Introduction to Operating Systems

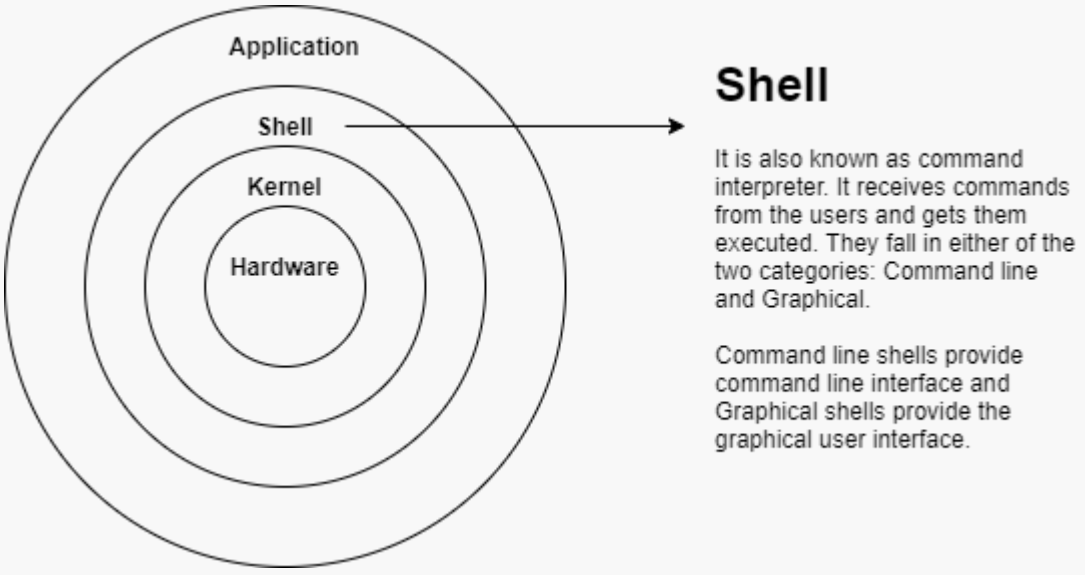
In this lecture, we have learnt about:

Detailed Definition of Operating Systems

```
>>> OS is a piece of code that -
        Has privileged access to underlying hardware
        Hides complexity of the hardware
        Manages the resources
        Makes sure that the apps are isolated and protected from one another
```

Components of OS

```
>>> The components of OS can be best summarised using image:
```



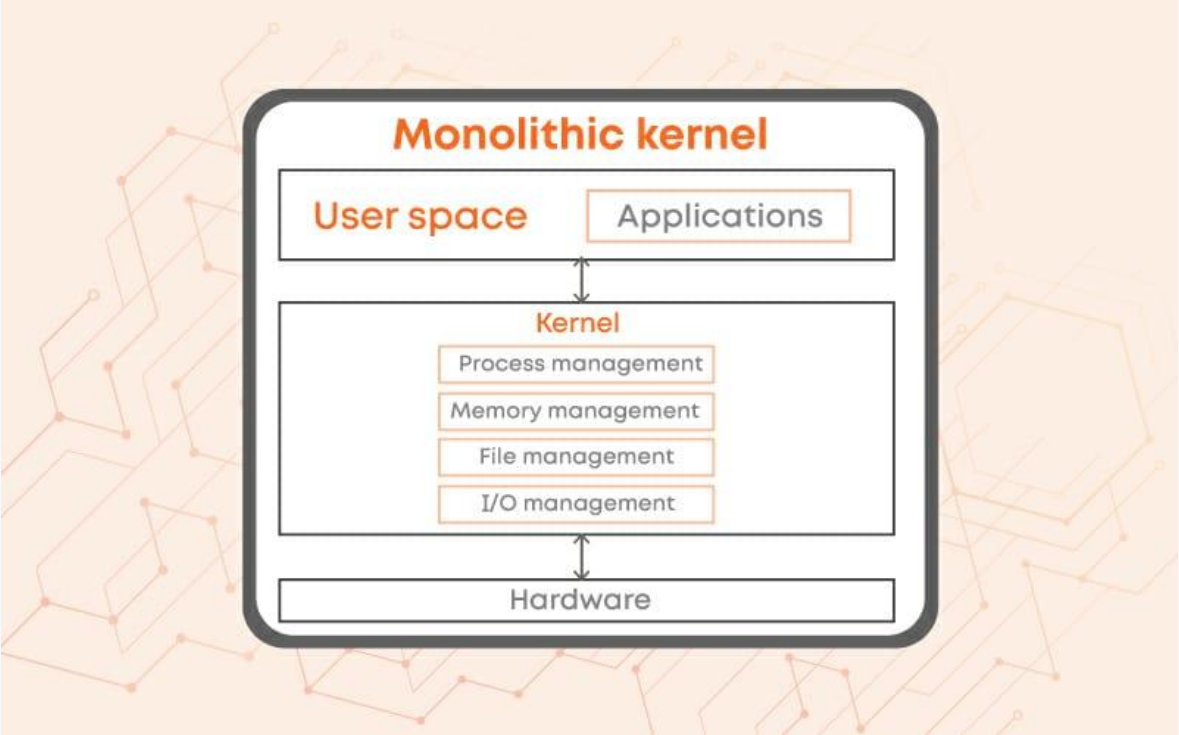
Kernel

A Kernel is the central component of an operating system that manages operations of computer and hardware. It basically manages operations of memory and CPU time. It is the core component of an operating system. Kernel acts as a bridge between applications and

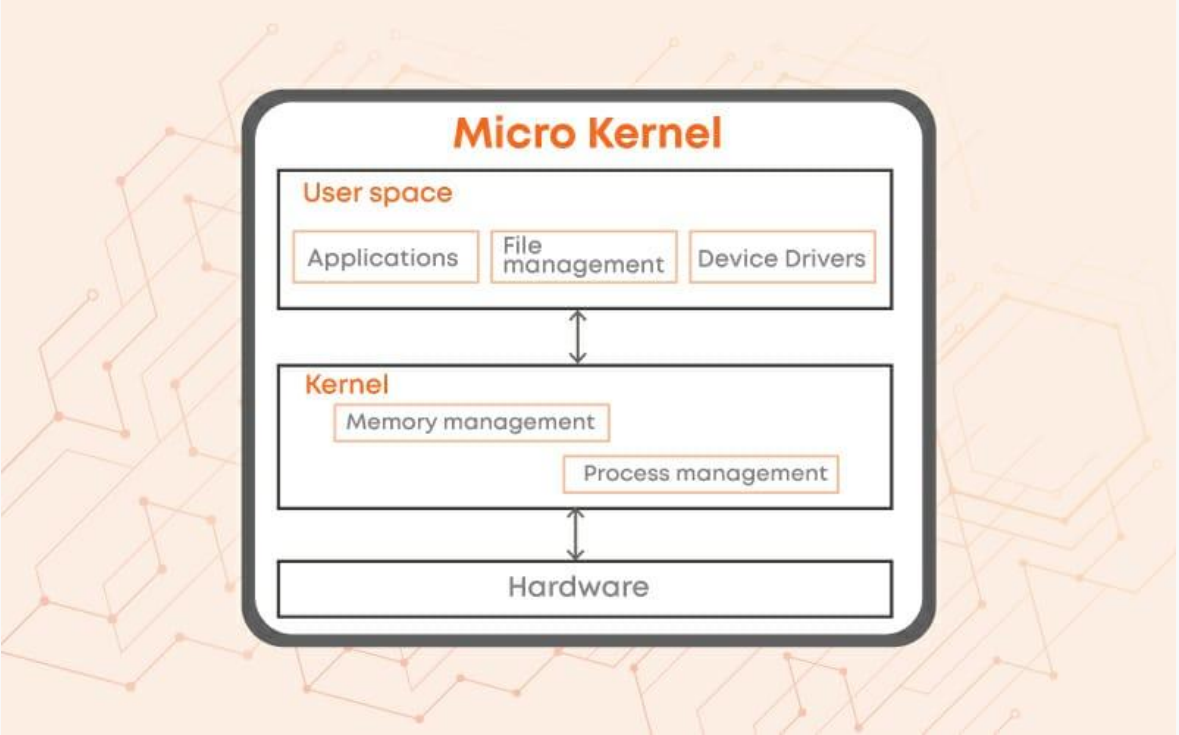
data processing performed at hardware level using inter-process communication and system calls.

Types of Kernel

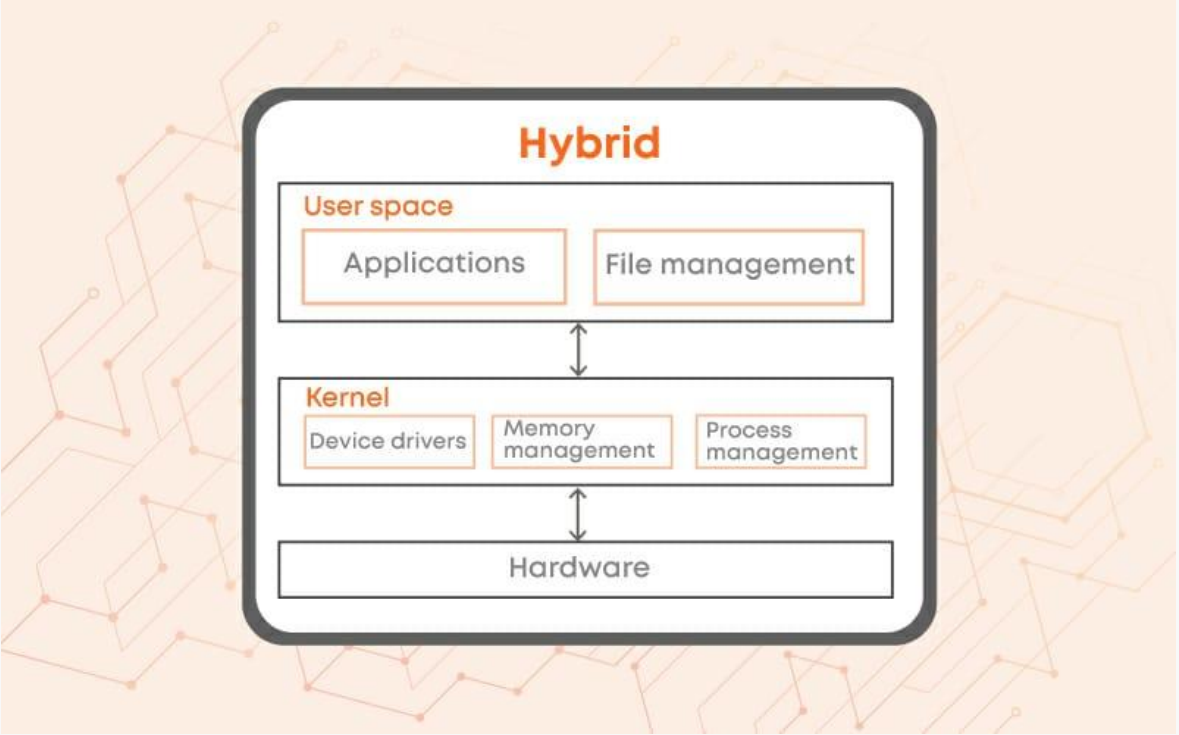
```
>>> Monolithic Kernel
Example: Unix, Linux
```




```
>>> MicroKernel
Examples: L4Linux, Minix
```



>>> Hybrid Kernel
Examples: Windows 7, Windows 10, Mac OS



Shell

A **shell**, also known as a command interpreter, is that part of the operating system that receives commands from the users and gets them executed.

System calls

A system call is a mechanism using which a user program can request a service from the kernel for which it does not have the permission to perform.User programs typically do not have permission to perform operations like accessing I/O devices and
communicating with other programs.
A user program invokes system calls when it requires such services.
System calls provide an interface between a program and the operating system.
System calls are of different types.

Example – fork, exec, getpid, getppid, wait, exit.

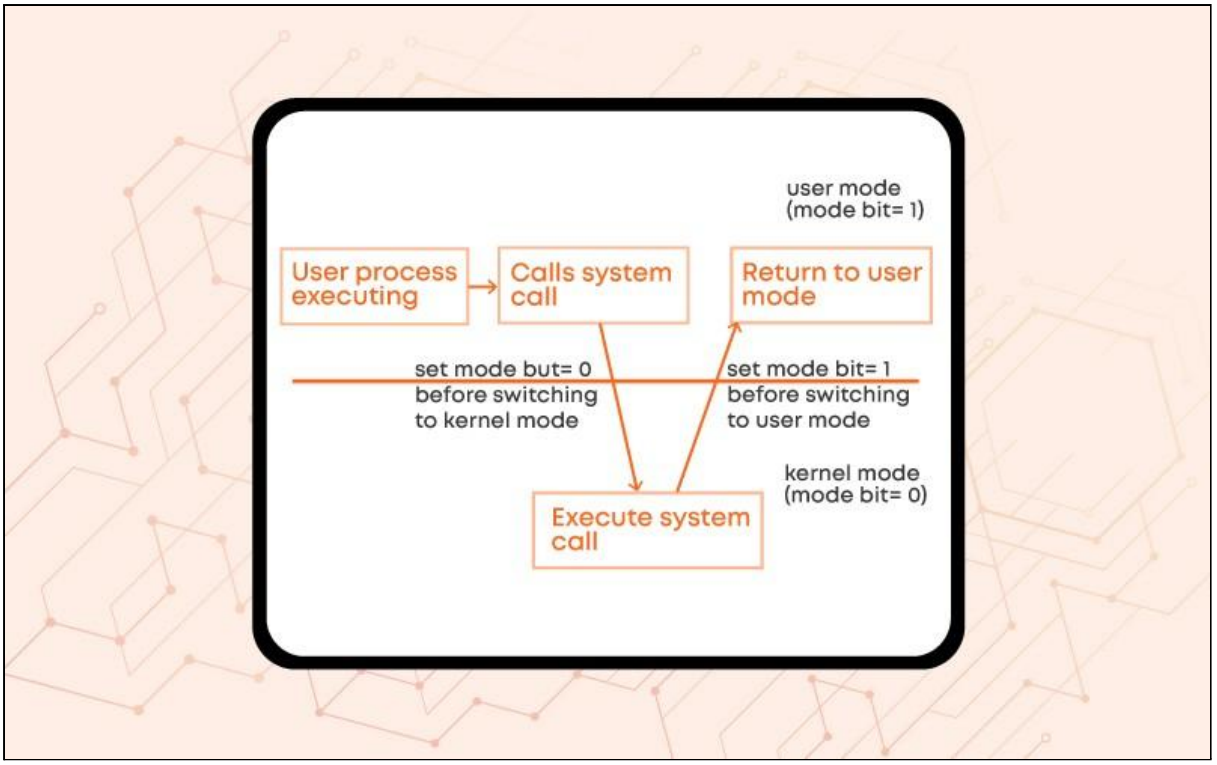
Operation mode

1. User Mode

When the computer system runs user applications like creating a text document or using any application program, then the system is in the user mode. When the user application requests for a service from the operating system or an interrupt occurs or system call, then there will be a transition from user to kernel mode to fulfill the requests.
To switch from kernel mode to user mode, mode bit should be 1.

2. Kernel Mode

When the system boots, hardware starts in kernel mode and when the operating system is loaded, it starts the user application in user mode. To provide protection to the hardware, we have privileged instructions which execute only in kernel mode. If a user attempts to run privileged instruction in user mode then it will treat the instruction as illegal and trap the OS.
To switch from user mode to kernel mode mode bit should be 0.



Note: In the above diagram, the mode bit is spelled wrong. The line should be: “set mode bit=0 before switching to kernel mode”