**Process Management**

The term "process" was first used by the designers of the MULTICS in 1960's. Since then, the term process, used somewhat interchangeably with 'task' or 'job'. The process has been given many definitions for instance

* A program in Execution.
* An asynchronous activity.
* The 'animated sprit' of a procedure in execution.
* The entity to which processors are assigned.
* The 'dispatchable' unit.

Process is not the same as program. A process is more than a program code. A process is an 'active' entity as oppose to program which consider to be a 'passive' entity. As we all know that a program is an algorithm expressed in some suitable notation, (e.g., programming language). Being a passive, a program is only a part of process.

Process, on the other hand, includes:

* Current value of Program Counter (PC)
* Contents of the processors registers
* Value of the variables
* The process stack (SP) which typically contains temporary data such as subroutine parameter, return address, and temporary variables.
* A data section that contains global variables.

The process state consist of everything necessary to resume the process execution if it is somehow put aside temporarily. The process state consists of at least following:

* Code for the program.
* Program's static data.
* Program's dynamic data.
* Program's procedure call stack.
* Contents of general purpose registers.
* Contents of program counter (PC)
* Contents of program status word (PSW).
* Operating Systems resource in use.

**Process State**

As a process executes, it changes state. The state of a process is defined in part by the current activity of that process. Each process may be in one of the following states:

**New State:** The process being created.

**Running State**: A process is said to be running if it has the CPU, that is, process actually using the CPU at that particular instant.

**Blocked (or waiting) State:** A process is said to be blocked if it is waiting for some event to happen such that as an I/O completion before it can proceed. Note that a process is unable to run until some external event happens.

**Ready State:** A process is said to be ready if it is waiting to be assigned to a processor.

**Terminated state:** The process has finished execution.

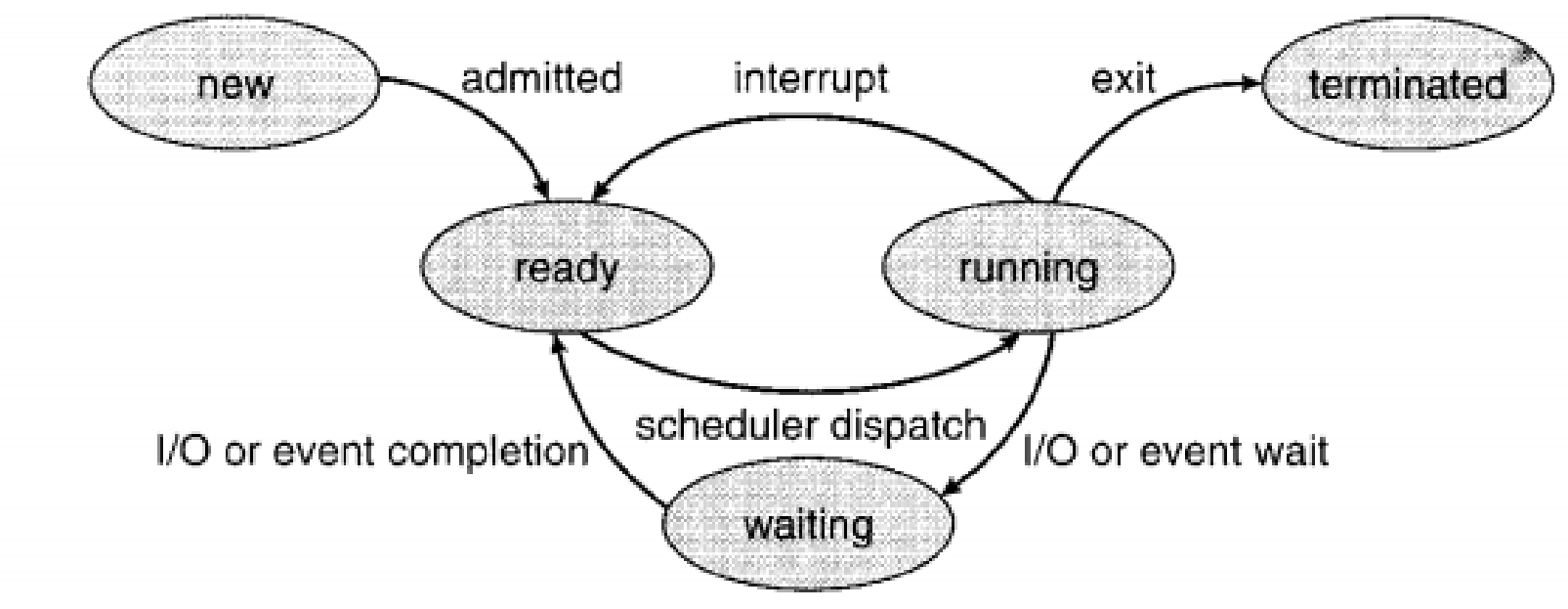


Figure : Diagram of process states.

**Process Control Block**

• Each process is represented in the operating system by a process control block PCS)—also

called a task control block

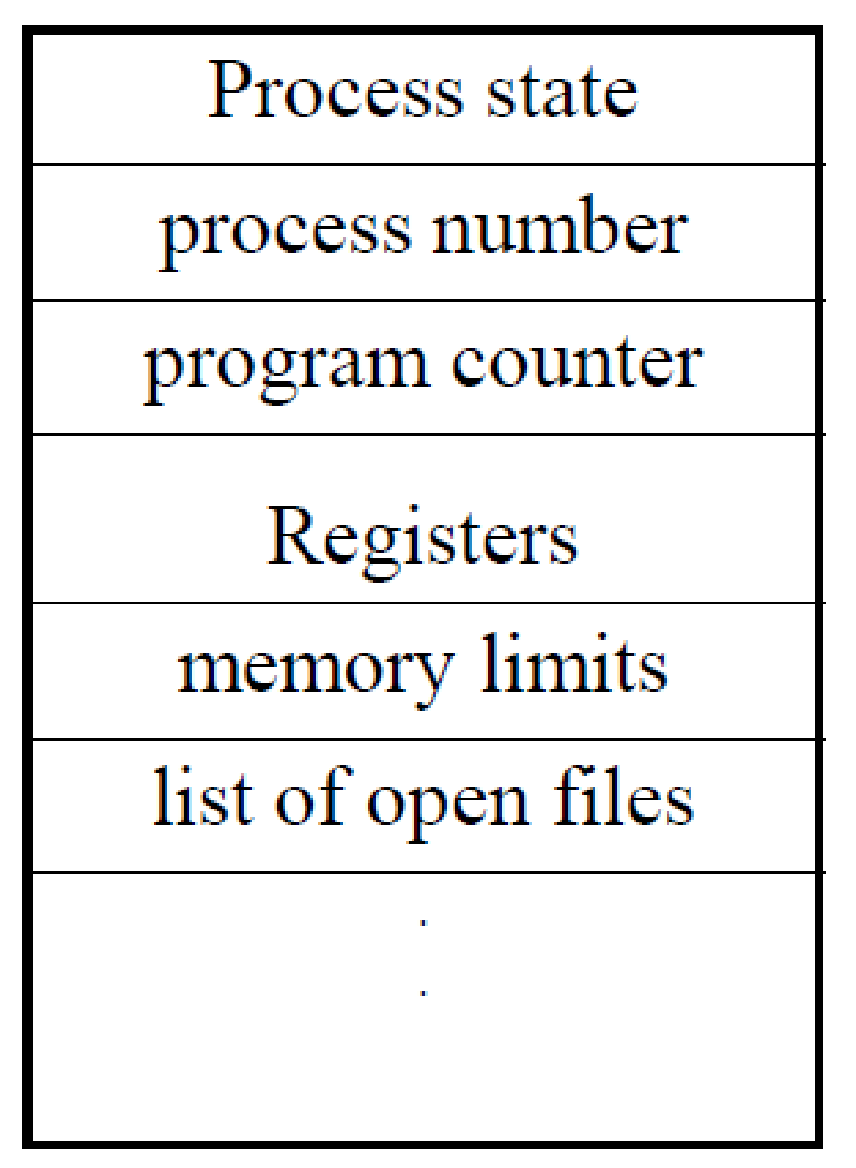
. 

Figure: Process control block.

While creating a process the operating system performs several operations. To identify these process, it must identify each process, hence 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 these information is required and must be saved when the process is switched from one state to another. When the process made transitions 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 PCB’s, that means logically contains a PCB for all of the current processes in the system.

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

**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 made transitions from one state to another, the operating system update 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.

**Remote Procedure Calls**

A remote procedure call is an inter-process communication technique that is used for client-server based applications. It is also known as a subroutine call or a function call.

A client has a request message that the RPC translates and sends to the server. This request may be a procedure or a function call to a remote server. When the server receives the request, it sends the required response back to the client. The client is blocked while the server is processing the call and only resumed execution after the server is finished.

The sequences of events in a remote procedure call are given as follows:

* The client stub is called by the client.
* The client stub makes a system call to send the message to the server and puts the parameters in the message.
* The message is sent from the client to the server by the client’s operating system.
* The message is passed to the server stub by the server operating system.
* The parameters are removed from the message by the server stub.
* Then, the server procedure is called by the server stub.

A diagram that demonstrates this is as follows:

**Advantages of Remote Procedure Call**

Some of the advantages of RPC are as follows:

* Remote procedure calls support process oriented and thread oriented models.
* The internal message passing mechanism of RPC is hidden from the user.
* The effort to re-write and re-develop the code is minimum in remote procedure calls.
* Remote procedure calls can be used in distributed environment as well as the local environment.
* Many of the protocol layers are omitted by RPC to improve performance.

**Disadvantages of Remote Procedure Call**

Some of the disadvantages of RPC are as follows:

* The remote procedure call is a concept that can be implemented in different ways. It is not a standard.
* There is no flexibility in RPC for hardware architecture. It is only interaction based.
* There is an increase in costs because of remote procedure call.

Video Link: <https://www.youtube.com/watch?v=OrM7nZcxXZU&list=PLBlnK6fEyqRiVhbXDGLXDk_OQAeuVcp2O&index=16>

<https://www.youtube.com/watch?v=jZ_6PXoaoxo&list=PLBlnK6fEyqRiVhbXDGLXDk_OQAeuVcp2O&index=17>

<https://www.youtube.com/watch?v=4s2MKuVYKV8&list=PLBlnK6fEyqRiVhbXDGLXDk_OQAeuVcp2O&index=18>

REFERENCES:

1. Galvin, Peter B., Silberchatz, A., “*Operating System Concepts*”, Addison Wesley, 8th Edition.
2. Flynn, *“Operating Systems”*, Cengage Learning.
3. Dhamdhere, D.M., "*Operating System:A Concept Based Approach*",  
   Tata Mc-Graw- Hill.
4. https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-828-operating-system-engineering-fall-2012/lecture-notes-and-readings/
5. <https://www.studytonight.com/operating-system/>
6. **https://www.youtube.com/watch?v=g\_kLI2pkNC4&list=PL3-wYxbt4yCjpcfUDz-TgD\_ainZ2K3MUZ&index=12**
7. https://nptel.ac.in/courses/106/105/106105214
8. <https://www.guru99.com/operating-system-tutorial.html>
9. <https://www.geeksforgeeks.org/operating-systems/>