innovate achieve lead

Operating Systems

BITS Pilani
Pilani Campus

Department of Computer Science and Information Systems

Key Contents

- Application and System Software
- Operating System
- Services of Operating System
- Execution process of a program
- UNIX Operating System
- Features of UNIX
- Layered Architecture of UNIX System
- Block diagram of the System Kernel
- System Calls

What is a Software ?

Software

A program comprising instructions that facilitates the end users with its desired functionality.

Application Software

Performs information processing tasks for end users.

Examples

Word Processing, Spreadsheets, Databases, etc.

What is a Software ? (Continued……)

System Software

- Manages and Supports operation of computer.

Types

- System Utilities and Operating System

System Programs

- System Programs are the programs that give service to other programs and interact with the hardware.
- Examples
  - Operating System – Set of system programs
  - Compiler
  - Assembler
  - Linker
  - Loader

System Utilities

Used to perform basic maintenance tasks on computer.

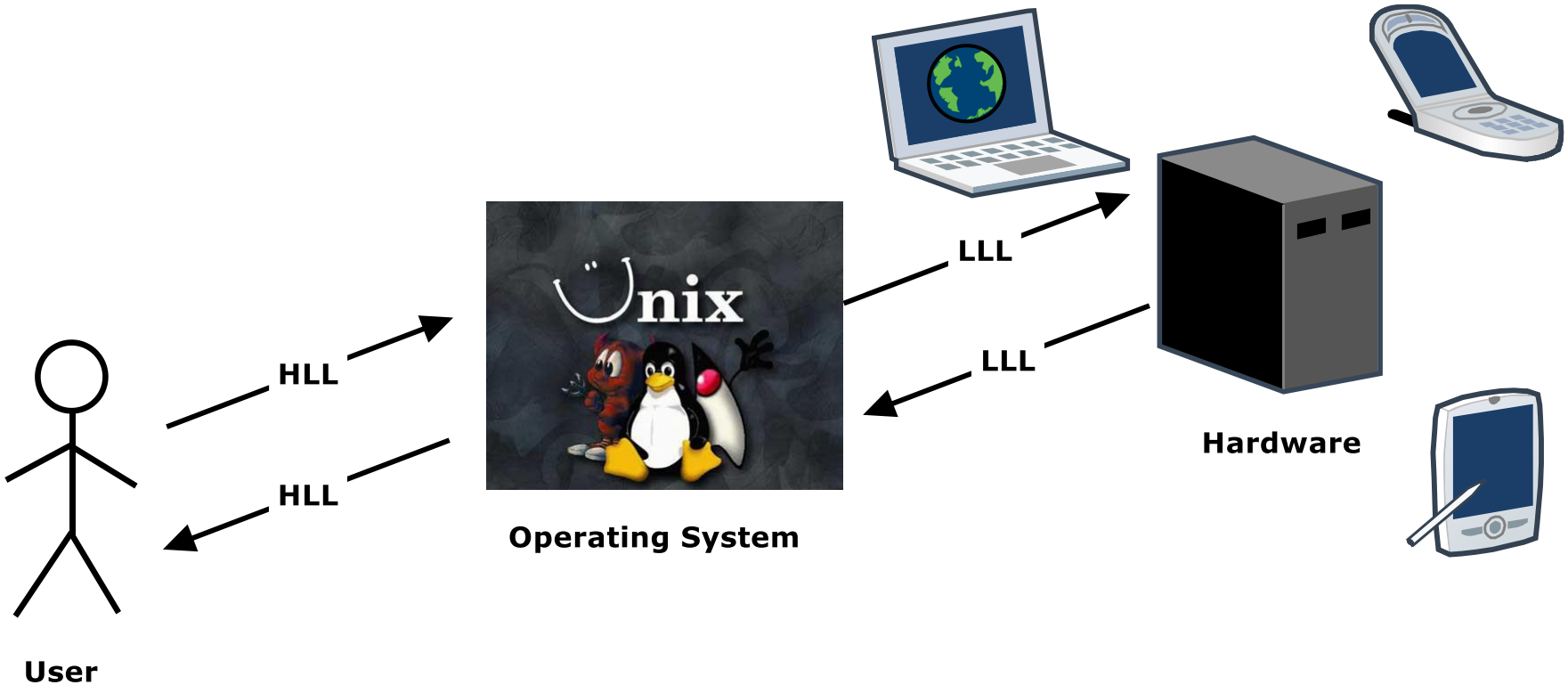
Examples: Disk Clean-up, System Restore, Disk defragmenters

What is an Operating System ?

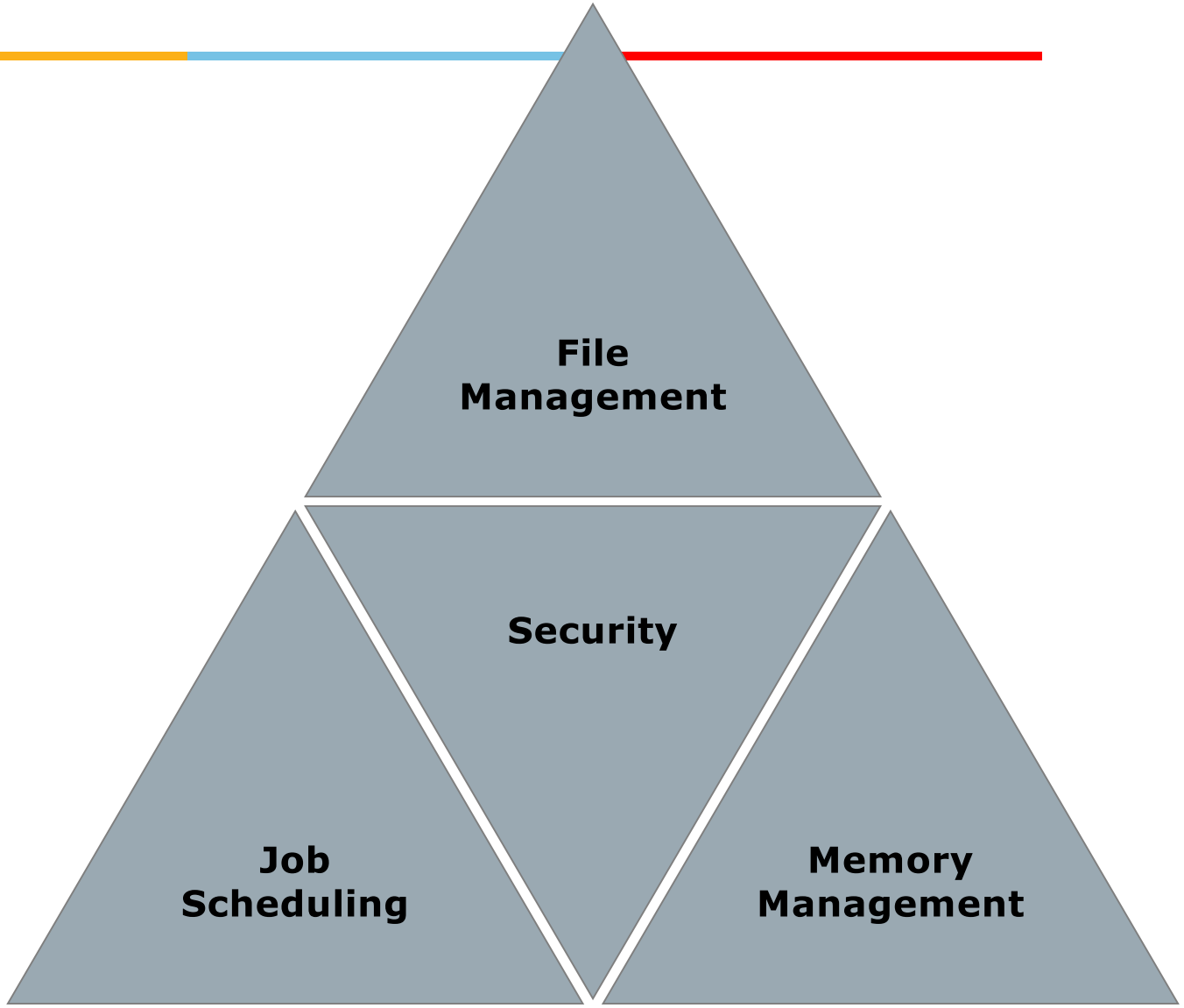
A software that controls the hardware resources of the computer and provides an environment under which programs can run.

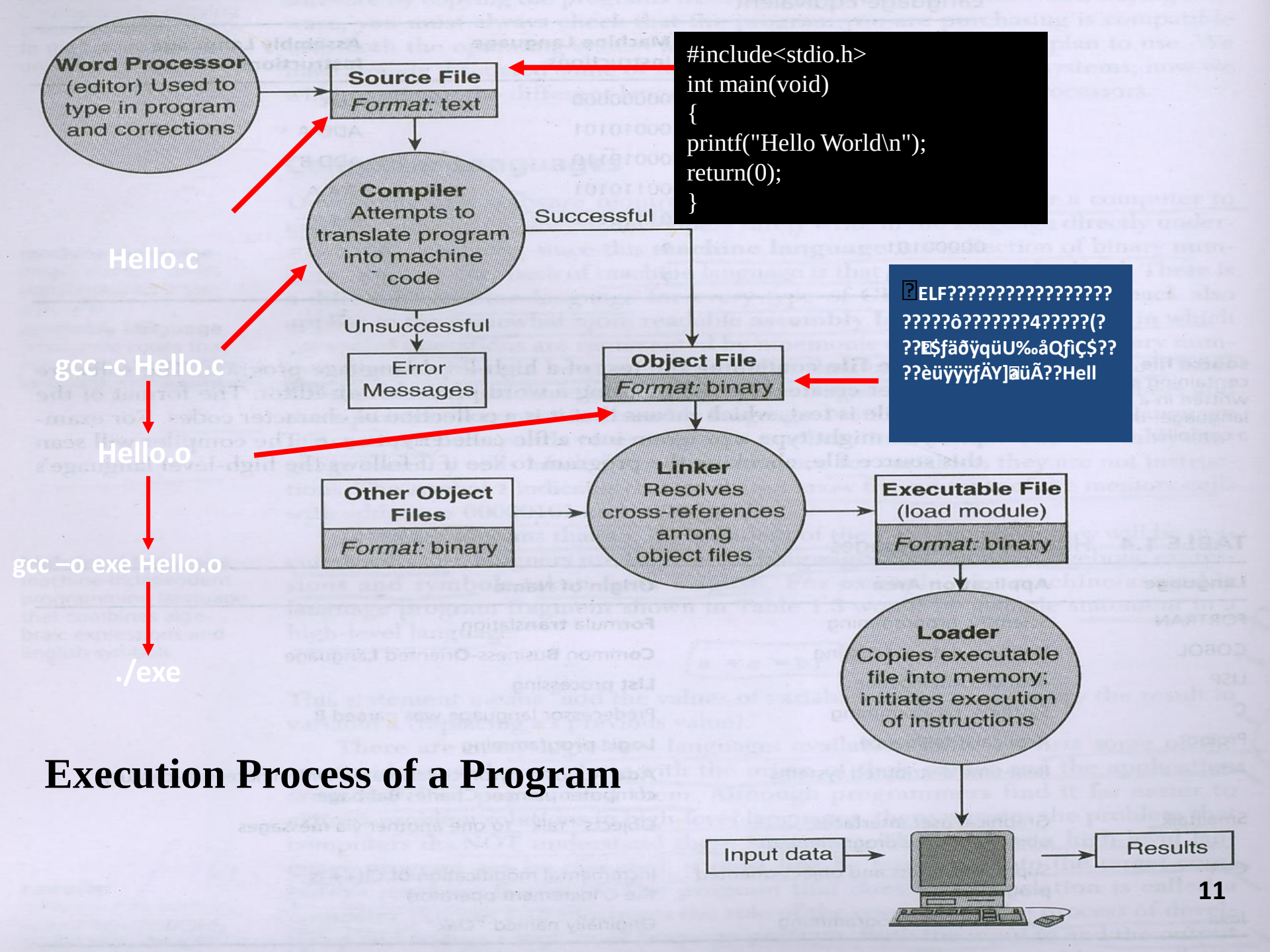
It acts as an interface between user of a system and the computer hardware

What is an Operating System ? (Continued…….)



Functions of Operating Systems





Execution Process of a Program

What is UNIX?

History of Unix

It is a command user interface OS

Executes on any computer

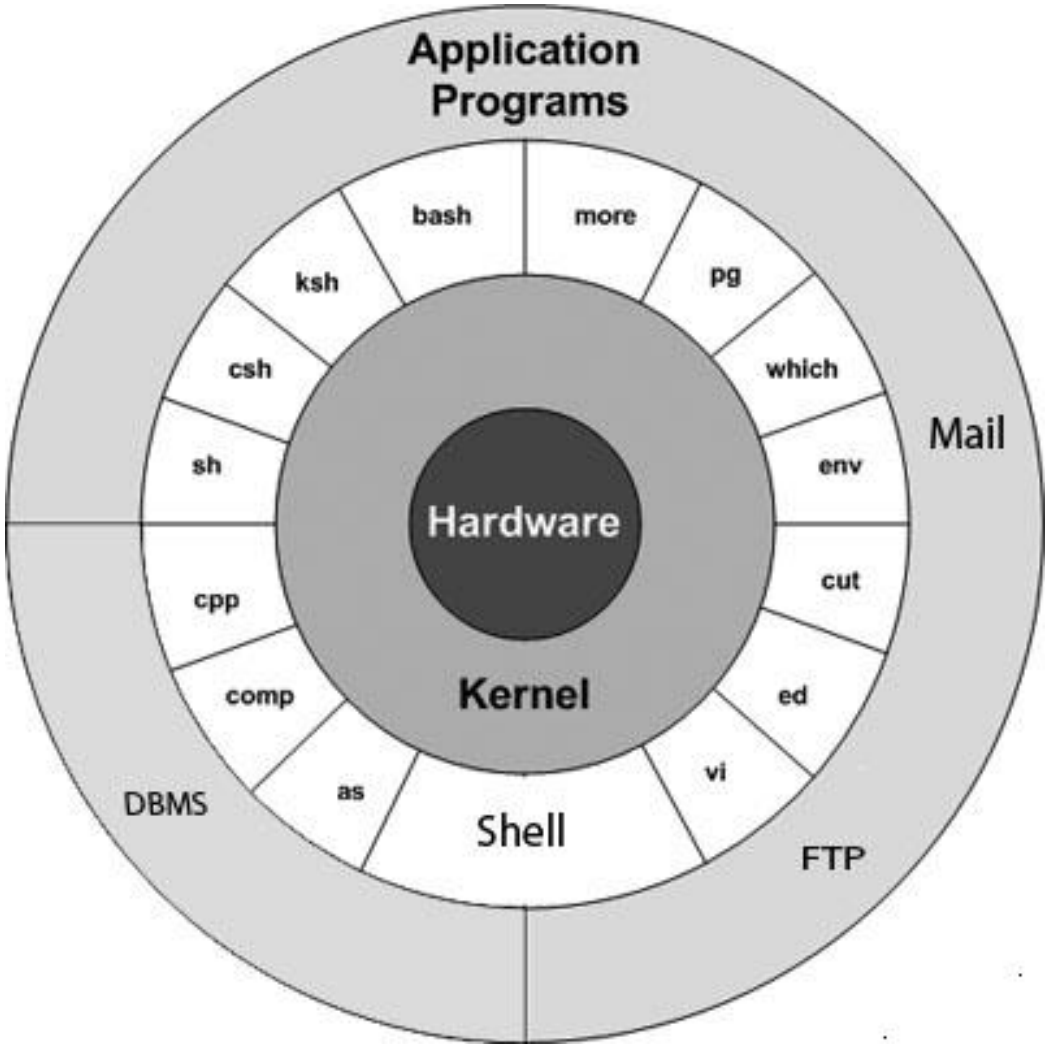
Features of UNIX

- Multi-User
- Multi-Programming
- Portability
- Machine Independent
- File Storage Capability

Different UNIX based OS

OS	Source & License	Release	Platform	Latest Release	Kernel	Status
Solaris **Oracle Corporation**	Mixed open source / closed source & Various	1992	SPARC, IA-32, X86-64, PowerPC	11.3, Oct 2015	Monolithic	Active
Darwin **Apple Inc & Open source community**	Open source & mostly APSL with proprietary drivers	2000	PowerPC, x86, ARM	17.3.0, Nov 2017	Hybrid	Active
AIX **IBM Corporation**	Closed source & proprietary	1986	ROMP, IBM POWER, PowerPC, x86 (IBM PS/2), System/370, ESA/390	7.2, Oct 2015	Monolithic	Active
HP-UX **Hewlett-Packard Company**	Closed source & proprietary	1982	PA-RISC, IA-64	11i v3 Update 16, March 2017	Monolithic	Active
FreeBSD **The FreeBSD Project**	Open source & FreeBSD license	1993	IA-32, x86-64, 64-bit SPARC, PowerPC, ARM, MIPS	11.1, Jul 2015	Monolithic	Active
NetBSD **The NetBSD Foundation**	Open source & 2-clauseBSD license	1993	Alpha, ARM, PA-RISC, 68k, MIPS, PowerPC, SH3, SPARC, RISC-V, VAX and x86	7.1.1 Dec 2017	Modular Monolithic AnyKernel (Rump Kernel)	Active
Xenix **Microsoft, SCO…**	Closed source & proprietary	1980	PC/XT, x86, PDP-11, Z8001, 68k	2.3.4, 1989	Monolithic	Discontinued
IRIX **Silicon Graphics**	Closed source & proprietary	1988	MIPS (Microprocessor without Interlocked Pipeline Stages)	6.5.30, Aug 2006	Monolithic	Discontinued
Tru64 **Digital Equipment Corporation**	Closed source & proprietary	1992	DEC Alpha	5.1B-6, Oct 2010	Hybrid Kernel	Discontinued
macOS **Apple Inc**	Closed source (with open source components)	2001	x86-64 (Discontinued PowerPC and IA-32)	10.13.2	Hybrid	Active

Layered architecture of the UNIX OS



Kernel (It's a heart of OS)

- Kernel is a program that constitutes the central core of the Operating System.
- Kernel provides basic services to all other parts of the Operating System including
  - Process Management: Process creation, termination, scheduling, execution etc
  - Memory Management: Allocation of memory to a program for execution
  - File Management: Creation of files, managing file permissions etc
  - I/O Management: Allocation of I/O devices
  - Network Management
- It can not directly interact with the user.
- But the interaction is done with the help of System Calls
- These services are requested by other parts of OS or by the application program through a set of program interfaces called as System Calls.

Shell

- Shell is an interface between user and the kernel.
- Its primary function is to read commands from the console and execute them.
- The term Shell comes from the fact that it is the outer most part of the OS.
- Several shells like Bourne, Korn, Bourne-again, C-Shell etc. are available.

System Call

- It's an interface with which a user application or any other part of OS request the kernel to provide services.
- Since user can not directly access the kernel, It can access it through system calls.
- There are many system calls for different types of services provided by the OS.
- For example,
  - File Management system calls – open(),read(),write(), close().
  - Process Management – fork(), exec()

C Library Function Calls and System Calls

- Functions defined in different C libraries are used in C programs.
- These functions internally invoke system calls to get the service from the kernel.
- C library function call can not directly invoke the functionalities in kernel.
- User application can directly invoke system call or through C library function.
- For example, printf() in turn calls a write() system call to print a string to stdout.

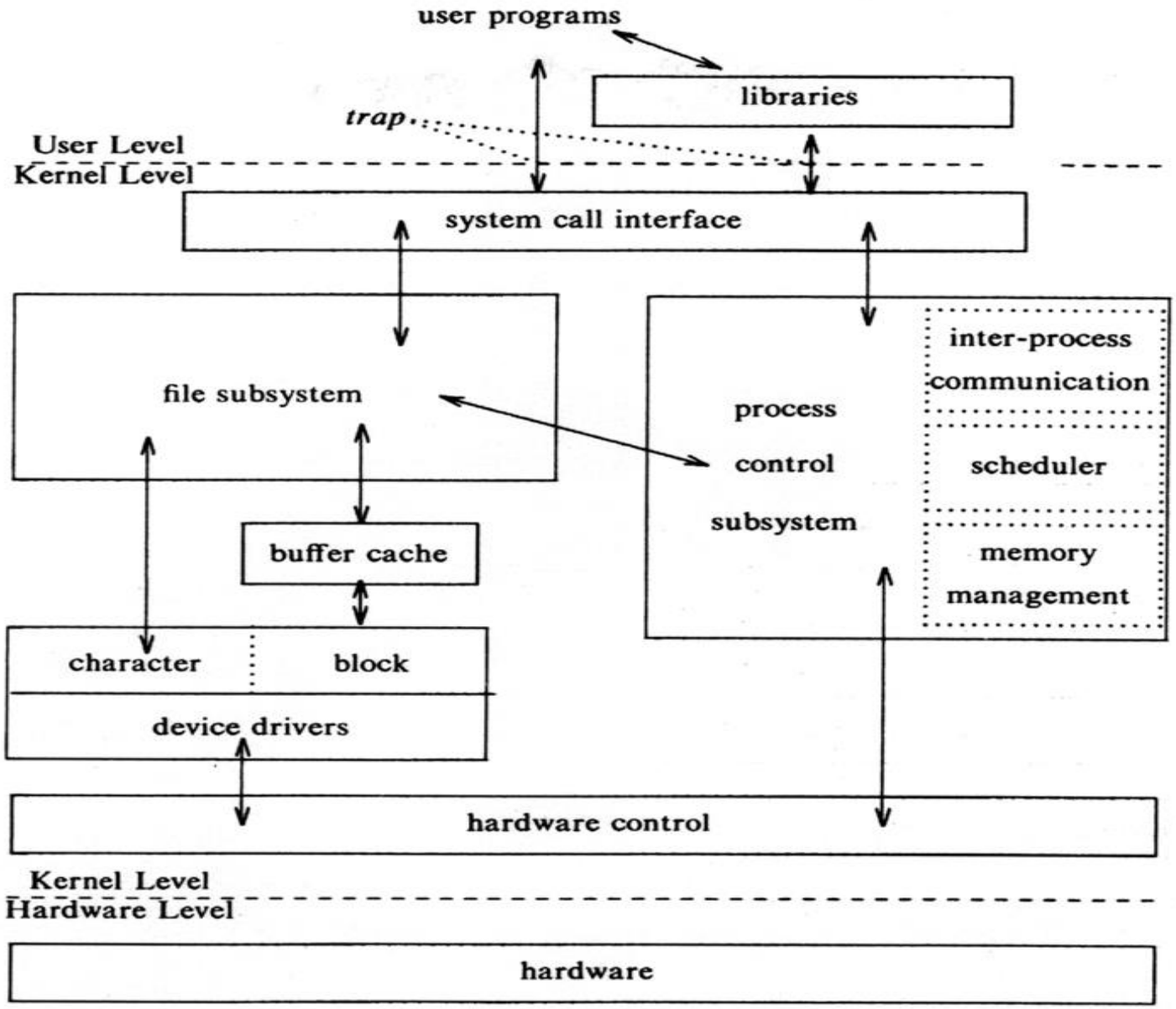
C Library Function Calls and System Calls

- malloc() in C is a library function in <stdlib> which is used to allocate memory. This in turn invokes a system call sbrk() which increases or decreases the address space of the process by the specified number of bytes.
- System calls like fork(), exec() can be directly invoked by the program.

Program Execution Process (exec() system call)

- An executable of a program is executed with the help of a system call exec().
- This system call loads the text and data of a program into the memory before execution.
- This system call takes the name of the executable along with command line arguments as input parameter and invokes the main() function of the program.

Block Diagram of System Kernel



Plan of Tutorial Classes

Tutorial No.	Topic	Description
1.	Introduction Unix Operating System	Architecture of Unix System, brief description of system kernel and its sub-modules, system calls, library functions etc.
2.	Internal Representation of Files	Internal structure of files and directories, Conversion of a path to an inode, superblock, inode assignment to a new file, allocation of disk block.
3.	System calls for Unix File System	File system calls: Open, read, write, create, close, pipes, dup etc.
4.	Programming using Unix file system calls	Programming Exercises

Any Queries?