MET's Institute of Engineering

Bhujbal Knowledge City, Adgaon, Nashik.

Department of Computer Engineering

"OPERATING SYSTEM"

Prepared By

Prof. Anand N. Gharu

(Assistant Professor)

PVGCOE Computer Dept.

CLASS : TE COMPUTER 2019

SUBJECT : SPOS (SEM-I)

UNIT : IV

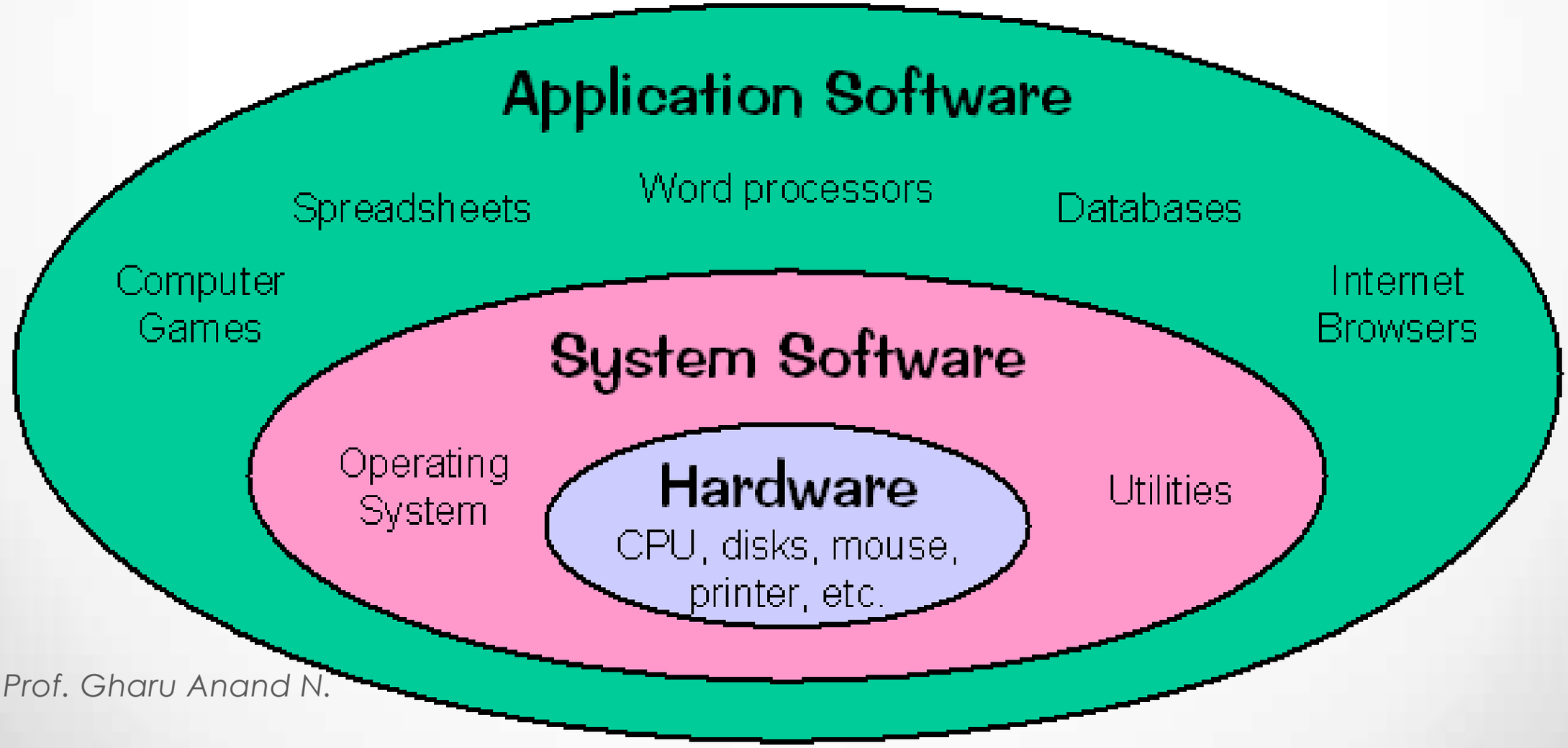
27 OCT 2023

CONTENTS :-

1. **Introduction**: Evolution of OS
2. Operating System Services
3. Functions of Operating System.
4. **Process Management:** Process, Process States: 5 and 7 state model
5. **Process control block**
6. **Threads,** Thread lifecycle, Multithreading Model, Process control system calls.
7. **Process Scheduling:** Uni-processor Scheduling, Scheduling: Preemptive, Non-preemptive, Long-term, Medium-term, Short term scheduling. Scheduling Algorithms: FCFS, SJF, RR, and Priority.

What’s operating system?

- An **Operating System** (**OS**) is an interface between a computer user and computer hardware. An **operating system** is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.



Evolution of OS

Operating system is divided into four generations, which are explained as follows –

1. First Generation (1945-1955) :

It is the beginning of the development of electronic computing systems which are substitutes for mechanical computing systems. Because of the drawbacks in mechanical computing systems like, the speed of humans to calculate is limited and humans can easily make mistakes. In this generation there is no operating system, so the computer system is given instructions which must be done directly.

Example − Type of operating system and devices used is Plug Boards.

Second Generation (1955-1965)

The Batch processing system was introduced in the second generation, where a job or a task that can be done in a series, and then executed sequentially. In this generation, the computer system is not equipped with an operating system, but several operating system functions exist like FMS and IBSYS.

Example − Type of operating system and devices used is Batch systems.

SERVICES OF OPERATING SYSTEM

Services of Operating System

- Program execution
- I/O operations
- File System manipulation
- Communication
- Error Detection
- Resource Allocation
- Protection

- **Program execution**

- Operating systems handle many kinds of activities from user programs to system programs like printer spooler, name servers, file server, etc. Each of these activities is encapsulated as a process.
- A process includes the complete execution context (code to execute, data to manipulate, registers, OS resources in use). Following are the major activities of an operating system with respect to program management −
- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

• I/O Operation

- An I/O subsystem comprises of I/O devices and their corresponding driver software. Drivers hide the peculiarities of specific hardware devices from the users.
- An Operating System manages the communication between user and device drivers.
- I/O operation means read or write operation with any file or any specific I/O device.
- Operating system provides the access to the required I/O device when required.

- **File system manipulation**
- A file represents a collection of related information. Computers can store files on the disk (secondary storage), for long-term storage purpose. Examples of storage media include magnetic tape, magnetic disk and optical disk drives like CD, DVD. Each of these media has its own properties like speed, capacity, data transfer rate and data access methods.
- Following are the major activities of an operating system with respect to file management −
- Program needs to read a file or write a file.
- The operating system gives the permission to the program for operation on file.
- Permission varies from read-only, read-write, denied and so on.
- Operating System provides an interface to the user to create/delete files.
- Operating System provides an interface to the user to create/delete directories.
- Operating System provides an interface to create the backup of file

- **Communication**
- In case of distributed systems which are a collection of processors that do not share memory, peripheral devices, or a clock, the operating system manages communications between all the processes. Multiple processes communicate with one another through communication lines in the network.
- Some activities are :
- Two processes often require data to be transferred between them
- Both the processes can be on one computer or on different computers, but are connected through a computer network.
- Communication may be implemented by two methods, either by Shared Memory or by Message Passing.

- **Error handling**
- Errors can occur anytime and anywhere. An error may occur in CPU, in I/O devices or in the memory hardware. Following are the major activities of an operating system with respect to error handling −
- The OS constantly checks for possible errors.
- The OS takes an appropriate action to ensure correct and consistent computing.

- **Resource Management**
- In case of multi-user or multi-tasking environment, resources such as main memory, CPU cycles and files storage are to be allocated to each user or job. Following are the major activities of an operating system with respect to resource management −
- The OS manages all kinds of resources using schedulers.
- CPU scheduling algorithms are used for better utilization of CPU.

- **Protection**
- Considering a computer system having multiple users and concurrent execution of multiple processes, the various processes must be protected from each other's activities.
- Protection refers to a mechanism or a way to control the access of programs, processes, or users to the resources defined by a computer system.
- The OS ensures that all access to system resources is controlled.
- The OS ensures that external I/O devices are protected from invalid access attempts.
- The OS provides authentication features for each user by means of passwords.

FUNCTION OF OPERATING SYSTEM

Function of Operating System

- Memory Management
- Processor Management
- Device Management
- File Management
- Security
- Resource sharing & protection
- Job accounting
- Error detection
- Coordination between other software and users

(Accounting)

Memory management

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.

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.

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.

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.

Other activities are :

Security − By means of password and similar other techniques, it prevents unauthorized access to programs and data.

Control over system performance − Recording delays between request for a service and response from the system.

Job accounting − Keeping track of time and resources used by various jobs and users.

Error detecting aids − Production of dumps, traces, error messages, and other debugging and error detecting aids.

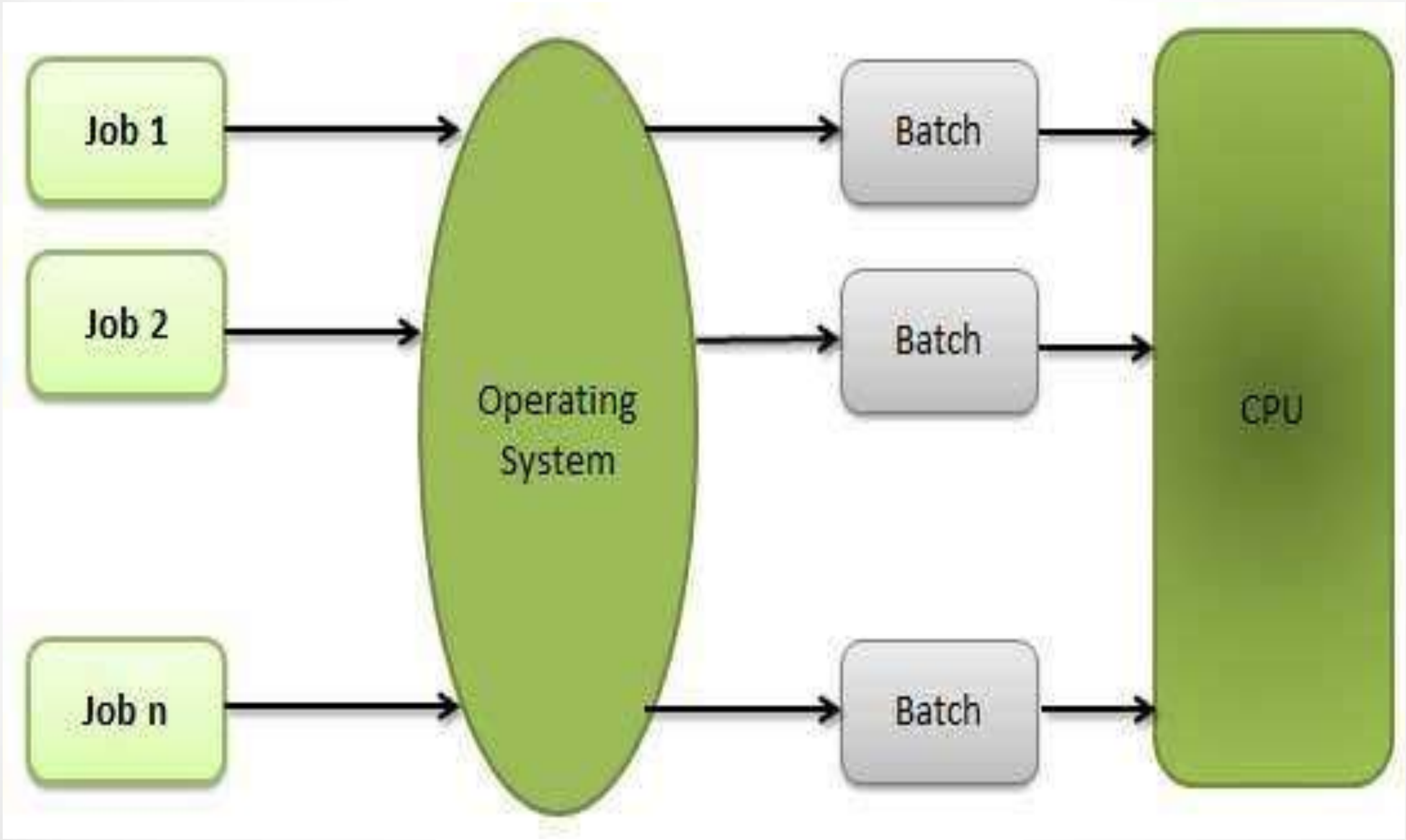
Coordination between other softwares and users − Coordination and assignment of compilers, interpreters, assemblers and other

TYPES OF OPERATING SYSTEM

Types of Operating System

1. Batch Operating System
2. Time-Sharing Operating System
3. Embedded Operating System
4. Multiprogramming Operating System
5. Network Operating System
6. Distributed Operating System
7. Multiprocessing Operating System
8. Real-Time Operating System

Batch Operating System



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.

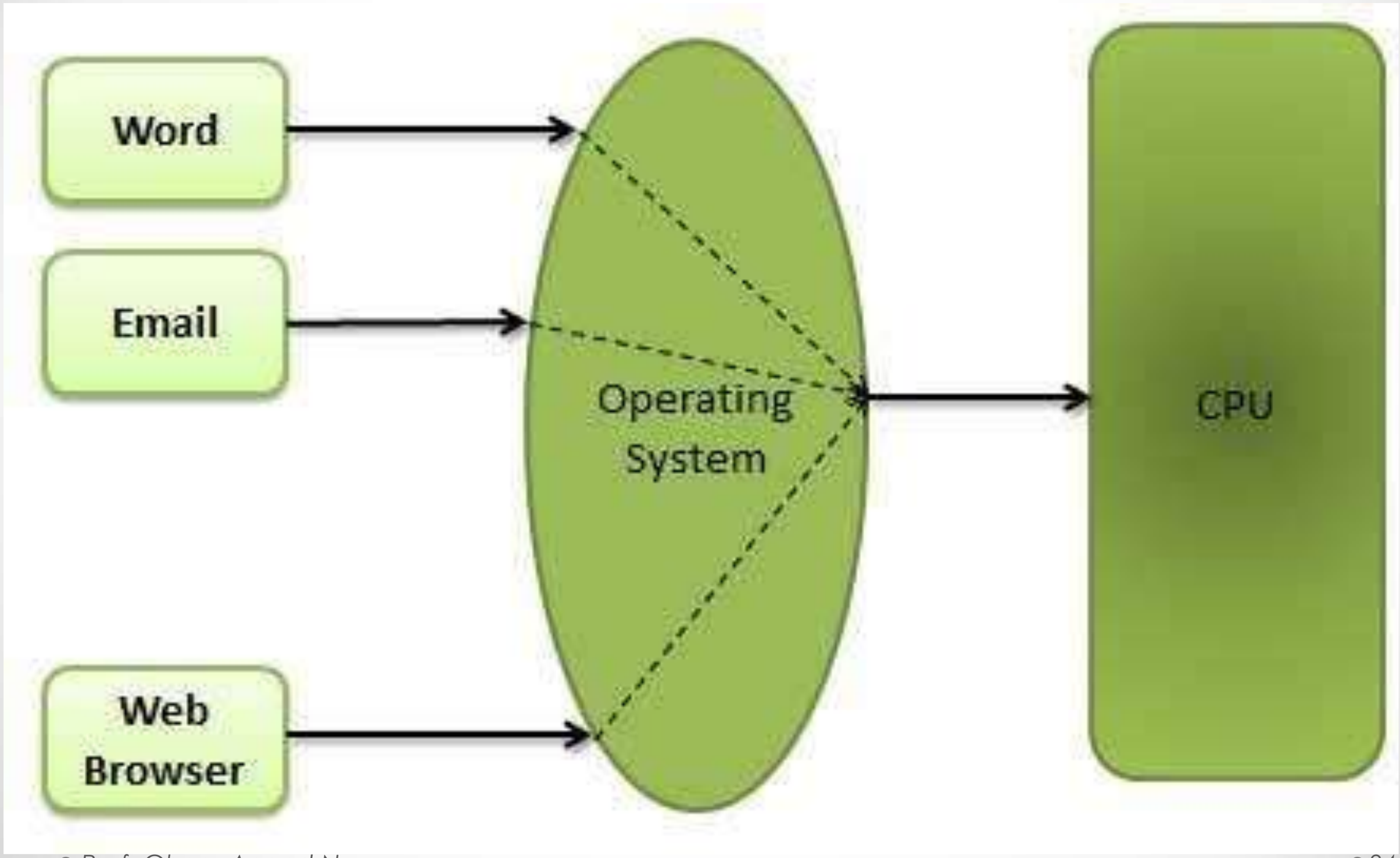
The problems with Batch Systems are as follows −

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.

Multitasking



Multitasking

Multitasking is when multiple jobs are executed by the CPU simultaneously by switching between them. Switches occur so frequently that the users may interact with each program while it is running. An OS does the following activities related to multitasking −

The user gives instructions to the operating system or to a program directly, and receives an immediate response.

The OS handles multitasking in the way that it can handle multiple operations/executes multiple programs at a time.

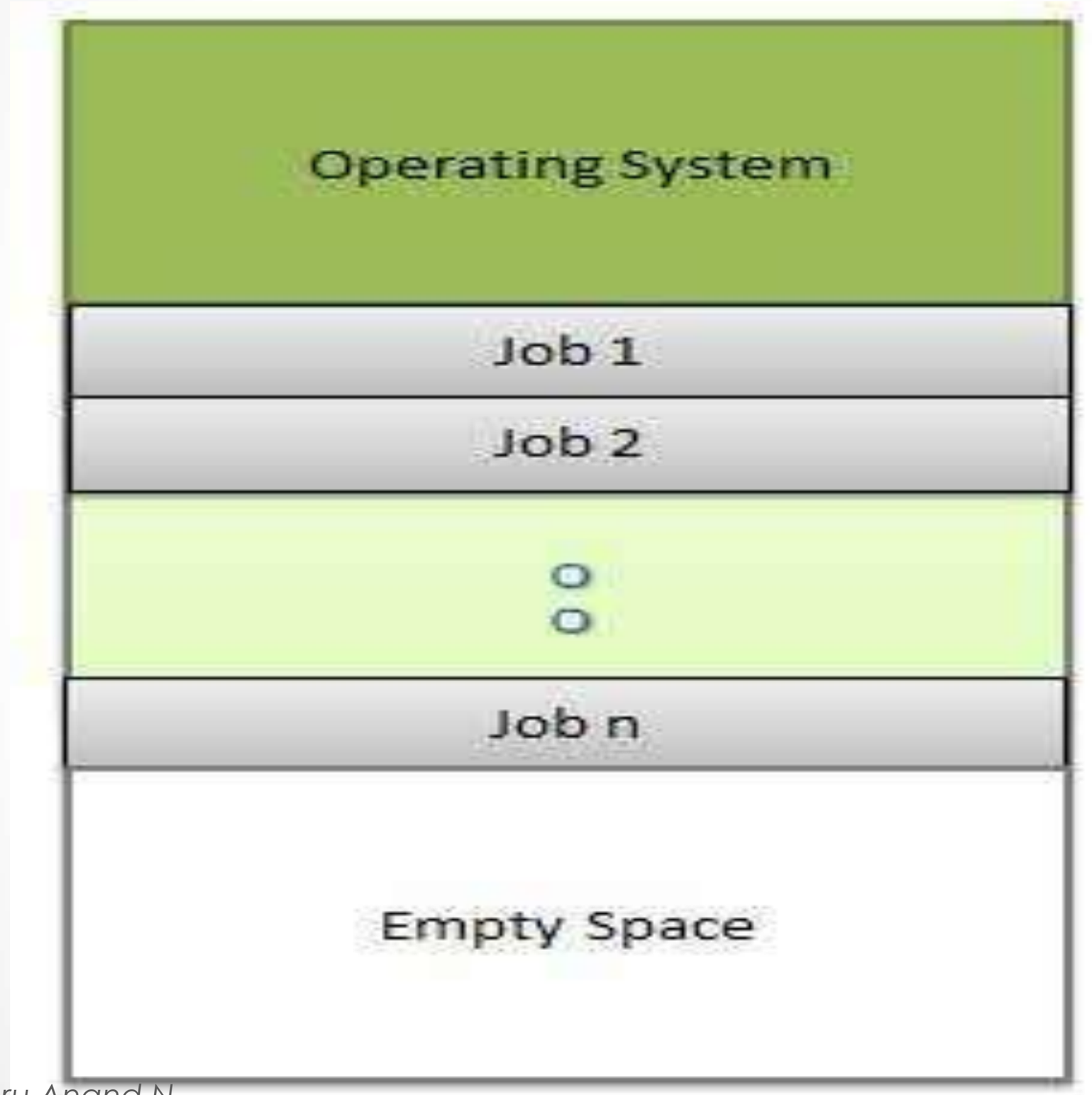
Multitasking Operating Systems are also known as Time-sharing systems.

These Operating Systems were developed to provide interactive use of a computer system at a reasonable cost.

A time-shared operating system uses the concept of CPU scheduling and multiprogramming to provide each user with a small portion of a time-shared CPU.

Each user has at least one separate program in memory.

Multiprogramming



Multiprogramming

When two or more programs reside in memory at the same time, is referred as **multiprogramming**. Multiprogramming assumes a single shared processor. Multiprogramming increases CPU utilization by organizing jobs so that the CPU always has one to execute.

An OS does the following activities related to multiprogramming.

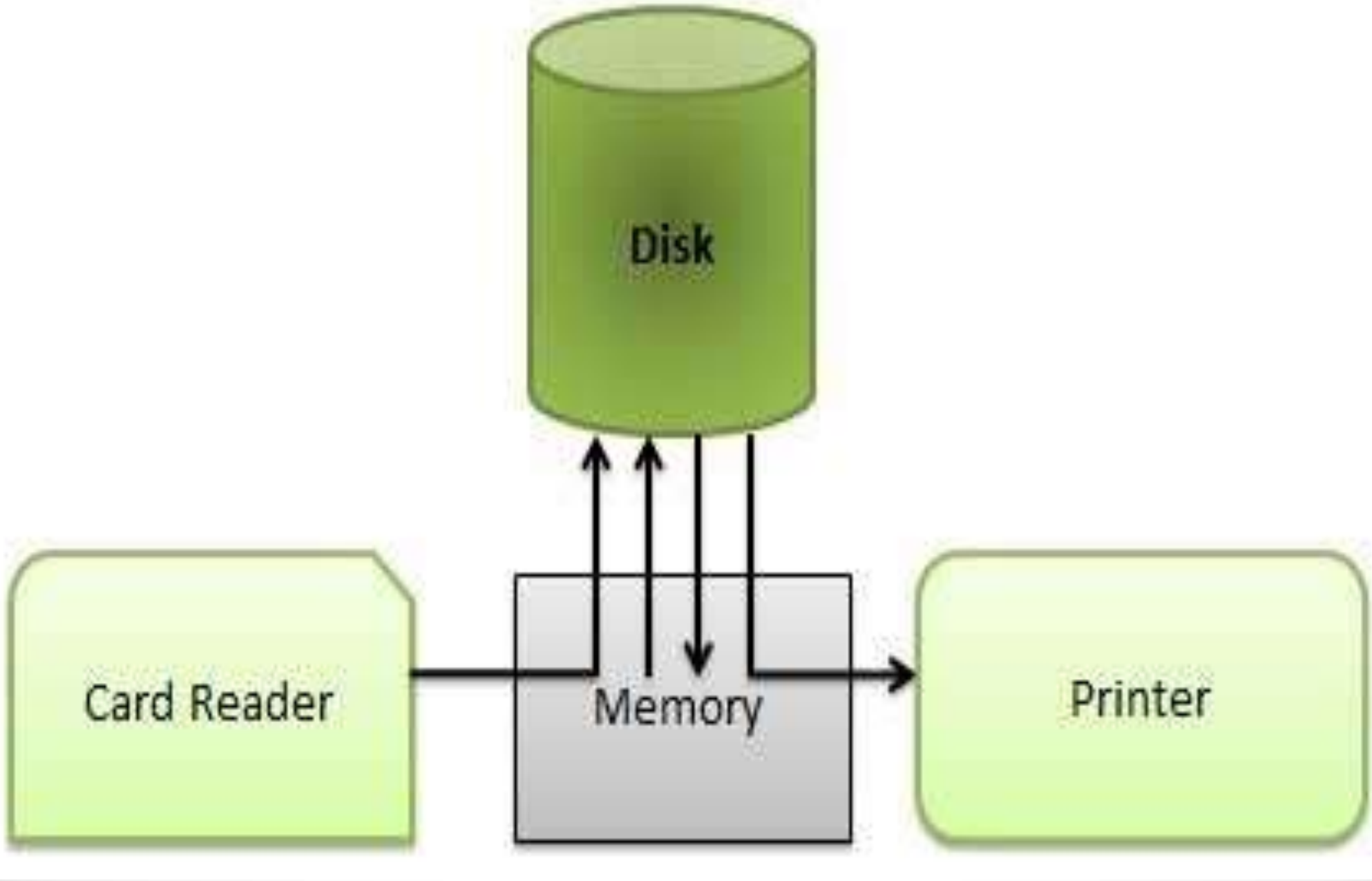
The operating system keeps several jobs in memory at a time.

This set of jobs is a subset of the jobs kept in the job pool.

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

Multiprogramming operating systems monitor the state of all active programs and system resources using memory management programs to ensures that the CPU is never idle, unless there are no jobs to process.

Spooling



Spooling

Spooling is an acronym for simultaneous peripheral operations on line.

Spooling refers to putting data of various I/O jobs in a buffer. This buffer is a special area in memory or hard disk which is accessible to I/O devices.

An operating system does the following activities related to distributed environment −

Handles I/O device data spooling as devices have different data access rates.

Maintains the spooling buffer which provides a waiting station where data can rest while the slower device catches up.

Maintains parallel computation because of spooling process as a computer can perform I/O in parallel fashion. It becomes possible to have the computer read data from a tape, write data to disk and to write out to a tape

Time sharing Operating System

Time-sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing.

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.

Distributed Operating System

Distributed systems use multiple central processors to serve multiple real-time applications and multiple users. Data processing jobs are distributed among the processors accordingly.

The processors communicate with one another through various communication lines (such as high-speed buses or telephone lines). These are referred as loosely coupled systems or distributed systems. Processors in a distributed system may vary in size and function. These processors are referred as sites, nodes, computers, and so on.

The advantages of distributed systems are as follows −

- Speedup the exchange of data with one another via electronic mail.
- If one site fails in a distributed system, the remaining sites can potentially continue operating.
- Better service to the customers.
- Reduction of the load on the host computer.
- Reduction of delays in data processing.

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.

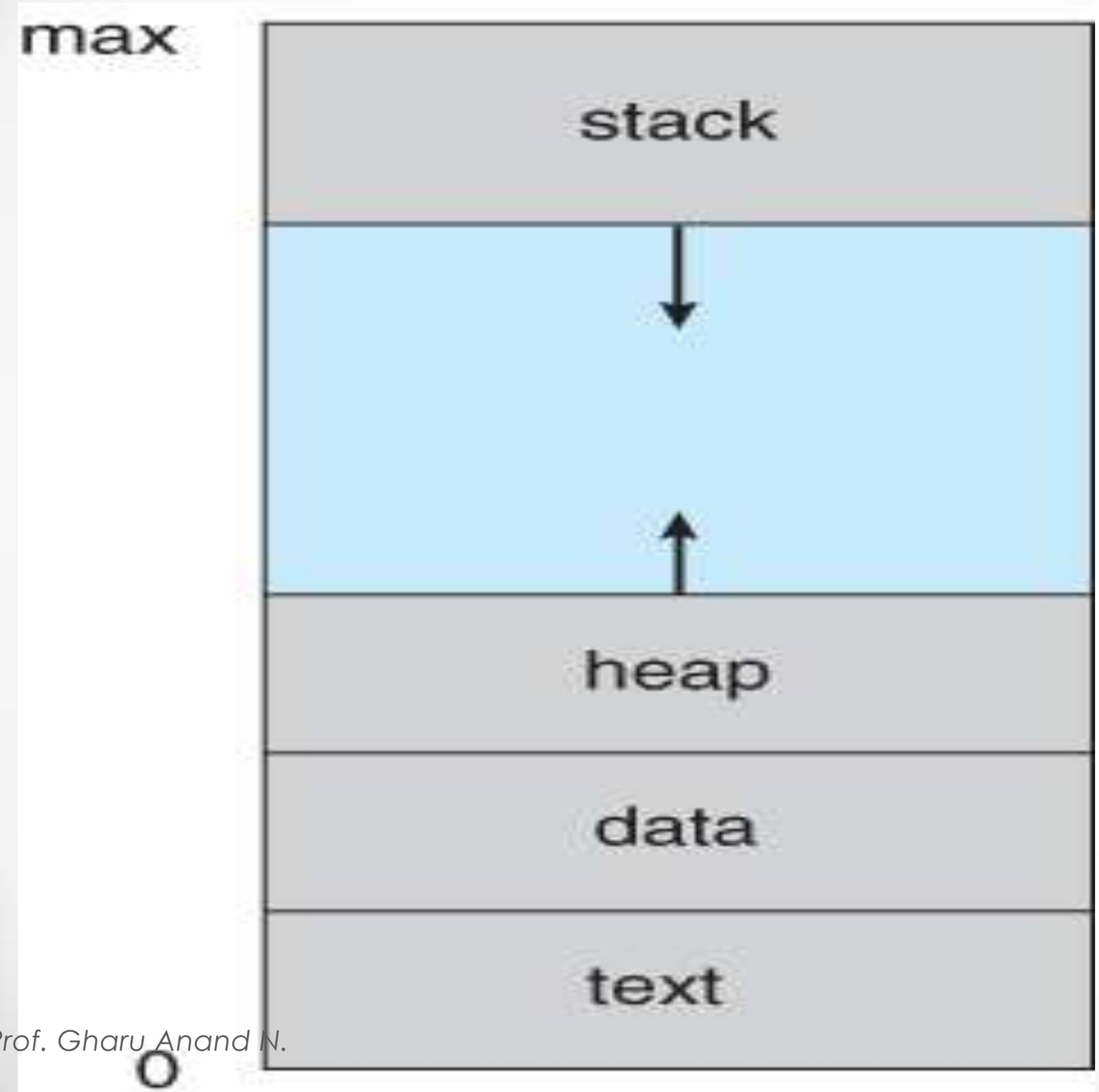
Real time Operating System

A real-time system is defined as a data processing system in which the time interval required to process and respond to inputs is so small that it controls the environment. The time taken by the system to respond to an input and display of required updated information is termed as the **response time**. So in this method, the response time is very less as compared to online processing.

Example, Scientific experiments, medical imaging systems, industrial control systems, weapon systems, robots, air traffic control systems, etc.

PROCESS MANAGEMENT CONCEPTS

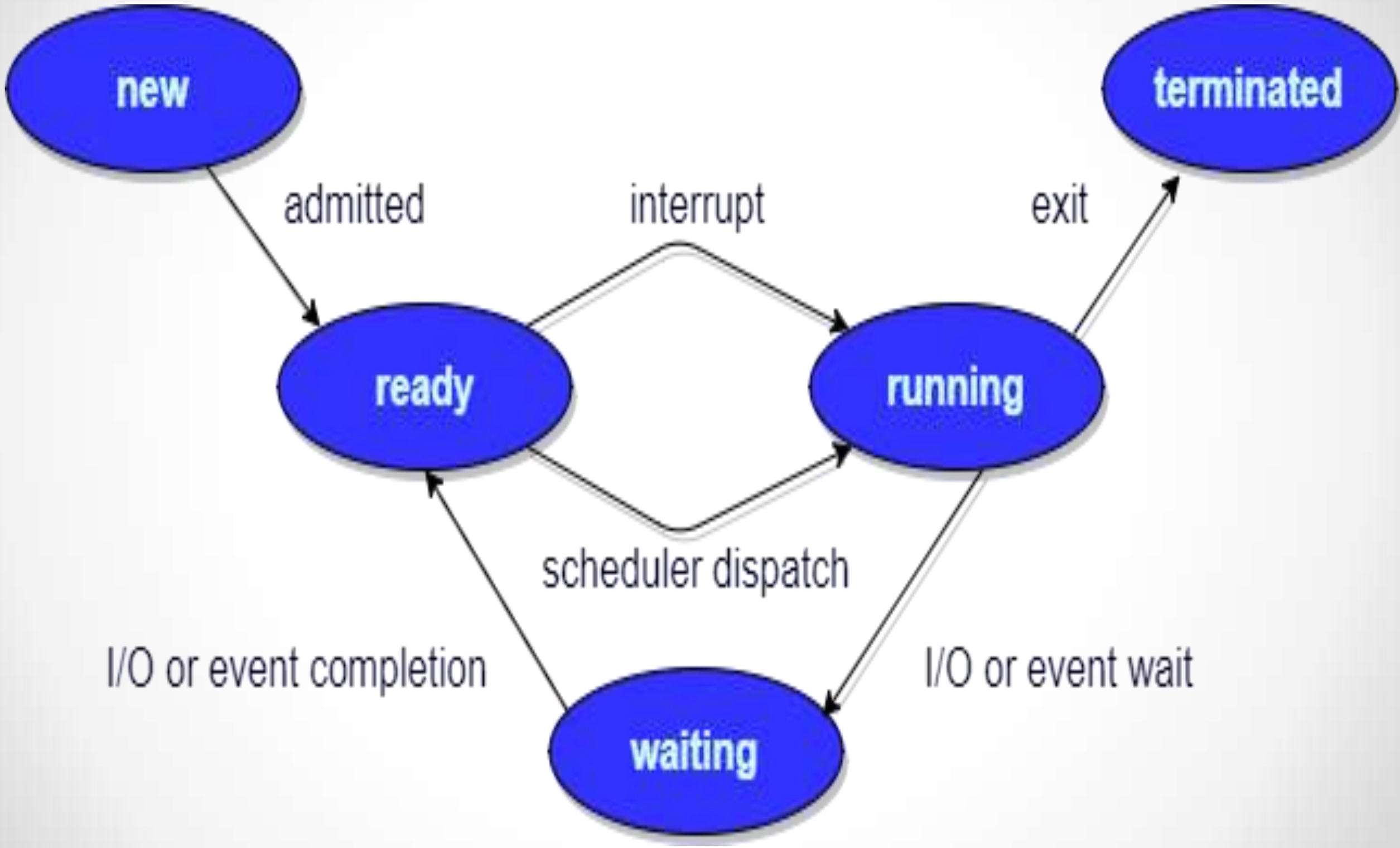
Process



Process

- **A process is a program in execution.** Process is not as same as program code but a lot more than it. A process is an 'active' entity as opposed to program which is considered to be a 'passive' entity. Attributes held by process include hardware state, memory, CPU etc.
- Process memory is divided into four sections for efficient working :
- The **text section** is made up of the compiled program code, read in from non-volatile storage when the program is launched.
- The **data section** is made up the global and static variables, allocated and initialized prior to executing the main.
- The **heap** is used for the dynamic memory allocation, and is managed via calls to new, delete, malloc, free, etc.
- The **stack** is used for local variables. Space on the stack is reserved for local variables when they are declared.

Five State Process



Five State Process

1. **New** − When a new process is created, It enter into the new state. Then it tries to load into RAM.

2. **Ready** − The processes that are loaded on RAM and waiting for CPU are in ready state.

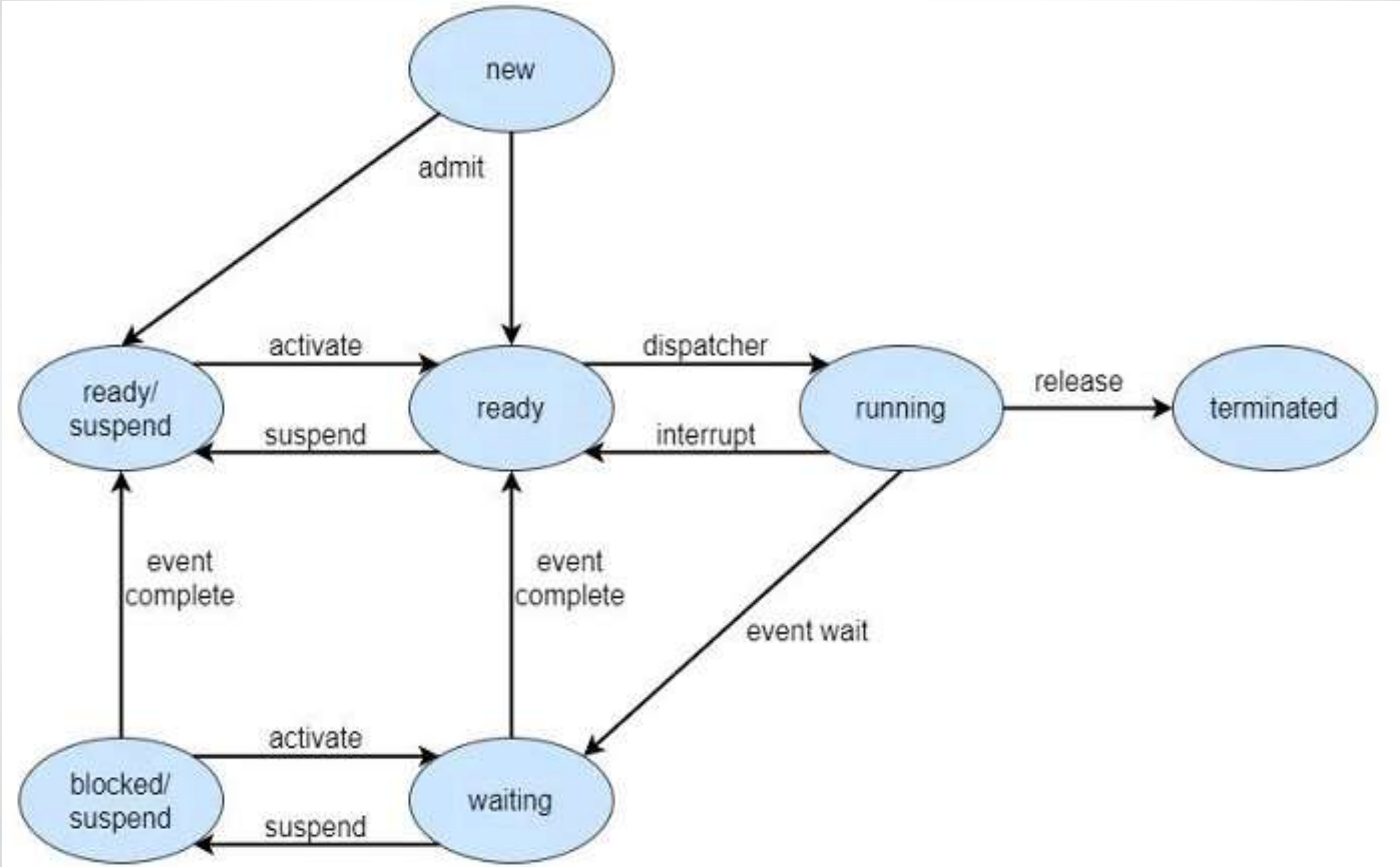
3. **Running** − The processes that are running on the CPU are in running state.

If the process is running in its critical section, then other processes need to wait in the ready state.

4. Blocked − All processes that are leaving the CPU and moving to the waiting state are in the blocked state. When the CPU becomes free, processes from the blocked state again move to the ready state, and from ready to Running state.

5. Exit / Terminated − A process that is terminated from CPU and RAM is in the terminate state..

Seven State Process



Seven State Process

1. **New** − Contains the processes that are newly coming for execution.
2. **Ready** − Contains the processes that are present in main memory and available for execution.
3. **Running** − Contains the process that is running or executing.
4. **Exit** − Contains the processes that complete its execution.
5. **Blocked** − Contains the processes that are present in main memory and awaiting an event to occur.
6. **Blocked Suspend** − It contains the process present in secondary memory and awaits an event to occur.
7. **Ready Suspend** − Contains the processes that are present in secondary memory but is available for execution as soon as it is loaded into main memory.

Process Control Block

- Process state
- Program Counter
- CPU registers
- CPU scheduling Information
- Accounting & Business information
- Memory-management information
- I/O status information

Process Control Block

Process State - This specifies the process state i.e. new, ready, running, waiting or terminated..

Process ID, and parent process ID.: This shows the number of the particular process

Program Counter - This contains the address of the next instruction that needs to be executed in the process.

Registers -

This specifies the registers that are used by the process. They may include accumulators, index registers, stack pointers, general purpose registers etc.

Process Control Block

List of Open Files :

These are the different files that are associated with the process

CPU Scheduling Information :

The process priority, pointers to scheduling queues etc. is the CPU scheduling information that is contained in the PCB. This may also include any other scheduling parameters.

Memory Management Information :

The memory management information includes the page tables or the segment tables depending on the memory system used. It also contains the value of the base registers, limit registers etc..

Process Control Block

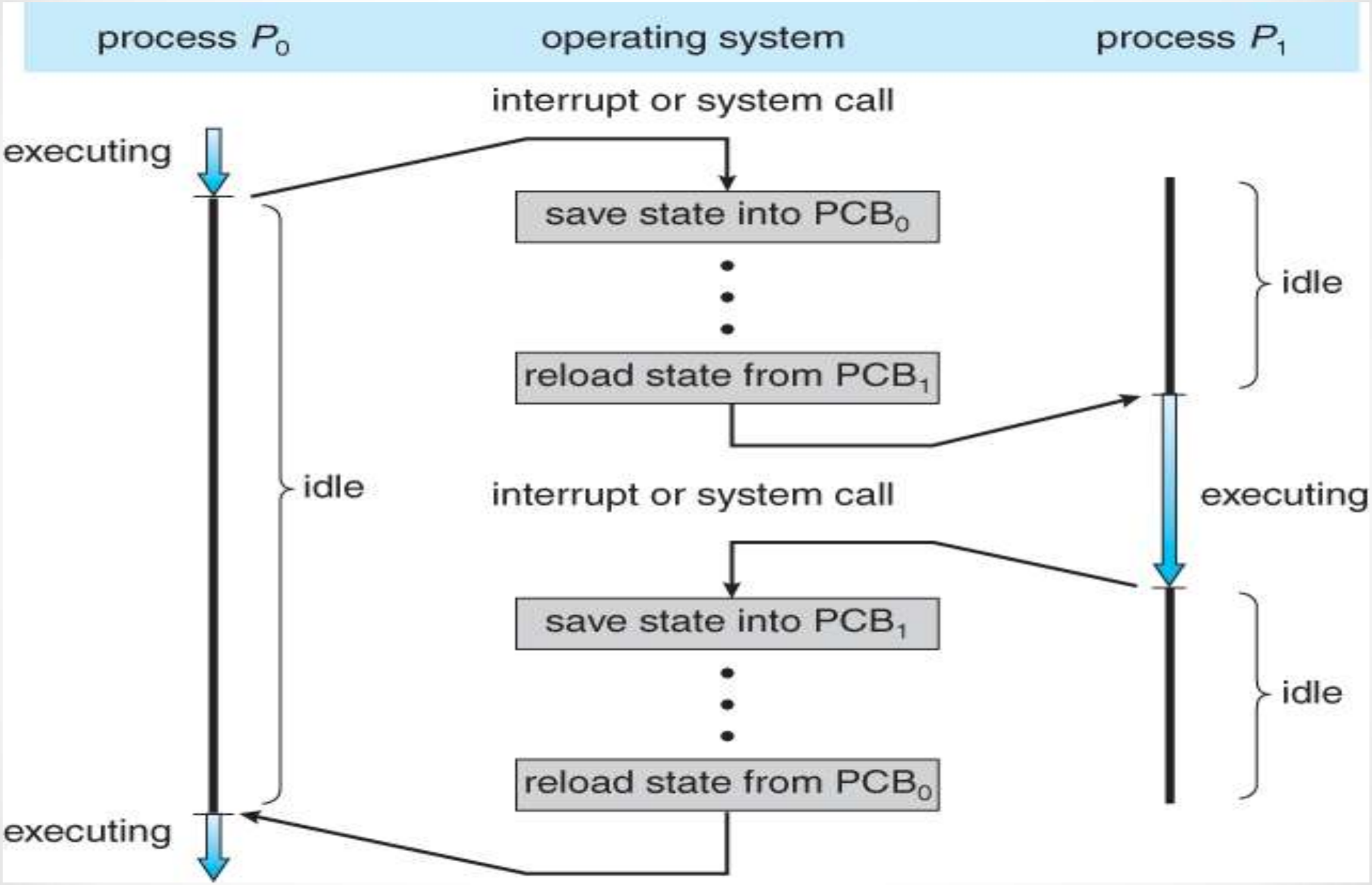
I/O Status Information :

This information includes the list of I/O devices used by the process, the list of files etc

Accounting information :

The time limits, account numbers, amount of CPU used, process numbers etc. are all a part of the PCB accounting information

Process Control Block



CPU switch from process to process using PCB

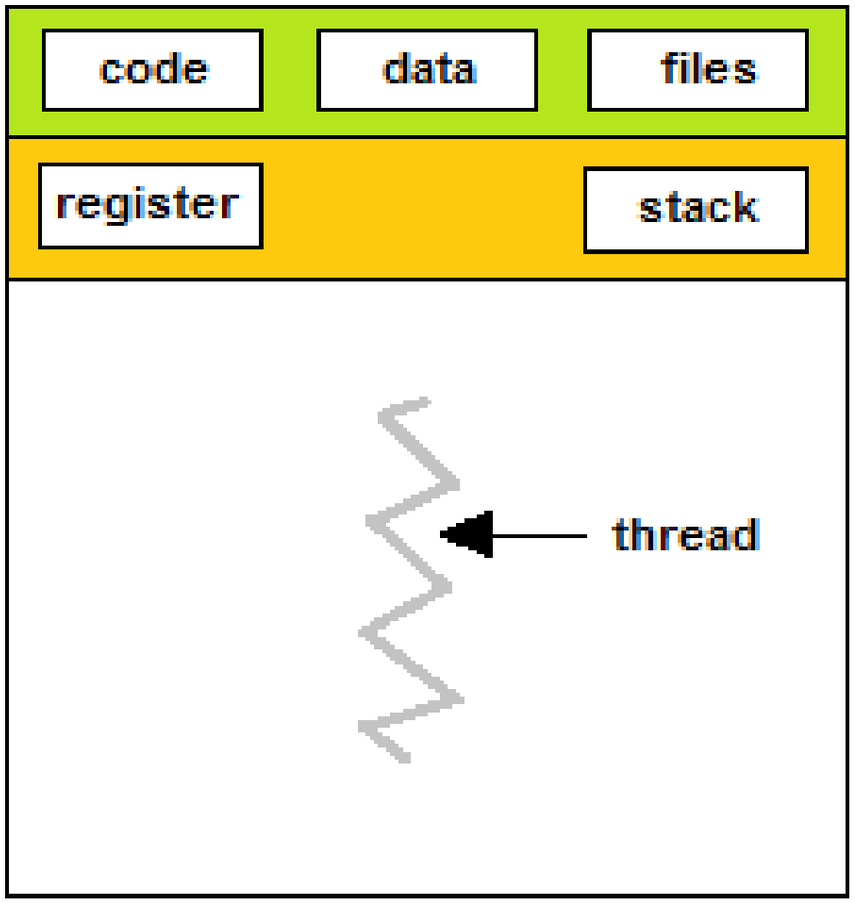
THREAD IN

OS

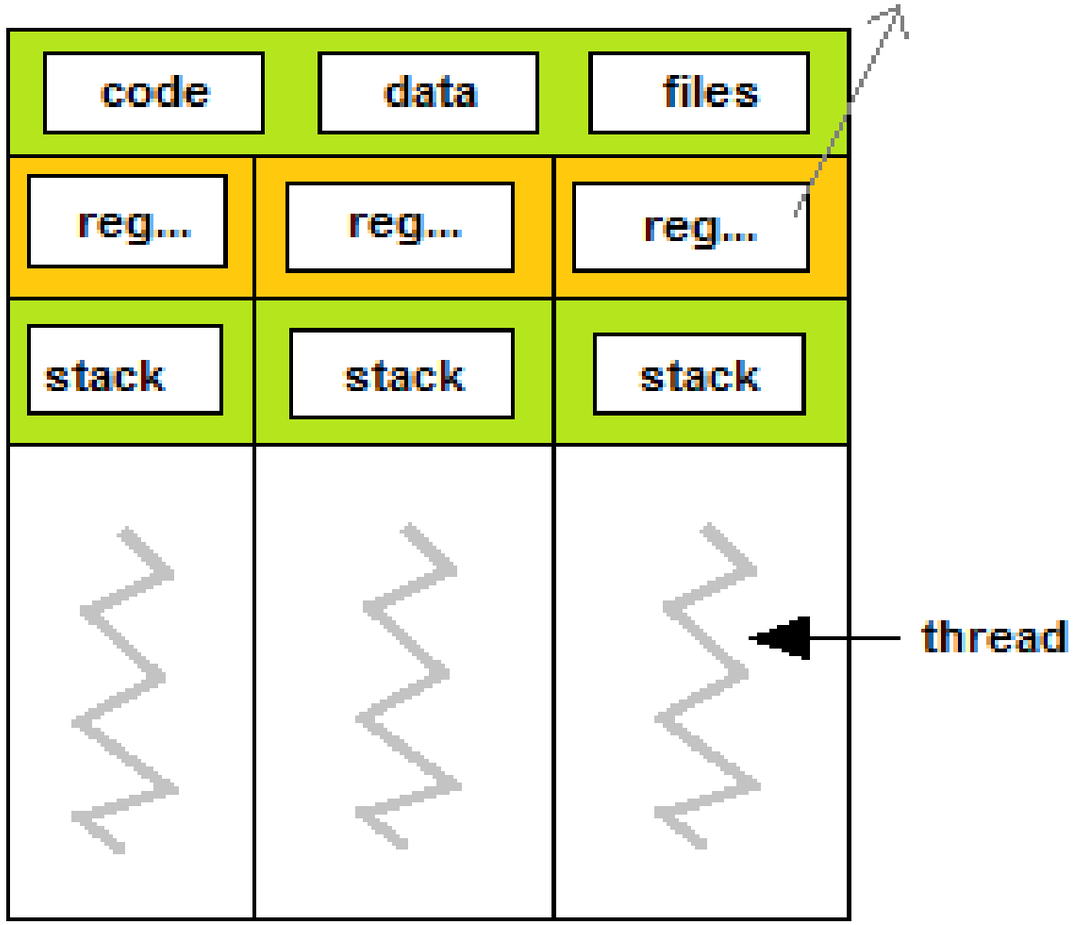
Thread concepts

Thread is an execution unit which consists of its own program counter, a stack, and a set of registers. Threads are also known as Lightweight processes. Threads are popular way to improve application through parallelism. The CPU switches rapidly back and forth among the threads giving illusion that the threads are running in parallel.

As each thread has its own independent resource for process execution, multpile processes can be executed parallely by increasing number of threads.



single-threaded process



multithreaded process

Types of Thread

There are two types of threads :

1. User Threads

2. Kernel Threads

User threads, are above the kernel and without kernel support. These are the threads that application programmers use in their programs.

Kernel threads are supported within the kernel of the OS itself. All modern OSs support kernel level threads, allowing the kernel to perform multiple simultaneous tasks and/or to service multiple kernel system calls simultaneously.

Advantages of Thread

Responsiveness

Resource sharing, hence allowing better utilization of resources.

Economy. Creating and managing threads becomes easier.

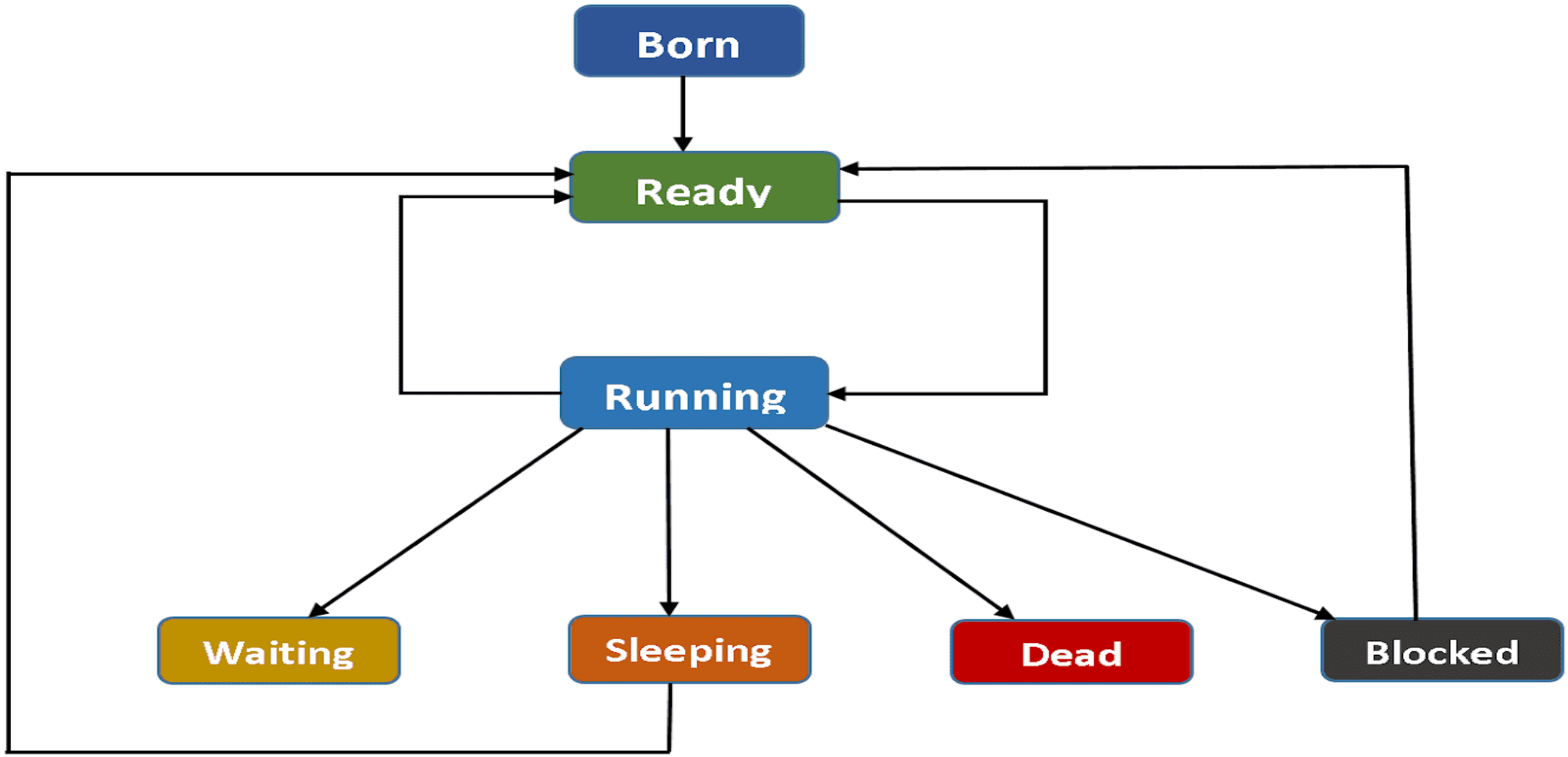
Scalability. One thread runs on one CPU. In Multithreaded processes, threads can be distributed over a series of processors to scale.

Context Switching is smooth. Context switching refers to the procedure followed by CPU to change from one task to another.

.

Lifecycle of Thread

The life cycle of a thread in an operating system involves the creation, scheduling, execution, blocking, and termination. The operating system plays a critical role in managing the life cycle of threads, ensuring that they run efficiently and effectively.



Process Vs Thread

Parameter	Process	Thread
Definition	Process means a program is in execution.	Thread means a segment of a process.
Lightweight	The process is not Lightweight.	Threads are Lightweight.
Termination time	The process takes more time to terminate.	The thread takes less time to terminate.
Creation time	It takes more time for creation.	It takes less time for creation.
Communication	Communication between processes needs more time compared to thread.	Communication between threads requires less time compared to processes.
Context switching time	It takes more time for context switching.	It takes less time for context switching.
Resource	Process consume more resources.	Thread consume fewer resources.
Treatment by OS	Different process are tread separately by OS.	All the level peer threads are treated as a single task by OS.
Memory	The process is mostly isolated.	Threads share memory.
Sharing	It does not share data	Threads share data with each other.

Process Vs Thread

Sr. No.	Process	Thread
1.	Process means any program is in execution.	Thread means segment of a process.
2.	Process takes more time to terminate.	Thread takes less time to terminate.
3.	It takes more time for creation.	It takes less time for creation.
4.	It also takes more time for context switching.	It takes less time for context switching.
5.	Process is less efficient in term of communication.	Thread is more efficient in term of communication.
6.	Process consume more resources.	Thread consume less resources.
7.	Process is isolated.	Threads share memory.
8.	Process is called heavy weight process.	Thread is called light weight process.
9.	Process switching uses interface in operating system.	Thread switching does not require to call a operating system and cause an interrupt to the kernel.
10.	If one process is blocked then it will not effect the execution of other	Second thread in the same task could not run, while one server thread is

Component of OS

- **System call :**

In computing, a **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

System call

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 : 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

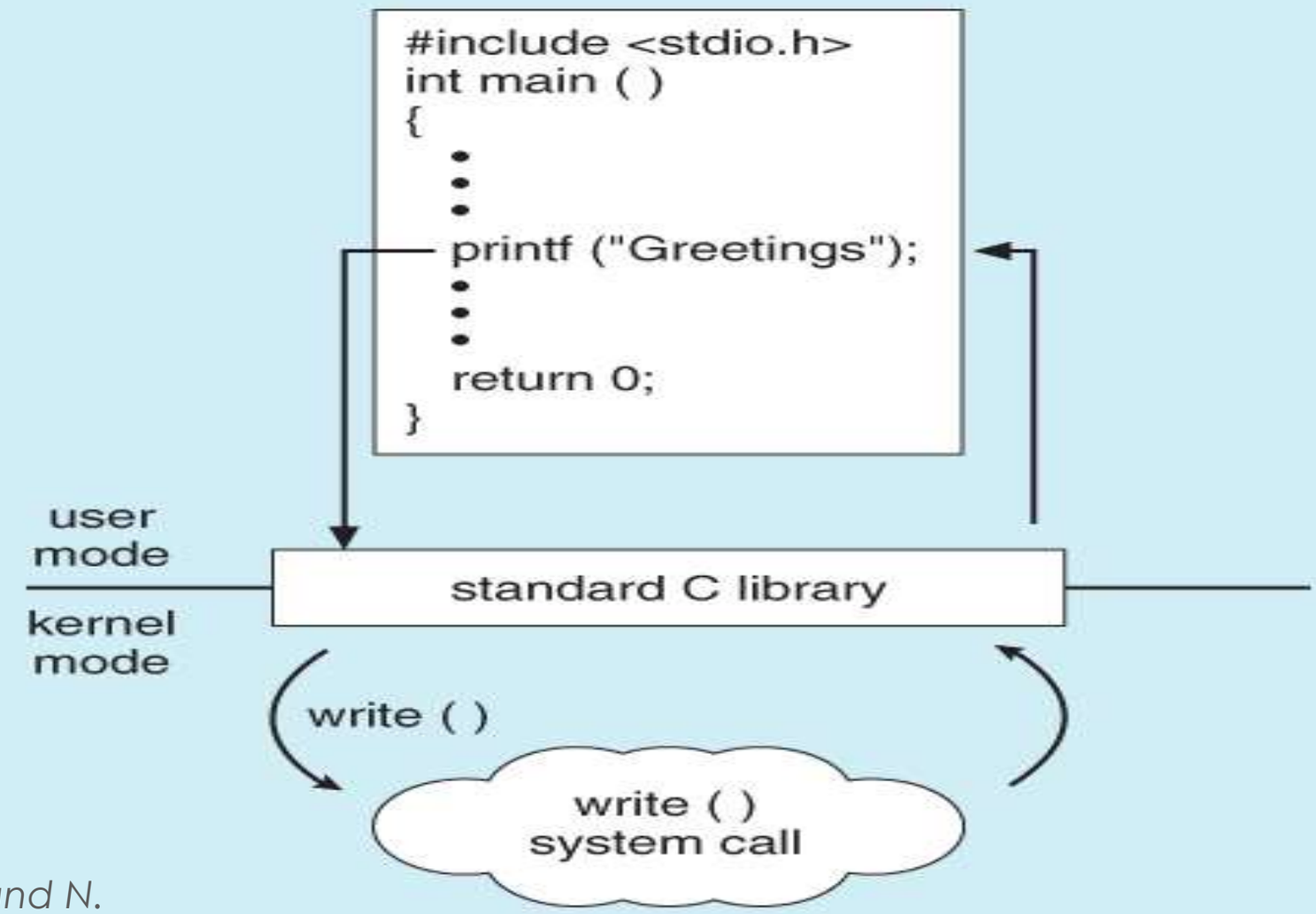
Information maintenance

Communication

System call

EXAMPLE OF STANDARD C LIBRARY

The standard C library provides a portion of the system-call interface for many versions of UNIX and Linux. For example, let's assume a C program invokes the printf( ) statement. The C library intercepts this call and invokes the necessary ssystem call(s) in the operating system - in this instance, the write( ) system call. The C library takes the value returned by write( ) and passes it back to the user program. This is shown below:



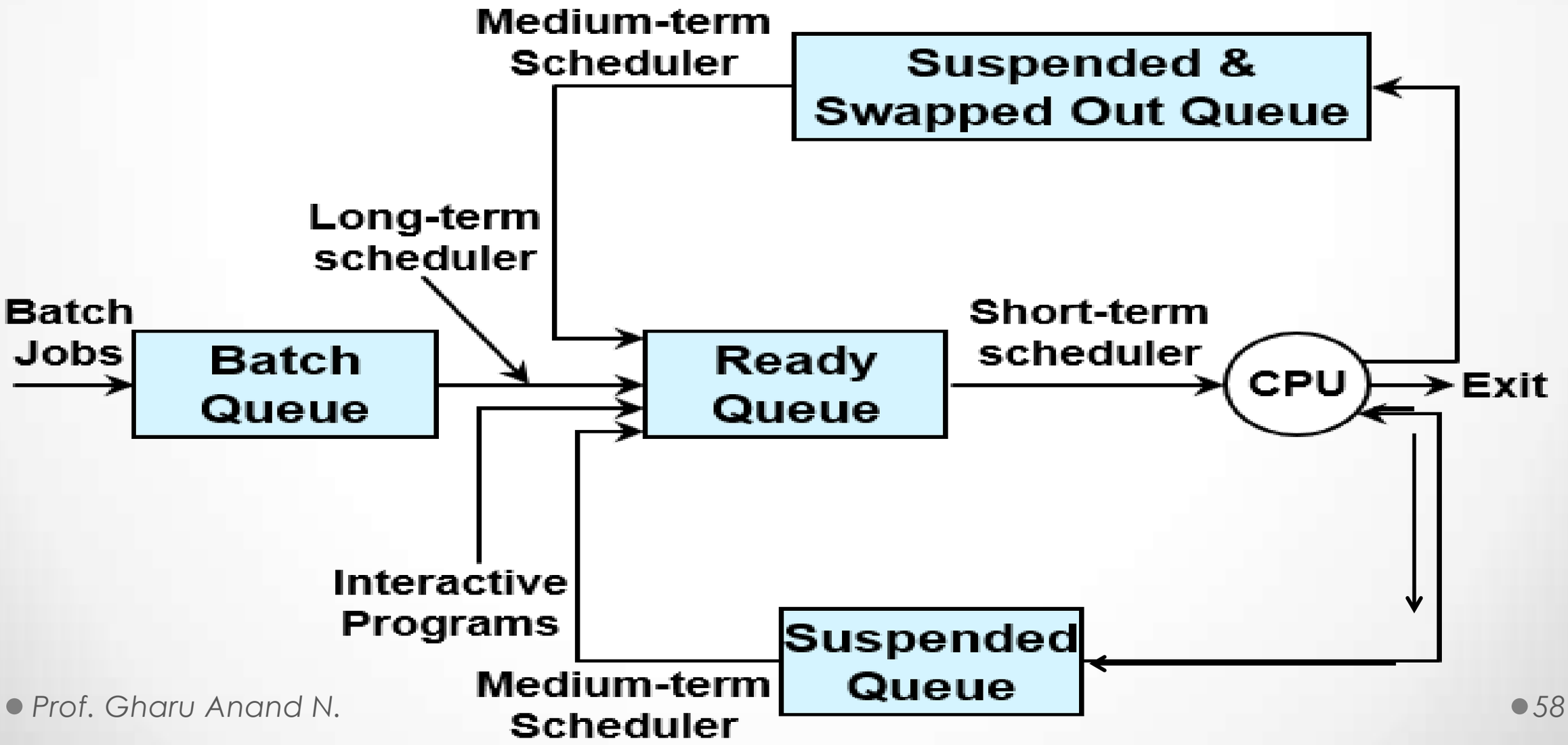
System call

WINDOWS	UNIX	
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
File Manipulation	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Device Manipulation	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleeo()	getpid() alarm() sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shmget() mmap()
Protection	SetFileSecurity() InitlializeSecurityDescriptor() SetSecurityDescriptorGroup()	chmod() umask() chown()

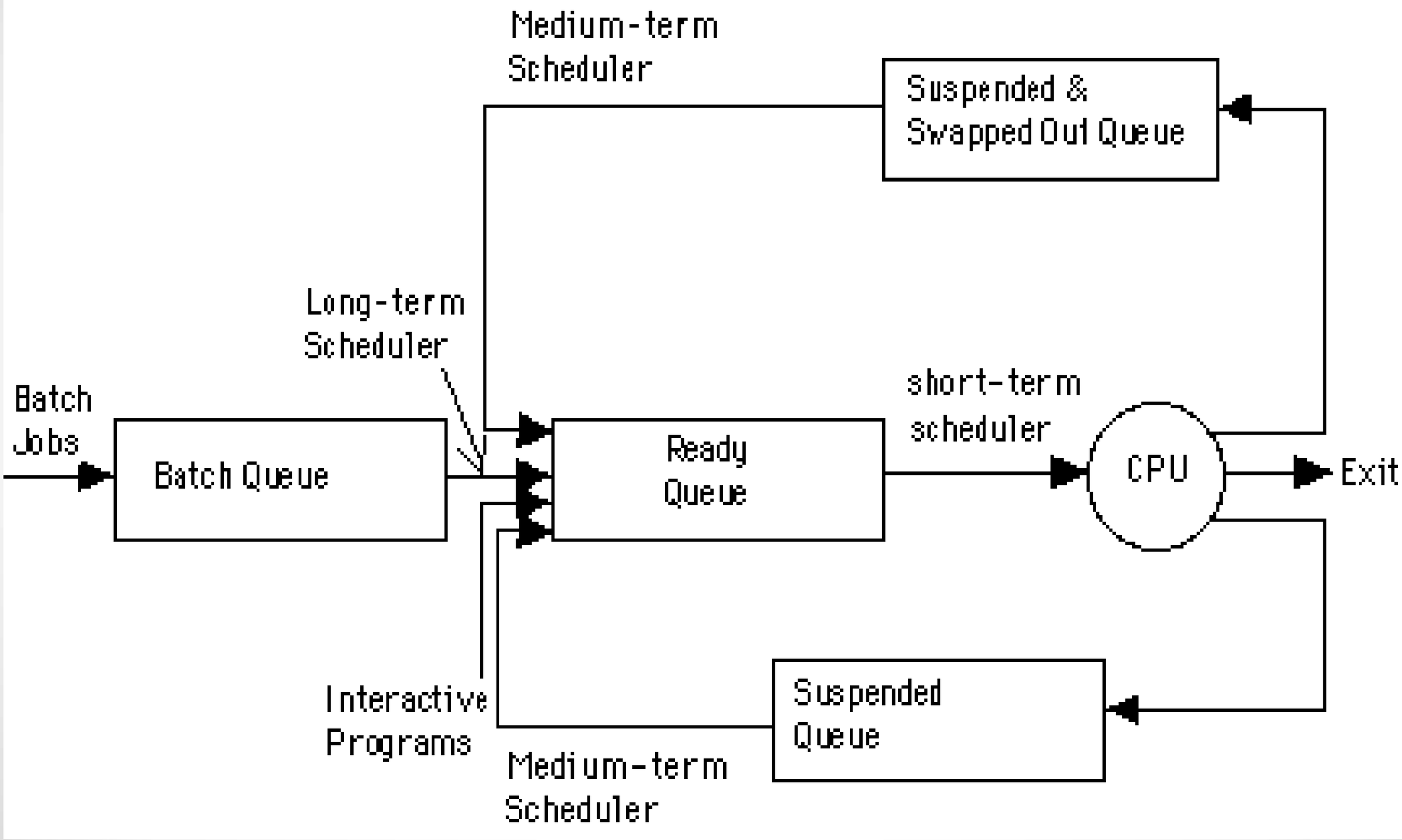
PROCESS SCHEDULLING

Process Schedulling

The **process scheduling** is the activity of the **process** manager that handles the removal of the running **process** from the CPU and the selection of another **process** on the basis of a particular strategies. **Process scheduling** is an essential part of a Multiprogramming **operating systems**.



Process Schedulling



Process Schedulling

An operating system uses two types of scheduling processes execution, **preemptive** and **non - preemptive**.

1. Preemptive process:

In preemptive scheduling policy, a low priority process has to be suspend its execution if high priority process is waiting in the same queue for its execution.

2. Non - Preemptive process:

In non - preemptive scheduling policy, processes are executed in first come first serve basis, which means the next process is executed only when currently running process finishes its execution.

.

Types of scheduler

1) Long Term Scheduler

It selects the process that are to be placed in ready queue. The long term scheduler basically decides the priority in which processes must be placed in main memory. Processes of long term scheduler are placed in the ready state because in this state the process is ready to execute waiting for calls of execution from CPU which takes time that's why this is known as long term scheduler.

Types of scheduler

2) Mid – Term Scheduler

It places the blocked and suspended processes in the secondary memory of a computer system. The task of moving from main memory to secondary memory is called **swapping out**.The task of moving back a swapped out process from secondary memory to main memory is known as **swapping in**. The swapping of processes is performed to ensure the best utilization of main memory.

3) Short Term Scheduler

It decides the priority in which processes is in the ready queue are allocated the central processing unit (CPU) time for their execution. The short term scheduler is also referred as central processing unit (CPU) scheduler.

Compare types of scheduler

S.N.	Long-Term Scheduler	Short-Term Scheduler	Medium-Term Scheduler
1	It is a job scheduler	It is a CPU scheduler	It is a process swapping scheduler.
2	Speed is lesser than short term scheduler	Speed is fastest among other two	Speed is in between both short and long term scheduler.
3	It controls the degree of multiprogramming	It provides lesser control over degree of multiprogramming	It reduces the degree of multiprogramming.
4	It is almost absent or minimal in time sharing system	It is also minimal in time sharing system	It is a part of Time sharing systems.
5	It selects processes from pool and loads them into memory for execution	It selects those processes which are ready to execute	It can re-introduce the process into memory and execution can be continued.

Schedulling criteria

CPU utilization : To make out the best use of CPU and not to waste any CPU cycle, CPU would be working most of the time(Ideally 100% of the time). Considering a real system, CPU usage should range from 40% (lightly loaded) to 90% (heavily loaded.)

Throughput : It is the total number of processes completed per unit time or rather say total amount of work done in a unit of time. This may range from 10/second to 1/hour depending on the specific processes.

Turnaround time : It is the amount of time taken to execute a particular process, i.e. The interval from time of submission of the process to the time of completion of the process(Wall clock time).

Schedulling criteria

Waiting time : The sum of the periods spent waiting in the ready queue amount of time a process has been waiting in the ready queue to acquire get control on the CPU.

Load average : It is the average number of processes residing in the ready queue waiting for their turn to get into the CPU.

Response time : Amount of time it takes from when a request was submitted until the first response is produced. Remember, it is the time till the first response and not the completion of process execution(final response).

SCHEDULLING ALGORITHMS

Schedulling Algorithms

1. First-Come, First-Served (FCFS) Scheduling
2. Shortest-Job-First (SJF) Scheduling
3. Priority Scheduling
4. Round Robin(RR) Scheduling
5. SJF Preemptive Scheduling

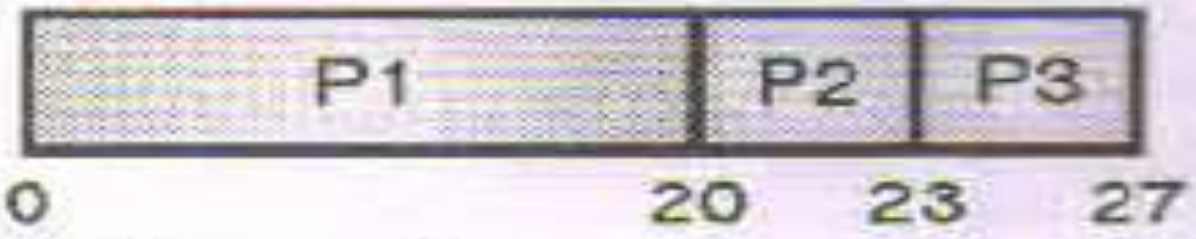
FCFS Algorithms

- Jobs are executed on first come, first serve basis.
- It is a non-preemptive, pre-emptive scheduling algorithm.
- Easy to understand and implement.
- Its implementation is based on FIFO queue.
- Poor in performance as average wait time is high

FCFS Algorithms

Consider the following example of three processes :

Process	Burst Time / Execution Time
P1	20
P2	3
P3	4



Start and finish times of processes are :

Process	Start time	CPU Time	Finish time
P1	0	20	20
P2	20	3	23
P3	23	4	27

FCFS Algorithms

The two important parameters :

1. Turnaround time.
2. Waiting time.

Which can be calculated as given below :

Turnaround time = Finish time – Arrival time

Waiting time = Turnaround time – CPU time.

Sr. No.	Process No.	CPU time	Start time	Finish time	Turnaround time	Waiting time
1.	P1	20	0	20	20	0
2.	P2	3	20	23	23	20
3.	P3	4	23	27	27	23
				Average →	70/3	43/3
					= 23.33	= 14.33

FCFS Algorithms

☞ **Conclusions**

1. FCFS algorithm is simple to implement.
2. FCFS algorithm results in poor performance. Average turnaround and waiting time are quite high.
3. FCFS meets the fairness criteria of scheduling.
4. FCFS algorithm can not be used in time sharing system, where it is important that each user gets a share of the CPU at regular interval. It is not desirable to allow one process to keep the CPU for a long time.

FCFS Algorithms

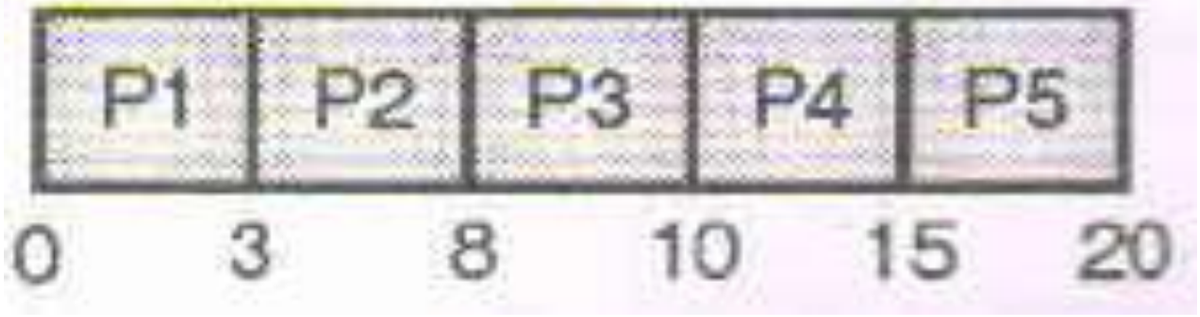
Example 6.10.1

Consider the following set of processes for FCFS scheduling

Process Name	Arrival Time	Processing Time
P1	0	3
P2	1	5
P3	3	2
P4	9	5
P5	12	5

Find waiting time and turnaround time of each process, also calculate the average turnaround time and waiting time.

FCFS Algorithms



TAT=FT-AT WT=TAT-BT

Sr. No.	Process No.	Arrival Time	CPU Time	Start Time	Finish Time	Turnaround Time	Waiting Time
1.	P1	0	3	0	3	3 – 0 = 3	3 – 3 = 0
2.	P2	1	5	3	8	8 – 1 = 7	7 – 5 = 2
3.	P3	3	2	8	10	10 – 3 = 7	7 – 2 = 5
4.	P4	9	5	10	15	15 – 9 = 6	6 – 5 = 1
5.	P5	12	5	15	20	20 – 12 = 8	8 – 5 = 3
			Average →			31/5	11/5
						= 6.2	= 2.2

FCFS Algorithms

UEx. 4.15.1 (SPPU - Q. 3(c), Dec. 16, 6 Marks)

Consider the following processes where Arrival and Burst time are as shown in Table P. 4.15.1 :

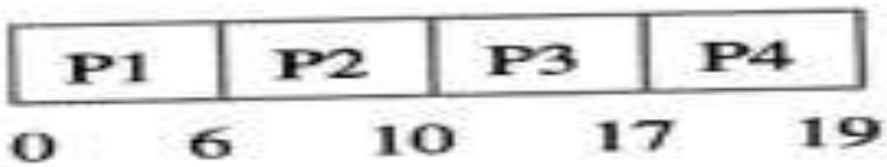
Calculate the Average Waiting Time and Average Turn-around Time if the processors are scheduled using FCFS.

Table P. 4.15.1

Process	Burst Time	Arrival Time
P1	06	0
P2	04	1
P3	07	3
P4	02	5

☑ **Soln. :**

Process	Burst Time	Arrival Time
P1	06	0
P2	04	1
P3	07	3
P4	02	5



FCFS Algorithms

Waiting time for

P1 = 0

P2 = 6

P3 = 10

P4 = 17

$$\text{Average waiting time} = \frac{0 + 6 + 10 + 17}{4}$$

$$= 8.25 \text{ ms}$$

Turn around time for P1 = 6

P2 = 10

P3 = 17

P4 = 19

$$\text{Average turnaround time} = \frac{6 + 10 + 17 + 19}{4}$$

$$= 13 \text{ ms}$$

The meaning of the lower turnaround time is; processes requests have to wait for a shorter time in order to obtain the required service from the system.

FCFS Algorithms

UEx. 4.15.2 (SPPU - Q. 1(c), Dec. 18, 6 Marks)

Consider the following process where the arrival and burst time as shown in Table P. 4.15.2. Calculate average waiting time and turnaround time using FCFS algorithm.

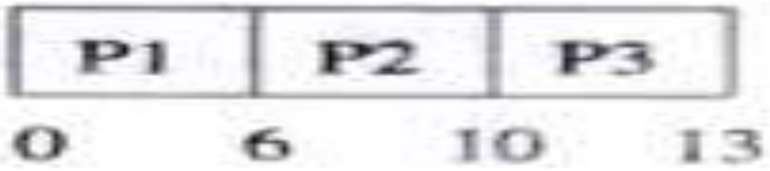
Table P. 4.15.2

Process	Burst time	Arrival time
P1	6	0
P2	4	4
P3	3	2

FCFS Algorithms

☑ **Soln. :**

Process	Burst time	Arrival time
P1	6	0
P2	4	4
P3	3	2



Waiting time for

P1 = 0

P2 = 6

P3 = 10

$$\text{Average waiting time} = \frac{0 + 6 + 10}{3} = \frac{16}{3}$$

$$= 5.3333 \text{ ms}$$

Turn around time for

P1 = 65

P2 = 10

P3 = 13

$$\text{Average turnaround time} = \frac{6 + 10 + 13}{3}$$

$$= \frac{29}{3} = 9.6666 \text{ ms}$$

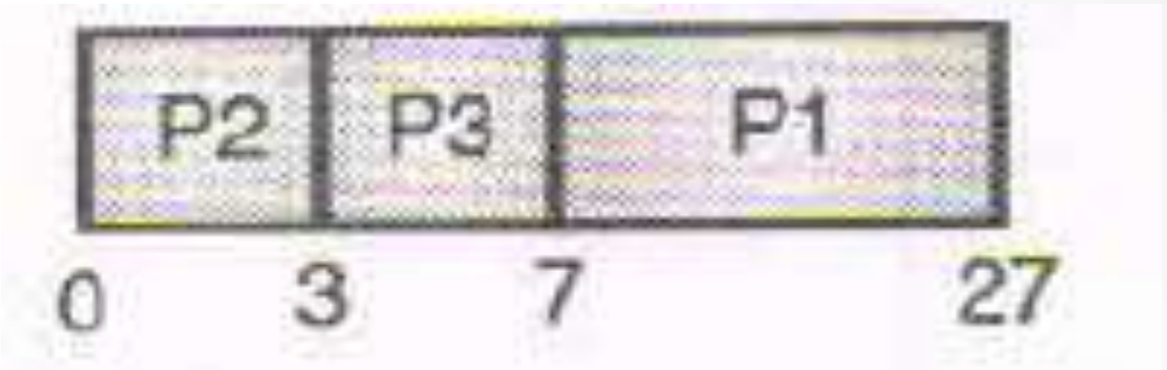
SJF(SJN) Algorithms

- This is also known as **shortest job first**, or SJF
- This is a non-preemptive, pre-emptive scheduling algorithm.
- Best approach to minimize waiting time.
- Easy to implement in Batch systems where required CPU time is known in advance.
- Impossible to implement in interactive systems where required CPU time is not known.
- The processor should know in advance how much time process will take.

SJF(SJN) Algorithms

Consider the following example of three processes :

Process	Burst time / Execution time
P1	20
P2	3
P3	4



Process	CPU Time	Start Time	Finish Time
P1	20	7	27
P2	3	0	3
P3	4	3	7

SJF(SJN) Algorithms

Sr. No.	Process No.	CPU Time	Start Time	Finish Time	Turnaround Time	Waiting Time
1	P1	20	7	27	27	7
2	P2	3	0	3	3	0
3	P3	4	3	7	7	3
				Average →	37/3	10/3
					= 12.33	= 3.33

- SJF algorithm is an optimal scheduling algorithm in terms of minimizing.
  1. The average waiting time.
  2. The average turnaround time.
- SJF algorithm does not guarantee fairness. If shorter jobs keep joining the ready then a relatively longer job may have to wait infinitely.
- The optimal performance of SJF scheduling is dependent upon future knowledge of the process/job behaviour. It is very difficult to estimate the exact CPU time requirement of a job/process.

SJF(SJN) Algorithms

Example 6.10.2

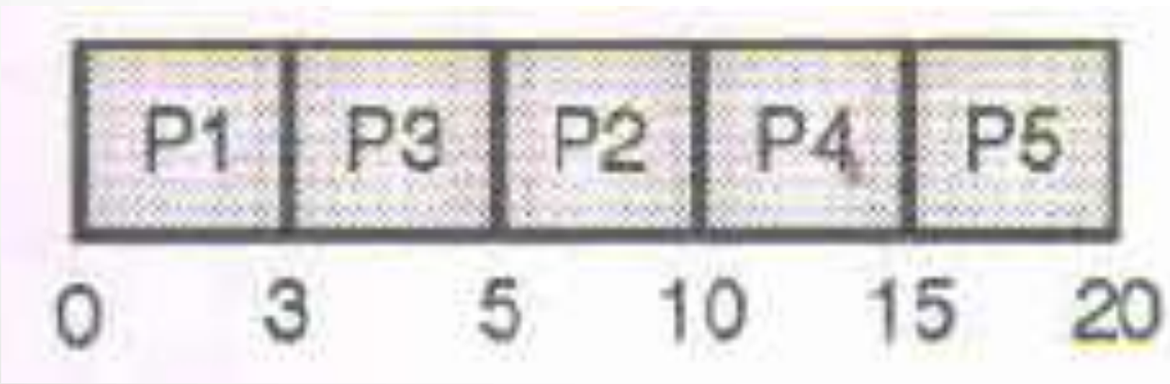
Consider the following set of processes for SJF scheduling :

Process Name	Arrival Time	Processing Time
P1	0	3
P2	1	5
P3	3	2
P4	9	5
P5	12	5

Find waiting time and turnaround time of each process.

Table Ex. 6.10.2

Elapse Time	Process in the ready queue	Process selected for execution	Comment
0	P1	P1	P1 arrives at time 0 and needs 3 units of time
3	P3, P2	P3	Both P2 and P3 arrive by time 3. Shorter of the two, P3 is selected.
5	P2	P2	P3 and P4 do not arrive by time 5.
10	P4	P4	P4 arrives at time 9.
15	P5	P5	



SJF(SJN) Algorithms

TAT=FT-AT WT=TAT-CT

Sr. No.	Process No.	Arrival Time	CPU Time	Start Time	Finish Time	Turnaround Time	Waiting Time
1.	P1	0	3	0	3	3	0
2.	P2	1	5	5	10	9	4
3.	P3	3	2	3	5	2	0
4.	P4	9	5	10	15	6	1
5.	P5	12	5	15	20	8	3
					Average →	28/5 = 5.6	8/5 = 1.6

SJF(SJN) Algorithms

UEx. 4.15.7 (SPPU - Q. 4(c), May 17, 6 Marks)

Consider the following processes where arrival and burst time (in seconds) are as shown in Table P. 4.15.7.

Calculate the average Waiting Time and Average turn-around Time if the processes are scheduled using SJF.

Table P. 4.15.7

Process	Burst Time	Arrival Time
P1	13	3
P2	15	3
P3	08	1
P4	12	1

SJF(SJN) Algorithms

☑ **Soln. :**

Process	Burst Time	Arrival Time
P1	13	3
P2	15	3
P3	08	1
P4	12	1

P3	P4	P1	P2

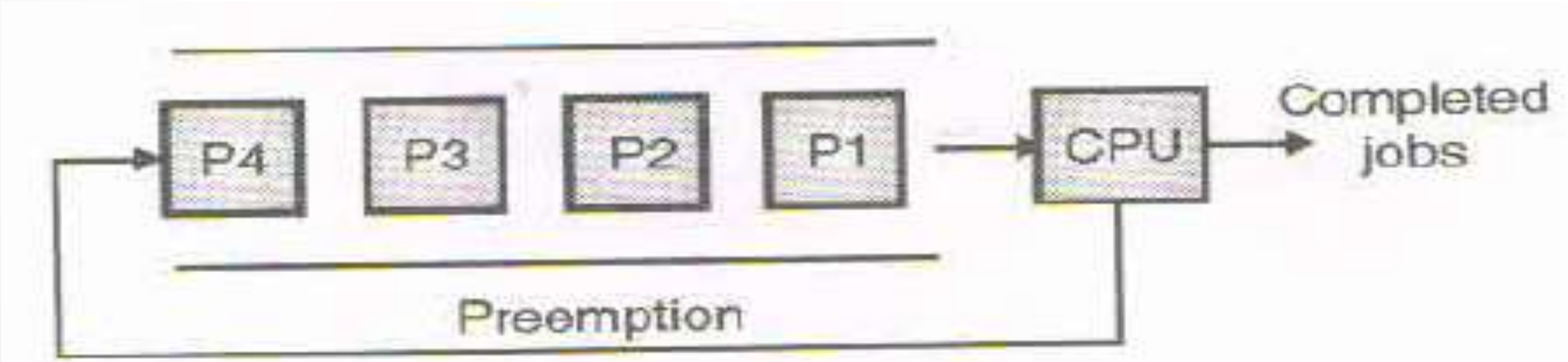
0 8 20 33 46

ST-AT FT-AT

Process	Burst time	Arrival time	Start time	Waiting time	Finish time	Turn around time
P1	13	3	20	17	33	30
P2	15	3	33	30	48	45
P3	8	1	0	−1	8	7
P4	12	1	8	7	20	19

RR Algorithms

- Round Robin is the preemptive process scheduling algorithm.
- Each process is provided a fix time to execute, it is called a **quantum**.
- Once a process is executed for a given time period, it is preempted and other process executes for a given time period.
- Context switching is used to save states of preempted processes.



(S8.17)**Fig. 6.10.3 : Round robin scheduling**

RR Algorithms

How to compute below times in Round Robin using a program?

- **Completion Time:** Time at which process completes its execution.
- **Turn Around Time:** Time Difference between completion time and arrival time.

  Turn Around Time = Completion Time – Arrival Time

- **Waiting Time(W.T):** Time Difference between turn around time and burst time.

  Waiting Time = Turn Around Time – Burst Time

RR Algorithms

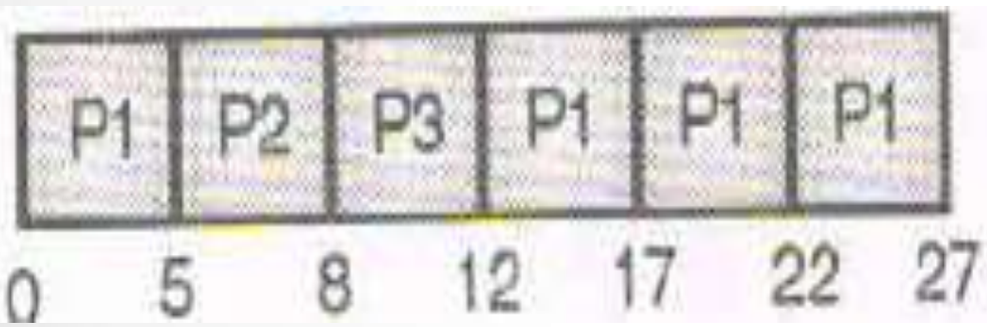
Consider the following example of three processes :

Process	Execution time
P1	20
P2	3
P3	4

Finish time of various processes are :

Process	CPU time	Finish time
P1	20	27
P2	3	8
P3	4	12

Time Quantum is 5



S. No.	Process No.	CPU time	Finish time	Turnaround time	Waiting time
1	P1	20	27	27	7
2	P2	3	8	8	5
3	P2	4	12	12	8
			Average →	47/3	20/3
				= 15.66	= 6.66

RR Algorithms

UEx. 4.15.8 **(SPPU - Q. 2(c), Dec. 18, 6 Marks)**

Consider the following process where the arrival and burst time as shown in Table P. 4.15.8. If the quantum slice time is 2 calculate average waiting time and turnaround time using Round Robin algorithm.

Table P. 4.15.8

	Burst time	Arrival time
P1	05	0
P2	04	2
P3	07	4
P4	06	6

RR Algorithms

	Burst time	Arrival time
P1	05	0
P2	04	2
P3	07	4
P4	06	6

Gnatt chart (slice = 2)

WT = {(ST-AT) + (ST - FT)}

2nd Te

P1	P2	P3	P4	P1	P2	P3	P4	P1	P3	P4	P3

0 2 4 6 8 10 12 14 16 17 19 21 22

FT-AT

RR Algorithms

Process	Burst time	Arrival time	Start time	Wait time	Finish time	Turn around time
P1	5	0	0	0 + 6 + 6 = 12	17	17
P2	4	2	2	0 + 6 = 6	12	10
P3	7	4	4	0 + 6 + 3 + 2 = 11	22	18
P4	6	6	6	0 + 6 + 3 = 9	21	15

$$\text{Average wait time} = \frac{12 + 6 + 11 + 9}{4} = 9.5 \text{ ms}$$

$$\text{Average turn around time} = \frac{17 + 10 + 18 + 15}{4} = \frac{60}{4} = 15 \text{ ms}$$

RR Algorithms

UEx. 4.15.9 (SPPU - Q. 5(b), March 18, 4 Marks)

Consider the following processes arrival time and burst time are as shown in Table 4.15.9. Calculate average waiting time and average turnaround time if Quantum time is 2. Use Round Robin algorithm.

Table P. 4.15.9

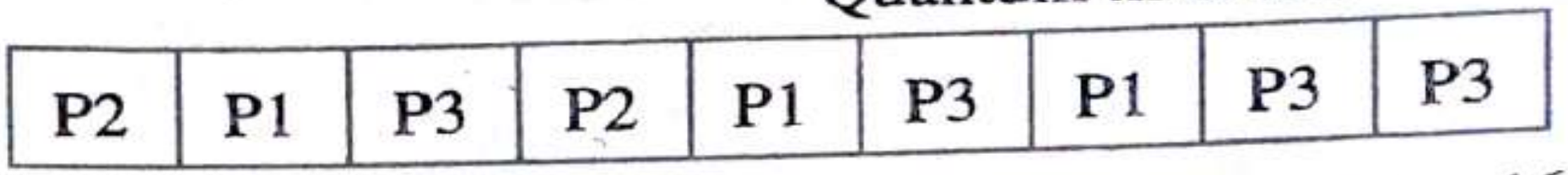
Process	Burst time	Arrival time
P1	5	1
P2	4	0
P3	7	2

☑ **Soln. :**

Process	Burst time	Arrival time
P1	5	1
P2	4	0
P3	7	2

RR Algorithms

Quantum time is 2



Process	Burst time	Arrival time	Start time	Wait time	Finish time	Turn around time
P1	5	1	2	1 + 4 + 4 = 9	13	13 – 1 = 12
P2	4	0	0	0 + 4 + 4 = 8	8	8 – 0 = 8
P3	7	2	4	2 + 4 + 3 + 3 = 12	16	16 – 2 = 14

$$\text{Average waiting time} = \frac{9 + 8 + 12}{3} = \frac{29}{3} = 9.67 \text{ ms}$$

$$\text{Average turn around time} = \frac{12 + 8 + 14}{3} = \frac{34}{3} = 11.33 \text{ ms}$$

RR Algorithms

UEx. 4.15.11 (SPPU - Q. 2(b), May 18, 7 Marks)

Consider the following process where the arrival and burst time as shown in Table P. 4.15.11. If the quantum slice time is 5 calculate average waiting time and turnaround time using Round Robin algorithm

Table P. 4.15.11

Process	Burst time	Arrival time
A	10	00
B	06	00
C	07	01
D	04	01
E	05	02

☑ **Soln. :**

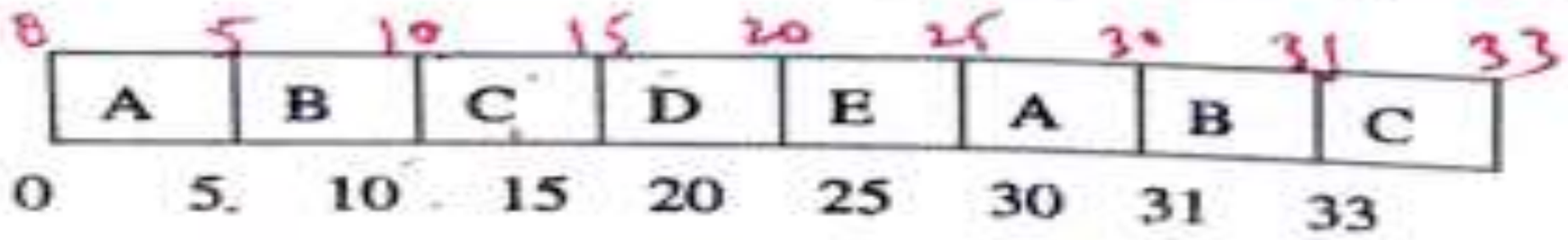
Process	Burst time	Arrival time
A	10	00
B	06	00
C	07	01
D	04	01
E	05	02

RR Algorithms

Quantum slice time is 5

WT = [(ST-AT) + (ST-FT)]
1st + 2nd term

Process	Burst time	Arrival time	Start time	Wait time	Finish time	Turn around time
A	10	0	0	0 + 20 = 20	30	30 – 0 = 30
B	6	0	5	5 + 20 = 25	31	31 – 0 = 31
C	7	1	10	9 + 16 = 25	33	33 – 1 = 32
D	4	1	15	14	20	20 – 1 = 19
E	5	2	20	18	25	25 – 2 = 23



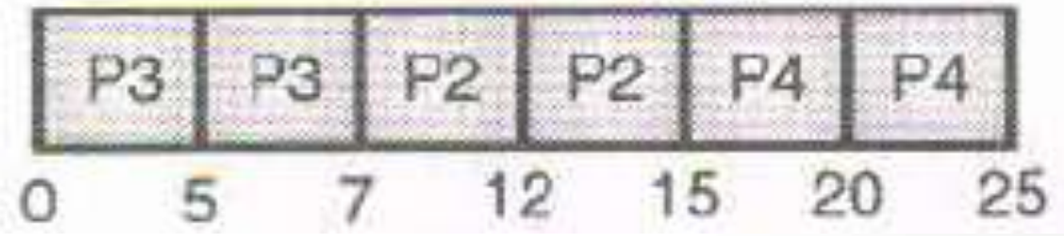
$$\text{Average waiting time} = \frac{20 + 25 + 25 + 14 + 18}{5} = 20.4 \text{ ms}$$

$$\text{Average turn around time} = \frac{30 + 31 + 32 + 19 + 23}{5} = 27 \text{ ms}$$

SJF Preemption Algorithms

Consider the following example of three processes :

Process	Execution
P1	10
P2	8
P3	7



Finish time of various processes are :

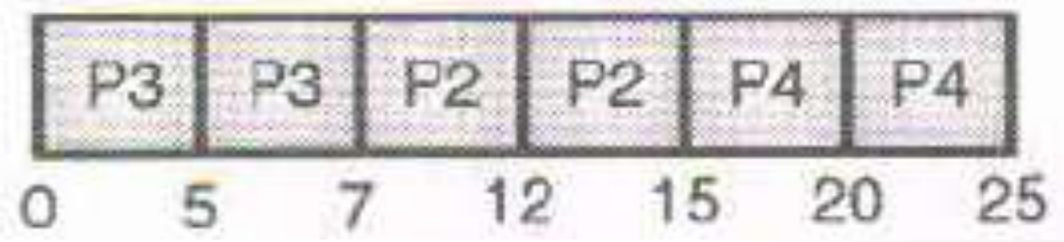
Process	CPU Time	Finish Time
P1	10	25
P2	8	15
P3	7	7

Sr. No	Process No.	CPU Time	Finish Time	Turnaround Time	Waiting Time
1.	P1	10	25	25	15
2.	P2	8	15	15	7
3.	P3	7	7	7	0

SJF Preemption Algorithms

Consider the following example of three processes :

Process	Execution
P1	10
P2	8
P3	7



Finish time of various processes are :

Process	CPU Time	Finish Time
P1	10	25
P2	8	15
P3	7	7

Sr. No	Process No.	CPU Time	Finish Time	Turnaround Time	Waiting Time
1.	P1	10	25	25	15
2.	P2	8	15	15	7
3.	P3	7	7	7	0

SJF Preemption Algorithms

Example 6.10.4

Consider the following set of processes for preemptive SJF scheduling (Time slab 2) :

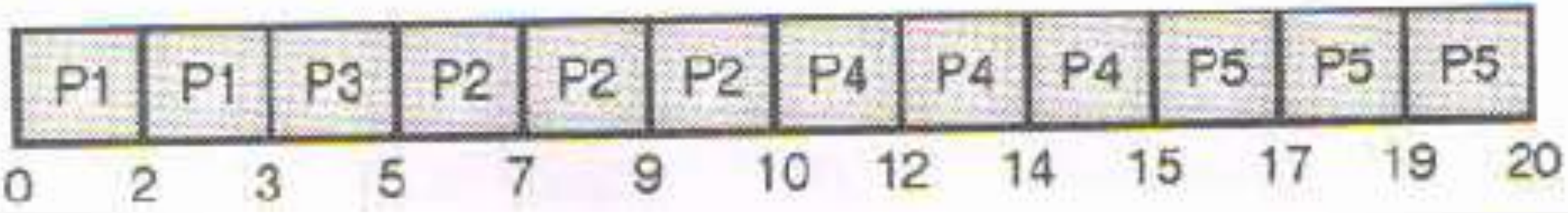
Process Name	Arrival Time	Processing Time
P1	0	3
P2	1	5
P3	3	2
P4	9	5
P5	12	5

SJF Preemption Algorithms

Time	Real Queue (smallest process at the front)	Next job to be executed
0	P1	P1 for 2 units. P1 needs additional 1 unit of time.
2	P1 P2	P1 for 1 unit. P1 completes
3	P3 P2	P3 for 2 units. P3 completes
5	P2	P2 for 2 unit
7	P2	P2 for 2 units
9	P2 P4	P2 for 1 unit P2 completes
10	P4	P4 for 2 units
12	P4 P5	P4 for 2 units
14	P4 P5	P4 for 1 unit P4 completes.
15	P5	P5 for 2 units
17	P5	P5 for 2 units
19	P5	P5 for 1 unit P5 completes.

SJF Preemption Algorithms

Gantt Chart



Sr. No	Process No.	Arrival Time	CPU Time	Finish Time	Turnaround Time	Waiting Time
1.	P1	0	3	3	3	0
2.	P2	1	5	10	9	4
3.	P3	3	2	5	2	0
4.	P4	9	5	15	6	1
5.	P5	12	5	20	8	3

Priority Algorithms

- Priority scheduling is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems.
- Each process is assigned a priority. Process with highest priority is to be executed first and so on.
- Processes with same priority are executed on first come first served basis.
- Priority can be decided based on memory requirements, time requirements or any other resource requirement.

THANK YOU!!!

My Blog : https://anandgharu.wordpress.com/

Email : gharu.anand@gmail.com