

Introduction to Process

A process is a program in execution. For example, when we write a program in C or C++ and compile it, the compiler creates binary code. The original code and binary code are both programs. When we actually run the binary code, it becomes a process. A process is an 'active' entity instead of a program, which is considered a 'passive' entity. A single program can create many processes when run multiple times; for example, when we open a .exe or binary file multiple times, multiple instances begin (multiple processes are created).

What is Process Management?

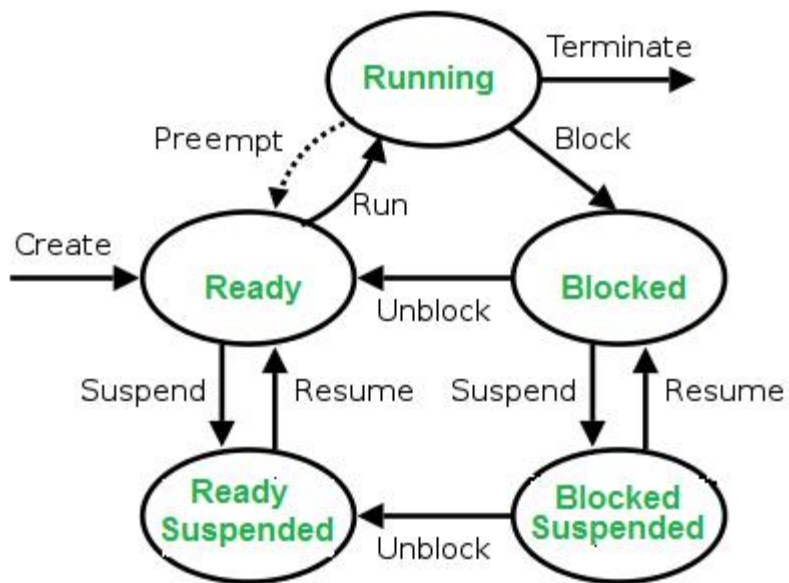
Process management is a key part of an operating system. It controls how processes are carried out, and controls how your computer runs by handling the active processes. This includes stopping processes, setting which processes should get more attention, and many more. You can manage processes on your own computer too.

The OS is responsible for managing the start, stop, and scheduling of processes, which are programs running on the system. The operating system uses a number of methods to prevent deadlocks, facilitate inter-process communication, and synchronize processes. Efficient resource allocation, conflict-free process execution, and optimal system performance are all guaranteed by competent process management. This essential component of an operating system enables the execution of numerous applications at once, enhancing system utilization and responsiveness.

States of Process

A process is in one of the following states:

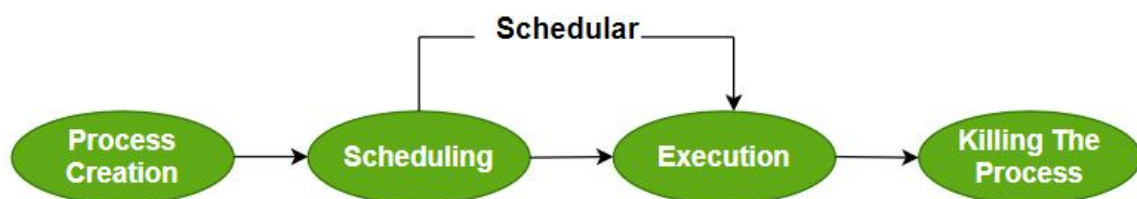
- **New:** Newly Created Process (or) being-created process.
- **Ready:** After the creation process moves to the Ready state, i.e. the process is ready for execution.
- **Running:** Currently running process in CPU (only one process at a time can be under execution in a single processor).
- **Wait (or Block):** When a process requests I/O access.
- **Complete (or Terminated):** The process completed its execution.
- **Suspended Ready:** When the ready queue becomes full, some processes are moved to a suspended ready state
- **Suspended Block:** When the waiting queue becomes full.



Process Operations

Process operations in an operating system refer to the various activities the OS performs to manage processes. These operations include process creation, process scheduling, execution and killing the process. Here are the key process operations:

Process Operations



Process Operations

Process Creation

Process creation in an operating system (OS) is the act of generating a new process. This new process is an instance of a program that can execute independently.

Scheduling

Once a process is ready to run, it enters the “ready queue.” The scheduler’s job is to pick a process from this queue and start its execution.

Execution

Execution means the CPU starts working on the process. During this time, the process might:

- Move to a waiting queue if it needs to perform an I/O operation.
- Get blocked if a higher-priority process needs the CPU.

Killing the Process

After the process finishes its tasks, the operating system ends it and removes its Process Control Block (PCB).

CPU-Bound vs I/O-Bound Processes

A CPU-bound process requires more CPU time or spends more time in the running state. An I/O-bound process requires more I/O time and less CPU time. An I/O-bound process spends more time in the waiting state.

Process planning is an integral part of the process management operating system. It refers to the mechanism used by the operating system to determine which process to run next. The goal of process scheduling is to improve overall system performance by maximizing CPU utilization, minimizing execution time, and improving system response time.

Process Control Block

An Operating System helps in process creation, scheduling, and termination with the help of Process Control Block. The Process Control Block (PCB), which is part of the Operating System, aids in managing how processes operate. Every OS process has a Process Control Block related to it. By keeping data on different things including their state, I/O status, and CPU Scheduling, a PCB maintains track of processes.

Now, let us understand the Process Control Block with the help of the components present in the Process Control Block.

A Process Control Block consists of:

1. Process ID
2. Process State
3. Program Counter
4. CPU Registers
5. CPU Scheduling Information
6. Accounting and Business Information
7. Memory Management Information

8. Input Output Status Information

Now, let us understand about each and every component in detail now.

Advertisement

1) Process ID

It is a Identification mark which is present for the Process. This is very useful for finding the process. It is also very useful for identifying the process also.

2) Process State

i) New State

A Program which is going to be taken up by the Operating System directly into the Main Memory is known as a New Process State

ii) Ready State

The ready state, when a process waits for the CPU to be assigned, is the first state it enters after being formed. The operating system pulls new processes from secondary memory and places them all in main memory.

The term "ready state processes" refers to processes that are in the main memory and are prepared for execution. Numerous processes could be active at the moment.

iii) Running State

The Operating System will select one of the processes from the ready state based on the scheduling mechanism. As a result, if our system only has one CPU, there will only ever be one process operating at any given moment. We can execute n processes concurrently in the system if there are n processors.

iv) Waiting or Blocking State

Depending on the scheduling mechanism or the inherent behavior of the process, a process can go from the Running state to the Block or Wait states.

The OS switches a process to the block or wait state and allots the CPU to the other processes while it waits for a specific resource to be allocated or for user input.

v) Terminated State

A process enters the termination state once it has completed its execution. The operating system will end the process and erase the whole context of the process (Process Control Block).

3) Program Counter

The address of the following instruction to be executed from memory is stored in a CPU register called a program counter (PC) in the computer processor. It is a digital counter required for both task execution speed and for monitoring the present stage of execution.

An instruction counter, instruction pointer, instruction addresses register, or sequence control register are other names for a program counter.

4) CPU Registers

When the process is in a running state, here is where the contents of the processor registers are kept. Accumulators, index and general-purpose registers, instruction registers, and condition code registers are the many categories of CPU registers.

5) CPU Scheduling Information

It is necessary to arrange a procedure for execution. This schedule determines when it transitions from ready to running. Process priority, scheduling queue pointers (to indicate the order of execution), and several other scheduling parameters are all included in CPU scheduling information.

6) Accounting and Business Information

The State of Business Addressing and Information includes information such as CPU use, the amount of time a process uses in real time, the number of jobs or processes, etc.

7) Memory Management Information

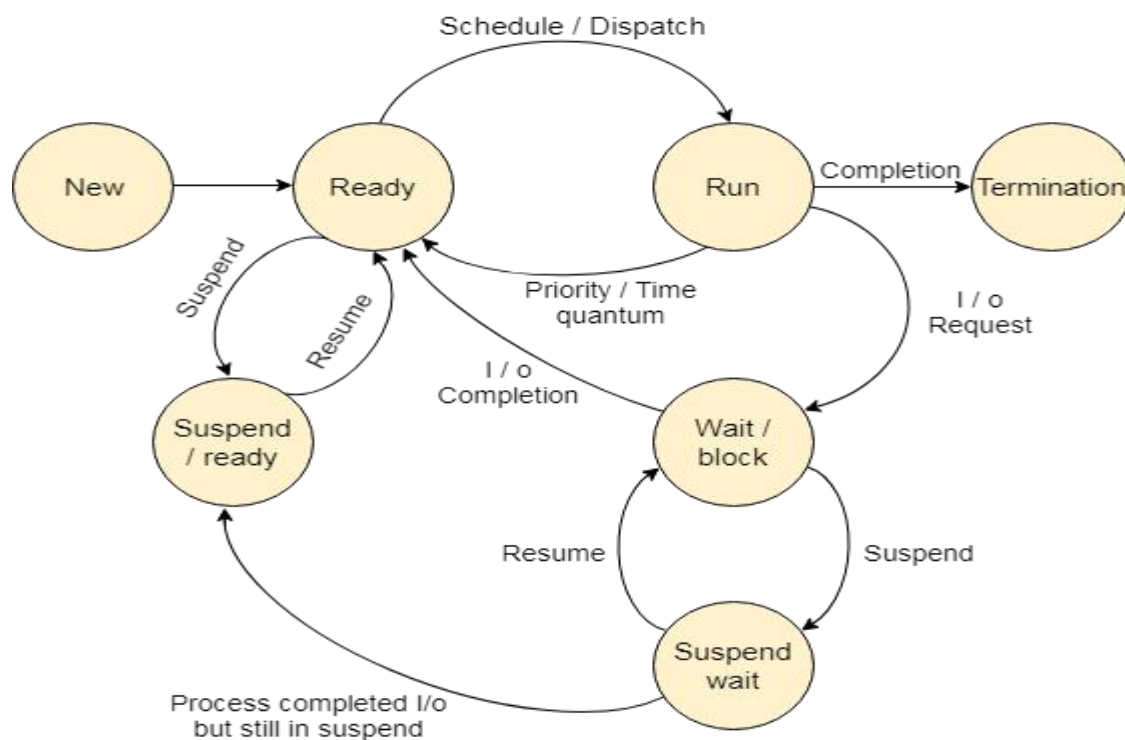
The Memory Management Information section contains information on the page, segment tables, and the value of the base and limit registers. It relies on the operating system's memory system.

8) Input Output Status Information

This Input Output Status Information section consists of Input and Output related information which includes about the process statuses, etc.

Process States

State Diagram



The process, from its creation to completion, passes through various states. The minimum number of states is five.

The names of the states are not standardized although the process may be in one of the following states during execution.

1. New

A program which is going to be picked up by the OS into the main memory is called a new process.

2. Ready

Whenever a process is created, it directly enters in the ready state, in which, it waits for the CPU to be assigned. The OS picks the new processes from the secondary memory and put all of them in the main memory.

The processes which are ready for the execution and reside in the main memory are called ready state processes. There can be many processes present in the ready state.

3. Running

One of the processes from the ready state will be chosen by the OS depending upon the scheduling algorithm. Hence, if we have only one CPU in our system, the number of running processes for a particular time will always be one. If we have n processors in the system then we can have n processes running simultaneously.

4. Block or wait

From the Running state, a process can make the transition to the block or wait state depending upon the scheduling algorithm or the intrinsic behavior of the process.

When a process waits for a certain resource to be assigned or for the input from the user then the OS move this process to the block or wait state and assigns the CPU to the other processes.

5. Completion or termination

When a process finishes its execution, it comes in the termination state. All the context of the process (Process Control Block) will also be deleted the process will be terminated by the Operating system.

6. Suspend ready

A process in the ready state, which is moved to secondary memory from the main memory due to lack of the resources (mainly primary memory) is called in the suspend ready state.

If the main memory is full and a higher priority process comes for the execution then the OS have to make the room for the process in the main memory by throwing the lower priority process out into the secondary memory. The suspend ready processes remain in the secondary memory until the main memory gets available.

7. Suspend wait

Instead of removing the process from the ready queue, it's better to remove the blocked process which is waiting for some resources in the main memory. Since it is already waiting for some resource to get available hence it is better if it waits in the secondary memory and make room for

the higher priority process. These processes complete their execution once the main memory gets available and their wait is finished.

Process Scheduling in OS (Operating System)

Operating system uses various schedulers for the process scheduling described below.

1. Long term scheduler

Long term scheduler is also known as job scheduler. It chooses the processes from the pool (secondary memory) and keeps them in the ready queue maintained in the primary memory.

Long Term scheduler mainly controls the degree of Multiprogramming. The purpose of long term scheduler is to choose a perfect mix of IO bound and CPU bound processes among the jobs present in the pool.

If the job scheduler chooses more IO bound processes then all of the jobs may reside in the blocked state all the time and the CPU will remain idle most of the time. This will reduce the degree of Multiprogramming. Therefore, the Job of long term scheduler is very critical and may affect the system for a very long time.

2. Short term scheduler

Short term scheduler is also known as CPU scheduler. It selects one of the Jobs from the ready queue and dispatch to the CPU for the execution.

A scheduling algorithm is used to select which job is going to be dispatched for the execution. The Job of the short term scheduler can be very critical in the sense that if it selects job whose CPU burst time is very high then all the jobs after that, will have to wait in the ready queue for a very long time.

This problem is called starvation which may arise if the short term scheduler makes some mistakes while selecting the job.

3. Medium term scheduler

Medium term scheduler takes care of the swapped out processes. If the running state processes needs some IO time for the completion then there is a need to change its state from running to waiting.

Medium term scheduler is used for this purpose. It removes the process from the running state to make room for the other processes. Such processes are the swapped out processes and this procedure is called swapping. The medium term scheduler is responsible for suspending and resuming the processes.

It reduces the degree of multi programming. The swapping is necessary to have a perfect mix of processes in the ready queue.

Various Times related to the Process

1. Arrival Time

The time at which the process enters into the ready queue is called the arrival time.

2. Burst Time

The total amount of time required by the CPU to execute the whole process is called the Burst Time. This does not include the waiting time. It is confusing to calculate the execution time for a process even before executing it.

3. Completion Time

The Time at which the process enters into the completion state or the time at which the process completes its execution, is called completion time.

4. Turnaround time

The total amount of time spent by the process from its arrival to its completion, is called Turnaround time.

5. Waiting Time

The Total amount of time for which the process waits for the CPU to be assigned is called waiting time.

6. Response Time

The difference between the arrival time and the time at which the process first gets the CPU is called Response Time.

CPU Scheduling

In the uni programming systems like MS DOS, when a process waits for any I/O operation to be done, the CPU remains idle. This is an overhead since it wastes the time and causes the problem of starvation. However, In Multi programming systems, the CPU doesn't remain idle during the waiting time of the Process and it starts executing other processes. Operating System has to define which process the CPU will be given.

In Multi programming systems, the Operating system schedules the processes on the CPU to have the maximum utilization of it and this procedure is called **CPU scheduling**. The Operating System uses various scheduling algorithm to schedule the processes.

This is a task of the short term scheduler to schedule the CPU for the number of processes present in the Job Pool. Whenever the running process requests some I/O operation then the short term scheduler saves the current context of the process (also called PCB) and changes its state from

running to waiting. During the time, process is in waiting state; the Short term scheduler picks another process from the ready queue and assigns the CPU to this process. This procedure is called **context switching**.

Why do we need Scheduling?

In Multi programming, if the long term scheduler picks more I/O bound processes then most of the time, the CPU remains idle. The task of Operating system is to optimize the utilization of resources.

If most of the running processes change their state from running to waiting then there may always be a possibility of deadlock in the system. Hence to reduce this overhead, the OS needs to schedule the jobs to get the optimal utilization of CPU and to avoid the possibility to deadlock.

Scheduling Algorithms in OS (Operating System)

There are various algorithms which are used by the Operating System to schedule the processes on the processor in an efficient way.

The Purpose of a Scheduling algorithm is :

1. Maximum CPU utilization
2. Fair allocation of CPU
3. Maximum throughput
4. Minimum turnaround time
5. Minimum waiting time
6. Minimum response time

There are two types of scheduling: preemptive scheduling and non-preemptive scheduling. Preemptive scheduling allows a running process to be interrupted by a high priority process, whereas in non-preemptive scheduling, any new process has to wait until the running process finishes its CPU cycle.

There are the following algorithms which can be used to schedule the jobs.

1. First Come First Serve

It is the simplest algorithm to implement. The process with the minimal arrival time will get the CPU first. The lesser the arrival time, the sooner will the process get the CPU. It is the non-preemptive type of scheduling.

2. Round Robin

In the Round Robin scheduling algorithm, the OS defines a time quantum (slice). All the processes will get executed in the cyclic way. Each of the process will get the CPU for a small amount of time (called time quantum) and then get back to the ready queue to wait for its next turn. It is a preemptive type of scheduling.

3. Shortest Job First

The job with the shortest burst time will get the CPU first. The lesser the burst time, the sooner will the process get the CPU. It is the non-preemptive type of scheduling.

4. Shortest remaining time first

It is the preemptive form of SJF. In this algorithm, the OS schedules the Job according to the remaining time of the execution.

5. Priority based scheduling

In this algorithm, the priority will be assigned to each of the processes. The higher the priority, the sooner will the process get the CPU. If the priority of the two processes is same then they will be scheduled according to their arrival time.

First Come First Serve CPU Process Scheduling in Operating System

The important concept name is First Come First Serve. This is the basic algorithm which every student must learn to understand all the basics of CPU Process Scheduling Algorithms.

First Come First Serve paves the way for understanding of other algorithms. This algorithm may have many disadvantages. But these disadvantages created very new and efficient algorithms. So, it is our responsibility to learn about First Come First Serve CPU Process Scheduling Algorithms.

Important Abbreviations

1. CPU - - - > Central Processing Unit
2. FCFS - - - > First Come First Serve
3. AT - - - > Arrival Time
4. BT - - - > Burst Time
5. WT - - - > Waiting Time
6. TAT - - - > Turn Around Time
7. CT - - - > Completion Time
8. FIFO - - - > First In First Out

First Come First Serve

First Come First Serve CPU Scheduling Algorithm shortly known as FCFS is the first algorithm of CPU Process Scheduling Algorithm. In First Come First Serve Algorithm what we do is to allow the process to execute in linear manner.

This means that whichever process enters process enters the ready queue first is executed first. This shows that First Come First Serve Algorithm follows First In First Out (FIFO) principle.

The First Come First Serve Algorithm can be executed in Preemptive and Non Preemptive manner. Before, going into examples, let us understand what is Preemptive and Non Preemptive Approach in CPU Process Scheduling.

Preemptive Approach

In this instance of Preemptive Process Scheduling, the OS allots the resources to a Process for a predetermined period of time. The process transitions from running state to ready state or from waiting state to ready state during resource allocation. This switching happens because the CPU may assign other processes precedence and substitute the currently active process for the higher priority process.

Non Preemptive Approach

In this case of Non Preemptive Process Scheduling, the resource cannot be withdrawn from a process before the process has finished running. When a running process finishes and transitions to the waiting state, resources are switched.

The Non preemptive way means that if a process or job is started execution, then the operating system must complete its process or job. Until, the process or job is zero the new or next process or job does not start its execution. The definition of Non Preemptive Scheduling in terms of Operating System means that the Central Processing Unit (CPU) will be completely dedicated till the end of the

process or job started first and the new process or job is executed only after finishing of the older process or job.

There may be a few cases, which might cause the Central Processing Unit (CPU) to allot a too much time. This is because in the First Come First Serve Scheduling Algorithm Non Preemptive Approach, the Processes or the jobs are chosen in serial order. Due, to this shorter jobs or processes behind the larger processes or jobs takes too much time to complete its execution. Due to this the Waiting Time, Turn Around Time, Completion Time is very high.



So, here as the first process is large or completion time is too high, then this **Convoy effect (The convoy effect is a phenomenon in operating systems where the entire system slows down due to a few slower processes)** in the First Come First Serve Algorithm is occurred.

Let us assume that Longer Job takes infinite time to complete. Then, the remaining processes have to wait for the same infinite time. Due to this Convoy Effect created by the Longer Job the Starvation of the waiting processes increases very rapidly. This is the biggest disadvantage of FCFS CPU Process Scheduling.

Characteristics of FCFS CPU Process Scheduling

The characteristics of FCFS CPU Process Scheduling are:

1. Implementation is simple.
2. It adopts a non pre emptive and pre emptive strategy.
3. It runs each procedure in the order that they are received.
4. Arrival time is used as a selection criterion for procedures.

Advantages of FCFS CPU Process Scheduling

1. In order to allocate processes, it uses the First In First Out queue.
2. The FCFS CPU Scheduling Process is straight forward and easy to implement.
3. In the FCFS situation pre emptive scheduling, there is no chance of process starving.
4. As there is no consideration of process priority, it is an equitable algorithm.

Disadvantages of FCFS CPU Process Scheduling

1. FCFS CPU Scheduling Algorithm has Long Waiting Time
2. FCFS CPU Scheduling favors CPU bound processes over Input/Output bound processes because it executes processes to completion.
3. In FCFS there is a chance of occurrence of Convoy Effect
4. Because FCFS is so straight forward, it often isn't very effective. Extended waiting periods go hand in hand with this. All other orders are left idle if the CPU is busy processing one time-consuming order.

Shortest Job Next (SJN)

- 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.

Priority Based Scheduling

- 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.

Shortest Remaining Time

- Shortest remaining time (SRT) is the preemptive version of the SJN algorithm.
- The processor is allocated to the job closest to completion but it can be preempted by a newer ready job with shorter time to completion.
- Impossible to implement in interactive systems where required CPU time is not known.

- It is often used in batch environments where short jobs need to give preference.

Round Robin Scheduling

- 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.

Multiple-Level Queues Scheduling

Multiple-level queues are not an independent scheduling algorithm. They make use of other existing algorithms to group and schedule jobs with common characteristics.

- Multiple queues are maintained for processes with common characteristics.
- Each queue can have its own scheduling algorithms.
- Priorities are assigned to each queue.

For example, CPU-bound jobs can be scheduled in one queue and all I/O-bound jobs in another queue. The Process Scheduler then alternately selects jobs from each queue and assigns them to the CPU based on the algorithm assigned to the queue.