

Generations of Operating systems: Third generation

Third Generation(1965-80)

• Technology	: Integrated Circuits
• Operating Systems	: Present
• Language	: High level language

Third Generation Computers

Generations of Operating systems: Third generation

Need of third generation computers:

- Developing and maintaining two completely different computers (IBM 1401 and IBM 7094) was *expensive*.
- Bigger companies wanted machines that can run their programs *faster*.
- IBM solved both the problem by introducing System/360.

Generations of Operating systems: Third generation

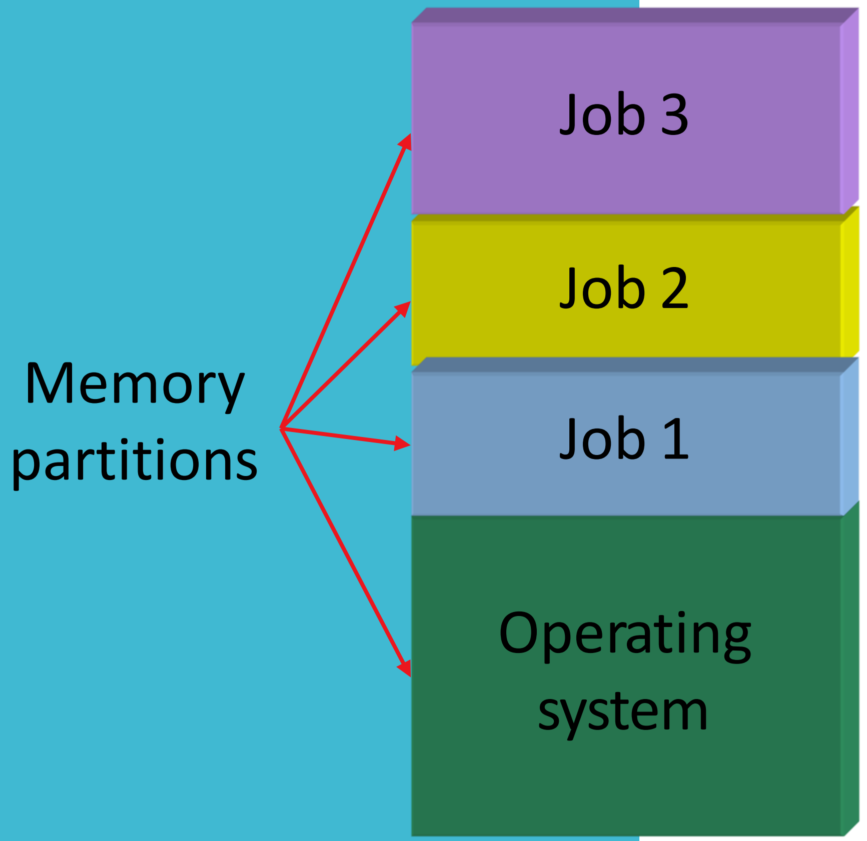
System/360

Generations of Operating systems: Third generation

Need of third generation computers:

- 2nd generation computer when current job paused to wait for I/O or tape then CPU simply set idle until the I/O finished.
- Thus some thing had to be done to avoid having CPU idle so much.
- The solution was multiprogramming.

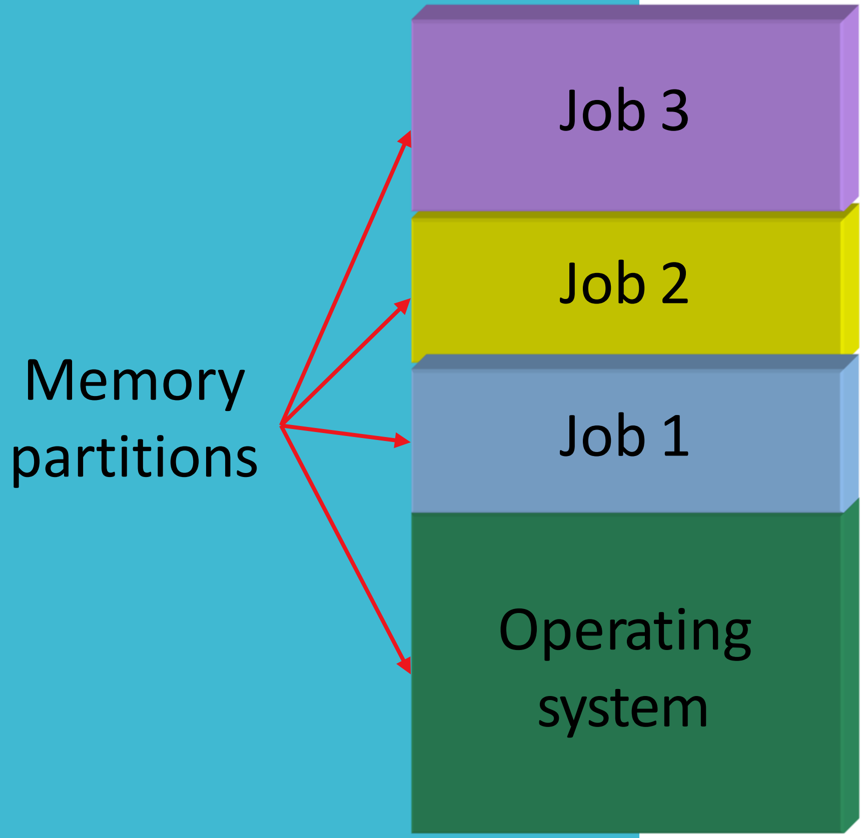
Generations of Operating systems: Third generation



Multiprogramming

- Partition the memory into several pieces with different job in each partition.
- While one job was waiting for I/O to complete, another job could use CPU.
- whenever running job finished, the OS load new job from the disk into the empty partition of memory. This is know as SPOOLING (Simultaneous Peripheral Operation On Line).

Generations of Operating systems: Third generation



- Still not interactive
  - User submit job.
  - Computer runs it.
  - User gets results after few minutes (hours, days) later.

Generations of Operating systems: Third generation

Timesharing

- Timesharing system is interactive.
- First general purpose time sharing system is CTSS (Compatible Time Sharing System).
- After the success of CTSS system, M.I.T , Bell labs and general Electrical decide to develop “computer utility”, a machine that supports hundreds of simultaneous time sharing users. This machine known as MULTICS (MULTiplexed Information and Computing Services).

Generations of Operating systems: Third generation

Another major development during the third generation was the *minicomputers*.

Generations of Operating systems: Third generation

Characteristics of Third Generation Computer

- Third Generation Computers were based on integrated circuit (IC) technology.
- Third Generation Computers were able to reduce computational time from microseconds to nanoseconds
- Third Generation Computers utilized operating systems to centrally control and made it *possible for multiple programs* to run on a computer
- *Users interacted* with computers through keyboards and mouse rather than traditional punched cards and printers.

Generations of Operating systems: Third generation

Characteristics of Third Generation Computer

- Third Generation Computers devices *consumed less power and generated less heat.* In some cases, air conditioning was still required.
- The size of Third Generation Computers was smaller as compared to previous computers

Generations of Operating systems: Fourth generation

Fourth Generation(1980-90)

Technology	: LSI
Operating systems	: Present
Language	: High level language

Generations of Operating systems: Fourth generation

- Large Scale Integrated Circuits (LSIC's) were introduced.
- Intel came out with 8080, first microcomputer with a disk.
- Kindall wrote disk based OS called CP/M (Control Program for Microcomputers).
- Tim Paterson had developed OS know as Disk Operating System (DOS). Microsoft revised system and was known as MS-DOS.

LSIC

Generations of Operating systems: Fourth generation

Later on around 1960, Engelbart invented GUI (Graphical User Interface) with windows icons, menus, and mouse.

One day Steve Jobs co-founder of apple saw GUI and instantly realized its potential value. Job then embark on building Apple with GUI. This project leads to Lisa, which was too expensive and failed commercially.

Job second attempt was Apple Macintosh, was huge success because its more user friendly and less expensive.

For about 10 years 1985 to 1995 , Windows was just graphical environment on top of MS-DOS.

Generations of Operating systems: Fourth generation

- Then in 1995 slightly modified version released windows 95, that incorporated many OS features.
- Then slightly changed version released know as windows 98. both 95 and 98 are 16-bit intel assembly language.
- Another Microsoft OS is windows NT (New Technology), its 32-bit system.
- Version 5 of Windows NT is renamed as Windows 2000.

Generations of Operating systems: Fourth generation

Another version of win 98 is windows ME (Millennium Edition).

In 2001 slightly upgraded version of windows 2000 known as windows XP was released.

Then in January 2007, Microsoft finally released the successor to win XP known as Vista. It came with new GUI , Aero, many upgraded user program.

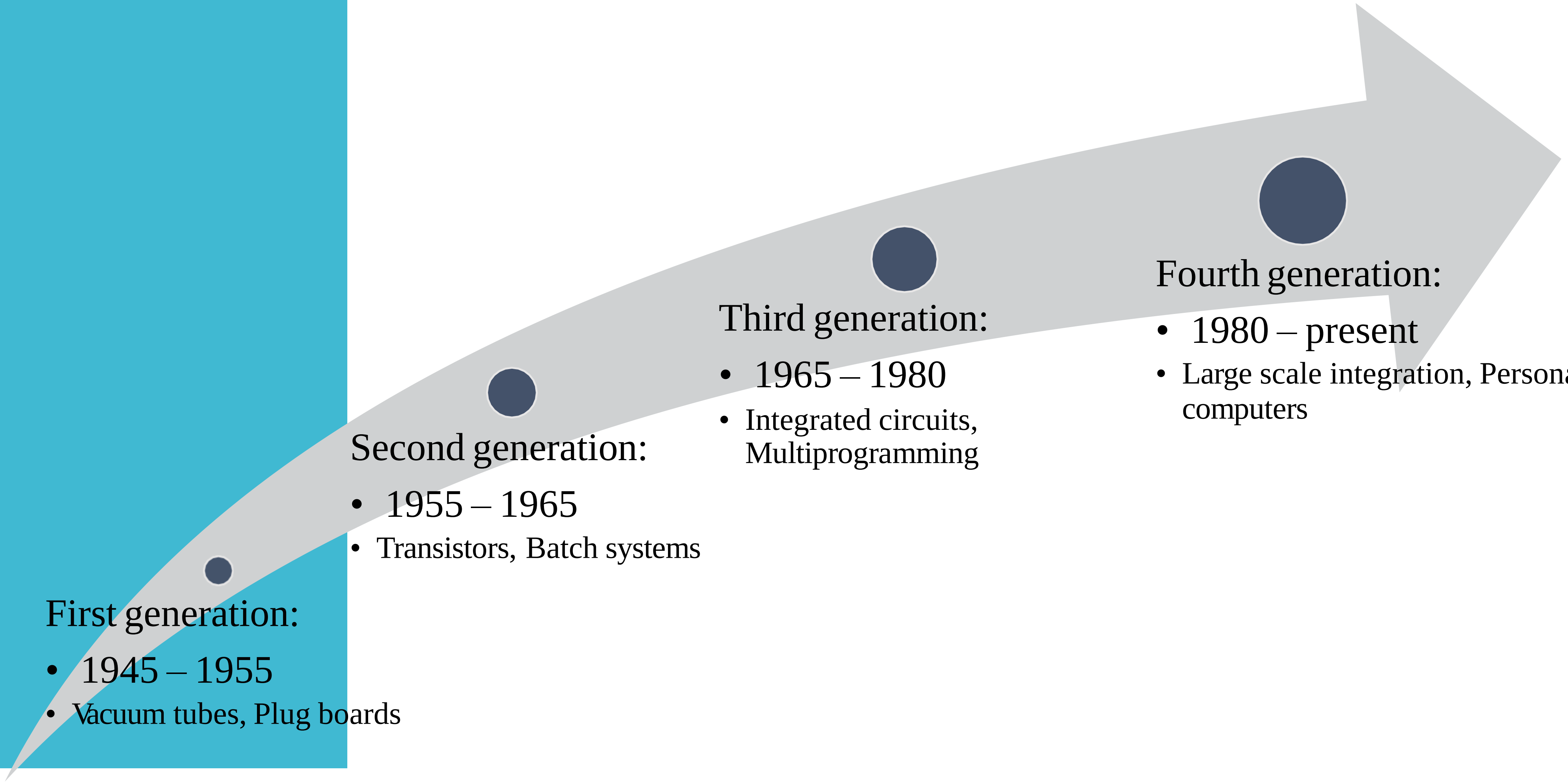
Generations of Operating systems: Fourth generation

An interesting development that began taking place during mid- 1980s is the growth of networks of personal computer running *network OS* and *distributed OS*

in NOS users are aware about existence of multiple users and can log into remote machine and copy files.

In contrast DOS is one that appears to its user as tradition uniprocessor system.

Generations of Operating systems: Summary



First generation:

- 1945 – 1955
  - Vacuum tubes, Plug boards

Second generation:

- 1955 – 1965
  - Transistors, Batch systems

Third generation:

- 1965 – 1980
  - Integrated circuits, Multiprogramming

Fourth generation:

- 1980 – present
  - Large scale integration, Personal computers

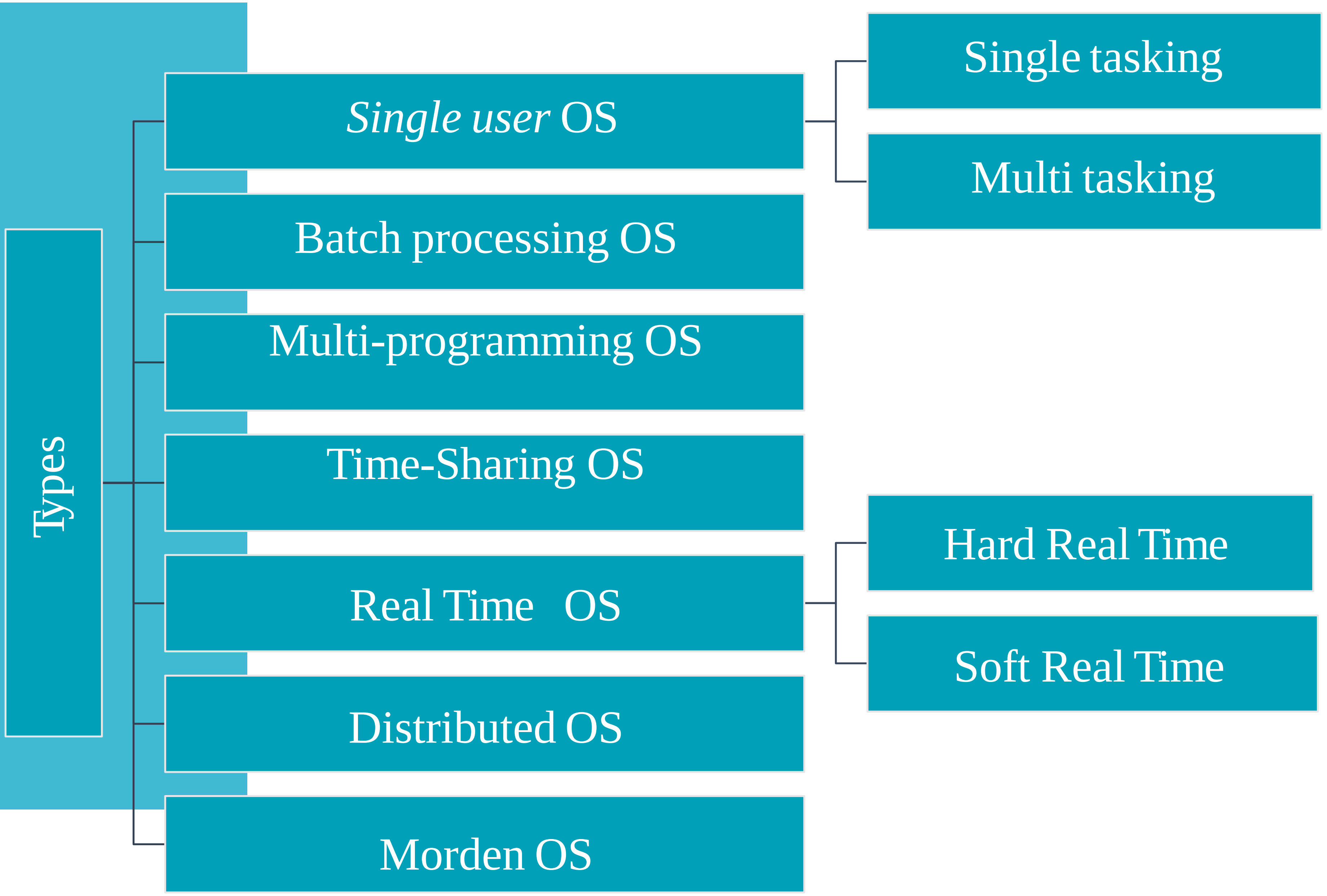
❖Types of Operating System

Types of Operating Systems

Types

- Single user OS
- Batch processing OS
- Multi-programming OS
- Time-Sharing OS
- Real Time OS
- Distributed OS
- Morden OS

Types of Operating Systems



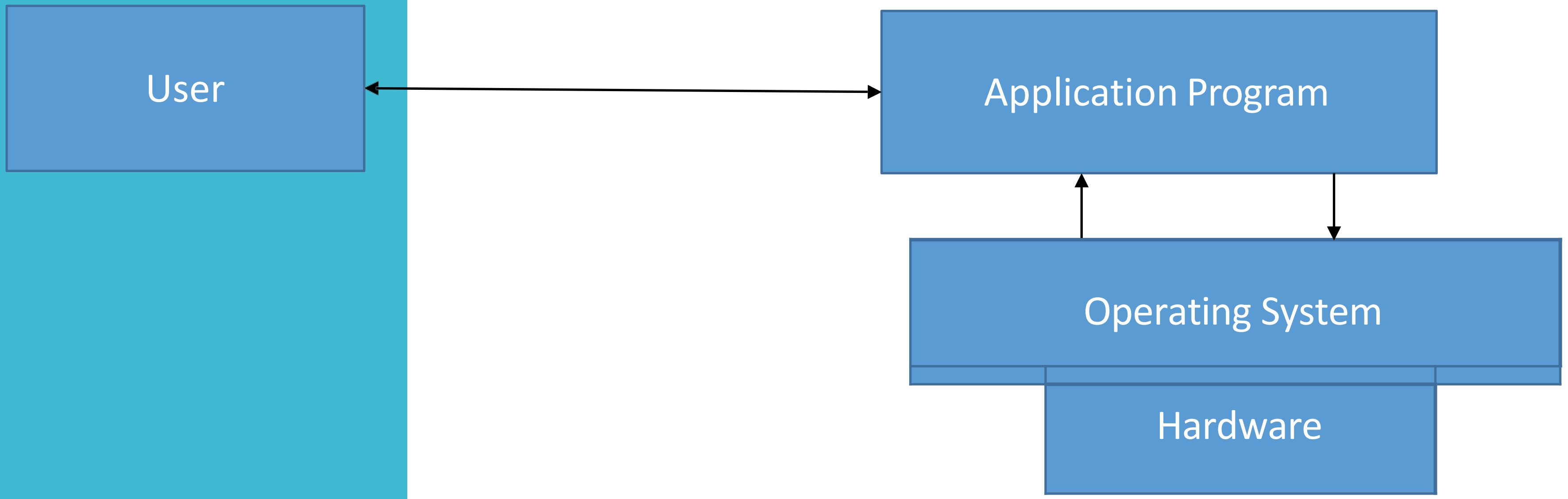
Types of Operating Systems

Single User OS

- One user can access computer at time.
- Computer have single processor and can execute single program.
- CPU sits idle for most of the time.
- Two types:
  - *Single user, single tasking:* allows only single user to execute one program at a time.
  - *Single user, multi tasking:* allows a single user to operate multiple program at a time.

Types of Operating Systems: Single User OS

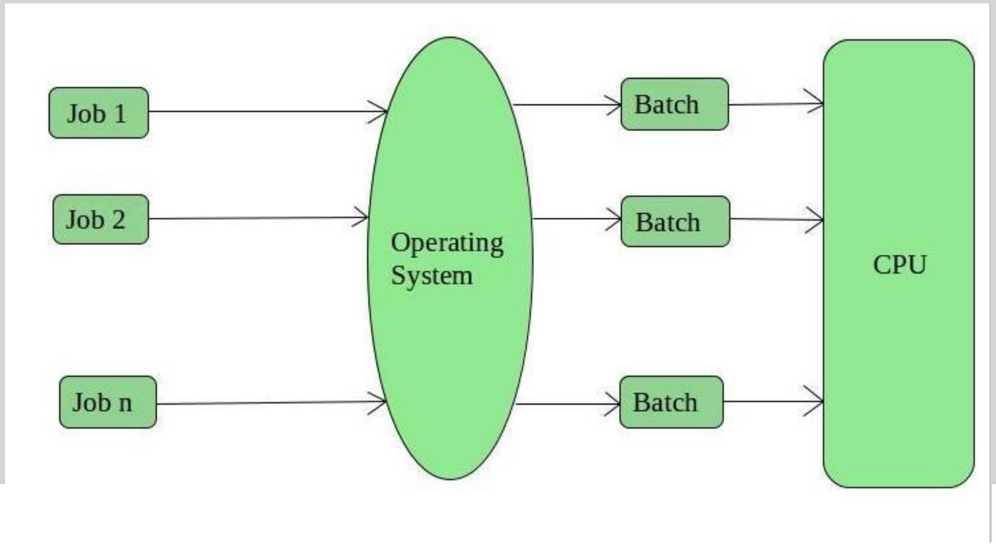
- single user OS executes an application program of user with help of hardware and gives results back to the user.



Types of Operating Systems

Batch processing

- A batch is sequence of user jobs.
- Each job in the batch is independent of other job in the batch, may belong to different user.



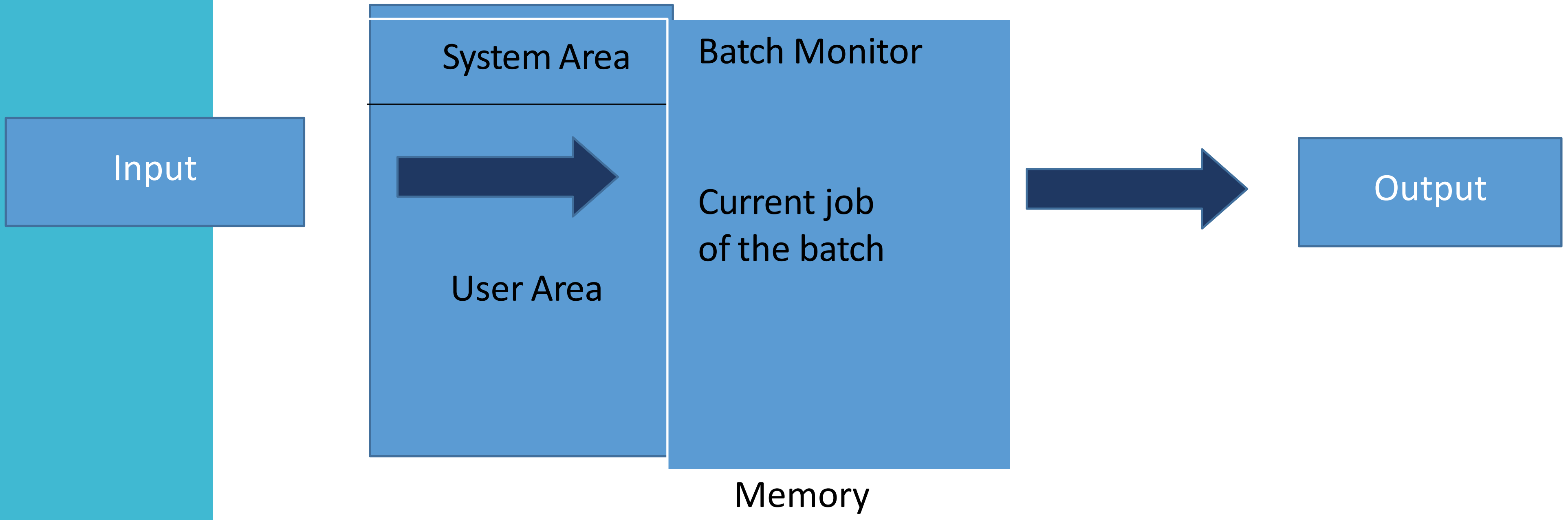
Types of Operating Systems

Batch processing

- **Working:**
  - Programmer prepare job and submit it to the operator. Job consist of program, data and some control information.
  - Operator sort them in batches with the similar requirements, and insert marker card to indicate start and end of the batch.
  - run them batch wise.
  - Some time later (after some hours later or may be even after some days) the output appeared.

Types of Operating Systems

- Batch processing:
  - Schematic of batch processing: part of the memory occupied by the batch monitor called system area, and the part of the memory occupied by the user job is called user area.



Types of Operating Systems

Multi-programming OS

- **Advantage**
- It allows *multiple user* to execute *multiple programs* using single CPU.
- In this system collection of job is maintained on the disk, and subset of them is kept in the main memory.
- These job can be execute simultaneously using *time-multiplexed CPU*.

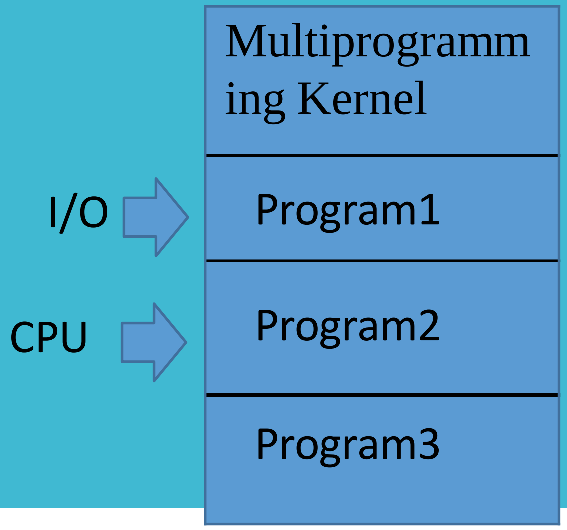
Types of Operating Systems

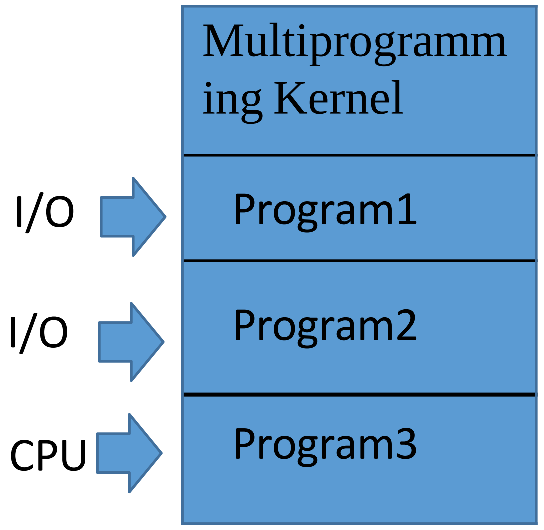
Batch processing

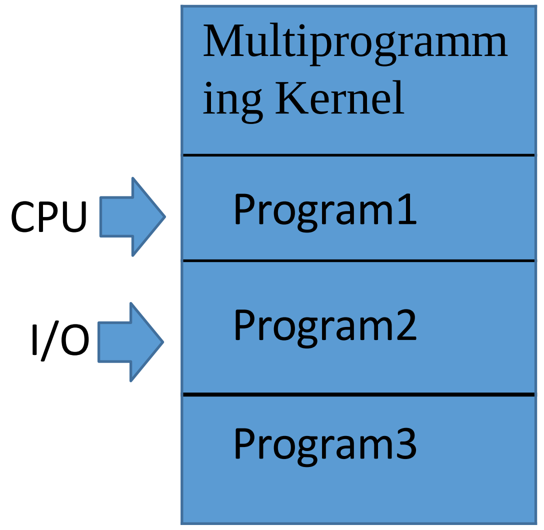
- **Disadvantage:**
  - *Low throughput*: CPU remain idle while I/O operations.
  - *High turn around time:* turn around time is time taken between submitting the job and getting the output, which includes both batch formation and its execution.

Types of Operating Systems

- Multi-programming OS:
  - Example:
    - When running program request for the I/O operation, then the CPU will be free thus CPU can be allocated to other program.







Types of Operating Systems

Multi-programming OS

- Here unlike batch OS more then one job can be loaded to the memory simultaneously.
- Kernel required to perform:
  - *Scheduling :* OS have to schedule programs in the main memory.
  - *Memory Management :* allocate memory to programs.
  - *I/O Management :* Devices must be allocated to programs as when required.

Types of Operating Systems

Time Sharing

- Time sharing kernel uses the *interval timer* to implement time slicing.
- Time sharing OS uses *round-robin scheduling* with time slicing.
- Features of Time-sharing OS:
  - Several jobs are kept in the main memory so memory management is required.
  - Memory is limited thus swapping is used.
  - CPU scheduling must be fair for all the process.
  - Examples of Time-Sharing OSs are: Multics, Unix, etc.

Types of Operating Systems

Time Sharing

Advantages of Time-Sharing OS:

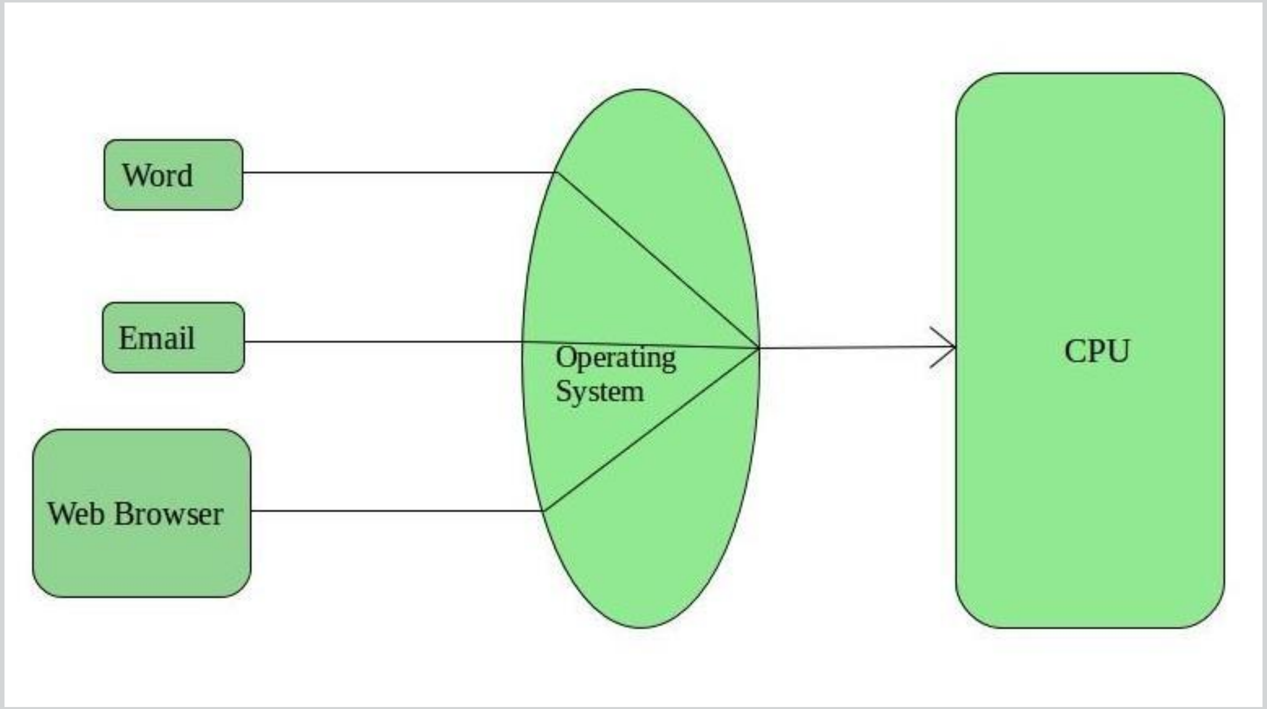
- Each task gets an equal opportunity
- Fewer chances of duplication of software
- CPU idle time can be reduced

Disadvantages of Time-Sharing OS:

- Reliability problem
- One must have to take care of the security and integrity of user programs and data
- Data communication problem

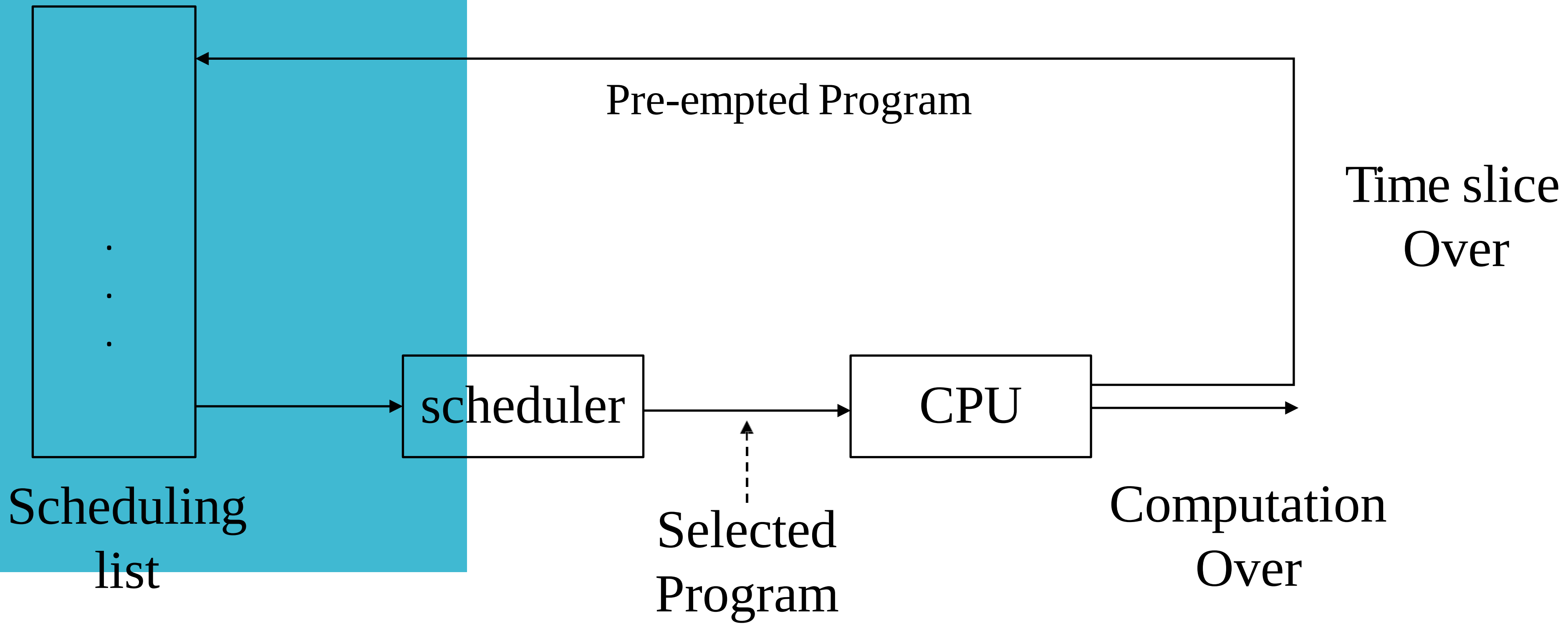
Types of Operating Systems

Time Sharing



Types of Operating Systems

- **Time-sharing / Multitasking OS:**
  - Here CPU is multiplexed by time among several job that are kept in main memory.
  - Time Slice: largest amount of CPU time program can execute on CPU.



Types of Operating Systems

Real Time OS

- A system is said to be Real Time if it is required to complete it's work & deliver it's services on time.
- A real time OS defines the completion of job within the *rigid time constraints* otherwise job looses its meaning.
- Example – Flight Control System, Air line reservation system.
  - All tasks in that system must execute on time.

Types of Operating Systems: Real Time OS

Two types:

Hard Real time :

- It ensures the complication of critical tasks within the well defined constraints.
- It can not afford missing even a single deadline, single miss can lead to the critical failure.
- Example: Flight controller system.

Soft Real Time

- Late completion of jobs is undesirable but not fatal(does not leads to any critical failure) .
- System performance degrades as more & more jobs miss deadlines.
- Example: Online Databases, DVD player cannot process a frame.

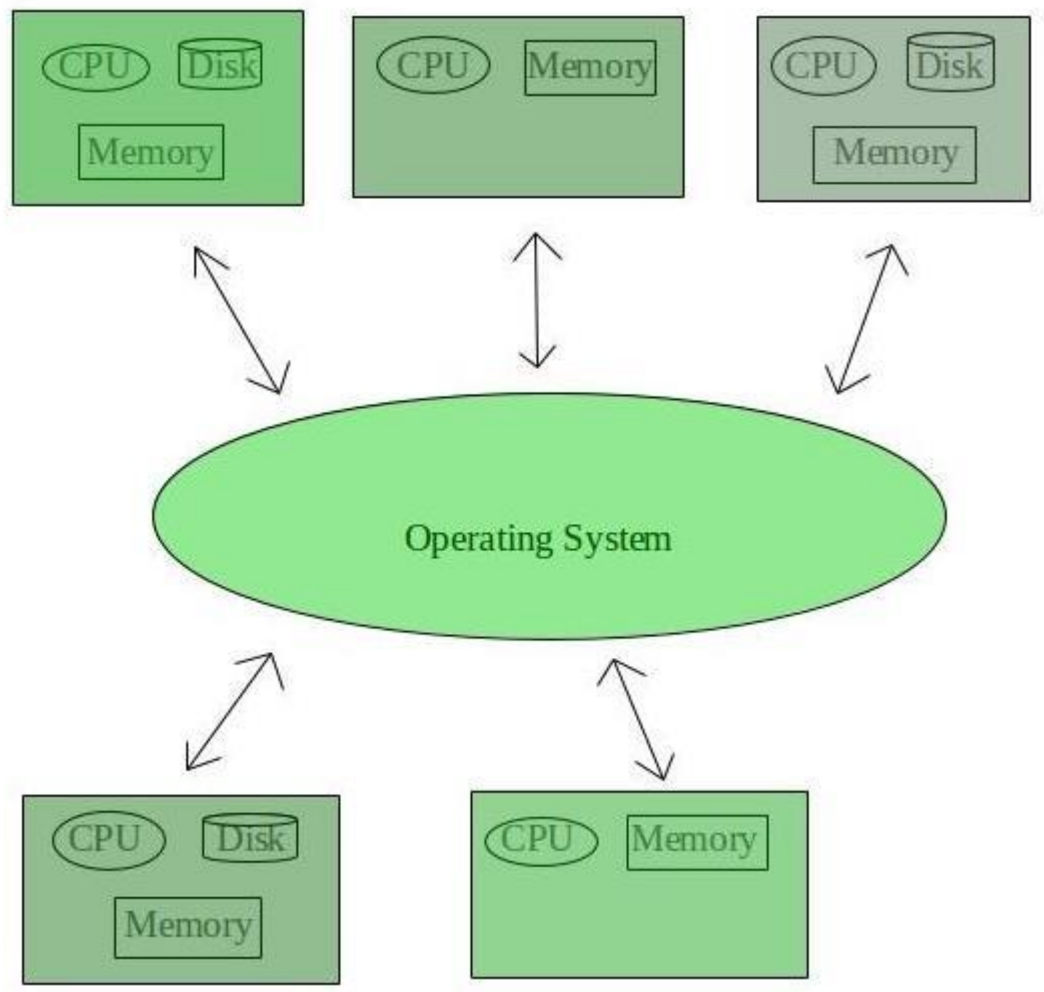
Types of Operating Systems

Distributed OS

- It consist of several individual computer system connected through network.
- A distributed system is system consisting of two or more nodes, where each node is computer system with its own memory, some network hardware, and able to compute some control function.
- It may connect devices such as memories, CPU and I/O devices.
- Examples of Distributed Operating System are- LOCUS, etc

Types of Operating Systems

Distributed OS



Types of Operating Systems

- Distributed OS
  - Features of DOS:

Feature	Description
Resource sharing	Improves resource utilization
Reliability	Availability of services and resources
Computation speed up	As a parts of communication system can be execute in different computer system.
Communication	Provides means of communication between remote entities.
Incremental Growth	Processing power can be enhanced.

Types of Operating Systems

- Distributed OS
  - Key concept and technique used in DOS:

Technique	Description
Distributed Control	Control function can be performed by possibly all nodes in the distributed system.
Transparency	Resources or services can be accessed without having knowledge about the location in the system. (to hide the fact that its process and resources are physically distributed across multiple computers.Eg. Access,location,migration,relocation,….)
Remote Procedure Call (RPC)	A procedure calls a procedure that is located in different computer system. The remote procedure may perform a part of computation in the application.

Types of Operating Systems

Distributed OS

Advantages of Distributed Operating System:

- Failure of one will not affect the other network communication, as all systems are independent from each other
- Since resources are being shared, computation is highly fast and durable
- Load on host computer reduces
- These systems are easily scalable as many systems can be easily added to the network
- Delay in data processing reduces

Types of Operating Systems

Distributed OS

Disadvantages of Distributed Operating System:

- Failure of the main network will stop the entire communication
- To establish distributed systems the language which is used are not well defined yet
- These types of systems are not readily available as they are very expensive. Not only that the underlying software is highly complex and not understood well yet

Types of Operating Systems

Modern OS

- User engage in diverse activities in modern computing environment, hence modern OS cannot use a uniform strategy for all processes.
- Here OS have to use a strategy that is appropriate for each individual process.

Types of Operating Systems

Modern OS

- Example:
  - User can open mail handler, edit files, execute some programs, and can listen music at the same time.
  - The execution of the program may be interactive and may involve an activity in he another node of DOS, listing music is a soft real time activity, so the OS must use round robin algorithm for program execution, have to use propriety scheduling for music and RPC to support distributed activities.
- Thus modern OS have to use most of the concept of all other types of OS.

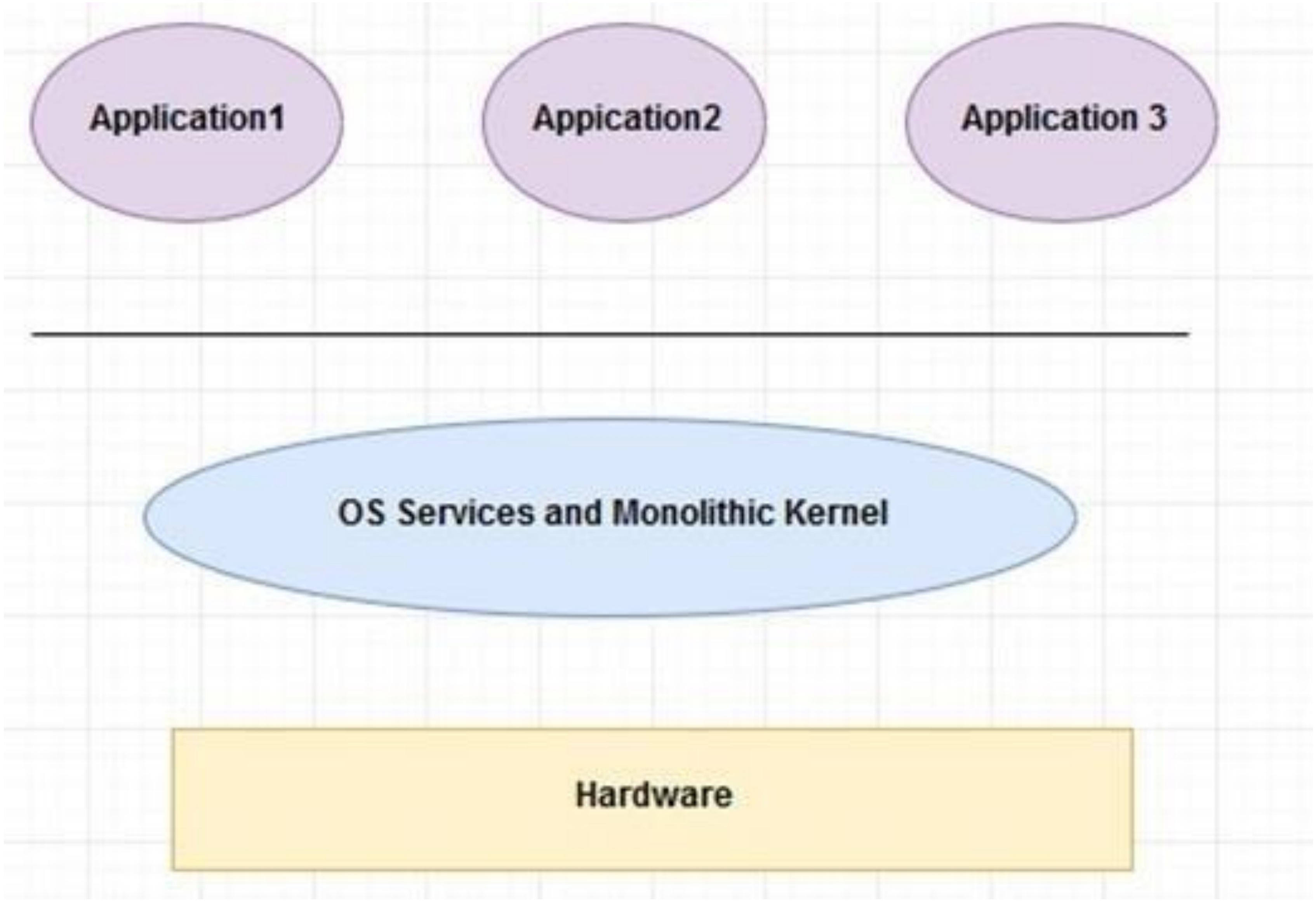
Types of Operating Systems

Modern OS

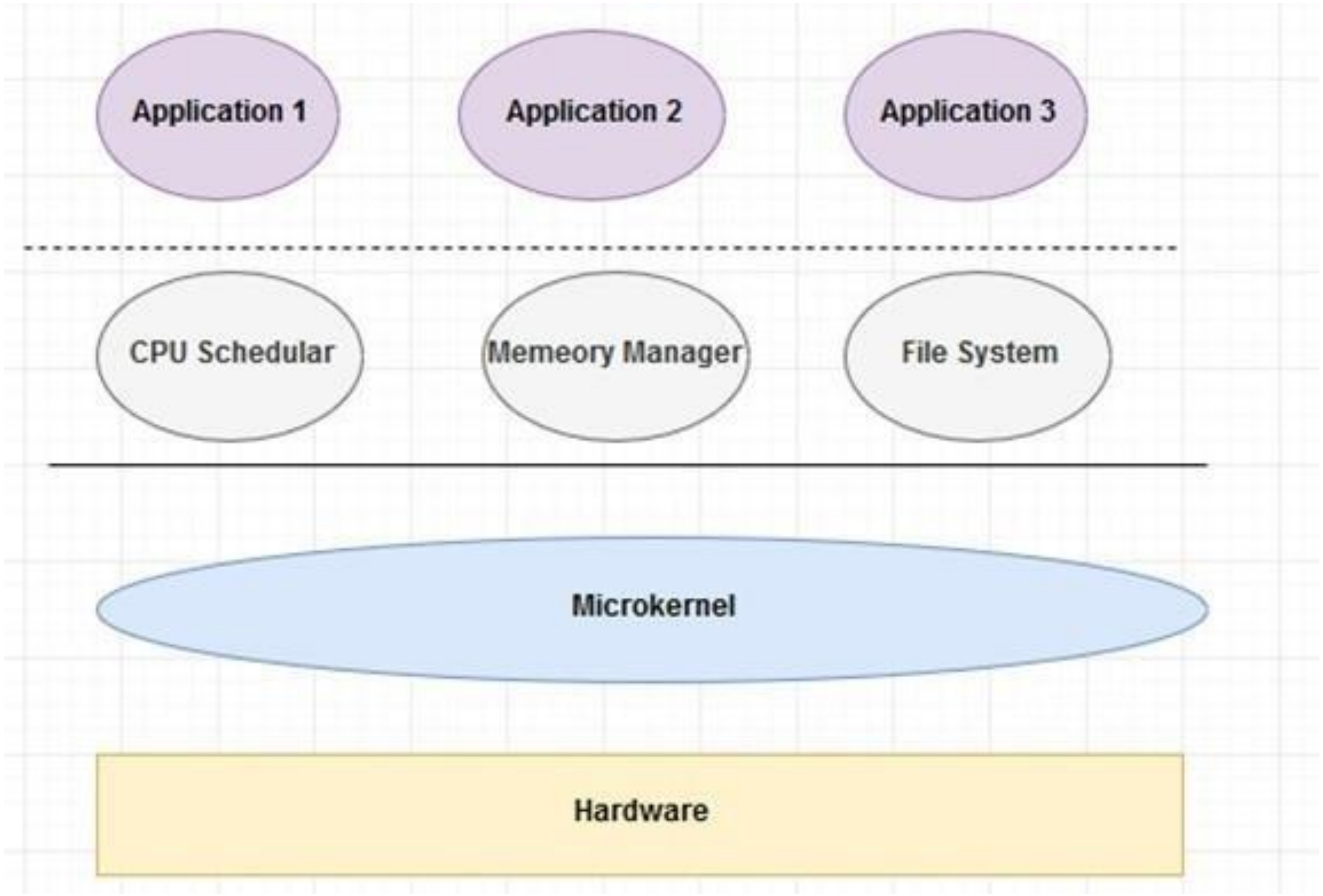
- But OS can not use different strategy for different processes so modern OS actually use single complex strategy.
- Example:
  - *priority based scheduling* where is the propriety is not fixed it is depend up on the *type of application.*
  - i.e. Real time process have high priority, all interactive process gets moderate priority and non-intractive process gets lower priority.

❖ Structure of Operating System

Monolithic Kernel



Microkernel



Monolithic Kernel VS Microkernel

S. No.	Parameters	Microkernel	Monolithic kernel
1.	**Address Space**	In microkernel, user services and kernel services are kept in separate address space.	In monolithic kernel, both user services and kernel services are kept in the same address space.
2.	**Design and Implementation**	OS is complex to design.	OS is easy to design and implement.
3.	**Size**	Microkernel are smaller in size.	Monolithic kernel is larger than microkernel.
4.	**Functionality**	Easier to add new functionalities.	Difficult to add new functionalities.
5.	**Coding**	To design a microkernel, more code is required.	Less code when compared to microkernel

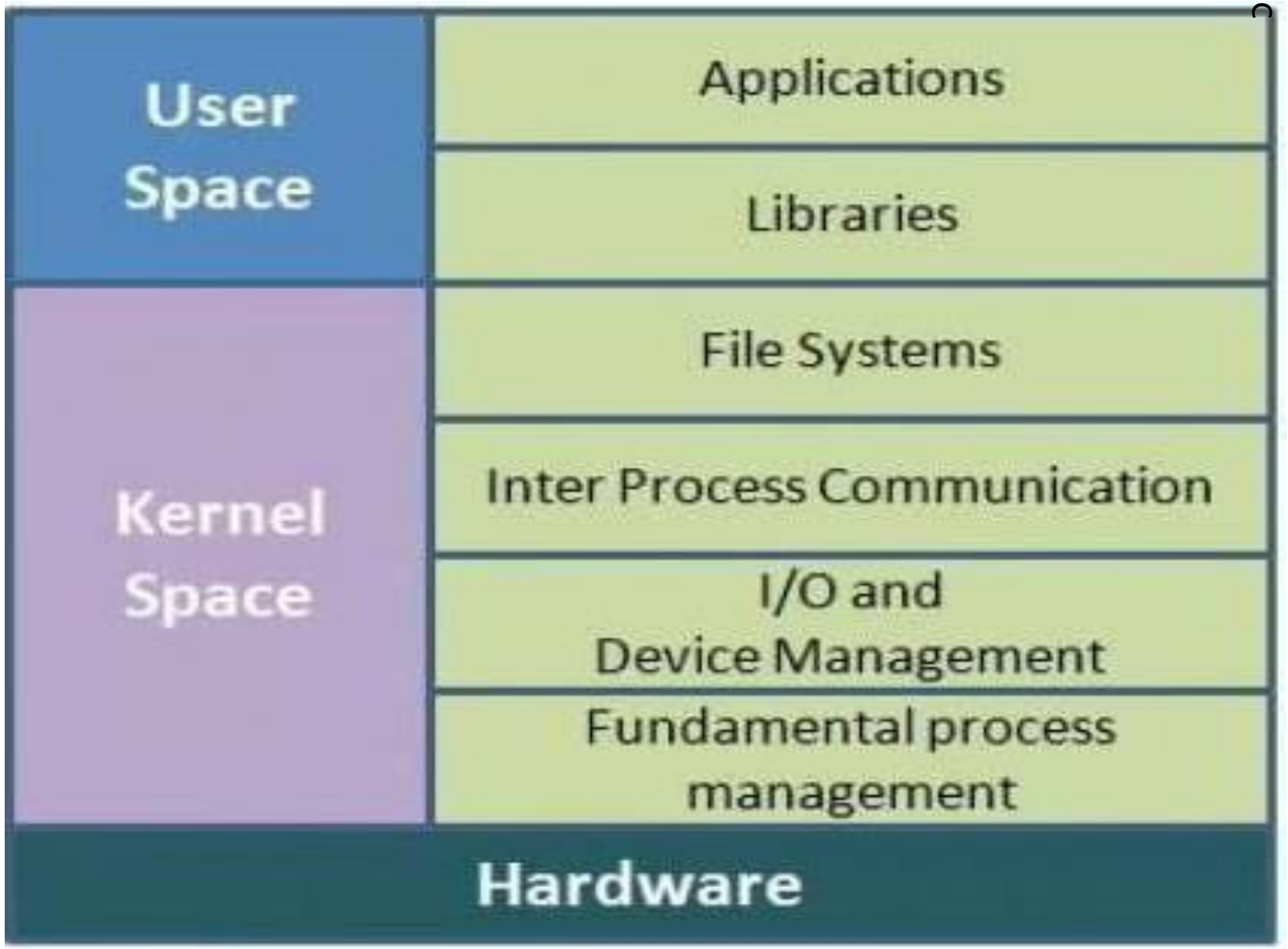
Monolithic Kernel VS Microkernel

S. No.	Parameters	Microkernel	Monolithic kernel
6.	**Failure**	Failure of one component does not effect the working of micro kernel.	Failure of one component in a monolithic kernel leads to the failure of the entire system.
7.	**Processing Speed**	Execution speed is low.	Execution speed is high.
8.	**Extend**	It is easy to extend Microkernel.	It is not easy to extend monolithic kernel.
9.	**Communication**	To implement IPC messaging queues are used by the communication microkernels.	Signals and Sockets are utilized to implement IPC in monolithic kernels.
10.	**Debugging**	Debugging is simple.	Debugging is difficult.

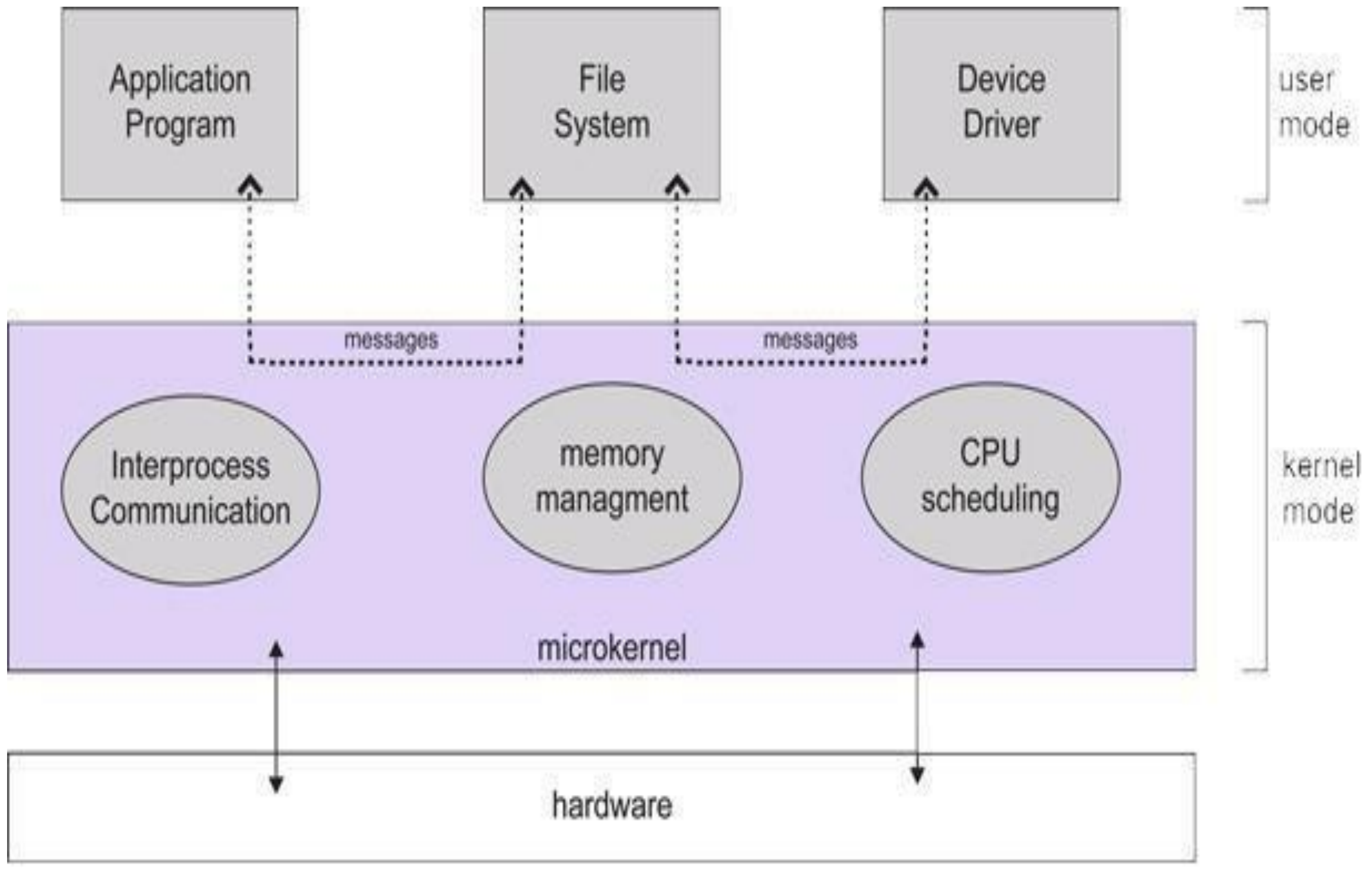
Monolithic Kernel VS Microkernel

S. No.	Parameters	Microkernel	Monolithic kernel
11.	**Maintain**	It is simple to maintain.	Extra time and resources are needed for maintenance.
12.	**Message passing and Context switching**	Message forwarding and context switching are required by the microkernel.	Message passing and context switching are not required while the kernel is working.
13.	**Services**	The kernel only offers IPC and low-level device management services.	The Kernel contains all of the operating system's services.
14.	**Example**	**Example :** Mac OS X.	**Example :** Microsoft Windows 95.

Monolithic systems


User Space
Applications
Libraries
Kernel Space
File Systems
Inter Process Communication
I/O and Device Management
Fundamental process management
Hardware

Microkernel systems



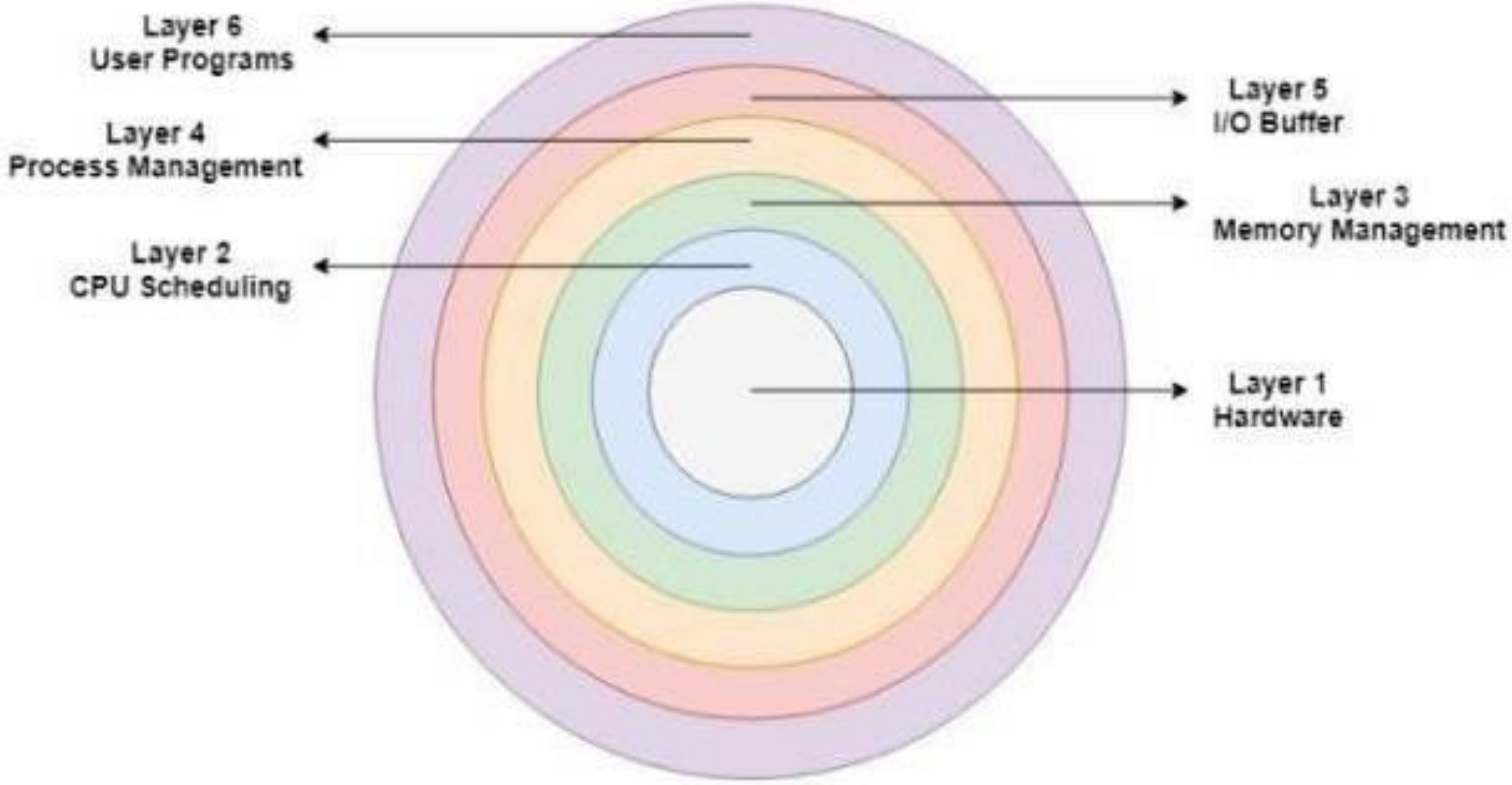
Architecture of a typical microkernel

LAYERED SYSTEM

- A layered system consists of a series of layers, each of which depends only on the correct operation of the layer immediately beneath it.
- The lowest layer represents the hardware interface, and the highest layer the application interface.
- All the layers can be defined separately and interact with each other as required. Also, it is easier to create, maintain and update the system if it is done in the form of layers. Change in one layer specification does not affect the rest of the layers.

LAYERED SYSTEM

- A key problem in a layered system is deciding on the ordering of the layers. (This is critical because of the requirement that each layer can ultimately only use services provided by layers below it - so the ordering cannot have any cycles.)
- In a strictly-layered structure, efficiency can also become a problem because when a higher-level layer requires a lower-level operation the request must work its way down layer by layer.
- Examples OS/2, window NT



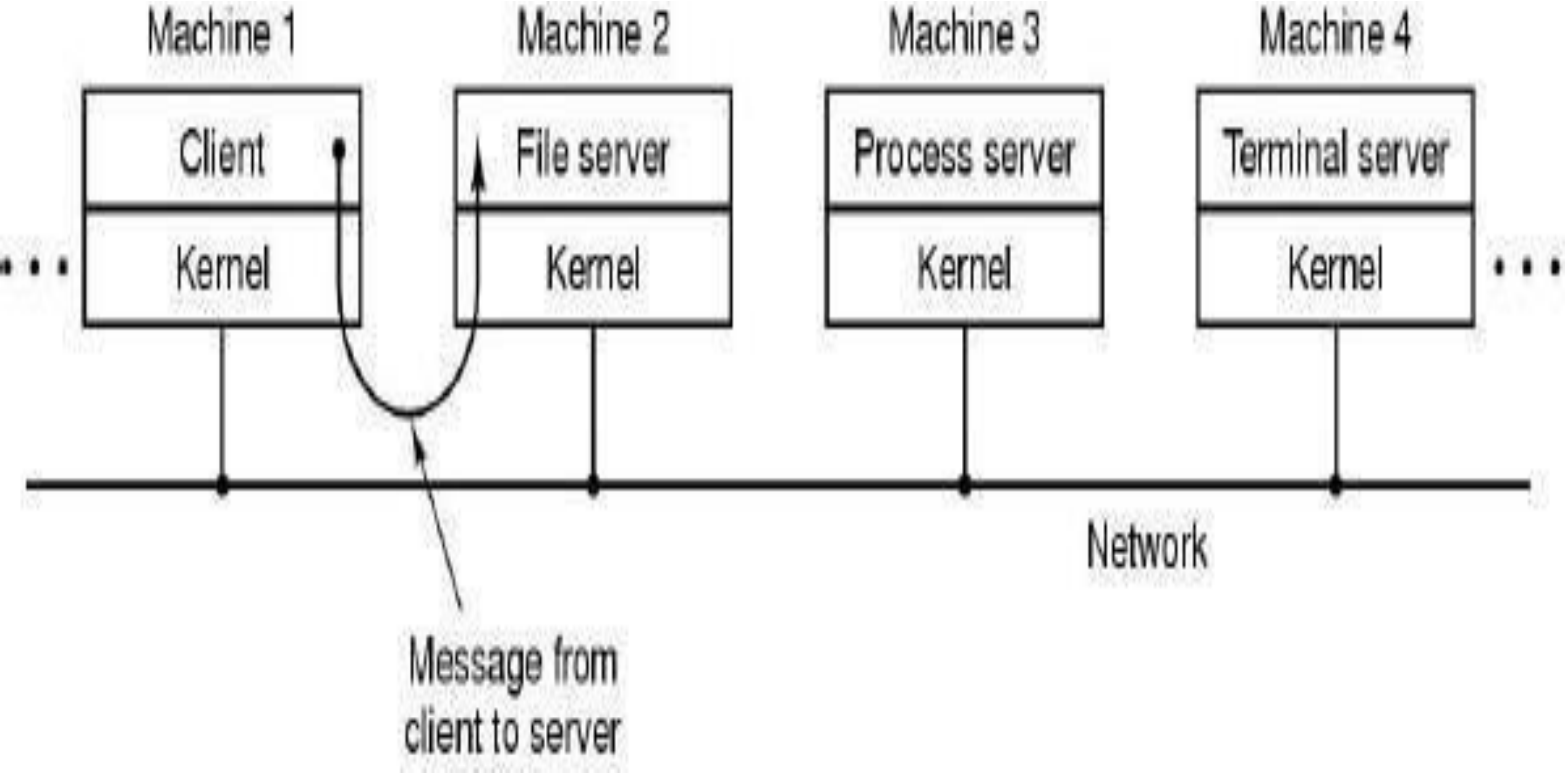
LAYERED OPERATING SYSTEM

CLIENT-SERVER MODEL

- To request a service, such as reading a block of

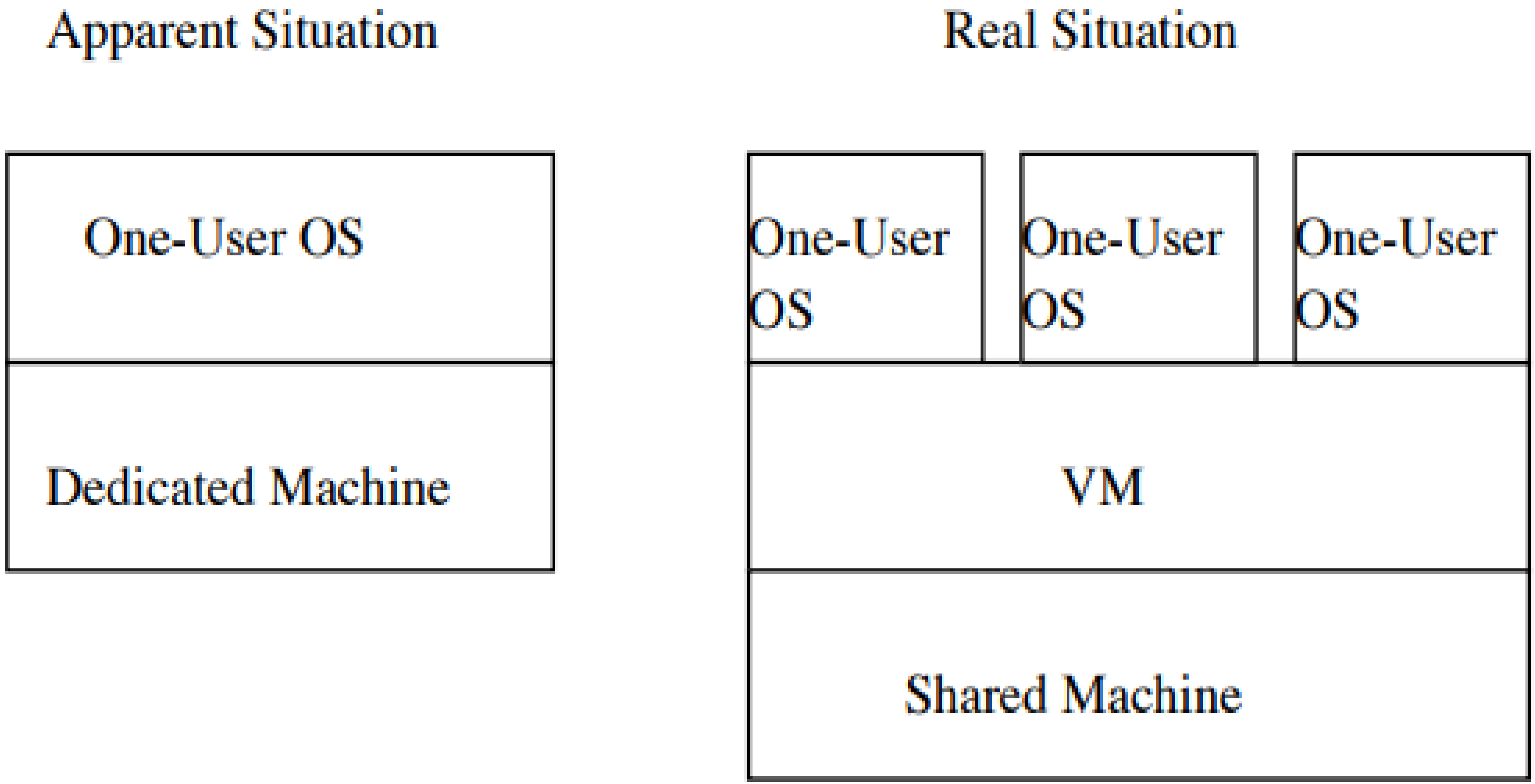
  a file, a user process (now known as the client process) sends the request to a server process, which then does the work and sends back the answer

- Kernel handle the communication between clients and servers. By splitting the operating system up into parts, each of which only handles one facet of the system, such as file service, process service, terminal service, or memory service, each part becomes small and manageable.

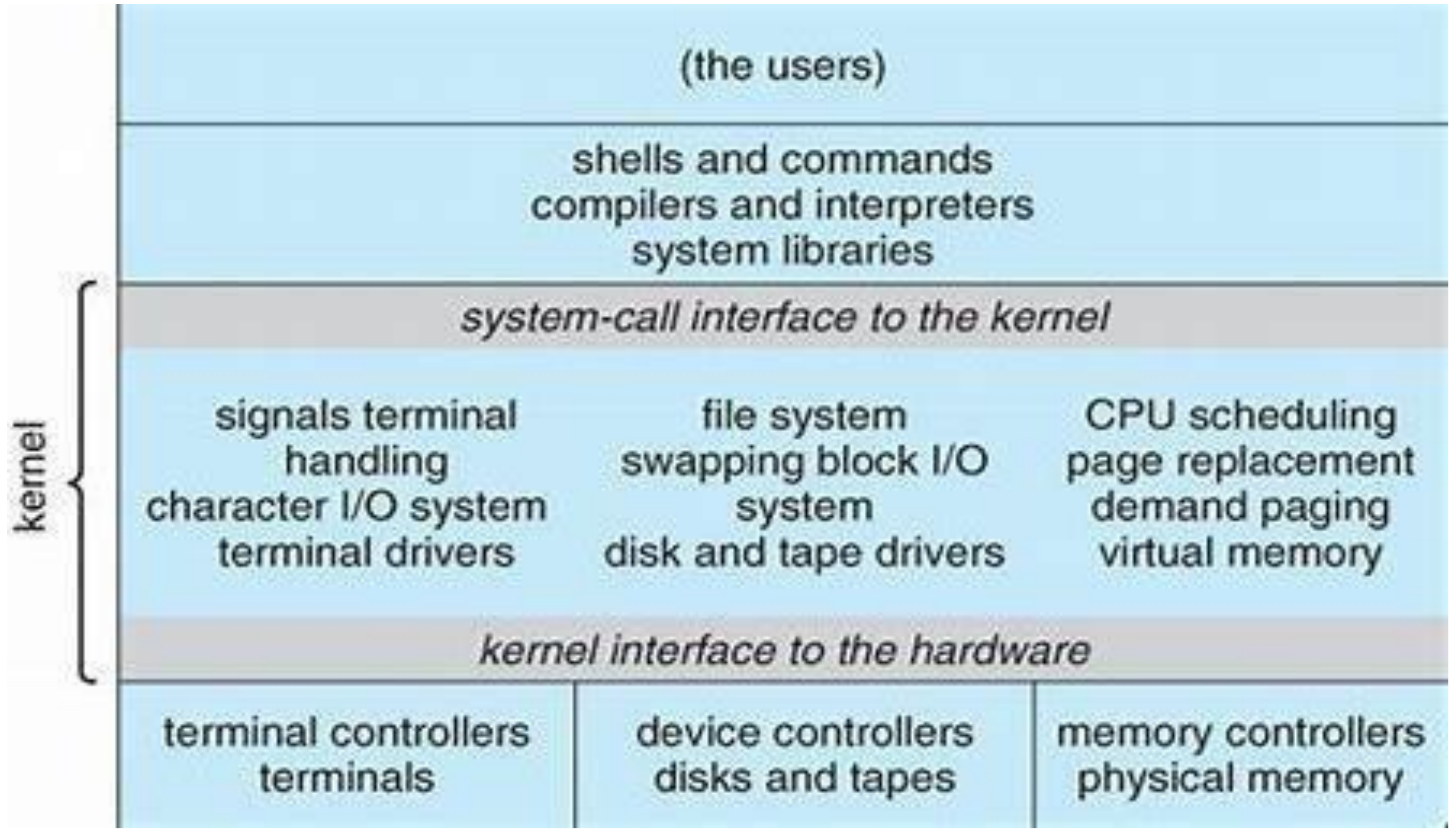

Machine 1
Machine 2
Machine 3
Machine 4
Client
File server
Process server
Terminal server
Kernel
Kernel
Kernel
Kernel
Network
Message from client to server

VIRTUAL MACHINE

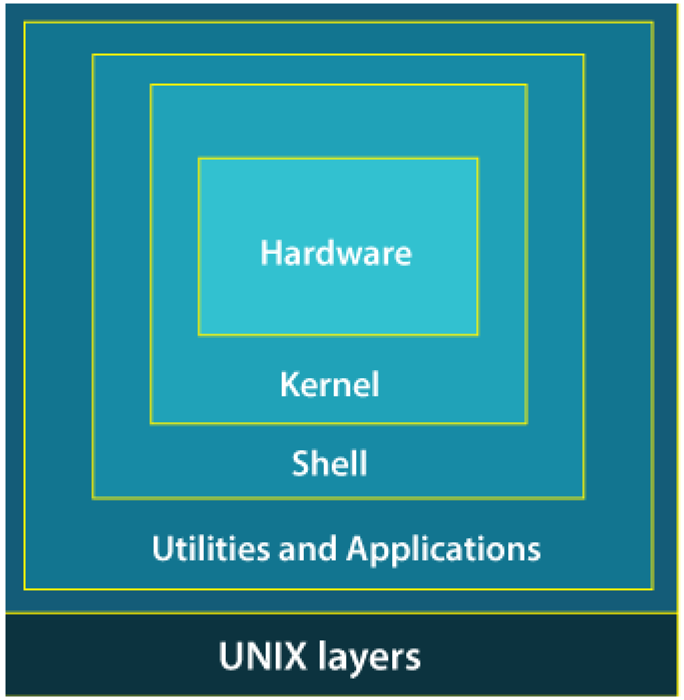
- A virtual machine (VM) is an efficient, isolated duplicate of a real machine
- Enables a single PC or server to simultaneously run multiple operating systems or multiple sessions of a single OS
- The individual user was quite conscious of sharing a particular machine with a number of other users. For example, the user would have to specifically request certain system resources when they were needed
- Virtual box and VMware is example of virtual machine

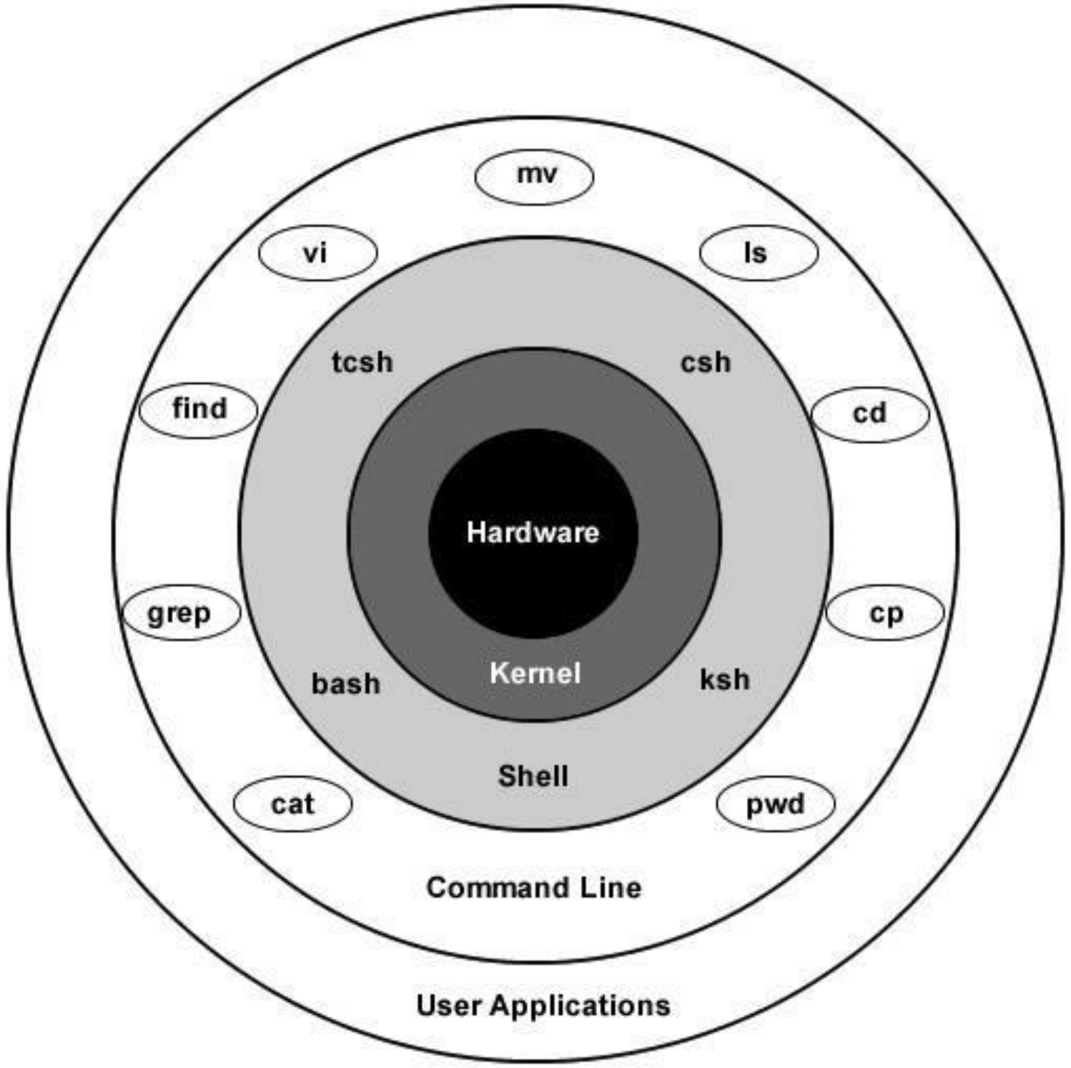

Apparent Situation
Real Situation
One-User OS
Dedicated Machine
One-User OS
One-User OS
One-User OS
VM
Shared Machine

UNIX SYSTEM STRUCTURE

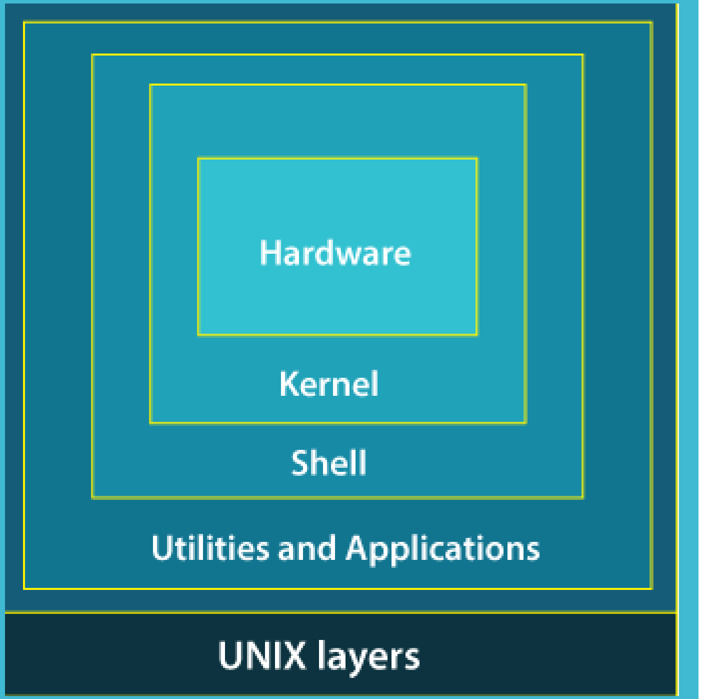


UNIX SYSTEM STRUCTURE





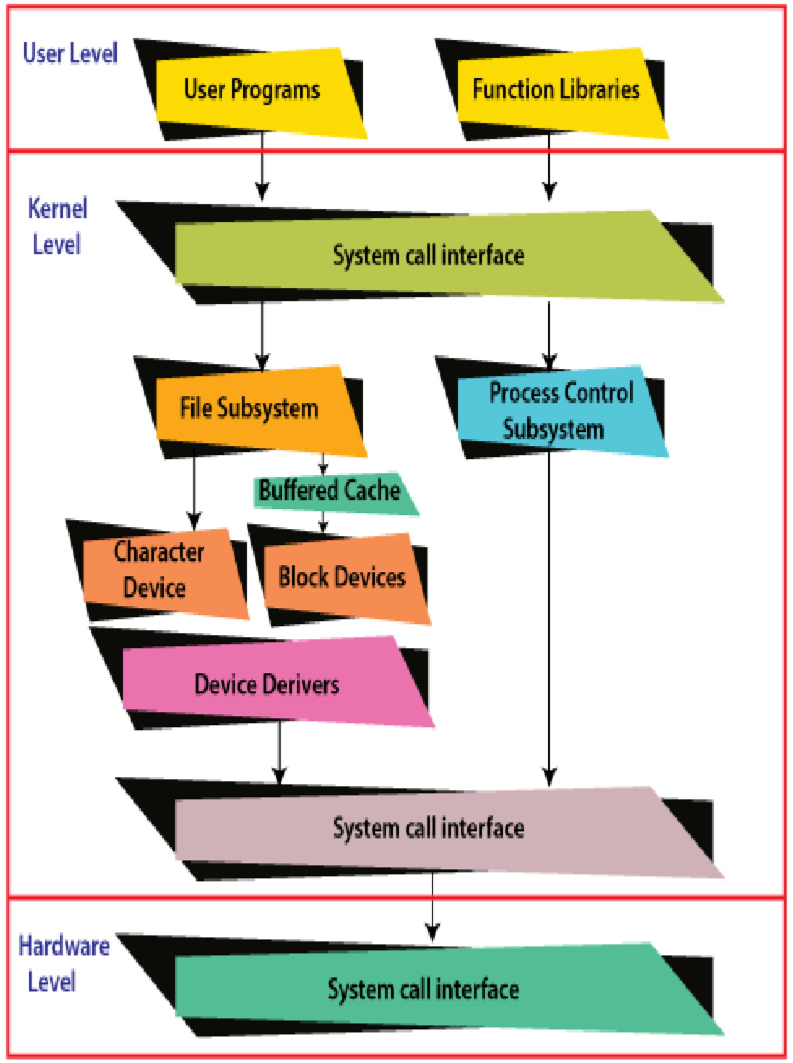
UNIX SYSTEM STRUCTURE



Layer-1: Hardware -

- This layer of UNIX consists of all hardware-related information in the UNIX environment.

UNIX SYSTEM STRUCTURE

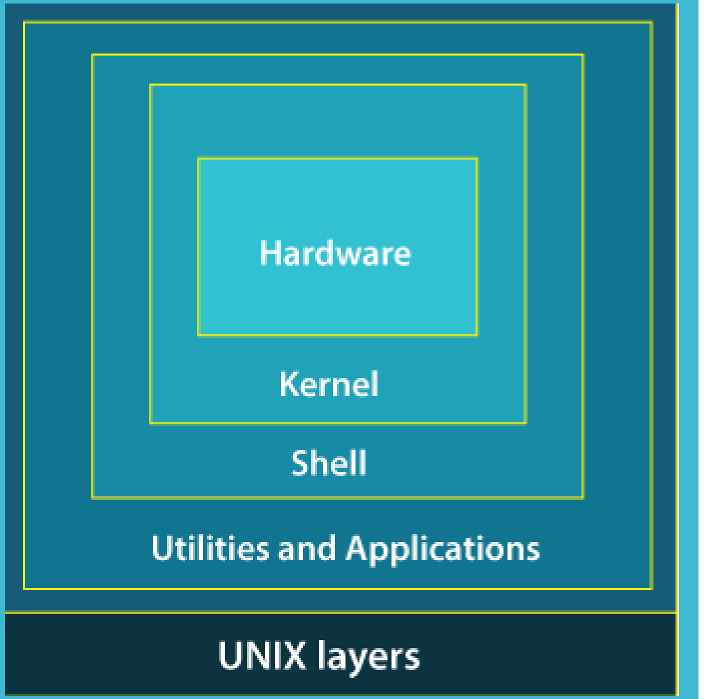


Kernel Architecture

Layer-2: Kernel -

- The core of the operating system that's liable for maintaining the full functionality is named the kernel.
- The kernel of UNIX runs on the particular machine hardware and interacts with the hardware effectively.
- The kernel is the heart of the operating system.
- The kernel ingests user input via the shell and accesses the hardware
  to perform things like memory allocation and file storage.
- It also works as a device manager and performs valuable functions for the processes which require access to the peripheral devices connected to the computer. The kernel controls these devices through device drivers.
- The kernel also manages the memory. Processes are executed programs that have owner's humans or systems who initiate their execution.
- The system must provide all processes with access to an adequate amount of memory, and a few processes require a lot of it.
- To make effective use of main memory and to allocate a sufficient amount of memory to every process. It uses essential techniques like paging, swapping, and virtual storage.

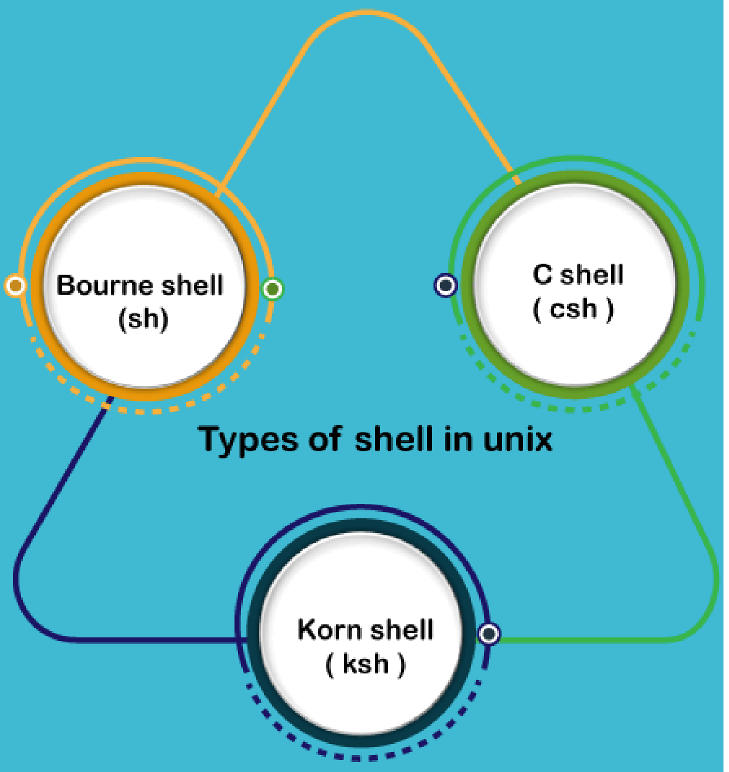
UNIX SYSTEM STRUCTURE



Layer-3: The Shell -

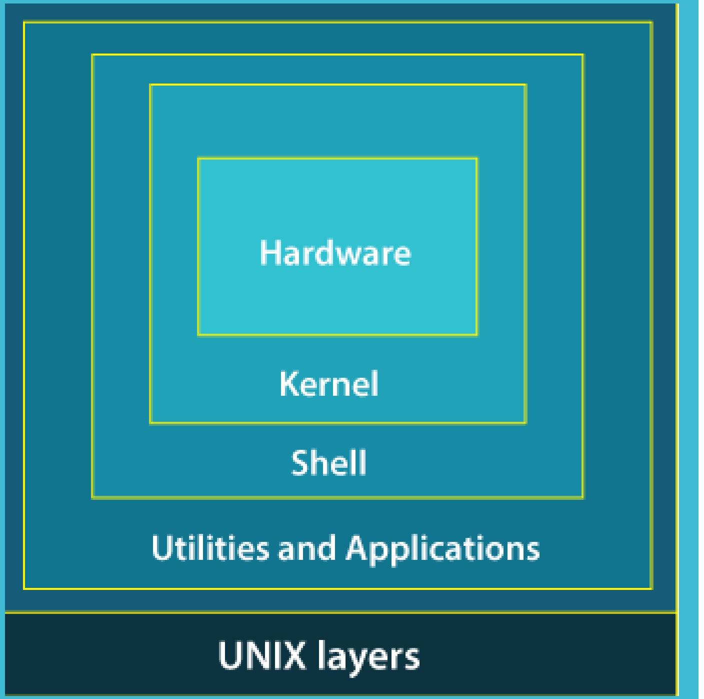
- The Shell is an interpreter that interprets the command submitted by the user at the terminal, and calls the program you simply want
- Shell are the Soul of the operating system..
- It also keeps a history of the list of the commands you have typed in.
- If you need to repeat a command you typed it, use the cursor keys to scroll up and down the list or type history for a list of previous commands.
- There are various commands like cat, mv, cat, grep, id and many more.

UNIX SYSTEM STRUCTURE

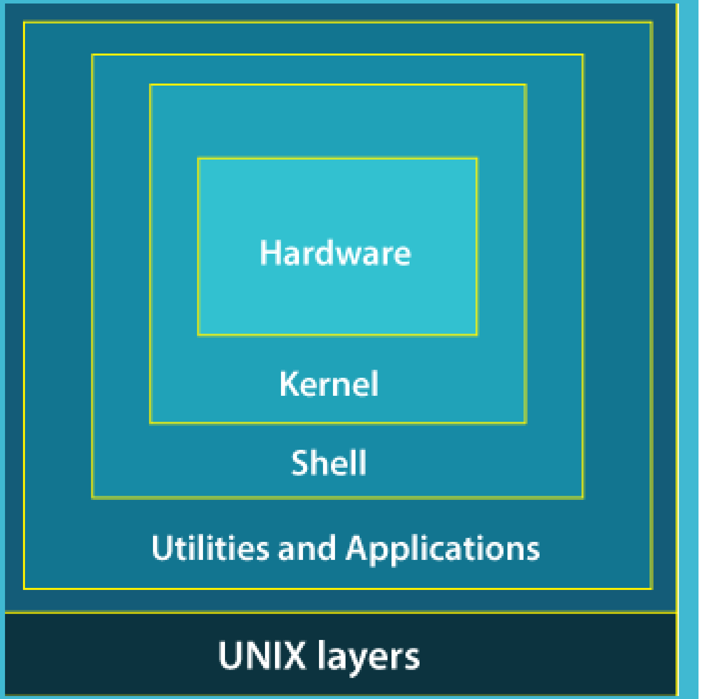


- **Type of Shell:**
- **Bourne Shell:** This Shell is simply called the Shell. It was the first Shell for UNIX OS. It is still the most widely available Shell on a UNIX system.
- **C Shell:** The C shell is another popular shell commonly available on a UNIX system. The C shell was developed by the University of California at Berkeley and removed some of the shortcomings of the Bourne shell.
- **Korn Shell:** This Shell was created by David Korn to address the Bourne Shell's user-interaction issues and to deal with the shortcomings of the C shell's scripting quirks.

UNIX SYSTEM STRUCTURE



- **Layer-4: Application Programs Layer -**
- It is the outermost layer that executes the given external applications. UNIX distributions typically come with several useful applications programs as standard.
- **For Example:** emacs editor, StarOffice, xv image viewer, g++ compiler etc.



- One example of how the shell and kernel work together is copying a file.
  - If you want to copy a file named "file1" and name the copy "file2", you would enter "cp file1 file2" at the command line.
  - The shell will search for the program "cp"
  - then tell the kernel to run that program on "file 1" and name the output "file 2".
  - When the copying is finished, the shell returns you to the prompt and awaits more commands.

Difference between Unix and Linux –

 Linux is essentially a clone of Unix. But, basic differences are shown below:

Linux	Unix
The source code of Linux is freely available to its users	The source code of Unix is not freely available general public
It has graphical user interface along with command line interface	It only has command line interface
Linux OS is portable, flexible, and can be executed in different hard drives	Unix OS is not portable
Different versions of Linux OS are Ubuntu, Linux Mint, RedHat Enterprise Linux, Solaris, etc.	Different version of Unix are AIS, HP-UX, BSD, Iris, etc.
The file systems supported by Linux are as follows: xfs, ramfs, vfat, cramfsm, ext3, ext4, ext2, ext1, ufs, autofs, devpts, ntfs	The file systems supported by Unix are as follows: zfs, js, hfx, gps, xfs, vxfs

❖System Call

System Calls

- **System call** is the programmatic way in which a computer program requests a service from the kernel of the operating system it is executed on.
- A system call is a way for programs to **interact with the operating system**. A computer program makes a system call when it makes a request to the operating system's kernel.
- System call **provides** the services of the operating system to
  the user programs via Application Program Interface(API).
- It provides an interface between a process and operating system to allow user-level processes to request services of the operating system.
- System calls are the only entry points into the kernel system. All programs needing resources must use system calls.

Services Provided by System Calls

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

- **Types of System Calls :**
- There are 5 different categories of system calls –
  - **Process control:** end, abort, create, terminate, allocate and free memory.
  - **File management:** create, open, close, delete, read file etc.
  - **Device management:** read, write, reposition, ioctl, functl
  - **Information maintenance:** getpid, getppid, attributes, get system time and date
  - **Communication:** pipe(), create/delete connection

fork() System Call

- **The** System call **fork()** is used to create child processes. It takes no arguments and returns a process ID.
- The purpose of **fork()** is to create a ***new*** process, which becomes the *child* process of the caller. After a new child process is created, ***both*** processes will execute the next instruction following the ***fork()*** system call.
- Therefore, we have to distinguish the parent from the child.
  This can be done by testing the returned value of **fork()**:

✓If **fork()** returns a negative value, the creation of a child process was unsuccessful.

✓**fork()** returns a zero to the newly created child process.

✓**fork()** returns a positive value, the ***process ID*** of the child process, to the parent.

- The returned process ID is of type **pid_t** defined in **sys/types.h**. Normally, the process ID is an integer. Moreover, a process can use function **getpid()** to retrieve the process ID assigned to this process.

Difference between fork() and exec()

Difference between fork() and exec()

- The **fork ()** system calls aids in the creation of processes. When a process uses the fork() system call, it creates a replicate of itself.
- The parent process is the existing process, and the child process is the new process. Although, the child process is equivalent to the parent process. When creating the child process, the parent state like open files, address space, and variables are copied to the child process.
- In other words, the child and parent processes are located in separate physical address spaces. As a result, the modification in the parent process doesn't appear in the child process.

Difference between fork() and exec()

- The **exec()** system call is used to make the processes. When the exec() function is used, the currently running process is terminated and replaced with the newly formed process. In other words, only the new process persists after calling exec().
- The parent process is shut down. This system call also substitutes the parent process's text segment, address space, and data segment with the child process.

System Calls: Interrupts

An interrupt is a signal to the processor emitted by hardware or software indicating an event that needs immediate attention

The processor responds by suspending its current activities, saving its state, and executing a function needed by interrupt

This interruption is temporary, and, after the interrupt handler finishes, the processor resumes normal activities.

There are two types of interrupts: hardware interrupts and software interrupts.

System Calls: Interrupts

- Hardware interrupt
  - pressing a key on the keyboard or moving the mouse triggers hardware interrupts that cause the processor to read the keystroke or mouse position.
- Software interrupt
  - errors or events occurring during program execution that are cannot be handled within the program itself.
  - For example, if the processor's arithmetic logic unit is commanded to divide a number by zero.

Hardware Interrupt VS Software Interrupt

SR.NO.	Hardware Interrupt	Software Interrupt
1	Hardware interrupt is an interrupt generated from an external device or hardware.	Software interrupt is the interrupt that is generated by any internal system of the computer.
2	It do not increment the program counter.	It increment the program counter.
3	Hardware interrupt can be invoked with some external device such as request to start an I/O or occurrence of a hardware failure.	Software interrupt can be invoked with the help of INT instruction.
4	It has lowest priority than software interrupts	It has highest priority among all interrupts.
5	Hardware interrupt is triggered by external hardware and is considered one of the ways to communicate with the outside peripherals, hardware.	Software interrupt is triggered by software and considered one of the ways to communicate with kernel or to trigger system calls, especially during error or exception handling.
6	It is an asynchronous event.	It is synchronous event.
7	Hardware interrupts can be classified into two types they are: 1. Maskable Interrupt. 2. Non Maskable Interrupt.	Software interrupts can be classified into two types they are: 1. Normal Interrupts. 2. Exception
8	Keystroke depressions and mouse movements are examples of hardware interrupt.	All system calls are examples of software interrupts