Processes

Dr Anas Al-dabbagh

In this lesson

- Process Concept
- Process Scheduling
- Operations on Processes
- Interprocess Communication
- Examples of IPC Systems

Process Concept

A process is a program in execution. It consists of the program code and its current activity, including the program counter, registers, and variables.

<u>**Process Components**</u>

1. **Text Section**: The program code (instructions to be executed).

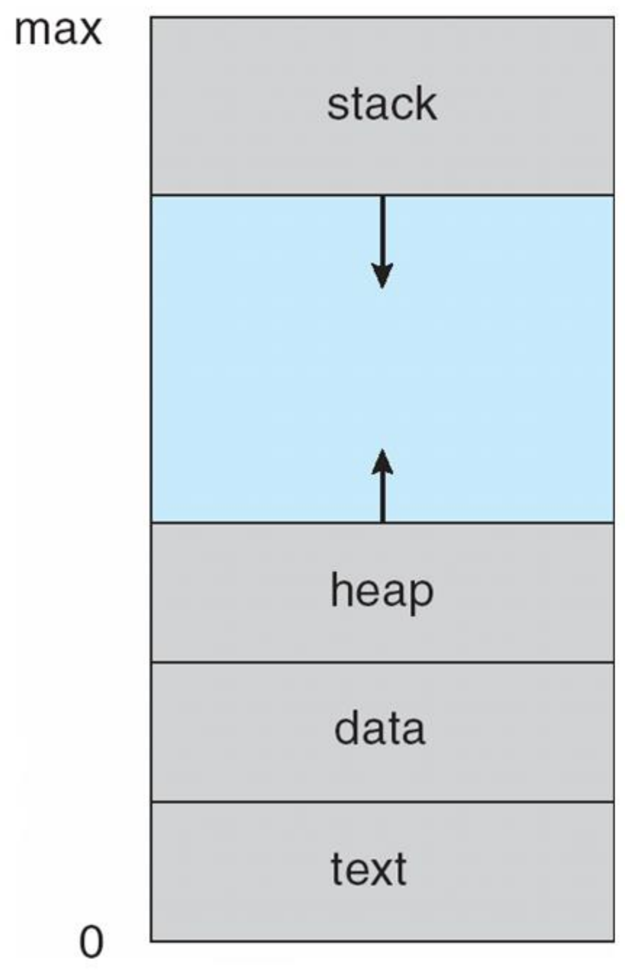
2. **Program Counter**: Indicates the next instruction to be executed.

3. **Stack**: Contains function parameters, return addresses, and local variables.

4. **Data Section**: Holds global and static variables.

5. **Heap**: Dynamically allocated memory during program execution.

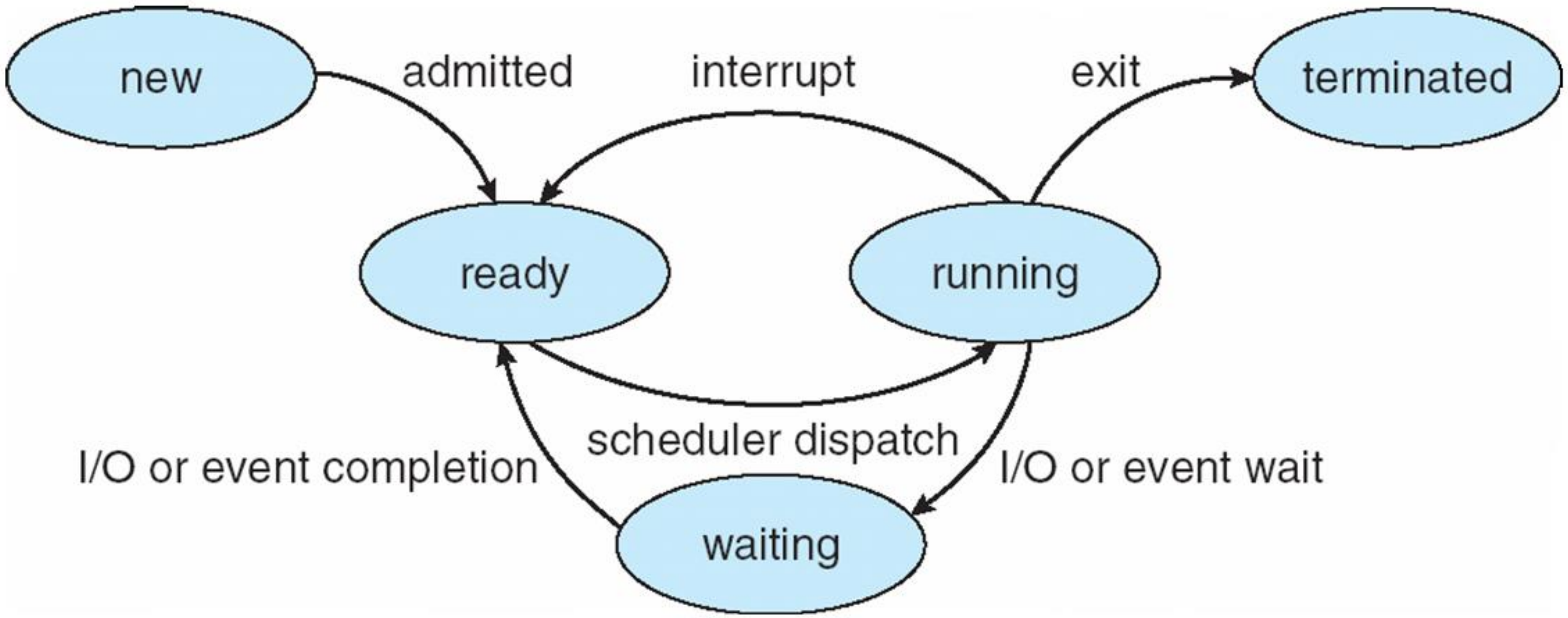
Process in Memory



Process States

- New: Process is being created.
- Running: Instructions are being executed.
- Waiting: Process is waiting for some event (like I/O).
- Ready: Process is ready to be assigned to a CPU.
- Terminated: Process has finished execution.

Diagram of Process State

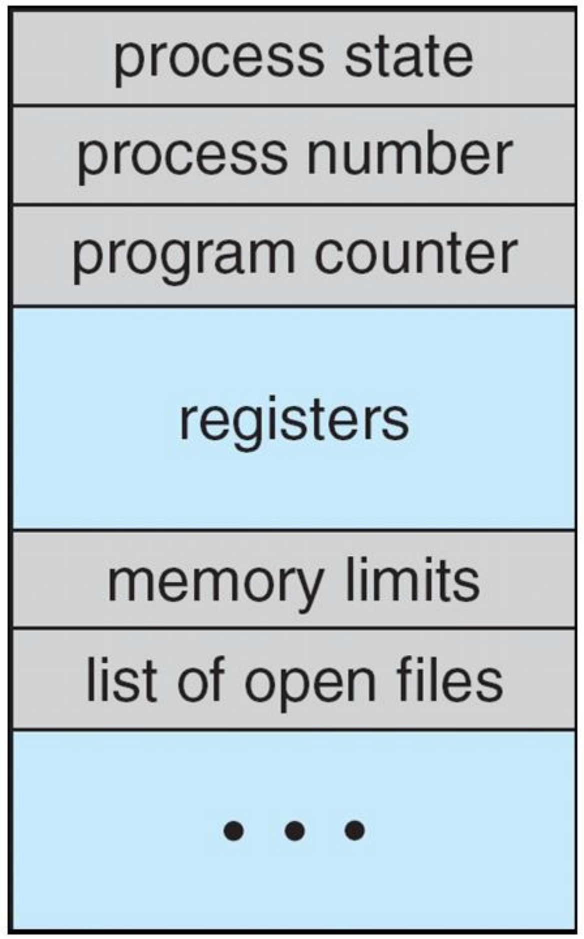


Process Control Block (PCB)

- A data structure maintained by the OS to manage process information.

Contents of PCB:

- **Process State**: Current status of the process.
- **Process ID (PID)**: Unique identifier for each process.
- **Program Counter**: Address of the next instruction.
- **CPU Registers**: Contents of all process-specific registers.
- **Memory Management Info**: Base and limit registers, page tables.
- **I/O Status**: Information about I/O devices allocated to the process.



Process Scheduling

- Process scheduling is the activity of the operating system that determines which process will execute on the CPU and for how long. It ensures that processes are executed efficiently, maximizing CPU utilization and system performance.

Objectives of Process Scheduling

1. **Maximize CPU Utilization**: Keep the CPU as busy as possible by switching between processes.

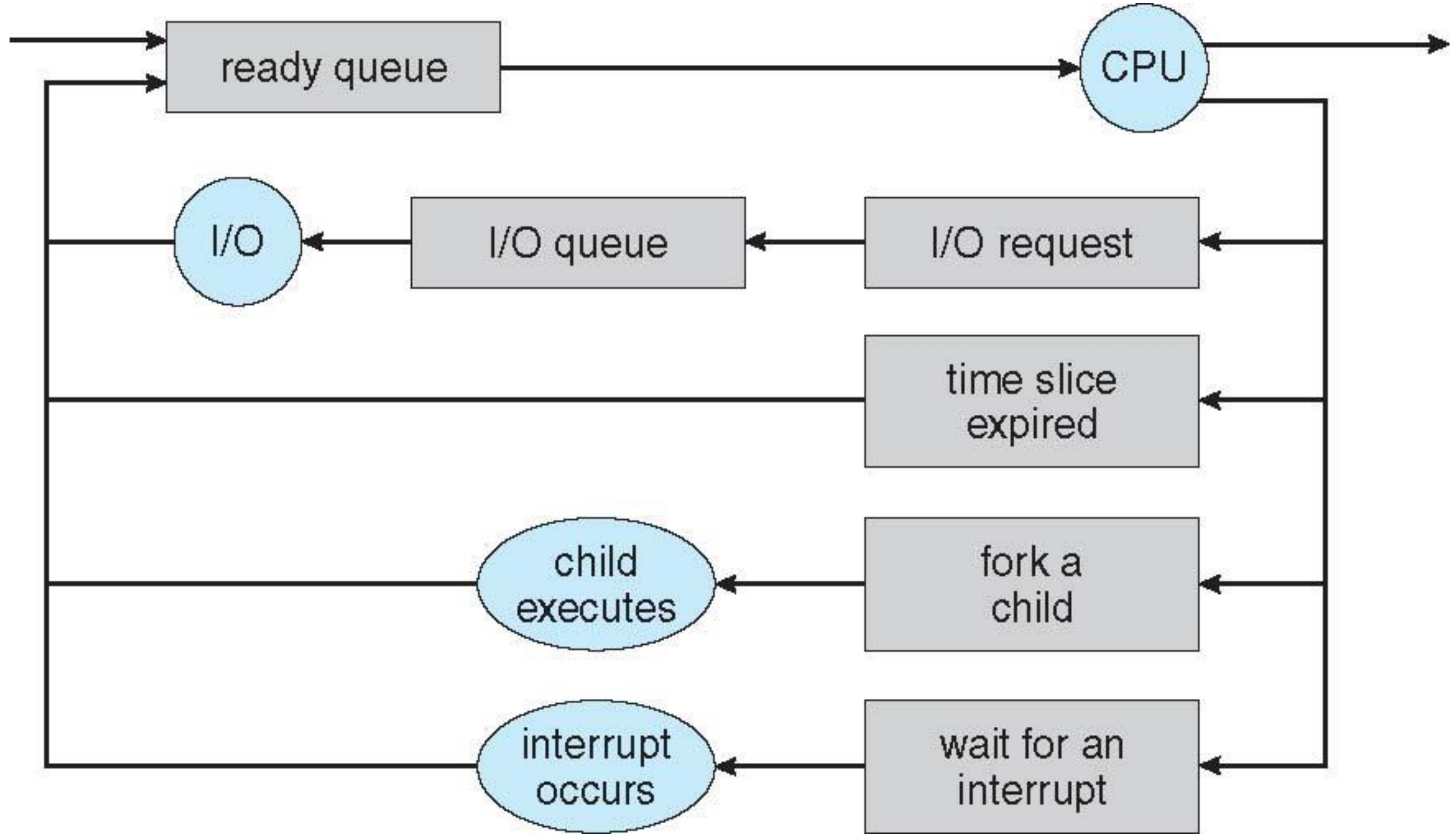
2. **Throughput**: Increase the number of processes completed per unit of time.

3. **Minimize Waiting Time**: Reduce the time a process spends waiting in the ready queue before execution.

4. **Response Time**: Minimize the time between a process request and its first execution.

5. **Fairness**: Ensure all processes get a fair share of CPU time and no process is starved.

Representation of Process Scheduling



Schedulers

- **Short-term scheduler (CPU Scheduler)**

  The short-term scheduler selects the next process to execute, allocating CPU resources. It’s responsible for managing the active processes on a frequent basis (milliseconds). As it needs to be fast, it's invoked more frequently than other schedulers and sometimes is the only scheduler in certain systems.

- **Long-term scheduler (Job Scheduler)**

  The long-term scheduler decides which processes are to be brought into the ready queue. It operates less frequently (seconds or minutes), which means it can afford to be slower. This scheduler controls the degree of multiprogramming in the system by balancing the number of processes in memory at any time.

Process Types

- **I/O-bound process:** A process that spends more time on I/O than on computations, typically having many short CPU bursts.

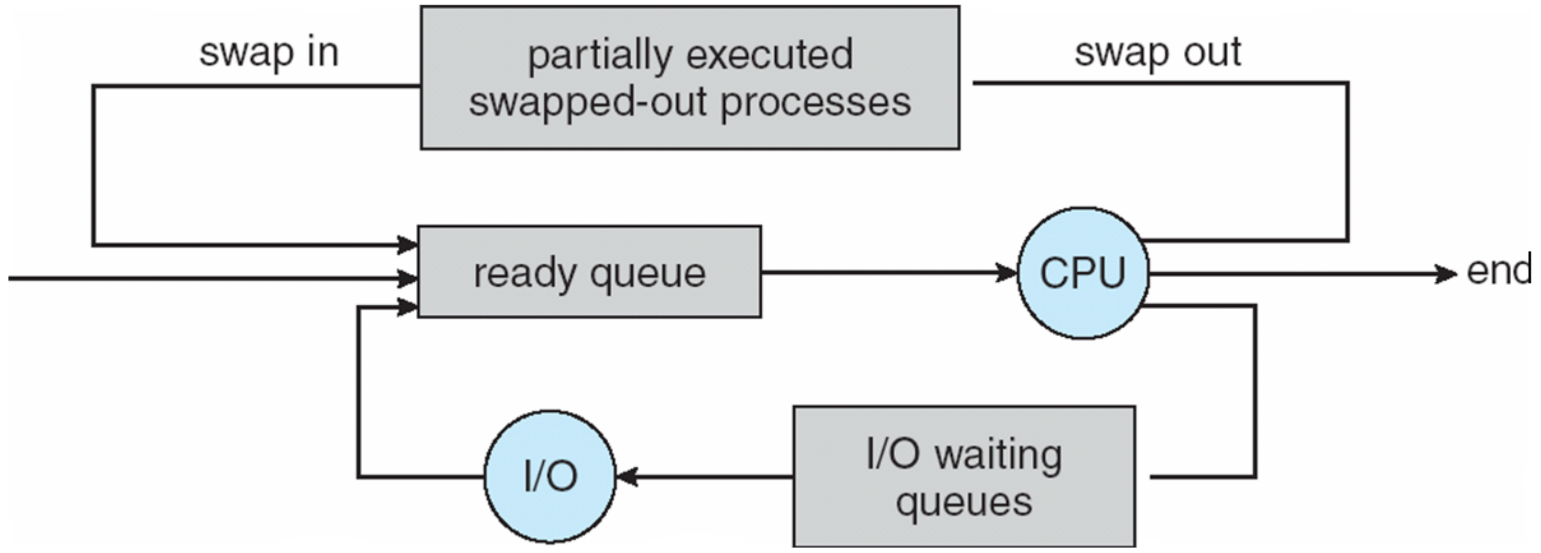
- **CPU-bound process:** A process that focuses more on computations, featuring fewer but longer CPU bursts.

- Goal of Long-term Scheduler

The long-term scheduler strives to maintain a good mix of I/O-bound and CPU-bound processes to ensure efficient system performance.

Schedulers

- **Medium-term scheduler** can be added if degree of multiple programming needs to decrease
  - Remove process from memory, store on disk, bring back in from disk to continue execution: **swapping**



Operations on Processes

1- Process Creation

Key Concepts:

Process: A running instance of a program, including its code, current activity, and allocated resources (memory, files, etc.).

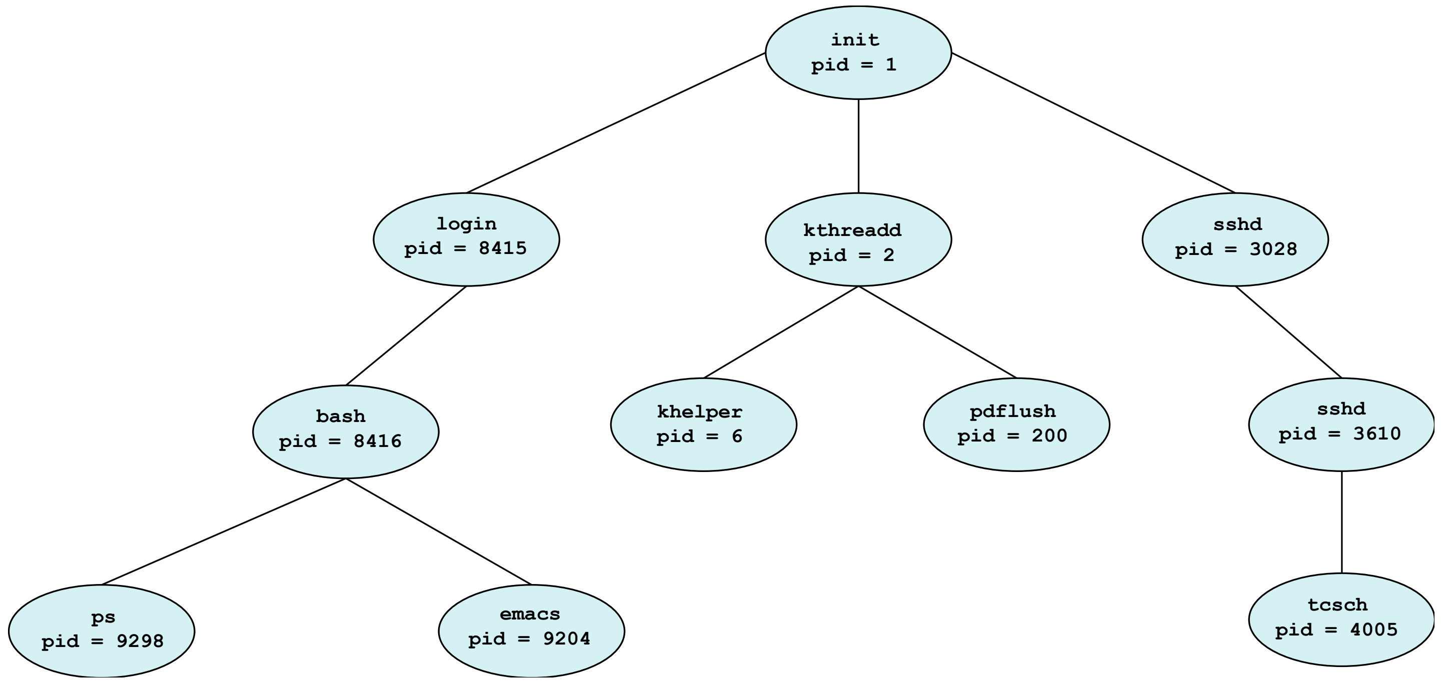
Parent Process: A process that initiates or creates one or more new processes.

Child Process: A process created by the parent process, which often inherits certain attributes from the parent.

Process Creation

- **Parent-Child Hierarchy**:
  - Each process in an operating system has a unique Process Identifier (**PID**).
  - A process can create one or more child processes using system calls.
  - Child processes inherit resources (like open files) from the parent but can run independently.
- **Process Tree**:
  - Processes form a hierarchical structure called a **process tree**, where a parent can have multiple children.
  - The tree starts with the **init process** (or systemd) at the top, which creates and manages all other processes.

A Tree of Processes in Linux



Process Termination

- Process executes last statement and then asks the operating system to delete it using the **`exit()`** system call.
  - Returns status data from child to parent (via **`wait()`**)
  - Process' resources are deallocated by operating system
- Parent may terminate the execution of children processes using the **`abort()`** system call. Some reasons for doing so:
  - Child has exceeded allocated resources
  - Task assigned to child is no longer required
  - The parent is exiting and the operating systems does not allow a child to continue if its parent terminates

Process Termination

- Some operating systems do not allow child to exists if its parent has terminated. If a process terminates, then all its children must also be terminated.
  - **cascading termination.** All children, grandchildren, etc. are terminated.
  - The termination is initiated by the operating system.
- The parent process may wait for termination of a child process by using the **`wait()`** system call . The call returns status information and the PID of the terminated process

```
pid = wait(&status);
```

- If no parent waiting (did not invoke **`wait()`**) process is a zombie
- If parent terminated without invoking **`wait`** , process is an orphan

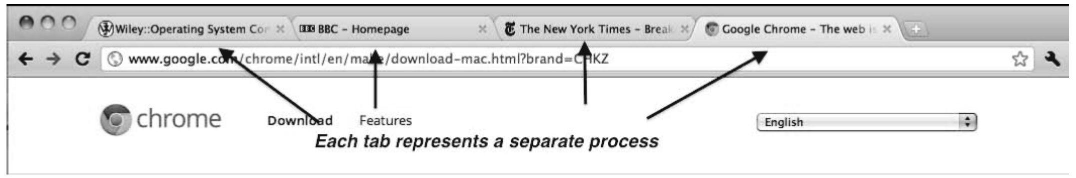
Multiprocess Architecture – Chrome Browser

Google Chrome is a multiprocess browser with three distinct types of processes:

•**Browser Process**: Manages the user interface, as well as disk and network input/output (I/O).

•**Renderer Process**: Responsible for rendering web pages, handling HTML and JavaScript.

•**Plug-in Process**: A separate process is created for each type of plug-in used in the browser.

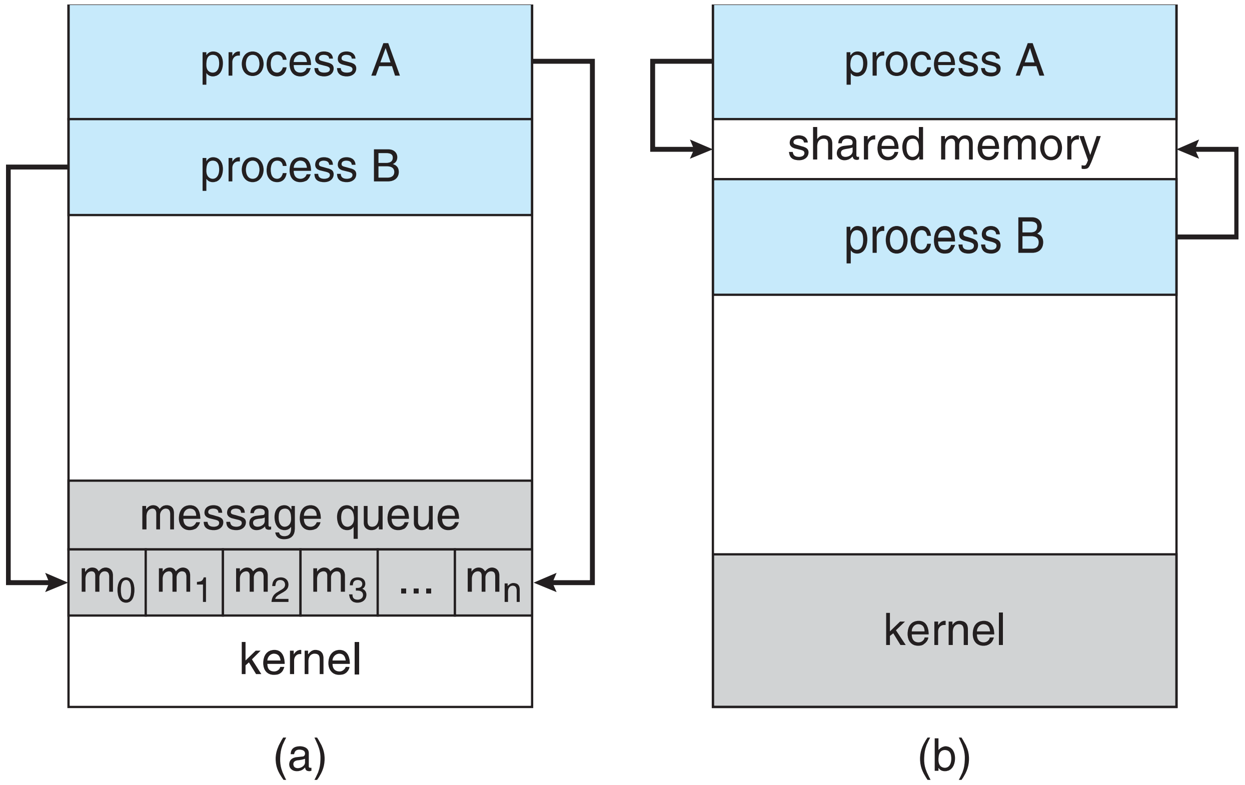

Wiley::Operating System Cor
BBC – Homepage
The New York Times – Break
Google Chrome – The web is
www.google.com/chrome/intl/en/make/download-mac.html?brand=CHKZ
chrome
Download
Features
Each tab represents a separate process
English

Interprocess Communication

- Processes within a system may be ***independent*** or ***cooperating***
- Cooperating process can affect or be affected by other processes, including sharing data
- Reasons for cooperating processes:
    - Information sharing
    - Computation speedup
    - Modularity
    - Convenience
- Cooperating processes need **interprocess communication** (**IPC**)
- Two models of IPC
    - **Shared memory**
    - **Message passing**

Communications Models

(a) Message passing. (b) shared memory.



Thank you