UNIX File System (UFS)

What is UNIX File System?

UNIX File System (UFS) is the **hierarchical file system** used by UNIX OS to manage and store files efficiently.

Key Features:

- Organizes files in a **tree-like structure**
- Provides **file attributes, permissions, and access control**
- Supports **long filenames, directories, and links**

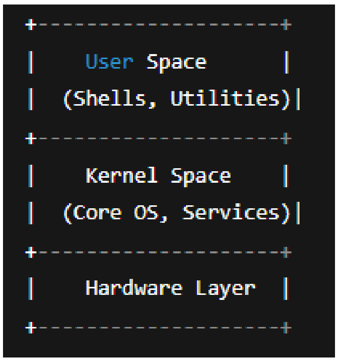
UNIX Architecture

UNIX is a multi-user, multitasking operating system. Its architecture is layered, separating hardware, kernel, and user components for efficiency and modularity.

---

1. Components of UNIX Architecture

UNIX architecture can be divided into three main layers:



```
+-------------------+
|    User Space     |
|  (Shells, Utilities)|
+-------------------+
|    Kernel Space   |
|  (Core OS, Services)|
+-------------------+
|    Hardware Layer  |
+-------------------+
```

---

1. Hardware Layer

- **Physical devices: CPU, memory, I/O devices**
- **Provides basic computing resources**
- **UNIX uses hardware via device drivers in the kernel**

---

2. Kernel

The kernel is the core of UNIX. It provides essential services like process management, memory management, file system management, and device management.

Components of Kernel:

1. **Process Management**
   - **Creates, schedules, and terminates processes**
   - **Handles inter-process communication (IPC)**
2. **Memory Management**
   - **Manages RAM and virtual memory**
   - **Provides paging and swapping**
3. **File System Management**
   - **Manages files, directories, and storage devices**
   - **Implements Unix File System (UFS)**
4. **Device Drivers**
   - **Interface for hardware devices**
   - **Handles I/O operations**
5. **System Calls**
   - **Interface for user programs to request OS services**
   - **Examples: read(), write(), fork(), exec()**
6. **Networking**
   - **Provides communication over TCP/IP**
   - **Supports sockets, network protocols**

---

3. User Space (Shell & Utilities)

- **Interface between user and kernel**
- **Contains shells and system utilities**

a. Shell

- **Command interpreter**
- **Types: Bourne Shell (sh), C Shell (csh), Korn Shell (ksh), Bash**

- Provides scripting and program execution

b. Utilities

- File management: ls, cp, mv, rm
- Text processing: grep, awk, sed
- System monitoring: ps, top, df, du

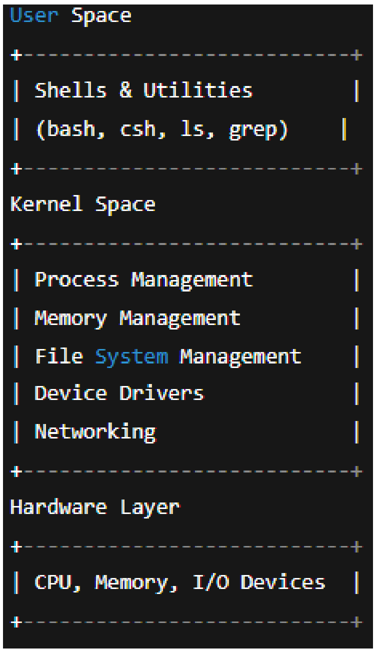
---

3. System Call Interface (SCI)

- Acts as a bridge between user programs and kernel
- Examples of system calls:
  - File: open(), read(), write(), close()
  - Process: fork(), exec(), wait()
  - Communication: pipe(), socket()

---

4. Layered UNIX Architecture Diagram



```
User Space
+----------------------------+
| Shells & Utilities         |
| (bash, csh, ls, grep)     |
+----------------------------+
Kernel Space
+----------------------------+
| Process Management         |
| Memory Management          |
| File System Management     |
| Device Drivers             |
| Networking                 |
+----------------------------+
Hardware Layer
+----------------------------+
| CPU, Memory, I/O Devices   |
+----------------------------+
```

Advantages of UNIX File System

- Efficient storage management
- Supports hierarchical directories

- File access control via permissions
- Allows sharing through links

Disadvantages

- Inode table fixed in size (limited number of files)
- Complex for very large filesystems
- No built-in encryption

Mobile Operating System (Mobile OS)

What is a Mobile OS?

A **Mobile OS** is an operating system designed to **run on smartphones, tablets, and embedded mobile devices**.

Key Features:

- Touchscreen interface
- Power and memory efficient
- Supports mobile connectivity (3G, 4G, 5G, Wi-Fi, Bluetooth)
- Runs mobile apps

2.1 Components of Mobile OS

```
+-------------------------+
| User Interface & Apps   |
+-------------------------+
| Application Framework   |
+-------------------------+
| Libraries & Runtime     |
+-------------------------+
| Kernel & Hardware Drivers|
+-------------------------+
| Mobile Hardware         |
+-------------------------+
```

1. **Kernel**
   - Core of OS
   - Manages **processes, memory, and devices**
   - Often based on **Linux (Android)** or **Darwin (iOS)**
2. **Hardware Abstraction Layer (HAL)**
   - Abstracts hardware specifics

    - Provides uniform interface for higher layers

3. **Libraries & Runtime**
    - Android: Java/Kotlin runtime, native C/C++ libraries
    - iOS: Cocoa Touch framework, Objective-C/Swift runtime
4. **Application Framework**
    - Allows apps to **access hardware & system services**
    - Example: camera, GPS, sensors
5. **User Interface & Apps**
    - Provides **GUI**
    - Runs user-installed apps

2.2 Popular Mobile OS

Mobile OS	**Kernel**	**App Platform**
Android	Linux	Java/Kotlin
iOS	XNU	Swift/Objective-C
Windows Mobile	Windows CE	C#/Native
HarmonyOS	LiteOS	Java/Kotlin

2.3 Features of Mobile OS

- Multitasking support
- Power management (battery efficiency)
- Security & sandboxing
- Connectivity support
- Sensor & multimedia support