Lecture 1: Introduction to Operating Systems & System Architecture

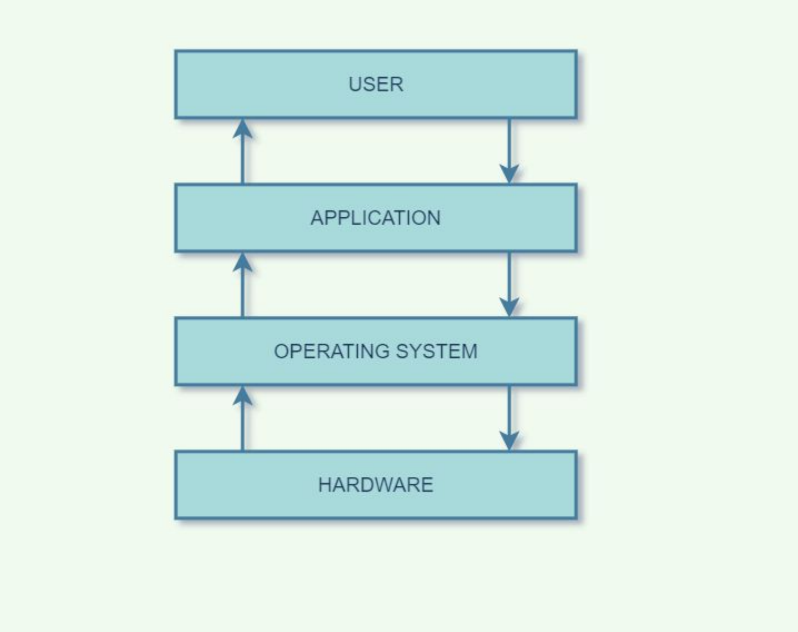
Core Topics:

- What is an Operating System?
- Functions of Operating System
- Types of OS: Batch, Time-sharing, Distributed, Real-time, etc
- Kernel, Shell, System Calls
- System Architecture: Monolithic, Microkernel, Layered

What is an Operating System?

- An Operating System (OS) is system software that manages computer hardware and software resources
- Acts as an intermediary between user and hardware.
- Examples: Windows, macOS, Linux, Android

Linux



Functions of Operating System?

- Process Management
- Memory Management
- File System Management
- Device Management
- Security & Access Control
- User Interface

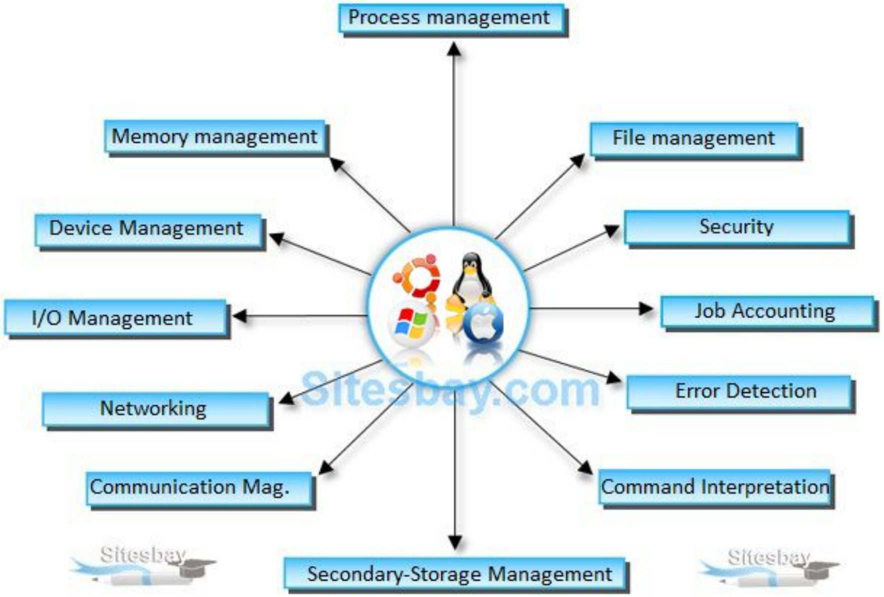
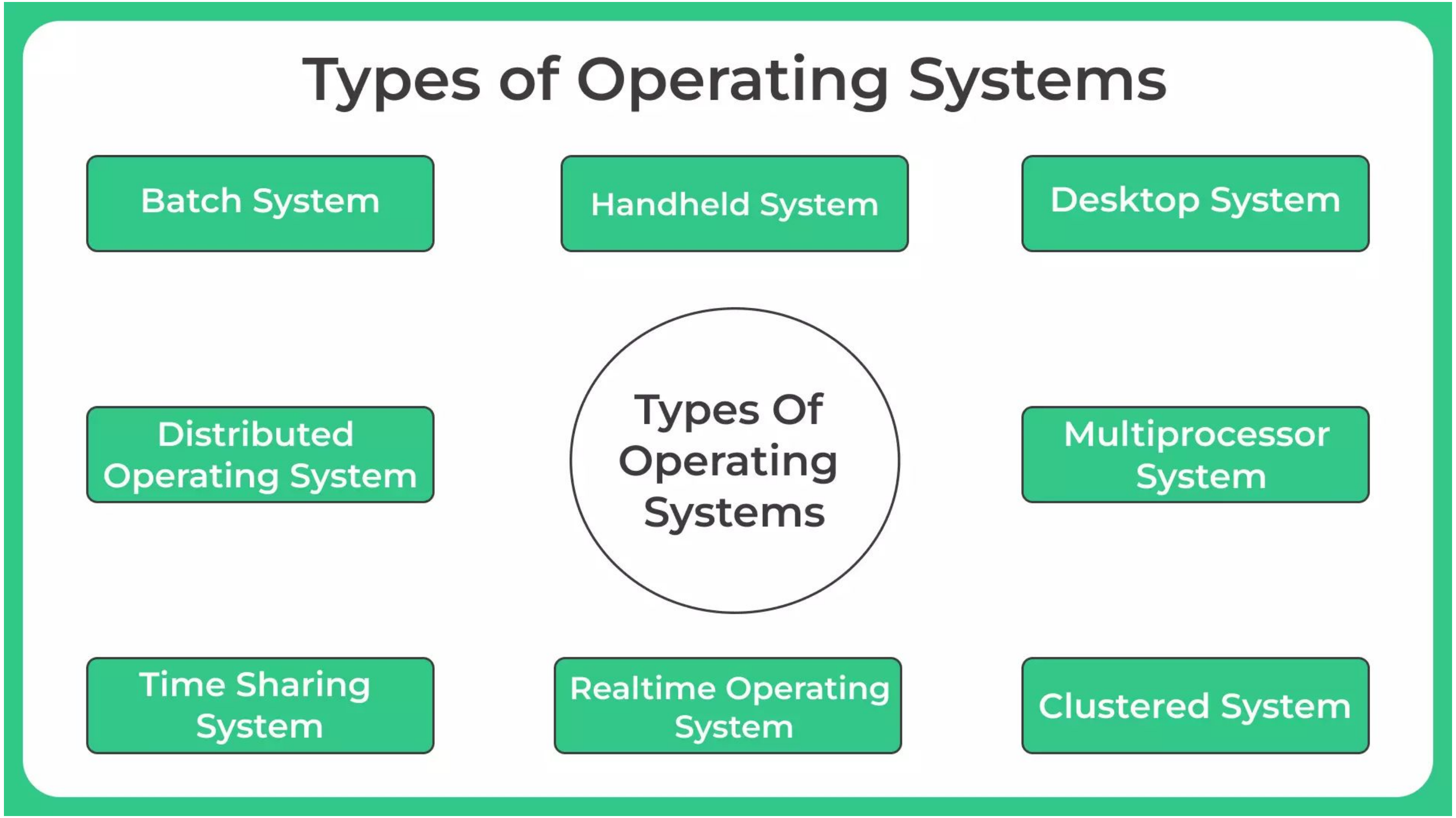


Fig: Fuctions of Operating System

Types of Operating Systems



Types of Operating Systems

Types Of Operating Systems

- Batch System
- Handheld System
- Desktop System
- Distributed Operating System
- Multiprocessor System
- Time Sharing System
- Realtime Operating System
- Clustered System

Types of Operating Systems

1. Batch Operating System
   - No direct interaction with the user
   - Jobs processed in batches
   - Example: IBM OS/360

2. Time-Sharing Operating System
   - Allows multiple users to share system resources simultaneously
   - Quick context switching
   - Example: UNIX

3. Distributed Operating System
   - Manages a group of independent computers as a single system
   - Promotes resource sharing
   - Example: LOCUS, Amoeba

Types of Operating Systems

4. Network Operating System (NOS)

- Provides services over a network
- Example: Novell NetWare, Windows Server
-

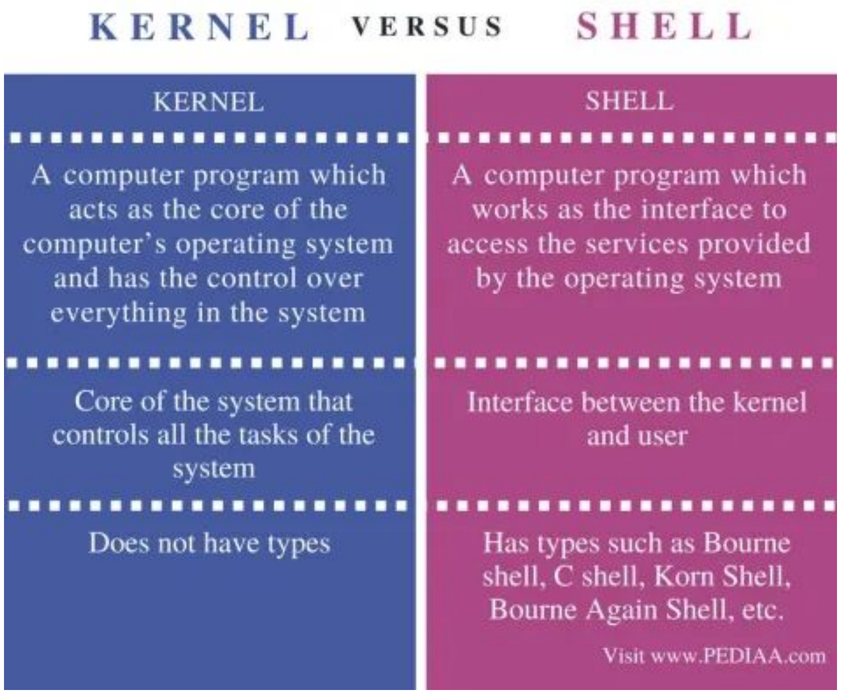
5. Real-Time Operating System (RTOS)

- Responds to input within strict time constraints
- Used in embedded systems, robots
- Example: VxWorks, FreeRTOS
-

6. Mobile Operating System

- Designed for mobile devices
- Example: Android, iOS

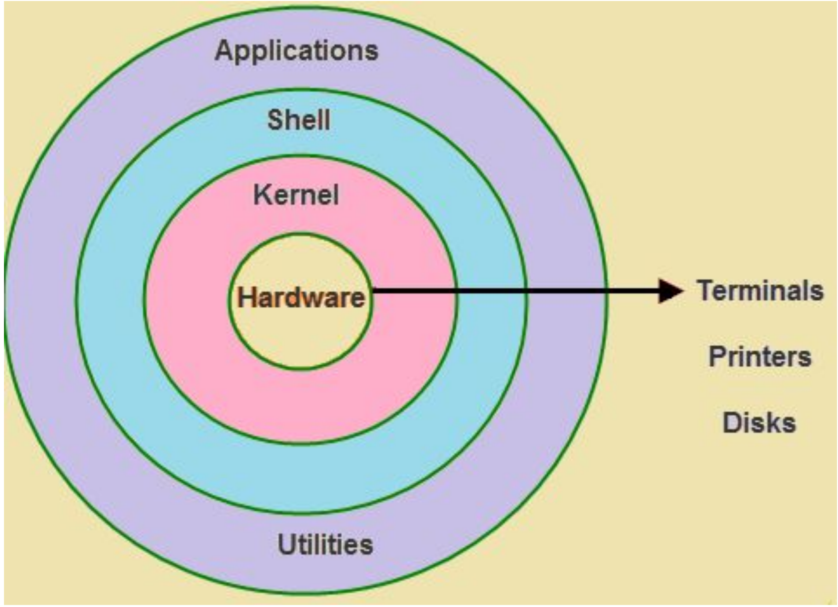
Kernel vs Shell



KERNEL VERSUS SHELL

KERNEL	SHELL
A computer program which acts as the core of the computer's operating system and has the control over everything in the system	A computer program which works as the interface to access the services provided by the operating system
Core of the system that controls all the tasks of the system	Interface between the kernel and user
Does not have types	Has types such as Bourne shell, C shell, Korn Shell, Bourne Again Shell, etc.

Visit www.PEDIAA.com



What is the shell advantages?

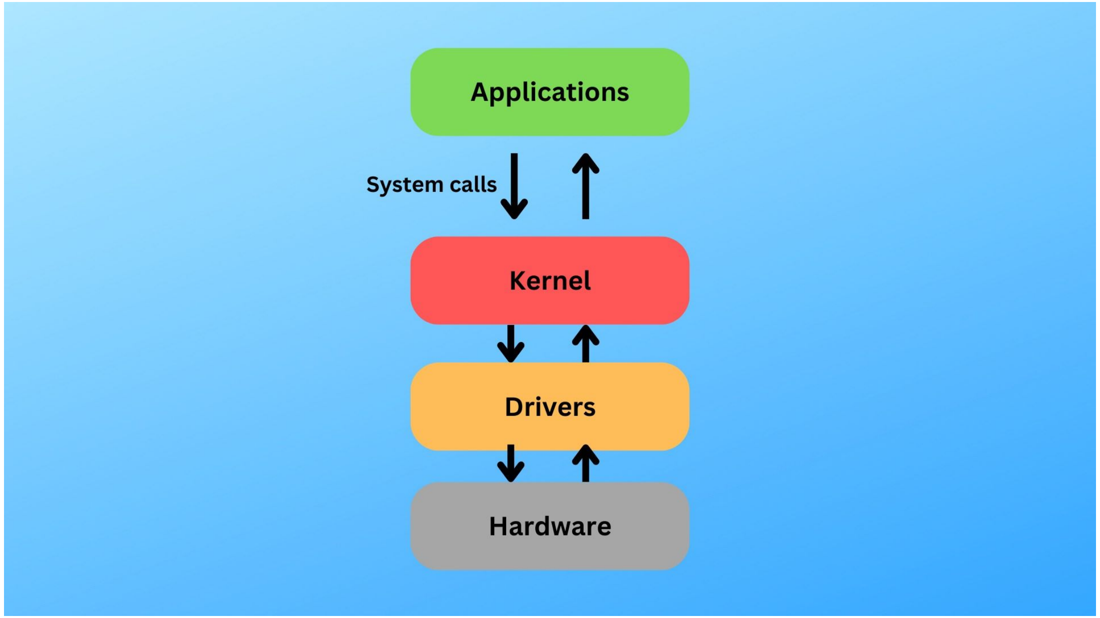
- It allows you to interact with the computer by typing commands and executing them.
- It acts as a command-line interpreter, taking your input, interpreting it, and executing the corresponding actions.
- It allows you to automate repetitive tasks by creating scripts or shell programs.
- It provides direct access to the system's utilities and functions, enabling efficient management and control over your computer.
- It offers flexibility, as you can customize and extend its functionality according to your needs.

What are System Calls?

Interface between user applications and the OS.

Provides services like:

- Process Control
- File Management
- Device Management
- Information Maintenance
- Communication



Shell vs System Calls

Feature	**Shell**	**System Calls**
Definition	User interface for the OS	Low-level interface to the kernel
Function	Interprets user commands	Requests services from the kernel
Level	Higher-level, user-facing	Lower-level, kernel-facing
Examples	Bash, PowerShell, Zsh	`open()`, `read()`, `write()`, `fork()`
Relationship	Interacts with kernel via system calls	Used by applications and shells

System Architecture Overview

- Defines how OS components are organized and interact.
- Three Common Architectures:

1.Monolithic Architecture

2. Layered Architecture:

3. Microkernel Architecture:

1. Monolithic Architecture

1. **Description:** All OS components (like the kernel, device drivers, file system, etc.) are integrated into a single, large program that runs in kernel space.

2. **Advantages:** Simple to implement, fast due to direct communication between components.

3. **Disadvantages:** Difficult to debug and update, a bug in one component can potentially crash the entire system.

4. **Examples**: Early versions of Unix, MS-DOS.

2. Layered Architecture:

1. **Description:** The OS is structured into multiple layers, with each layer built upon the lower layers.

2. **Advantages:** Modularity, easier debugging and maintenance, improved security due to isolation of layers.

3. **Disadvantages:** Can be slower than monolithic due to inter-layer communication overhead.

4. **Examples:** Some implementations of Unix, many modern operating systems.

3. Microkernel Architecture:

1. **Description:**The kernel provides only minimal functionalities (like memory management, inter-process communication), while other services (like device drivers, file systems) run as user-level processes.

2. **Advantages:**High modularity, improved security and reliability, easier to extend and maintain.

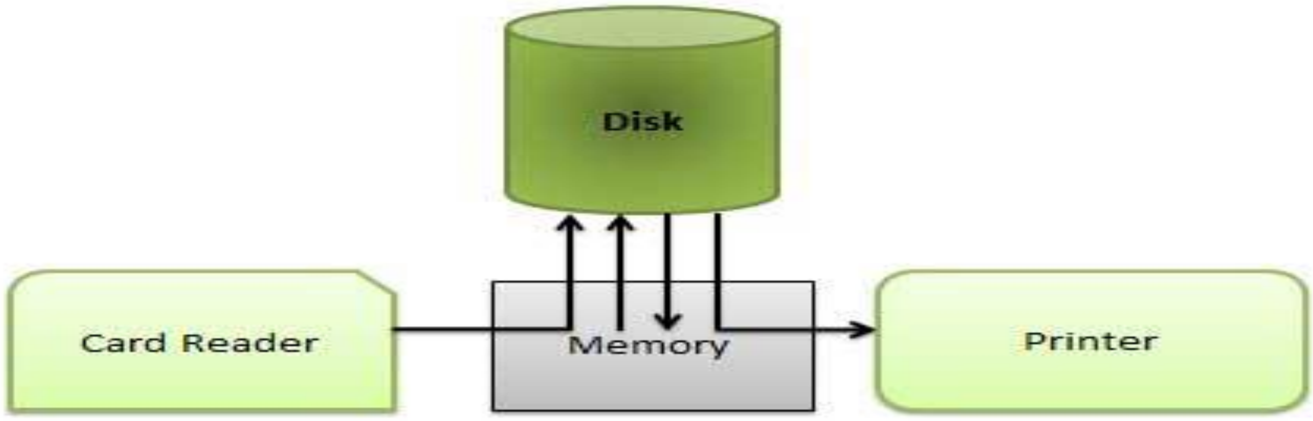
3. **Disadvantages:**Can have performance overhead due to increased inter-process communication.

4. **Examples:**Mach, MINIX.

Bootstrapping

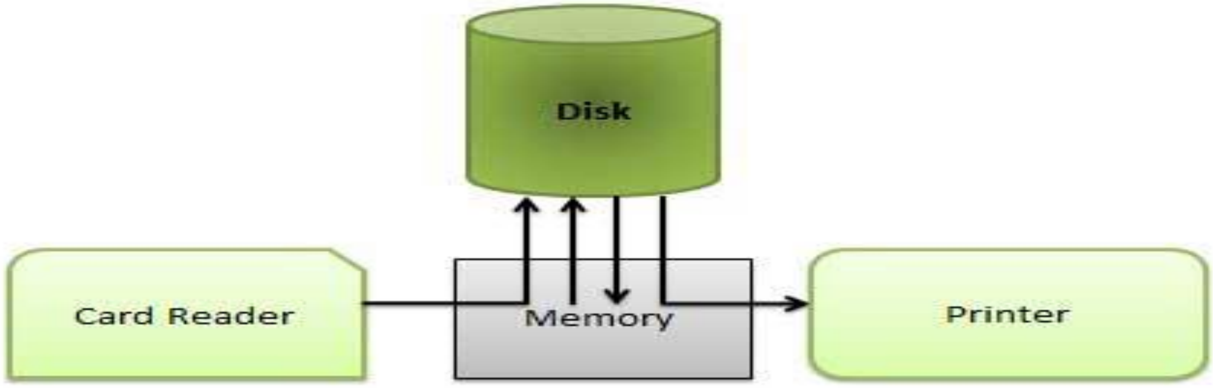
- Booting is the process of loading operating system into main memory.
- For a computer to start running, it needs to have an initial program to load and execute the boot program, which in turn loads the operating system.
- The primitive loader program that can load and execute the Boot program is called Bootstrap Program.
- Boot strap program is generally stored in ROM.
- On- Start up, the computer automatically reads the **bootstrap program**.

Spooling (Simultaneous Peripheral Operation On-Line)



- Random-Access Device like disk system was introduction.
- Location of card images is recorded in a table kept by the operating system.
- When the job is completed, the output is actually printed.
- This form of processing is called Spooling.

Spooling (Simultaneous Peripheral Operation On-Line)

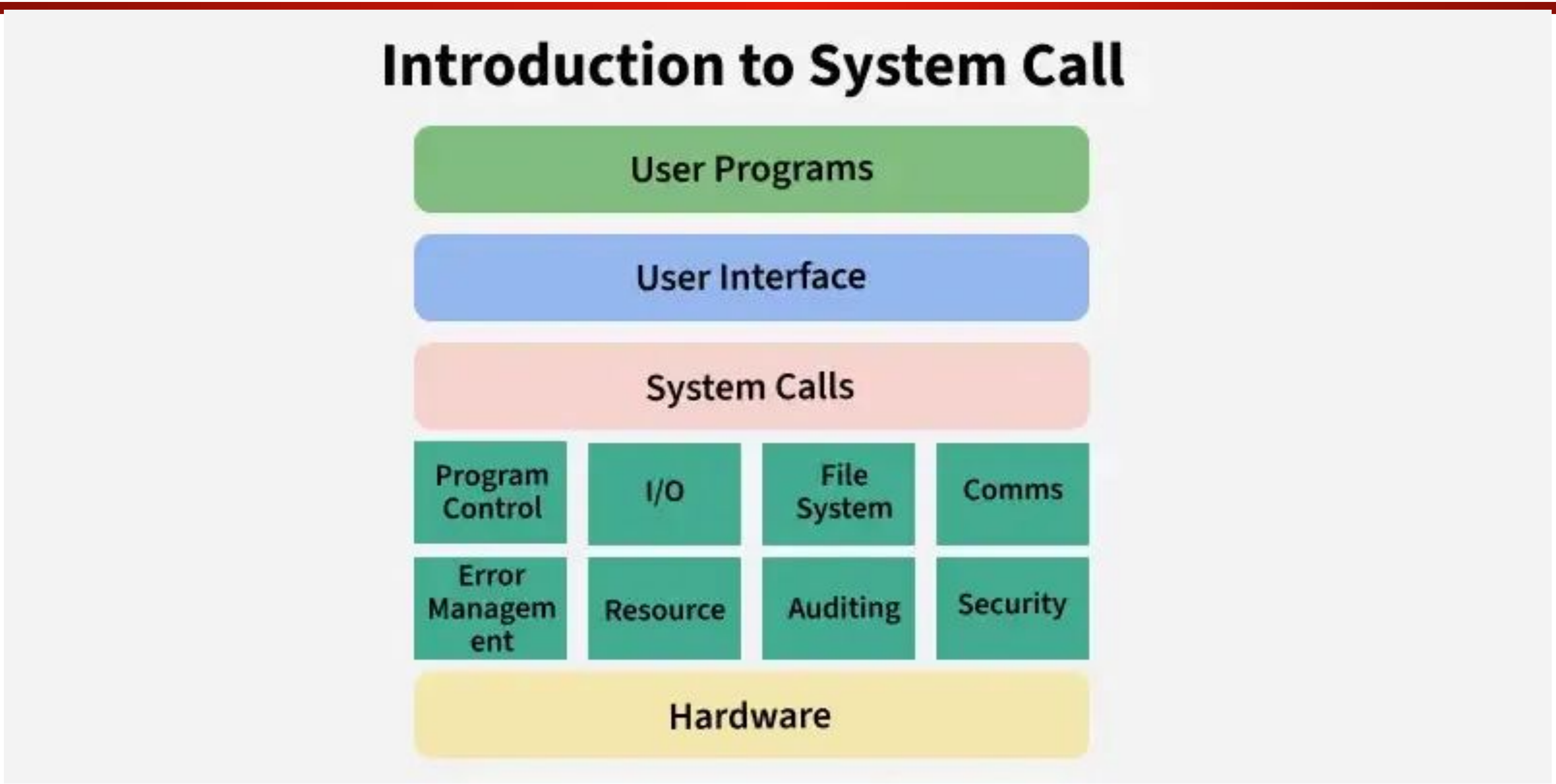


- Spooling overlaps the I/O of one job with the computation of other jobs.
- Spooling has a direct beneficial effect on the performance of the system
- Spooling can keep the CPU and the I/O devices working at same time at much higher rates.

System Call

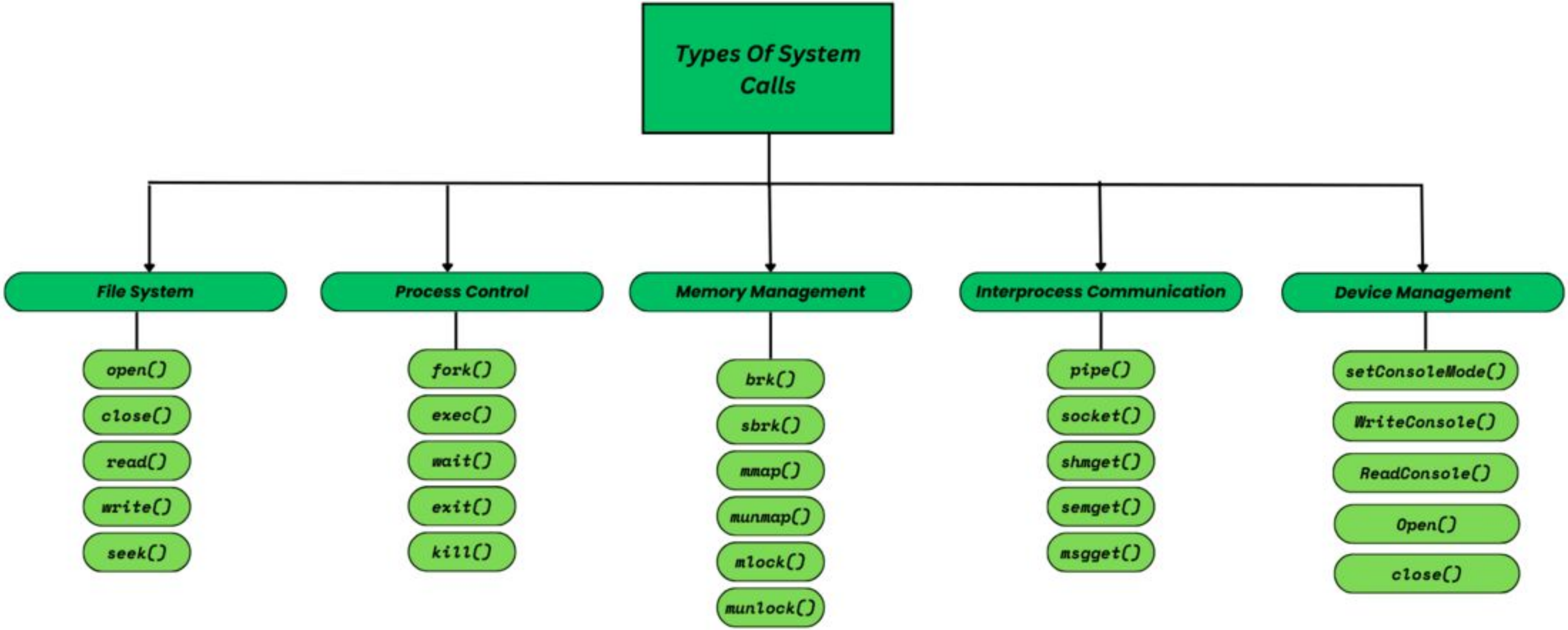
- A system call is a programmatic way in which a computer program requests a service from the kernel of the operating system on which it is executed.
- A computer program makes a system call when it requests the operating system's kernel.
- System call provides the services of the operating system to the user programs via the Application Program Interface(API).
- An API, or Application Programming Interface, is a set of rules and specifications that allows different software systems to communicate and interact with each other

System Call



Introduction to System Call

Types of system call


Types Of System Calls
File System
open()
close()
read()
write()
seek()
Process Control
fork()
exec()
wait()
exit()
kill()
Memory Management
brk()
sbrk()
mmap()
munmap()
mlock()
munlock()
Interprocess Communication
pipe()
socket()
shmget()
semget()
msgget()
Device Management
setConsoleMode()
WriteConsole()
ReadConsole()
Open()
close()