Operating Systems Fundamentals

Real-Time Operating Systems Programming (RTOSP)
Master in Critical Computing Systems Engineering (MCCSE)

2022/23

Paulo Baltarejo Sousa
pbs@isep.ipp.pt

Disclaimer

Material and Slides

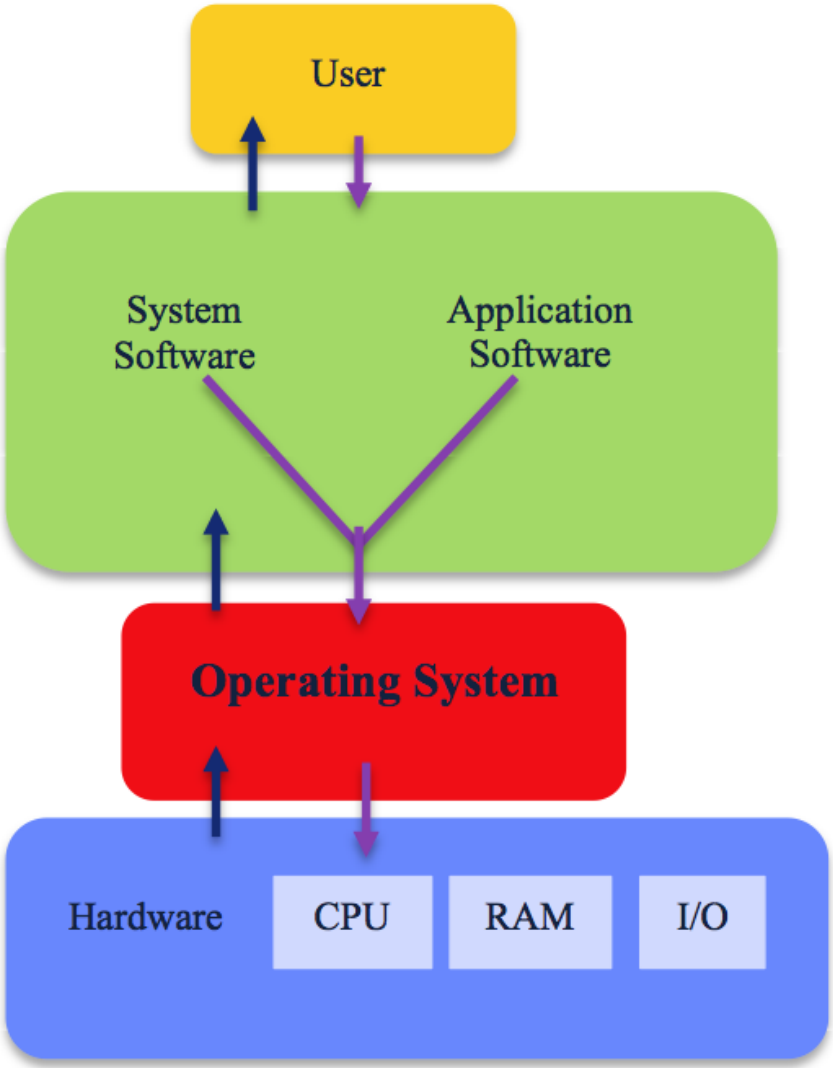
Some of the material/slides are adapted from various:

- Presentations found on the internet;
- Books;
- Web sites;
- ...

Outline

Operating System

Logical View



What is an Operating System (OS)?

- An OS **is the program that**, after being initially loaded into the computer by a boot program, **manages all of the software and hardware on the computer** (generally referred as **computer resources**).
  - OS is a program **that acts as an Interface between the system hardware and the user** making the tasks easier.
    - It performs basic tasks such as file, memory and process management, handling input and output, and controlling peripheral devices such as disk drives and printers.
  - The OS **coordinates all of this to make sure each program gets what it needs**.
    - Most of the time, there are several different programs (software) running at the same time, and they all need to access your computer's central processing unit (CPU), memory, storage and so on.
- The computer programs **make use of the OS by making requests for services through a defined application program interface (API)**.
  - Such API is, typically, implemented through **system calls**.

Why do we need an OS?

- **Convenience**:
  - OSs enable users to get started on the things they wish to complete quickly without having to cope with the stress of first configuring the system.
  - High-level **abstraction of physical resources**;
- **Efficiency**:
  - An OS enables the efficient use of resources.
    - This is due to less time spent configuring the system.
- **Ability to evolve**
  - An OS should be designed in such a way that it allows for the effective development, testing, and introduction of new features without interfering with service.
  - Enable **portable code**.
- **Management of system resources**
  - OS guarantees that resources are shared fairly among various processes and users.
    - Fairly?????? Depends on ....

Tasks of the Operating System (I)

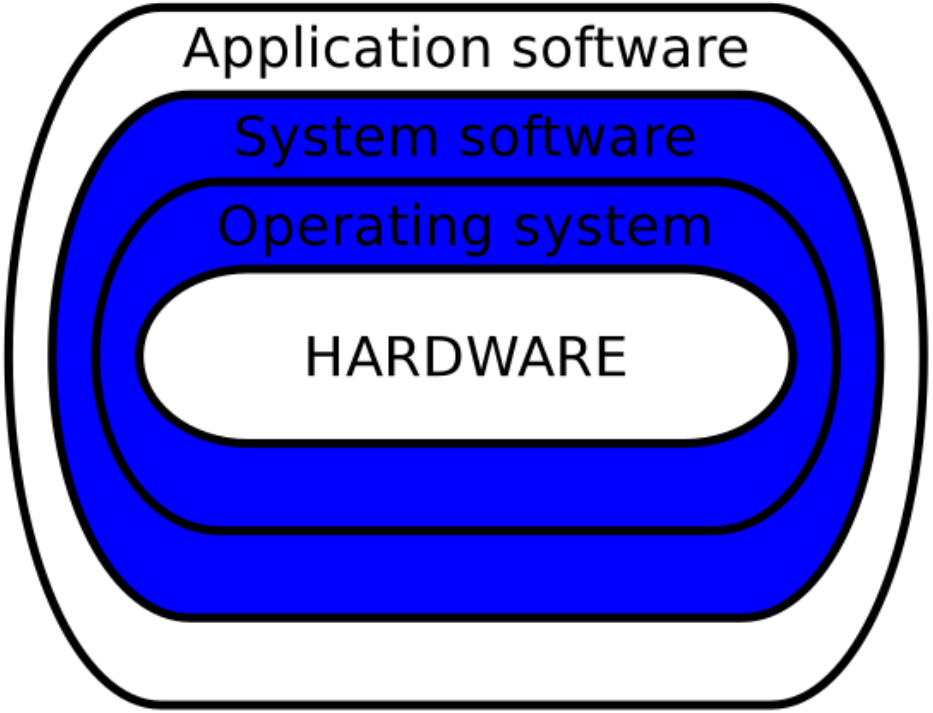
- **Interface between the user and the computer**:
  - An OS provides a very easy way to interact with the computer.
- **Booting**
  - Booting is basically the process of starting the computer.
  - It helps to start the computer when the power is switched ON.
- **Managing the input/output devices**
  - It helps to operate the different input/output devices.
  - It decides which program or process can use which device.
  - It controls the allocation and deallocation of devices.
- **Platform for other application software**
  - Users require different application programs to perform specific tasks on the system.
  - IT manages and controls these applications so that they can work efficiently.
  - It loads and runs application software as well as system software.

Tasks of the Operating System (II)

- **Manages the memory**
  - It allocates and deallocates memory to all the applications/tasks.
- **Manages the system files**
  - It helps to manage files on the system.
  - All the data on the system is in the form of files.
  - It makes interaction with the files easy.
- **Provides Security**
  - The operating system provides various techniques which assure the integrity and confidentiality of user data.
  - It keeps the system and applications safe through authorization/permission.

The System and Application Programming

- System components:
  - System software
    - Command Line Interface (Shell);
    - Graphical User Interface (GUI);
    - ...
  - Operating system
    - Process management;
    - Memory management;
    - ...

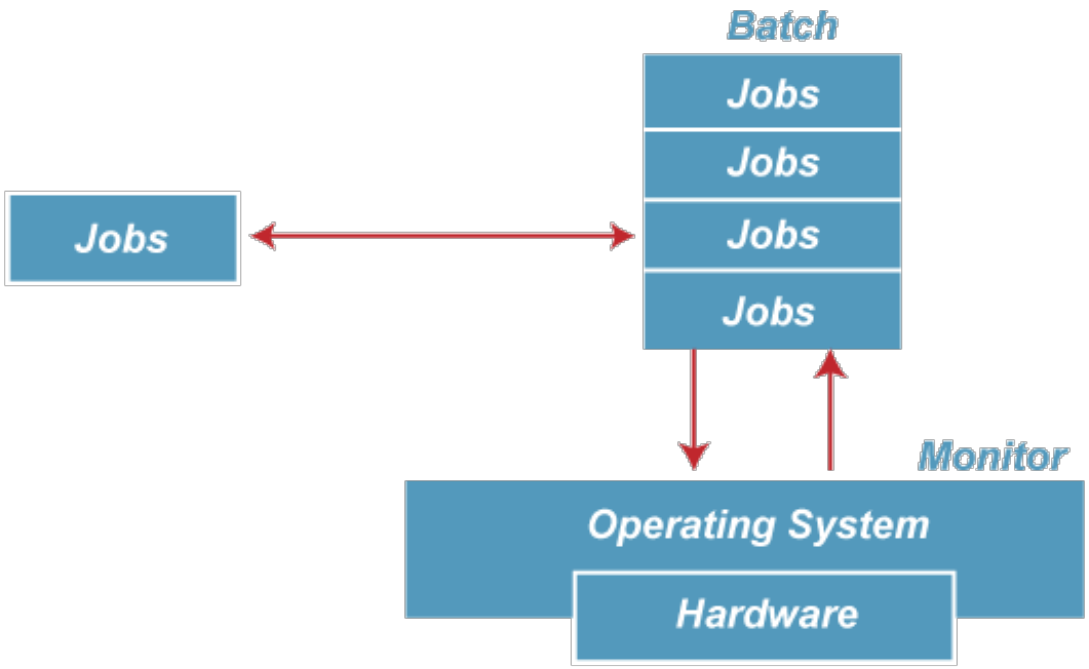


- Application programming
  - There is a trend in application programming, away from system-level programming and towards very high-level (application) development;
  - There is no programmer, however, who does not benefit from some understanding of system programming;

Operating Systems Jargon

Batch Processing

- The system put **all of the jobs in a queue on the basis of first come first serve and then executes the jobs one by one**.
  - Executes one task or activity at a time (single processing)
- The users collect their respective output when all the jobs get executed.

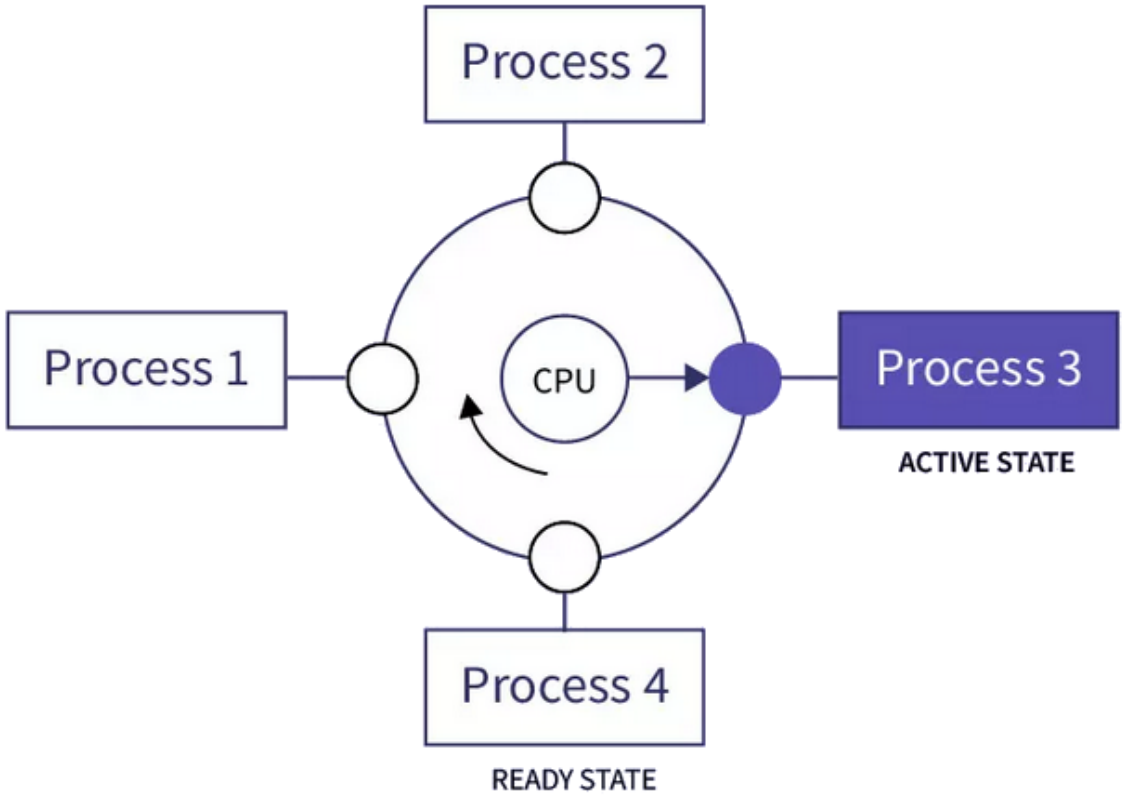


Multiprogramming

- Multiprogramming **is an extension to batch processing where the processor or Central Process Unit (CPU) is always kept busy**.
- Each program needs two types of system time: CPU time and Input/output (I/O) time.
- When a program is being performed, it is known as a **Task**/**Job** and **Process**.
- **Concurrent program executions improve system resource utilization** as compared to batch processing systems.

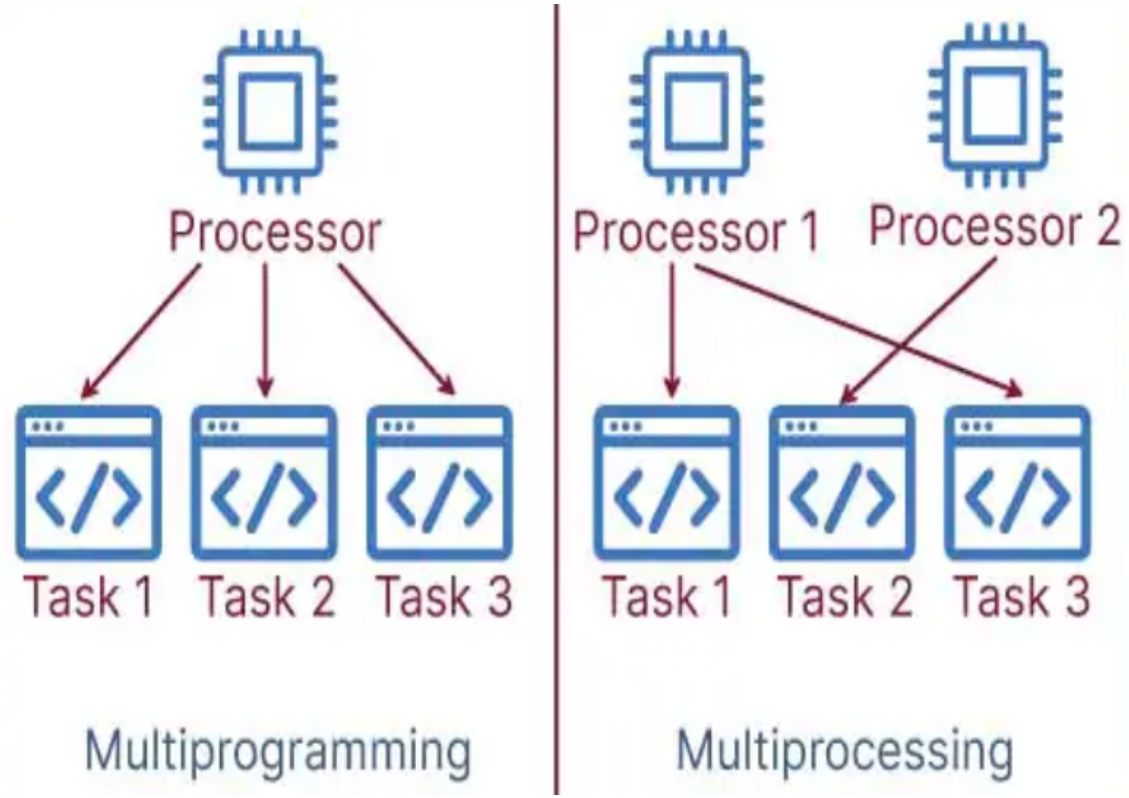
Time-Sharing

- It uses the concept of CPU scheduling and multiprogramming so that the time of the CPU is divided equally(typically) among the running tasks/processes (or users).



Multiprocessing

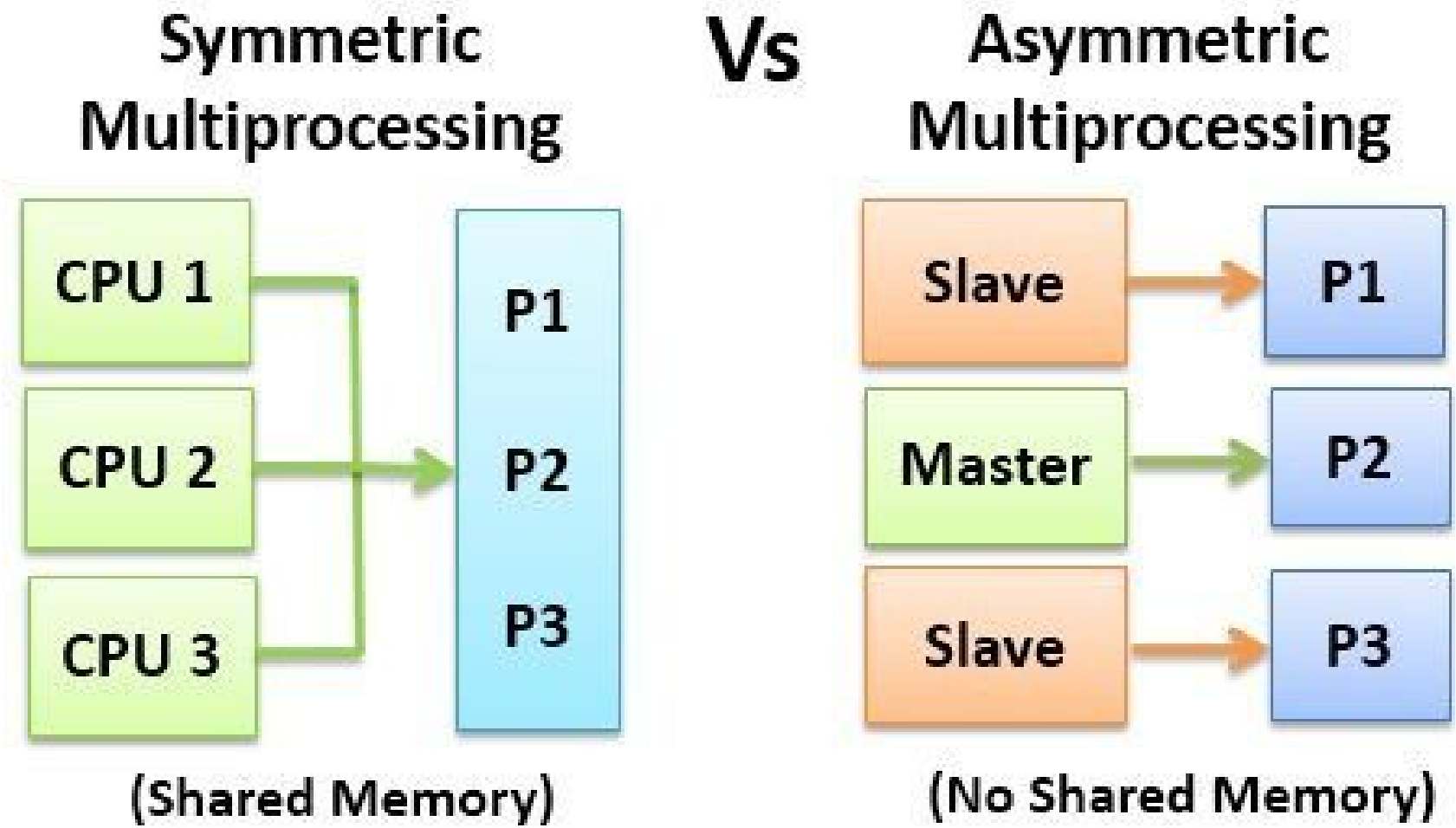
- In Multiprocessing, **Parallel computing is achieved**.
- There are more than one processors present in the system which can execute more than one process at the same time.
  - Note, multicore systems have multiple processors in a single package



Asymmetric vs Symmetric Multiprocessing (I)

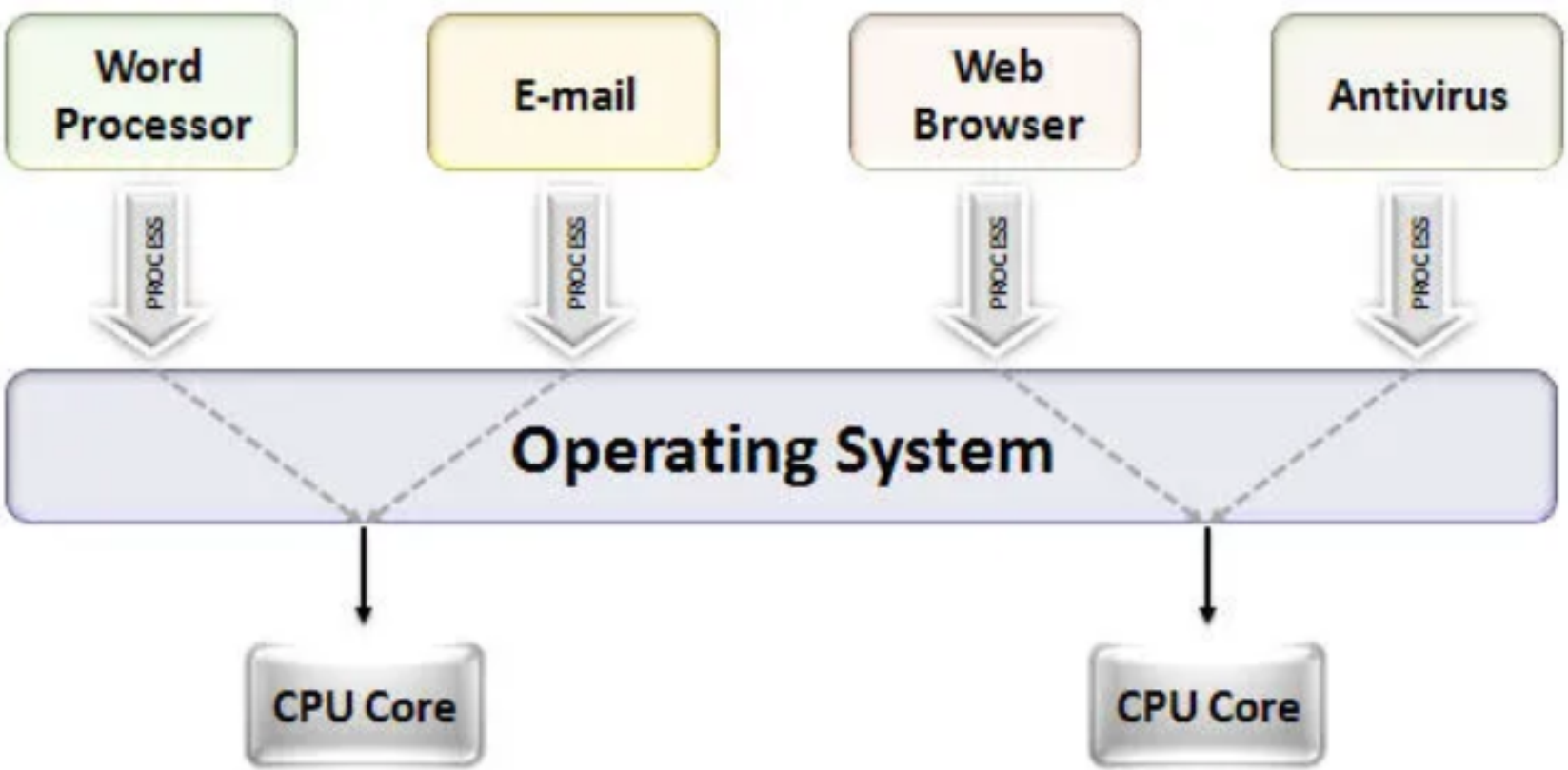
- **Asymmetric Multiprocessing**:
  - It is a multiprocessor computer system where not all of the multiple interconnected CPUs are treated equally.
  - In asymmetric multiprocessing, only **a master processor runs the tasks of the operating system**.
- **Symmetric Multiprocessing**:
  - It involves a multiprocessor computer hardware and software architecture where two or more identical processors are connected to a single, shared main memory, and have full access to all input and output devices.
  - In other words, Symmetric Multiprocessing is a type of multiprocessing where **each processor is self-scheduling**.

Asymmetric vs Symmetric Multiprocessing (II)



Multitasking

- The multitasking operating system is a logical extension of a multiprogramming system that enables multiple programs simultaneously.
- Multiprocessing also enables multitasking.



Types of Operating Systems

Based on Device (I)

- **Desktop Operating System**
  - Windows, Linux, and MacOS are the most popular.
- **Server Operating System**
  - Linux or UNIX is more preferred for servers
- **Mobile Operating System**
  - Android and iOS
- **Supercomputer Operating Systems**
  - Supercomputers are used in weather forecasting, space research and wherever a huge amount of data needs to be parallel processed in a short amount of time
  - Computer Node Kernel and Input Node Kernel based on modified Linux operating system

Based on Device (II)

- **Embedded, Industrial Robots and Controllers Operating Systems**
  - VxWorks, PSOS, VRTX, RTLinux, Lynx, QNX, eCos
- **Cloud based Operating System**
  - These are lightweight operating systems used for cloud based use cases.
  - Chrome OS(linux based and uses google chrome as interface), EasyPeasy (debian and unbuntu) linux based OS for netbooks), Joli OS (Ubuntu Linux based), EyeOS(gives desktop interface within web page), OSW3, DOKY(linux based)

Based on Capability

- **Single user vs multi-user OS**
- **Batch OS**
- **Multitasking/Time-sharing OS**
- **Mobile OS**
- **Distibuted OS**
  - Distibuted OS is an operating system model in which multiple machines can work and communicate together.
  - In this type of OS, OS is installed in multiple machines facilitating the interaction.
- **Network OS**
  - Network operating systems are the systems that run on a server and manage all the networking functions
  - This type of OS is used in file sharing, printer sharing, data, users, user groups, applications across local or enterprise network.
- **Real-Time OS**
  - This type of OS is intended for real-time applications.

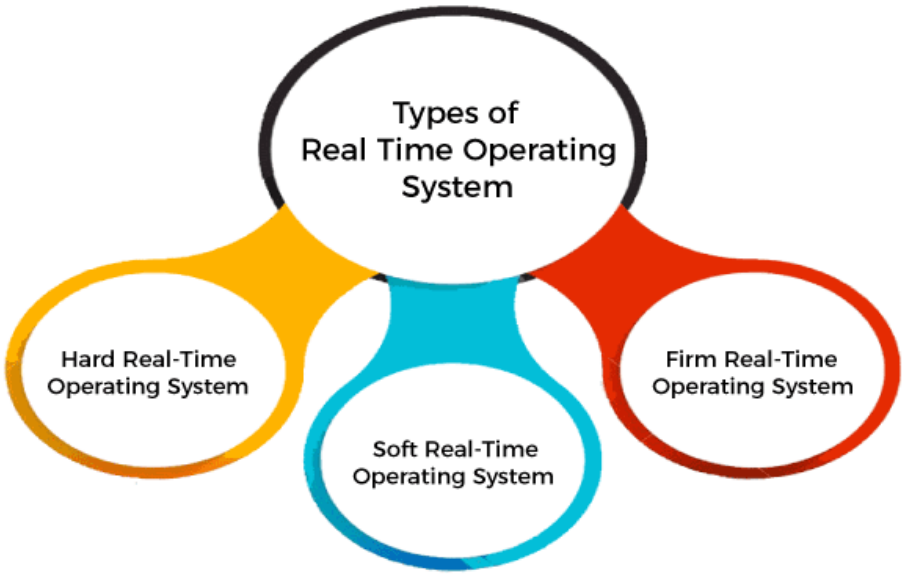
General *vs* Real Time Purpose

General-Purpose Operating System (GPOS)

- The GPOS is an operating system that can manage a high number of processes and complete an execution per unit time (defined as **throughput**).
- GPOS use a **fairness policy** to carry processes and threads to the CPU.
  - The task scheduler uses a fairness policy, allowing the overall high throughput but not ensuring that high-priority jobs will be executed first.
- GPOS is designed to perform non-time-critical general tasks.

Real-Time Operating System (RTOS)

- A RTOS is a type of operating system that is designed to meet strict time constraints, with a guaranteed response time for critical tasks.
- In RTOS, each job/task/process carries a certain **deadline within which the job is supposed to be completed**, otherwise, the huge loss will be there, or even if the result is produced, it will be completely useless.



Types of RTOS

- **Hard**
  - Hard real-time is when a system **will cease to function if a task deadline is missed**, which can result in catastrophic consequences.
  - These systems typically require deterministic and predictable behavior, with a guaranteed response time for critical tasks.
- **Soft**
  - Soft real time system is a system that **will not cease to function if a task deadline is missed**.
    - Result obtained **after deadline is not incorrect**.
    - Its operation is degraded if results are not produced according to the specified timing requirement.
- **Firm**
  - Firm real-time is a **system that will not cease to function if a task deadline is missed**, but the result of such task is discarded.
    - The utility of result becomes zero after the deadline.
    - Result obtained **after deadline is considered incorrect**.

The Kernel

What is a Kernel in OS?

- Kernel is the **core of the OS**.
  - It provides basic services for all other parts of the OS.
    - It is the main layer between the OS and underlying computer hardware, and it helps with tasks such as process and memory management, file systems, device control and networking.
- During normal system startup, a computer's basic input/output system, or BIOS, completes a hardware bootstrap or initialization.
- It then runs a bootloader which loads the kernel from a storage device – such as a hard drive – into a protected memory space.
- Once the kernel is loaded into computer memory, the BIOS transfers control to the kernel.
- Finally, it then loads other OS components to complete the system startup and make control available to users through a desktop or other user interface.

Kernel Architecture: Monolithic

- In Monolithic kernel mode, OS runs in a **single address space**.
- Monolithic kernel **has all the OS functions or services** within a single kernel.
- Monolithic kernel **runs as a single process in a single address space in memory**.
- Monolithic architecture has two modes: **Kernel mode** and **User mode**.
  - Kernel mode has all the primary or core operation system features
  - User mode has the operating system features not added in the kernel mode
- Monolithic architecture enables higher performance however less flexible for modifications to add new features or enhance existing features.
  - Linux, Windows, MacOS

Kernel Architecture: Microkernel

- Microkernel architecture is an architecture with**kernel having the basic interaction with hardware** and the basic Inter-Process Communication (IPC) mechanisms.
- In the microkernel architecture, Kernel **has the least amount of operating system core services**.
  - IPC, scheduling and memory management are the core services in a micro-kernel.
  - All the other OS services exist outside the Kernel.
    - It is broken down into several processes, known as **servers**;
    - All servers are kept separate and run in different address spaces;
- Microkernel provides the flexibilities to add new features or modify existing features while slightly affecting performance as it increases amount of interactions between kernel and user mode features.
  - QNX

Monolithic *vs* Microkernel