

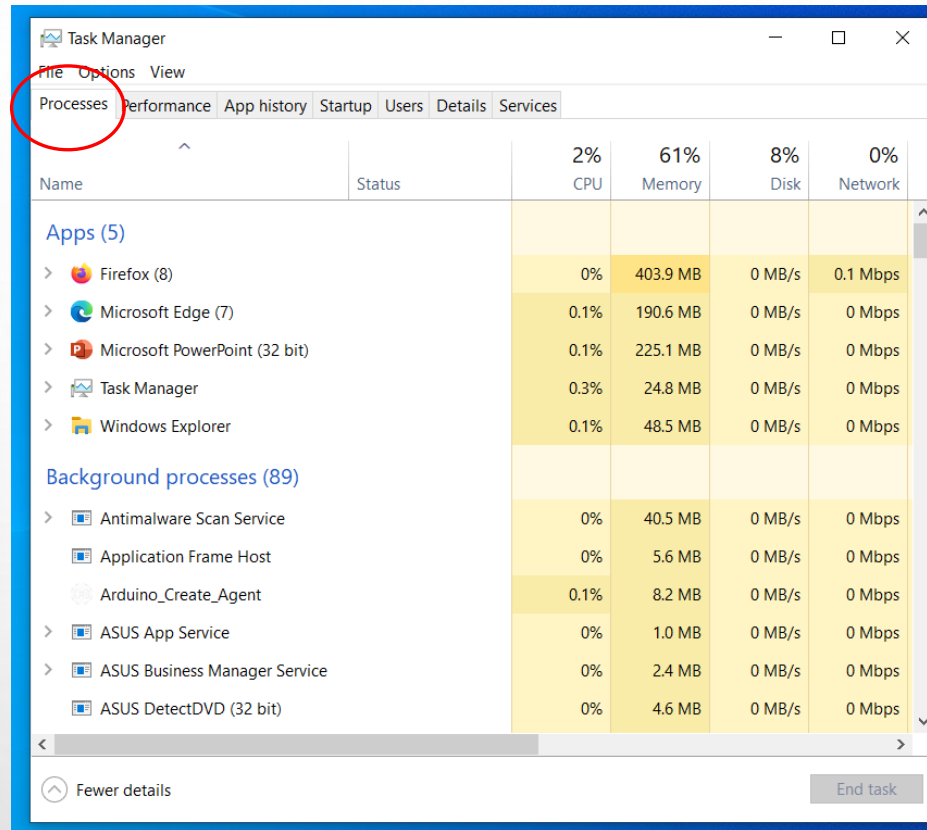
The background of the slide is a light gray gradient. It is decorated with numerous realistic water droplets of various sizes. Some droplets are large and prominent, while others are small and subtle. They are scattered across the slide, with a higher concentration in the top-left and bottom-right corners. The droplets have highlights and shadows, giving them a three-dimensional appearance.

PROCESS MANAGEMENT

PROCESS CONCEPT

- A process can be thought of as a **program in execution**
- To accomplish its task a process will need certain resources, such as:
CPU time, memory, files, and I/O devices
- These resources are allocated to the process either when it is created or while it is executing.

PROCESSES



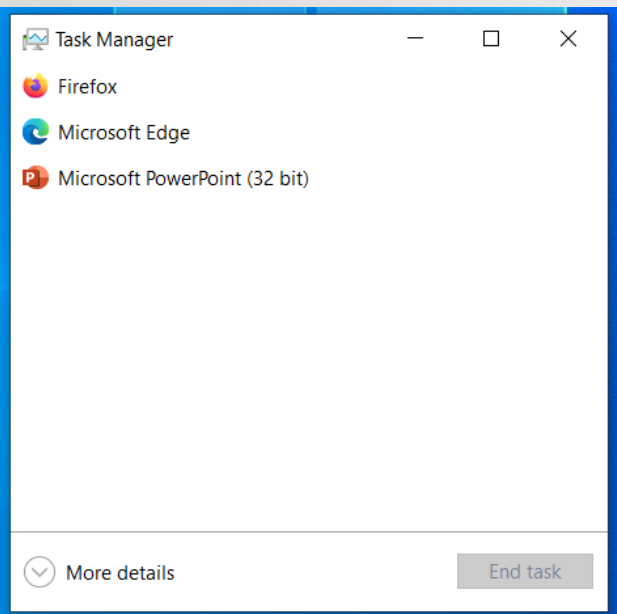
Task Manager

File Options View

Processes Performance App history Startup Users Details Services

Name	Status	2% CPU	61% Memory	8% Disk	0% Network
Apps (5)					
> Firefox (8)		0%	403.9 MB	0 MB/s	0.1 Mbps
> Microsoft Edge (7)		0.1%	190.6 MB	0 MB/s	0 Mbps
> Microsoft PowerPoint (32 bit)		0.1%	225.1 MB	0 MB/s	0 Mbps
> Task Manager		0.3%	24.8 MB	0 MB/s	0 Mbps
> Windows Explorer		0.1%	48.5 MB	0 MB/s	0 Mbps
Background processes (89)					
> Antimalware Scan Service		0%	40.5 MB	0 MB/s	0 Mbps
Application Frame Host		0%	5.6 MB	0 MB/s	0 Mbps
Arduino_Create_Agent		0.1%	8.2 MB	0 MB/s	0 Mbps
> ASUS App Service		0%	1.0 MB	0 MB/s	0 Mbps
> ASUS Business Manager Service		0%	2.4 MB	0 MB/s	0 Mbps
ASUS DetectDVD (32 bit)		0%	4.6 MB	0 MB/s	0 Mbps

Fewer details End task



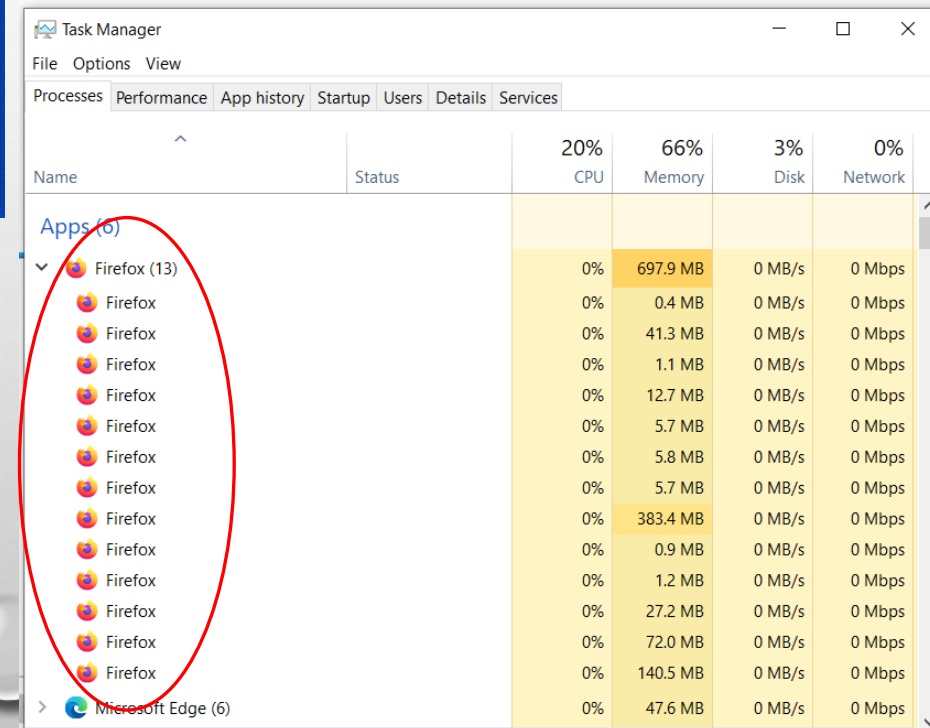
Task Manager

Firefox

Microsoft Edge

Microsoft PowerPoint (32 bit)

More details End task



Task Manager

File Options View

Processes Performance App history Startup Users Details Services

Name	Status	20% CPU	66% Memory	3% Disk	0% Network
Apps (6)					
> Firefox (13)		0%	697.9 MB	0 MB/s	0 Mbps
Firefox		0%	0.4 MB	0 MB/s	0 Mbps
Firefox		0%	41.3 MB	0 MB/s	0 Mbps
Firefox		0%	1.1 MB	0 MB/s	0 Mbps
Firefox		0%	12.7 MB	0 MB/s	0 Mbps
Firefox		0%	5.7 MB	0 MB/s	0 Mbps
Firefox		0%	5.8 MB	0 MB/s	0 Mbps
Firefox		0%	5.7 MB	0 MB/s	0 Mbps
Firefox		0%	383.4 MB	0 MB/s	0 Mbps
Firefox		0%	0.9 MB	0 MB/s	0 Mbps
Firefox		0%	1.2 MB	0 MB/s	0 Mbps
Firefox		0%	27.2 MB	0 MB/s	0 Mbps
Firefox		0%	72.0 MB	0 MB/s	0 Mbps
Firefox		0%	140.5 MB	0 MB/s	0 Mbps
> Microsoft Edge (6)		0%	47.6 MB	0 MB/s	0 Mbps

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.
- A process may be in one of the following states:

New: The process is being created.

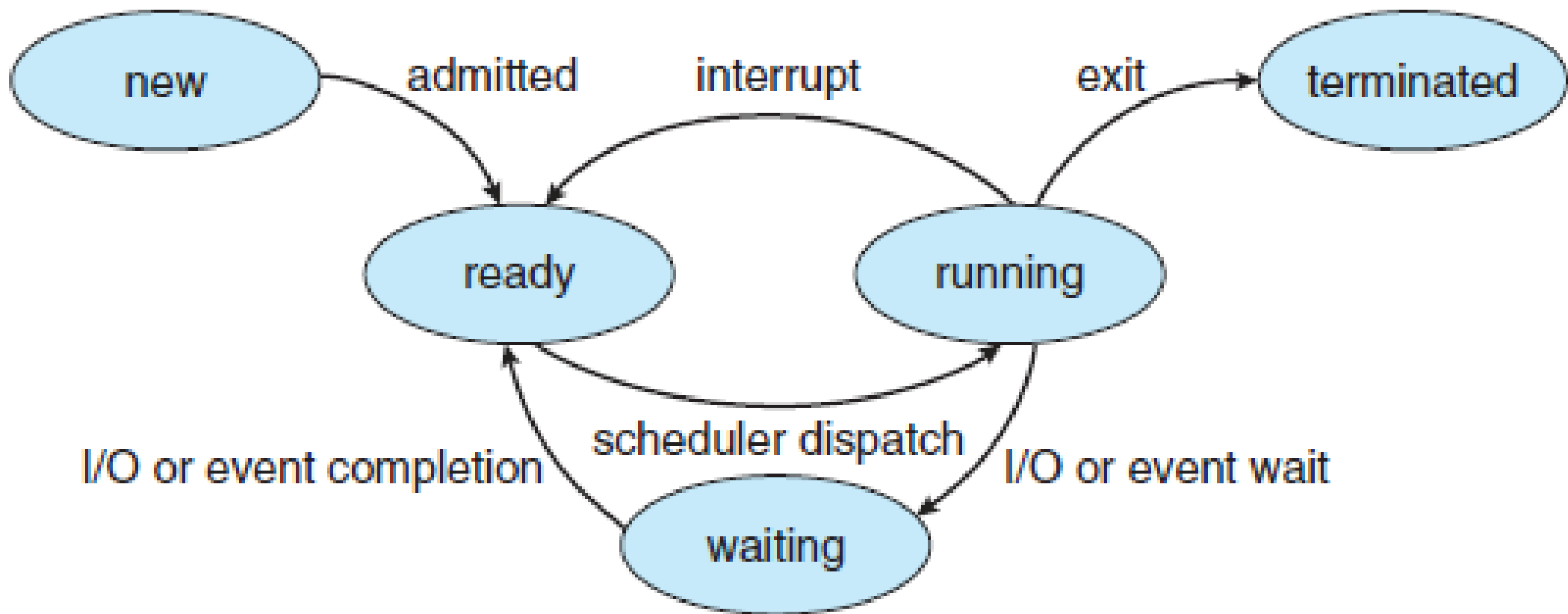
Running: Instructions are being executed.

Waiting: The process is waiting for some event to occur (such as an I/O completion or reception of a signal).

Ready: The process is waiting to be assigned to a processor.

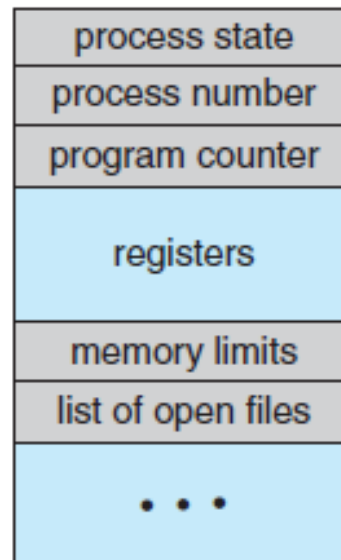
Terminated: The process has finished execution.

PROCESS STATE



PROCESS CONTROL BLOCK

- Each process is represented in the operating system by a **Process Control Block (PCB)** also called a **Task Control Block**.



PROCESS CONTROL BLOCK

- **Process number:** Unique ID to identify the process
- **Process state:** The state may be new, ready, running, waiting, and terminated.
- **Program counter:** The counter indicates the address of the next instruction to be executed for this process.
- **CPU registers:** The registers vary in number and type, depending on the computer architecture. They include accumulators, index registers, stack pointers, and general-purpose registers, plus any condition-code information. Along with the program counter, this state information must be saved when an interrupt occurs, to allow the process to be continued correctly afterward

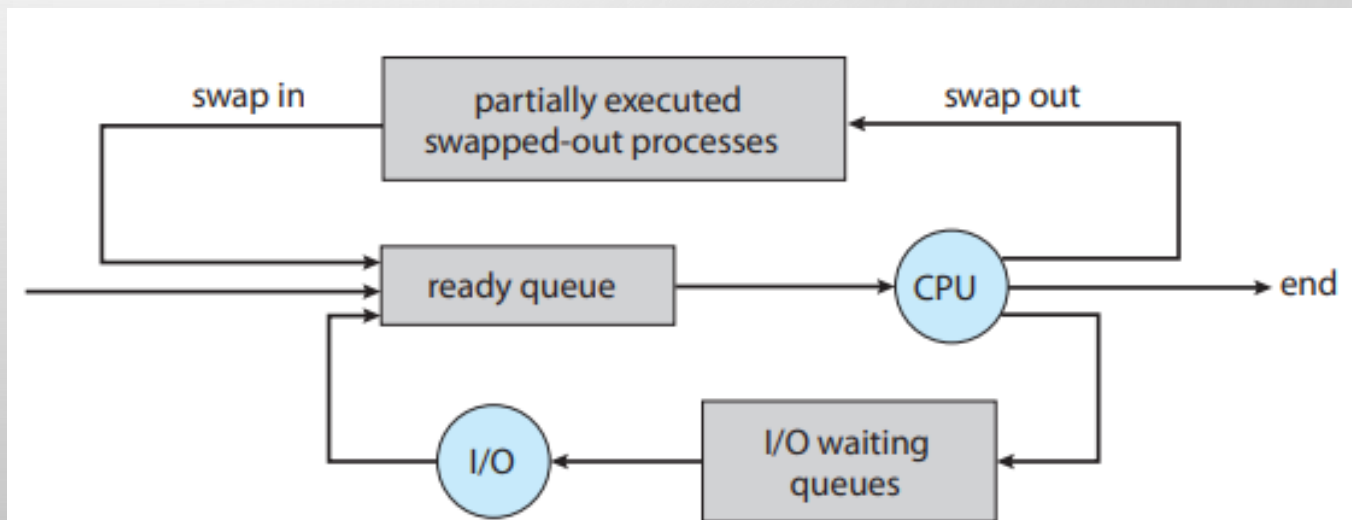
- CPU-Scheduling Information : This information includes a process priority, pointers to scheduling queues, and any other scheduling parameters.
- Memory-Management Information: This information may include such items as the value of the base and limit registers and the page tables, or the segment tables, depending on the memory system used by the operating system.
- Accounting Information: This information includes the amount of CPU and real time used, time limits, account numbers, job or process numbers, and so on.
- I/O Status Information: This information includes the list of I/O devices allocated to the process, a list of open files, and so on

PROCESS SCHEDULING

- The objective of multiprogramming is to have some process running at all times, to maximize CPU utilization.
- The objective of time sharing is to switch the CPU among processes so frequently that users can interact with each program while it is running.
- To meet these objectives, the **Process Scheduler** selects an available process (possibly from a set of several available processes) for program execution on the CPU

SCHEDULING QUEUES

- Job Queue: As processes enter the system, they are put into a job queue, which consists of all processes in the system.
- Ready Queue: The processes that are residing in main memory and are ready and waiting to execute are kept on a list called the ready queue.



CONTEXT SWITCH

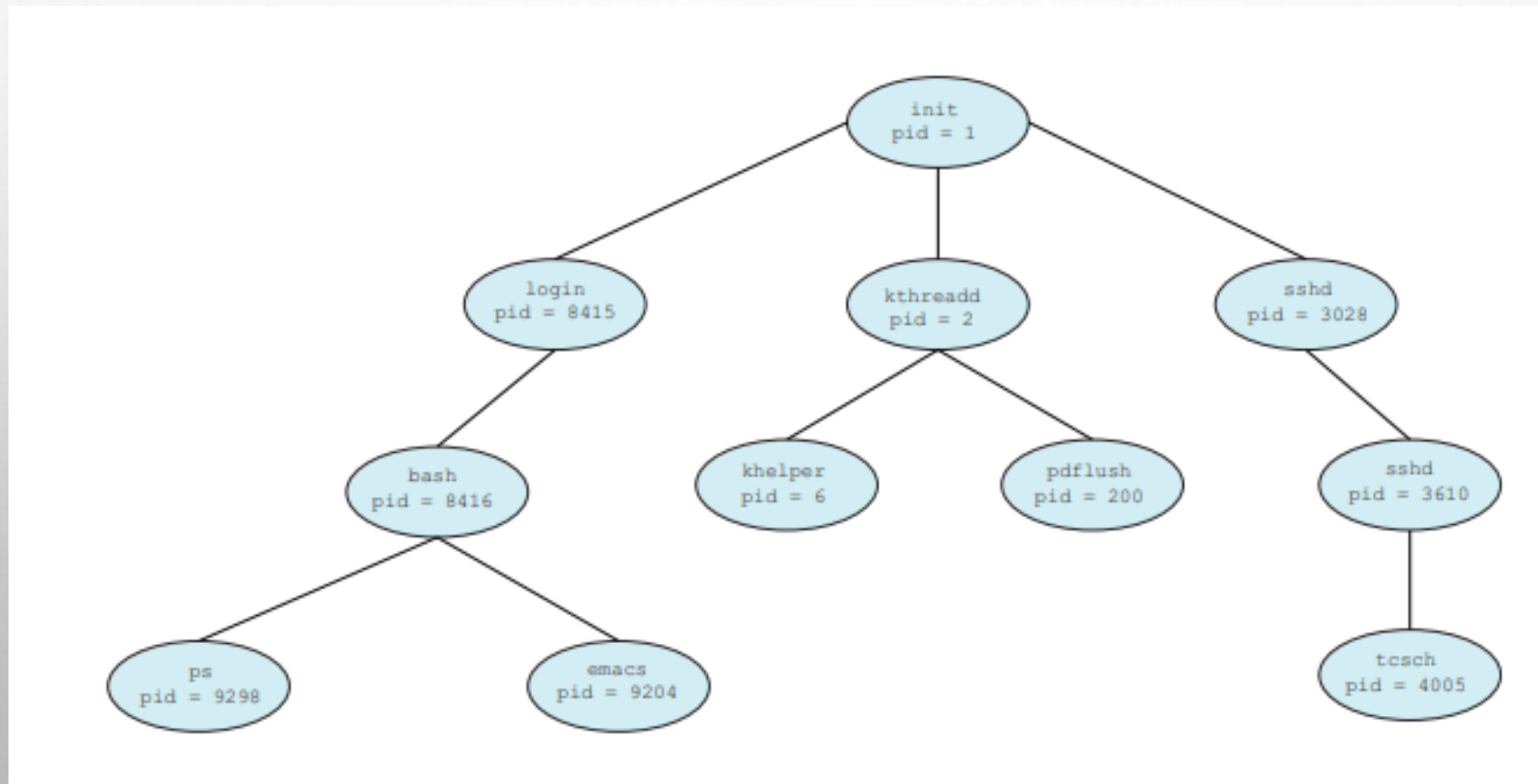
- Interrupts cause the operating system to change a CPU from its current task and to run a kernel routine.
- Such operations happen frequently on general-purpose systems.
- When an interrupt occurs, the system needs to save the current context of the process running on the CPU so that it can restore that context when its processing is done, essentially suspending the process and then resuming it.
- The context is represented in the PCB of the process.
- Switching the CPU to another process requires performing a state save of the current process and a state restore of a different process. This task is known as a **CONTEXT SWITCH**.

OPERATIONS ON PROCESSES

PROCESS CREATION

- During the course of execution, a process may create several new processes.
- The **creating process** is called a **parent process**, and the new processes are called the **children of that process**.
- Each of these new processes may in turn create other processes, forming **a tree of processes**.
- Most operating systems (including unix, linux, and windows) identify processes according to a unique **Process Identifier (or PID)**, which is typically an integer number.
- The **PID** provides a unique value for each process in the system, and it can be used as an index to access various attributes of a process within the kernel.

A TREE OF PROCESSES ON A TYPICAL LINUX SYSTEM






When a process creates a new process, two possibilities for execution exist:

1. The parent continues to execute concurrently with its children.
2. The parent waits until some or all of its children have terminated.

There are also two address-space possibilities for the new process:


1. The child process is a duplicate of the parent process (it has the same program and data as the parent).
 2. The child process has a new program loaded into it
- 

PROCESS TERMINATION

- A process terminates when it finishes executing its final statement and asks the operating system to delete it by using the `exit()` system call.
- At that point, the process may return a status value (typically an integer) to its parent process (via the `wait()` system call).
- All the resources of the process including physical and virtual memory, open files, and I/O buffers are deallocated by the operating system.
- A process can cause the termination of another process via an appropriate system call (for example, `TerminateProcess()` in Windows).
- Usually, such a system call can be invoked only by the parent of the process that is to be terminated.




A parent may terminate the execution of one of its children for a variety of reasons, such as these:


- The child has exceeded its usage of some of the resources that it has been allocated.
 - The task assigned to the child is no longer required.
 - The parent is exiting, and the operating system does not allow a child to continue if its parent terminates
- 

INTER PROCESS COMMUNICATION

- Processes executing concurrently in the operating system may be either **independent processes** or **cooperating processes**.
- A process is independent if it cannot affect or be affected by the other processes executing in the system.
- Any process that does not share data with any other process is independent.
- A process is cooperating if it can affect or be affected by the other processes executing in the system.
- Clearly, any process that shares data with other processes is a cooperating process.

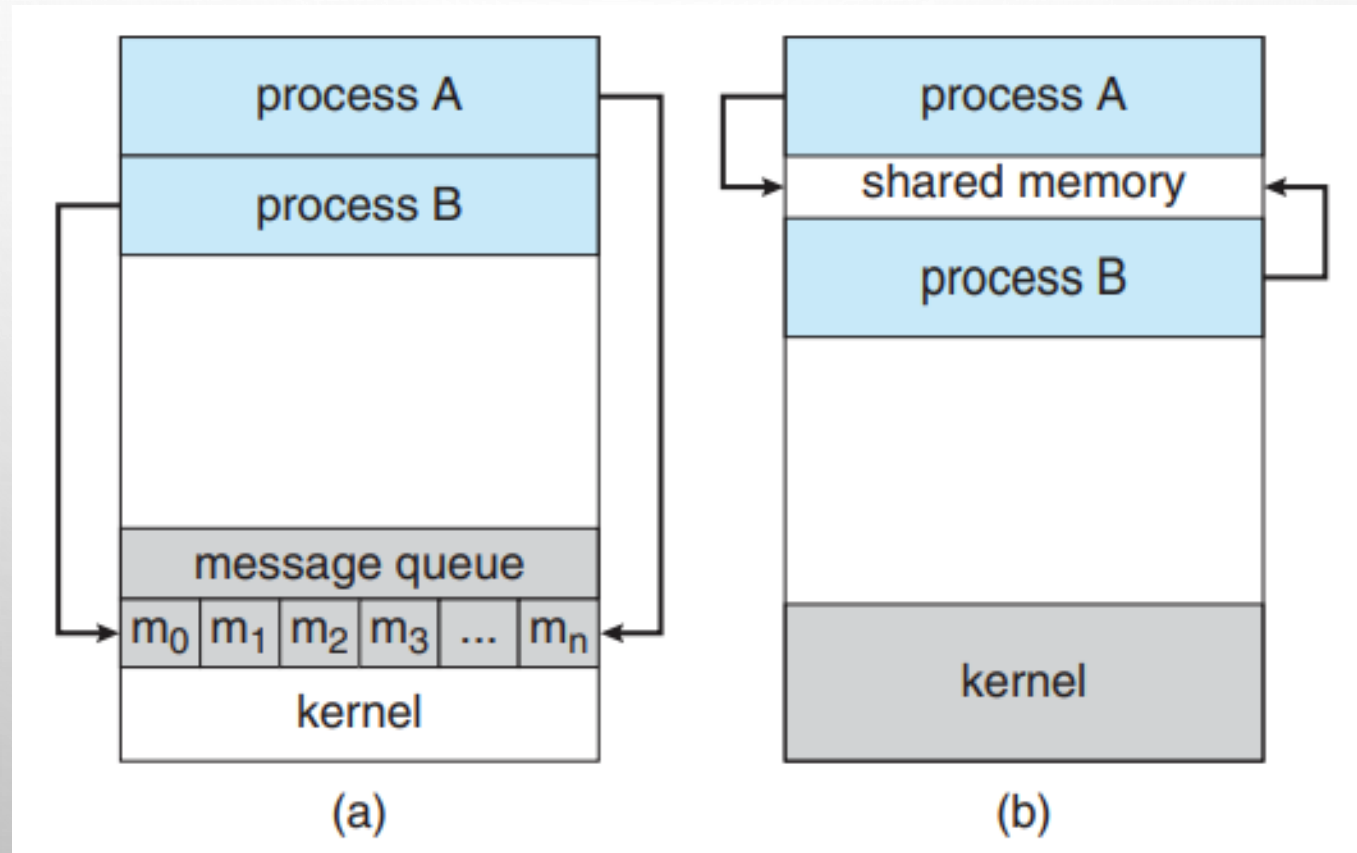
- 
- Cooperating processes require an Inter Process Communication (IPC) mechanism that will allow them to exchange data and information.
 - There are two fundamental models of Inter Process Communication:

Shared memory and Message passing

- In the shared-memory model, a region of memory that is shared by cooperating processes is established.
 - Processes can then exchange information by reading and writing data to the shared region.
 - In the message-passing model, communication takes place by means of messages exchanged between the cooperating processes.
- 

COMMUNICATIONS MODELS

(A) Message Passing. (B) Shared Memory





• Topics for Independent learning:

- Schedulers (3.2.2)
 - Shared-memory systems (3.4.1)
 - Message-passing systems (3.4.2)
- 