Introduction of Process Management

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).
Process management refers to the techniques and strategies used by organizations to design, monitor, and control their business processes to achieve their goals efficiently and effectively. It involves identifying the steps involved in completing a task, assessing the resources required for each step, and determining the best way to execute the task.
Process management can help organizations improve their operational efficiency, reduce costs, increase customer satisfaction, and maintain compliance with regulatory requirements. It involves analyzing the performance of existing processes, identifying bottlenecks, and making changes to optimize the process flow.
Process management includes various tools and techniques such as process mapping, process analysis, process improvement, process automation, and process control. By applying these tools and techniques, organizations can streamline their processes, eliminate waste, and improve productivity.

Process Management

If the operating system supports multiple users then services under this are very important. In this regard, operating systems have to keep track of all the completed processes, Schedule them, and dispatch them one after another. But the user should feel that he has full control of the CPU.
Some of the systems call in this category are as follows.

1. Create a child's process identical to the parent's.
2. Terminate a process
3. Wait for a child process to terminate
4. Change the priority of the process
5. Block the process
6. Ready the process
7. Dispatch a process
8. Suspend a process
9. Resume a process
10. Delay a process
11. Fork a process

What does a process look like in memory?

Explanation of Process

1. **Text Section**: A Process, sometimes known as the Text Section, also includes the current activity represented by the value of the Program Counter.

2. **Stack**: The stack contains temporary data, such as function parameters, returns addresses, and local variables.
3. **Data Section**: Contains the global variable.
4. **Heap Section**: Dynamically allocated memory to process during its run time.

Attributes or Characteristics of a Process

A process has the following attributes.

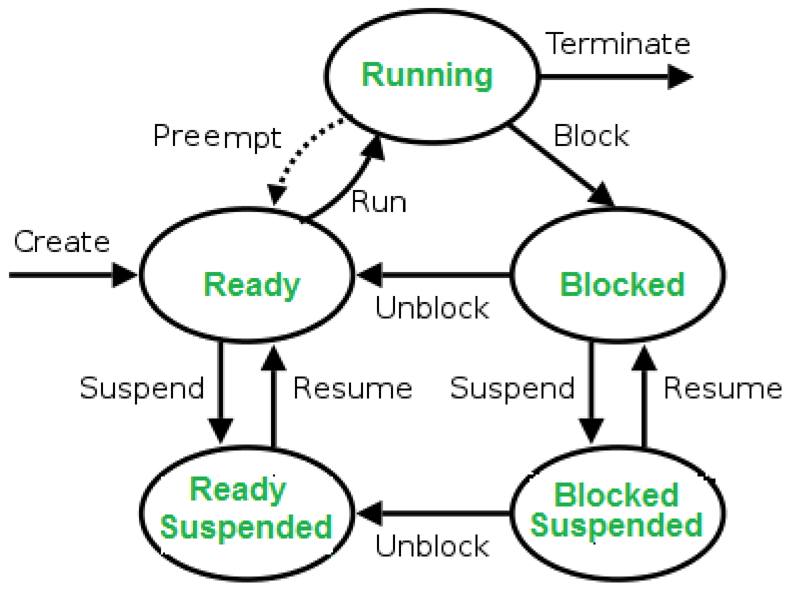
1. **Process Id:** A unique identifier assigned by the operating system
2. **Process State:** Can be ready, running, etc.
3. **CPU registers:** Like the Program Counter (CPU registers must be saved and restored when a process is swapped in and out of the CPU)
4. **Accounts information:** Amount of CPU used for process execution, time limits, execution ID, etc
5. **I/O status information:** For example, devices allocated to the process, open files, etc
6. **CPU scheduling information:** For example, Priority (Different processes may have different priorities, for example, a shorter process assigned high priority in the shortest job first scheduling)

All of the above attributes of a process are also known as the **context of the process**. Every process has its own process control block(PCB), i.e. each process will have a unique PCB. All of the above attributes are part of the PCB.

States of Process

A process is in one of the following states:

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



Context Switching: The process of saving the context of one process and loading the context of another process is known as Context Switching. In simple terms, it is like loading and unloading the process from the running state to the ready state.

When Does Context Switching Happen?

1. When a high-priority process comes to a ready state (i.e., with higher priority than the running process)
2. An Interrupt occurs
3. User and kernel-mode switch (It is not necessary though)
4. Preemptive CPU scheduling is used.

Context Switch vs Mode Switch

A mode switch occurs when the CPU privilege level is changed, for example when a system call is made or a fault occurs. The kernel works in more a privileged mode than a standard user task. If a user process wants to access things that are only accessible to the kernel, a mode switch must occur. The currently executing process need not be changed during a mode switch. A mode switch typically occurs for a process context switch to occur. Only the kernel can cause a context switch.

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.

Scheduling Algorithms

The operating system can use different scheduling algorithms to schedule processes. Here are some commonly used timing algorithms:

1. **First-come, first-served (FCFS):** This is the simplest scheduling algorithm, where the process is executed on a first-come, first-served basis.

FCFS is non-preemptive, which means that once a process starts executing, it continues until it is finished or waiting for I/O.

2. **Shortest Job First (SJF):** SJF is a proactive scheduling algorithm that selects the process with the shortest burst time. The burst time is the time a process takes to complete its execution. SJF minimizes the average waiting time of processes.
3. **Round Robin (RR):** RR is a proactive scheduling algorithm that reserves a fixed amount of time in a round for each process. If a process does not complete its execution within the specified time, it is blocked and added to the end of the queue. RR ensures fair distribution of CPU time to all processes and avoids starvation.
4. **Priority Scheduling:** This scheduling algorithm assigns priority to each process and the process with the highest priority is executed first. Priority can be set based on process type, importance, or resource requirements.
5. **Multilevel queue:** This scheduling algorithm divides the ready queue into several separate queues, each queue having a different priority. Processes are queued based on their priority, and each queue uses its own scheduling algorithm. This scheduling algorithm is useful in scenarios where different types of processes have different priorities.

Advantages of Process Management

1. **Improved Efficiency:** Process management can help organizations identify bottlenecks and inefficiencies in their processes, allowing them to make changes to streamline workflows and increase productivity.
2. **Cost Savings:** By identifying and eliminating waste and inefficiencies, process management can help organizations reduce costs associated with their business operations.
3. **Improved Quality:** Process management can help organizations improve the quality of their products or services by standardizing processes and reducing errors.
4. **Increased Customer Satisfaction:** By improving efficiency and quality, process management can enhance the customer experience and increase satisfaction.
5. **Compliance with Regulations:** Process management can help organizations comply with regulatory requirements by ensuring that processes are properly documented, controlled, and monitored.

Disadvantages of Process Management

1. **Time and Resource Intensive:** Implementing and maintaining process management initiatives can be time-consuming and require significant resources.
2. **Resistance to Change:** Some employees may resist changes to established processes, which can slow down or hinder the implementation of process management initiatives.
3. **Overemphasis on Process:** Overemphasis on the process can lead to a lack of focus on customer needs and other important aspects of business operations.

4. **Risk of Standardization:** Standardizing processes too much can limit flexibility and creativity, potentially stifling innovation.
5. **Difficulty in Measuring Results:** Measuring the effectiveness of process management initiatives can be difficult, making it challenging to determine their impact on organizational performance.

Process Table and Process Control Block (PCB)

While creating a process, the operating system performs several operations. To identify the processes, it assigns a process identification number (PID) to each process. As the operating system supports multi-programming, it needs to keep track of all the processes. For this task, the process control block (PCB) is used to track the process's execution status. Each block of memory contains information about the process state, program counter, stack pointer, status of opened files, scheduling algorithms, etc.
All this information is required and must be saved when the process is switched from one state to another. When the process makes a transition from one state to another, the operating system must update information in the process's PCB. A process control block (PCB) contains information about the process, i.e. registers, quantum, priority, etc. The process table is an array of PCBs, that means logically contains a PCB for all of the current processes in the system.

1. **Pointer:** It is a stack pointer that is required to be saved when the process is switched from one state to another to retain the current position of the process.
2. **Process state:** It stores the respective state of the process.
3. **Process number:** Every process is assigned a unique id known as process ID or PID which stores the process identifier.
4. **Program counter:** It stores the counter,: which contains the address of the next instruction that is to be executed for the process.
5. **Register:** These are the CPU registers which include the accumulator, base, registers, and general-purpose registers.
6. **Memory limits:** This field contains the information about memory management system used by the operating system. This may include page tables, segment tables, etc.
7. **Open files list :** This information includes the list of files opened for a process.

Additional Points to Consider for Process Control Block (PCB)

- **Interrupt handling:** The PCB also contains information about the interrupts that a process may have generated and how they were handled by the operating system.
- **Context switching:** The process of switching from one process to another is called context switching. The PCB plays a crucial role in context switching by saving the state of the current process and restoring the state of the next process.

- **Real-time systems:** Real-time operating systems may require additional information in the PCB, such as deadlines and priorities, to ensure that time-critical processes are executed in a timely manner.
- **Virtual memory management:** The PCB may contain information about a process's virtual memory management, such as page tables and page fault handling.
- **Inter-process communication:** The PCB can be used to facilitate inter-process communication by storing information about shared resources and communication channels between processes.
- **Fault tolerance:** Some operating systems may use multiple copies of the PCB to provide fault tolerance in case of hardware failures or software errors.

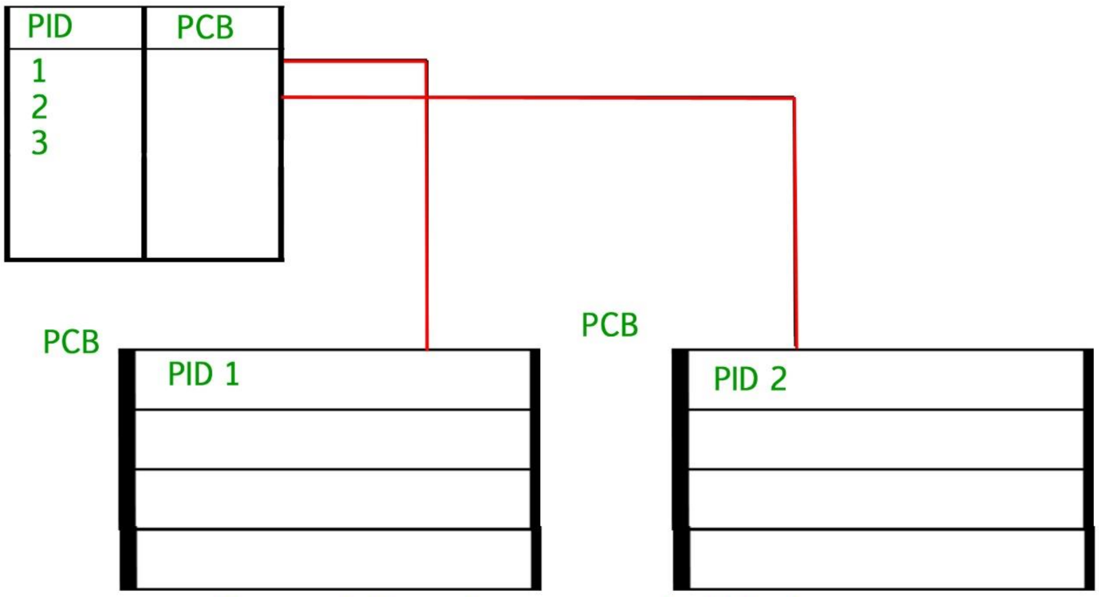
Advantages-

1. **Efficient process management:** The process table and PCB provide an efficient way to manage processes in an operating system. The process table contains all the information about each process, while the PCB contains the current state of the process, such as the program counter and CPU registers.
2. **Resource management:** The process table and PCB allow the operating system to manage system resources, such as memory and CPU time, efficiently. By keeping track of each process's resource usage, the operating system can ensure that all processes have access to the resources they need.
3. **Process synchronization:** The process table and PCB can be used to synchronize processes in an operating system. The PCB contains information about each process's synchronization state, such as its waiting status and the resources it is waiting for.
4. **Process scheduling:** The process table and PCB can be used to schedule processes for execution. By keeping track of each process's state and resource usage, the operating system can determine which processes should be executed next.

Disadvantages-

1. **Overhead:** The process table and PCB can introduce overhead and reduce system performance. The operating system must maintain the process table and PCB for each process, which can consume system resources.
2. **Complexity:** The process table and PCB can increase system complexity and make it more challenging to develop and maintain operating systems. The need to manage and synchronize multiple processes can make it more difficult to design and implement system features and ensure system stability.
3. **Scalability:** The process table and PCB may not scale well for large-scale systems with many processes. As the number of processes increases, the process table and PCB can become larger and more difficult to manage efficiently.
4. **Security:** The process table and PCB can introduce security risks if they are not implemented correctly. Malicious programs can potentially access or modify the process table and PCB to gain unauthorized access to system resources or cause system instability.

5. **Miscellaneous accounting and status data –** This field includes information about the amount of CPU used, time constraints, jobs or process number, etc. The process control block stores the register content also known as execution content of the processor when it was blocked from running. This execution content architecture enables the operating system to restore a process's execution context when the process returns to the running state. When the process makes a transition from one state to another, the operating system updates its information in the process's PCB. The operating system maintains pointers to each process's PCB in a process table so that it can access the PCB quickly.



Process table and process control block

1: What is a Process Control Block (PCB)?

A process control block (PCB) is a data structure used by operating systems to store important information about running processes. It contains information such as the unique identifier of the process (Process ID or PID), current status, program counter, CPU registers, memory allocation, open file descriptions and accounting information. The circuit is critical to context switching because it allows the operating system to efficiently manage and control multiple processes.

2: What information does a Process Control Block (PCB) contain?

A process control board (PCB) stores various information about a process so that the operating system can manage it properly.

A typical printed circuit board contains the following components:

Process ID (PID): a unique identifier for each process. **Process Status:** Indicates whether the process is currently running, ready, pending, or stopped. The program counter stores the address of the next instruction to be executed. **CPU Registers:** Stores the current CPU register values for context switching. **Memory Management Information:** Information about memory allocation and process usage. I/O Information: Tracks the status of the I/O devices assigned to the process. Accounting information: resource usage statistics such as CPU time, I/O time, etc.

3: How does the Process Control Block (PCB) facilitate context switching?

Context switching is the process of saving the current state of a running process and loading the state of another process so that the CPU can switch its execution from one process to another. The process control block (PCB) plays a key role in context switching because it contains all relevant information about the process. When the operating system decides to switch to another process, it stores the current process in the circuit’s memory, including CPU registers and program counters. It then loads the chip to start the next process, resets its state and resumes execution from where it left off. This seamless switching between processes allows the operating system to create the illusion of simultaneous execution, even though the processor can only run one process at a time.