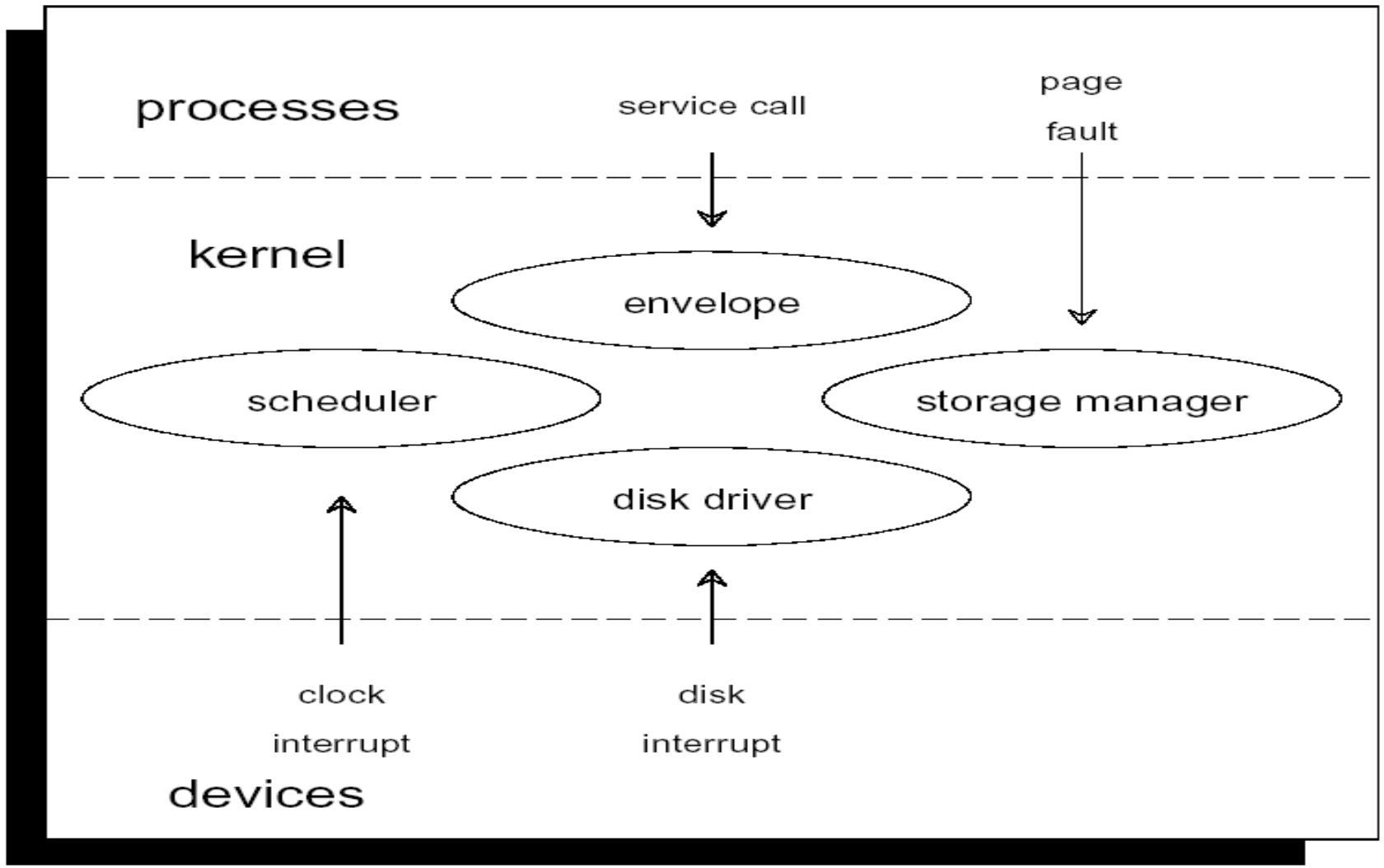
ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

OPERATING SYSTEMS & PARALLEL COMPUTING

Operating system kernels

One Kernel Point of View



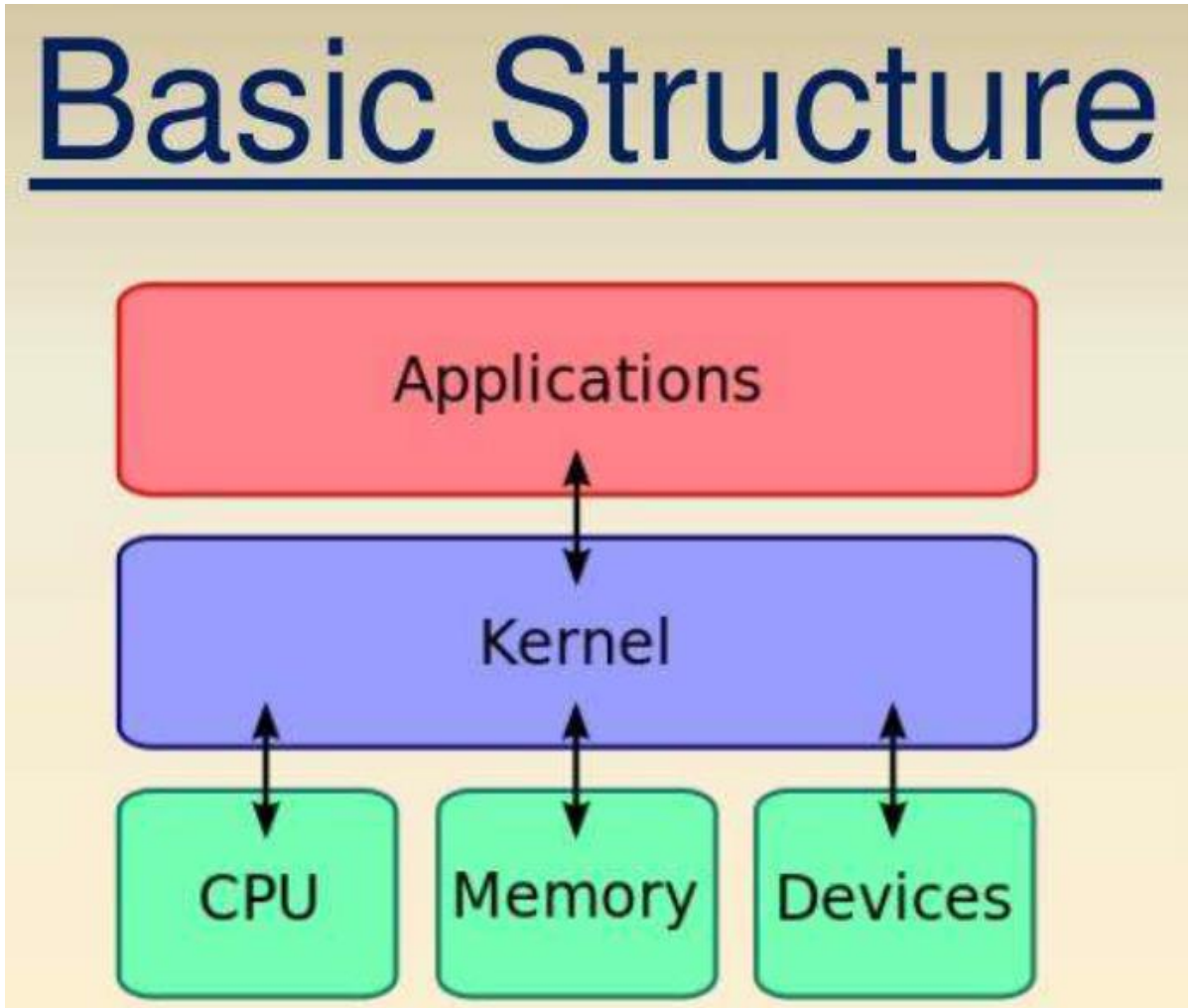
ie
SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

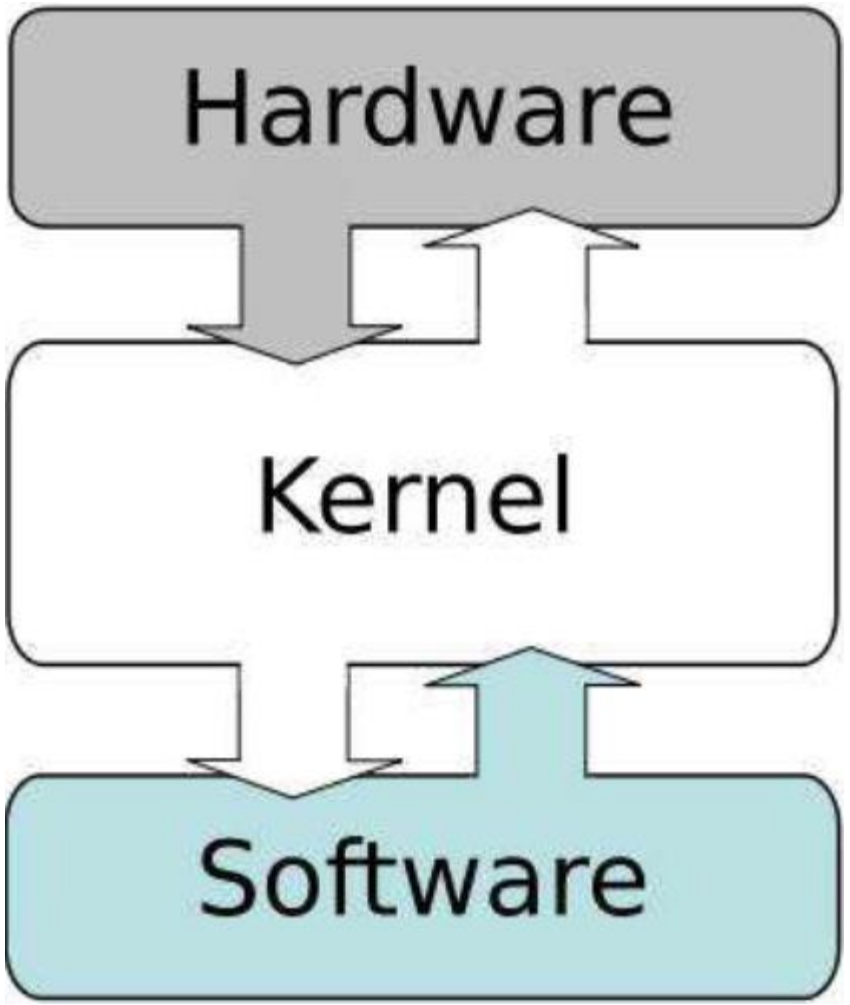
What is the OS/Kernel?

- Is the Operating System just the Kernel (not the utilities and application programs)?!
- The Command Line Interface (CLI) (or command layer/interpreter or shell) allows direct command entry by the user.
- The shell used to be in the kernel but now is a (first between equals) utility outside of it:
  - Easy to change/debug
  - Many of them (sh, bsh, csh, ksh, tcsh, wsh, bash)
  - Possible to switch between them (chsh)

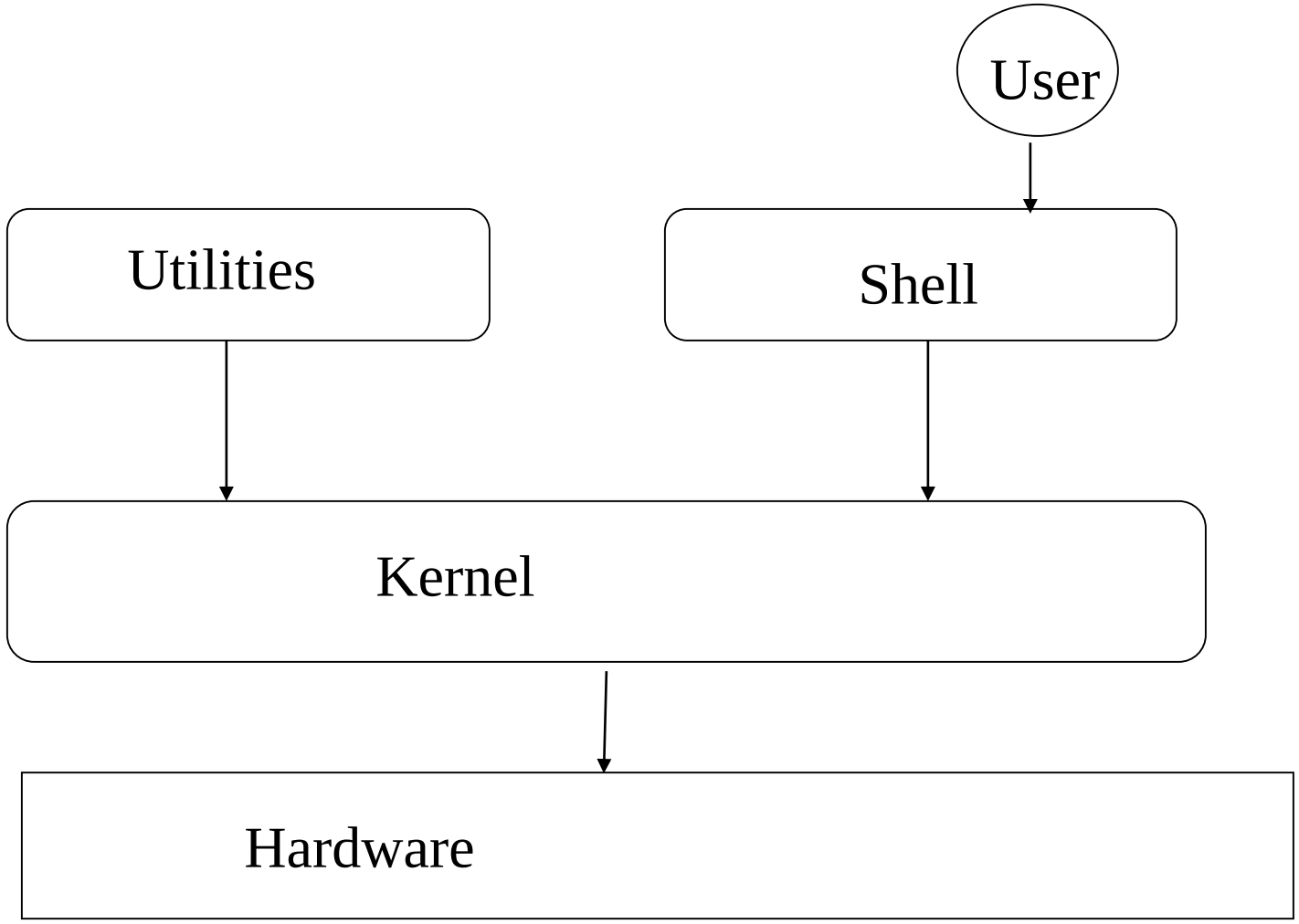
WHAT IS KERNEL?

- In computing, the kernel is the central component of most computer operating systems; it is a bridge between applications and the actual data processing done at the hardware level.
- It acts as an interface between the user applications and the hardware.
- The sole aim of kernel is to manage the communication between the software (user level applications) and the hardware (CPU, disk memory etc)
- When a process makes requests of the kernel, the request is called a system call. Various kernel designs differ in how they manage system calls and resources.


Basic Structure
Applications
Kernel
CPU
Memory
Devices

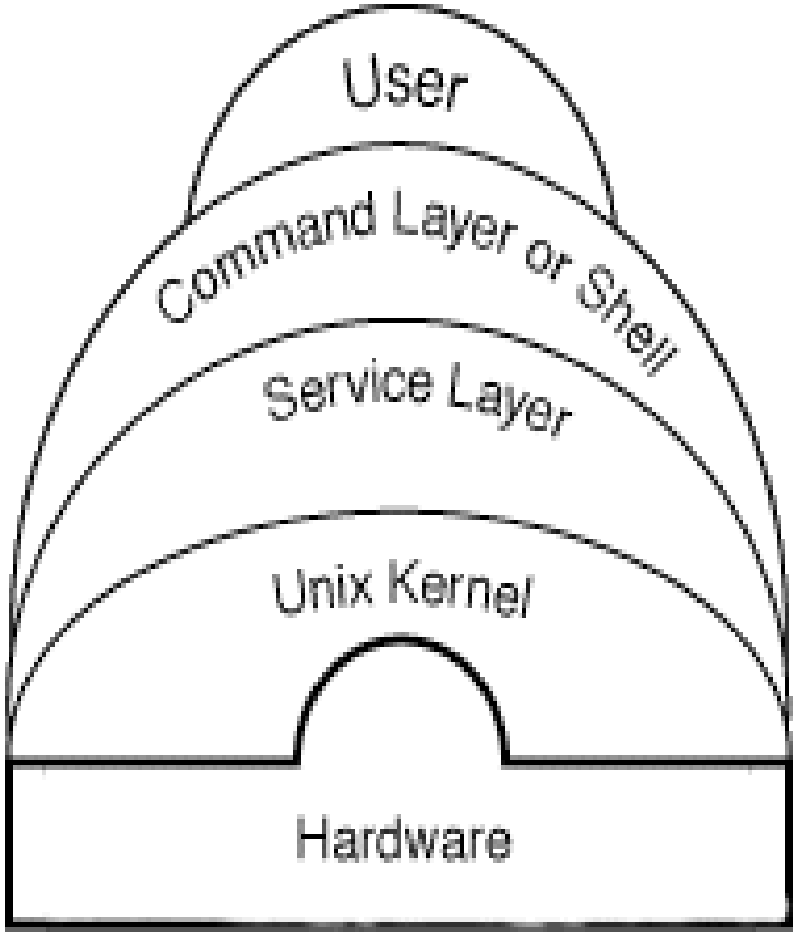

Hardware
Kernel
Software

UNIX Shell and Utilities

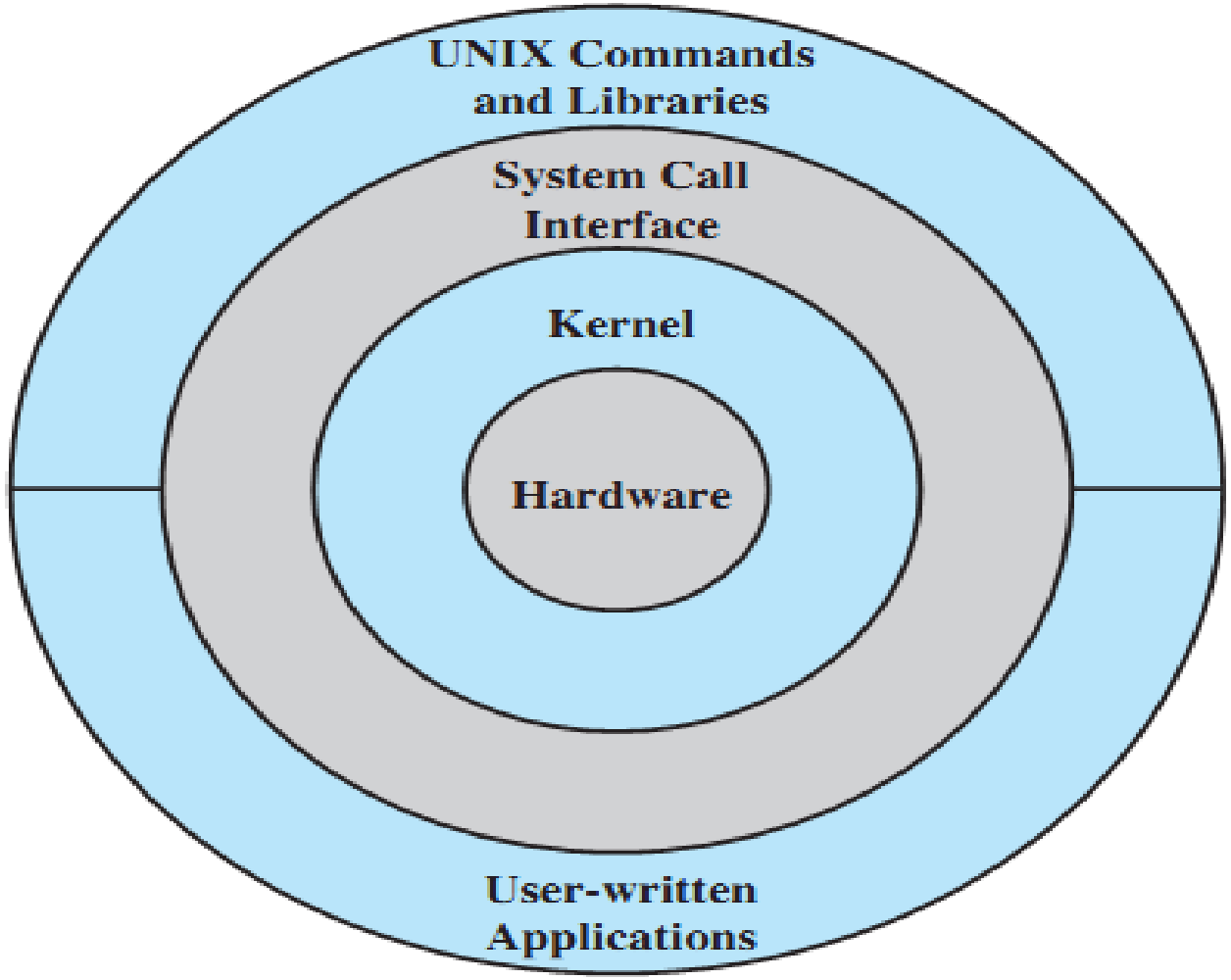


SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

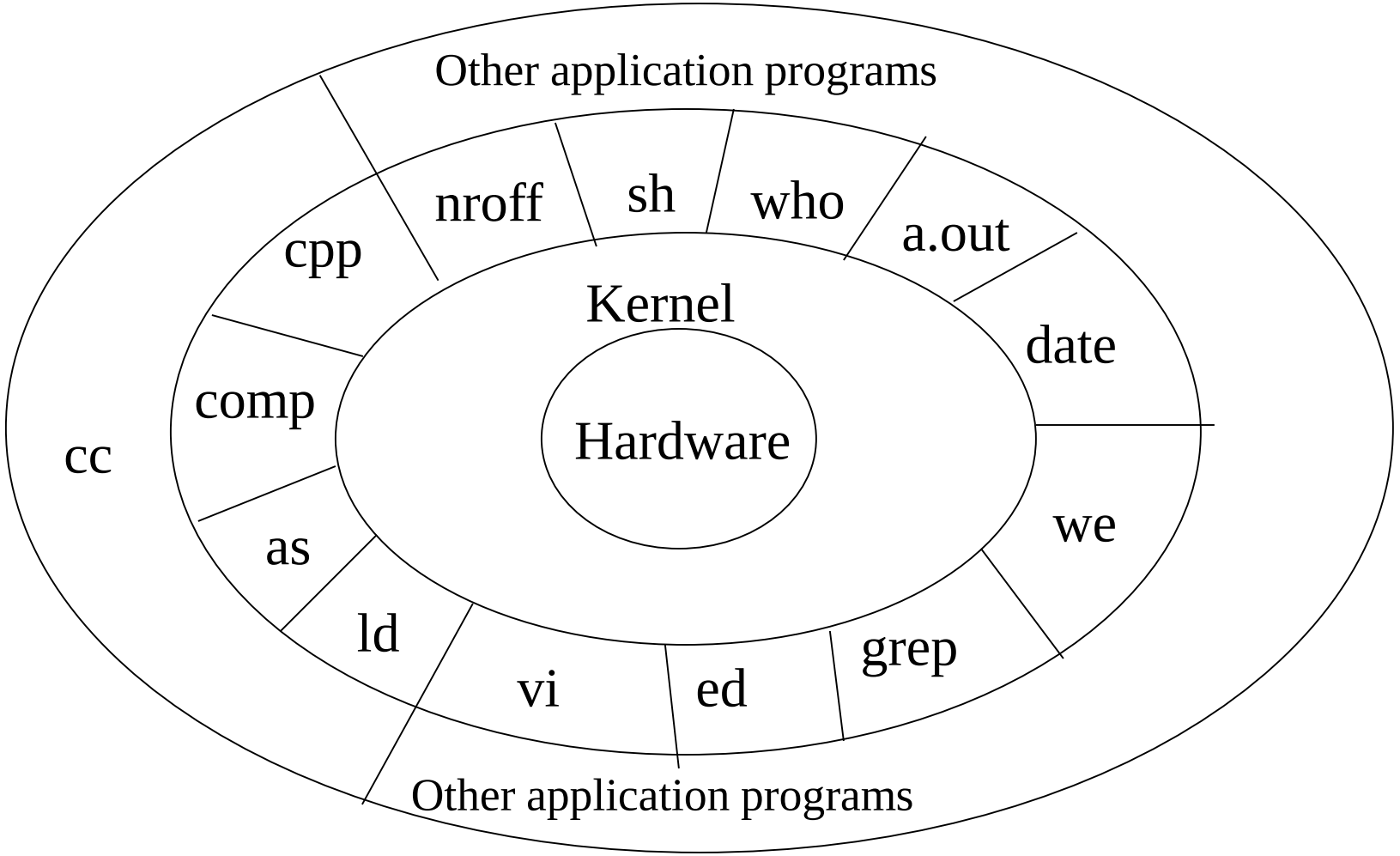
UNIX System Layout



General UNIX Architecture (1)



General UNIX Architecture (2)



TYPE OF KERNELS

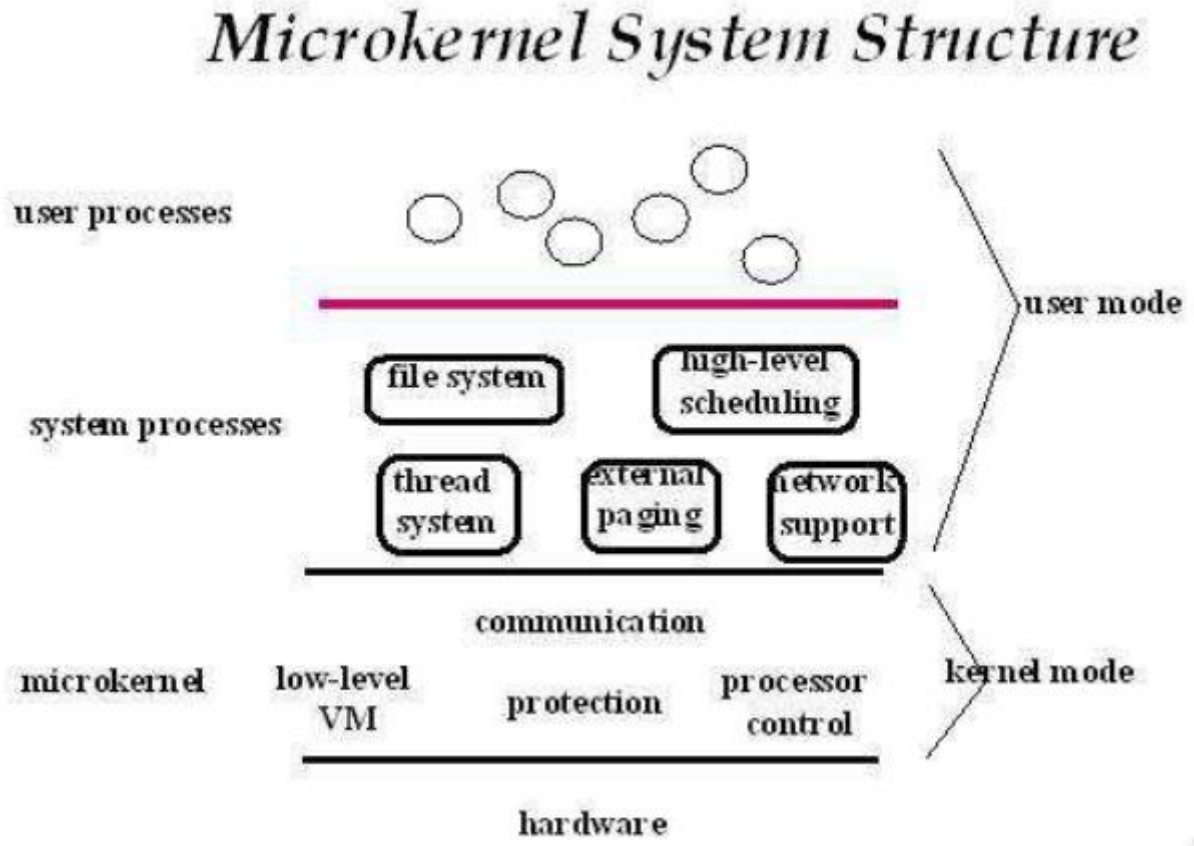
- Micro Kernel:
  - Only the very important parts like INTER process communication(IPC) , basic schedular, basic memory handling , basic I/O primitives are placed in kernel. Others are maintained as server processes in user space . Communication is done by message passing . •
- The kernel is broken down into separate processes, known as servers.
  - Some of the servers run in kernel space and some run in user-space. All servers are kept separate and run in different address spaces. The communication in microkernels is done via message passing. The servers communicate through InterProcess Communication IPC. Servers invoke “services” from each other by sending messages.

ADVANTAGES OF MICROKERNEL

- Microkernels are easier to maintain than monolithic kernels, but the large number of system calls and context switches might slow down the system because they typically generate more overhead than plain function calls. A microkernel allows the implementation of the remaining part of the operating system as a normal application program written in a high-level language
- Crash resistant (if one server fails, other servers can still work efficiently.)
- Portable
- Smaller in Size

DISADVANTAGES OF MICROKERNEL

- Slower processing because of additional message passing.

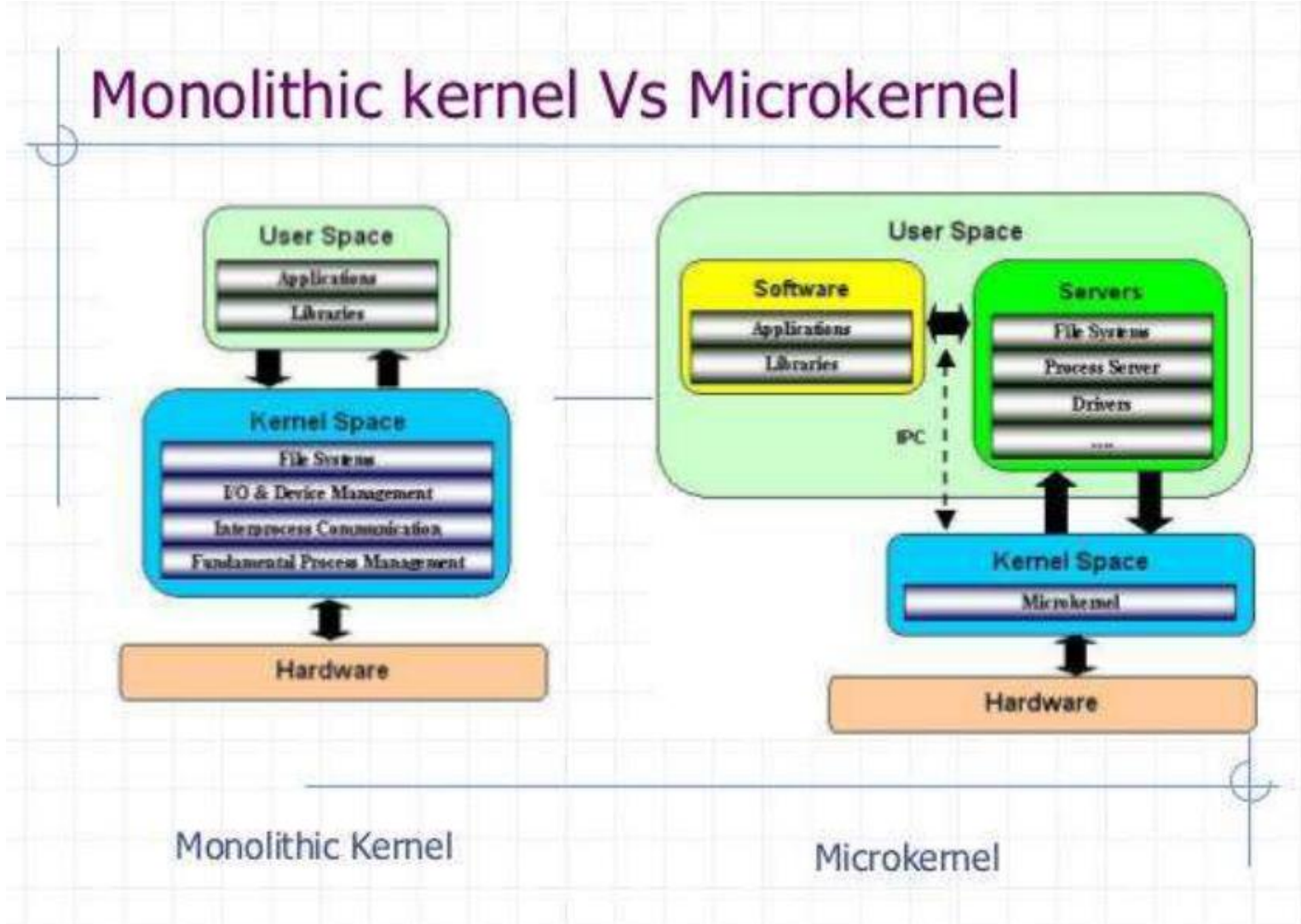


MONOLITHIC KERNEL

- **MONOLITHIC KERNEL**: It runs every basic system service like process and memory management, interrupt handling and I/O communication, file system etc. in kernel space. Examples are Linux, Unix.

  - Advantages: performance
  - Disadvantages: difficult to debug and maintain

Monolithic kernel Vs Microkernel



HYBRID KERNEL

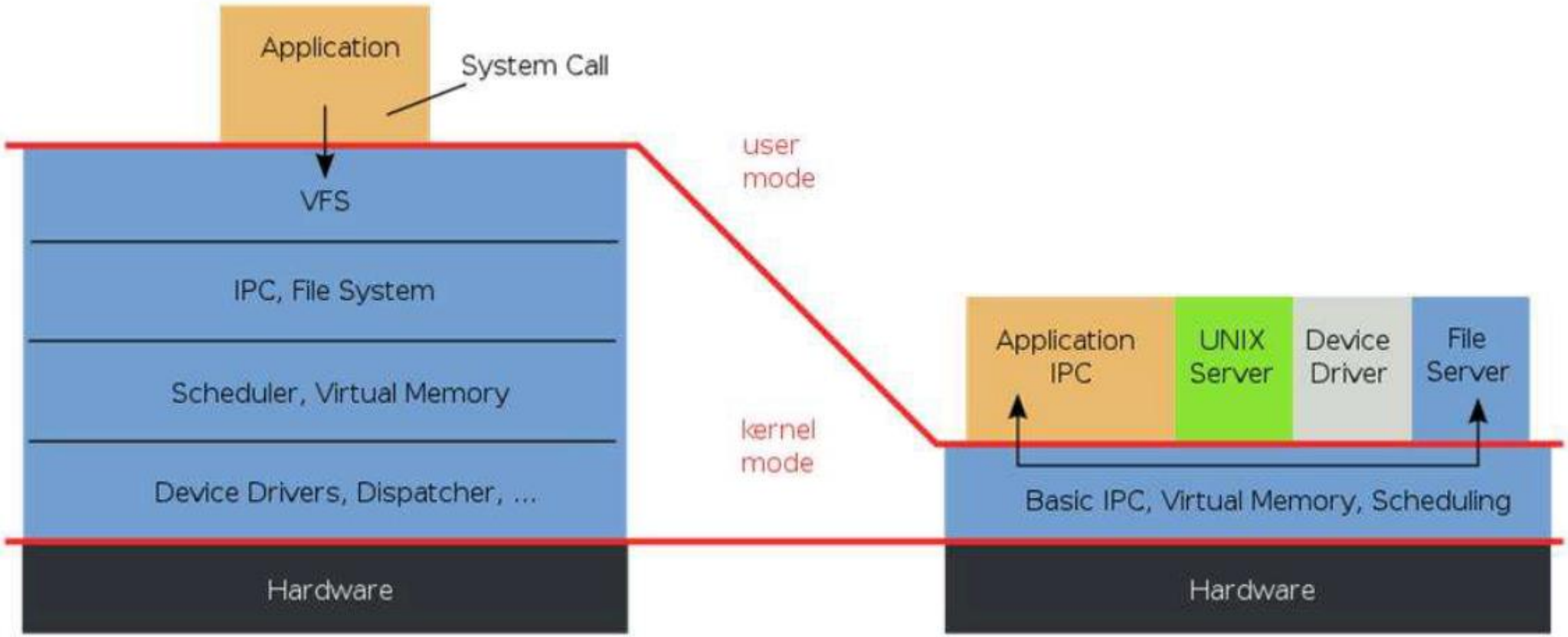
- One other type of kernel is called a **hybrid kernel** which lies on the boundary between monolithic kernels and microkernels. This means it has qualities of both, but hybrid kernels cannot be classified as a monolithic kernel or microkernel exclusively.

- Combine the best of both worlds
  - Speed and simple design of a monolithic kernel
  - Modularity and stability of a microkernel

- Advantages:
  - Benefits of monolithic and microkernels
- Disadvantages:
  - Same as monolithic kernels

Monolithic Kernel based Operating System

Microkernel based Operating System



SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

KERNEL'S RESPONSIBILITIES

- The central processing unit
- Random access memory
- INPUT/OUTPUT devices
- Memory management
- Device management
- System calls

THE CENTRAL PROCESSING UNIT

- This central component of a computer system is responsible for *running* or *executing* programs. The kernel takes responsibility for deciding at any time which of the many running programs should be allocated to the processor or processors (each of which can usually run only one program at a time).

ie
SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

FEATURES THAT KERNEL PROVIDES

- low-level scheduling of processes (dispatching)
- inter-process communication
- process synchronization
- context switching
- manipulation of process control blocks
- interrupt handling
- process creation and destruction
- process suspension and resumption

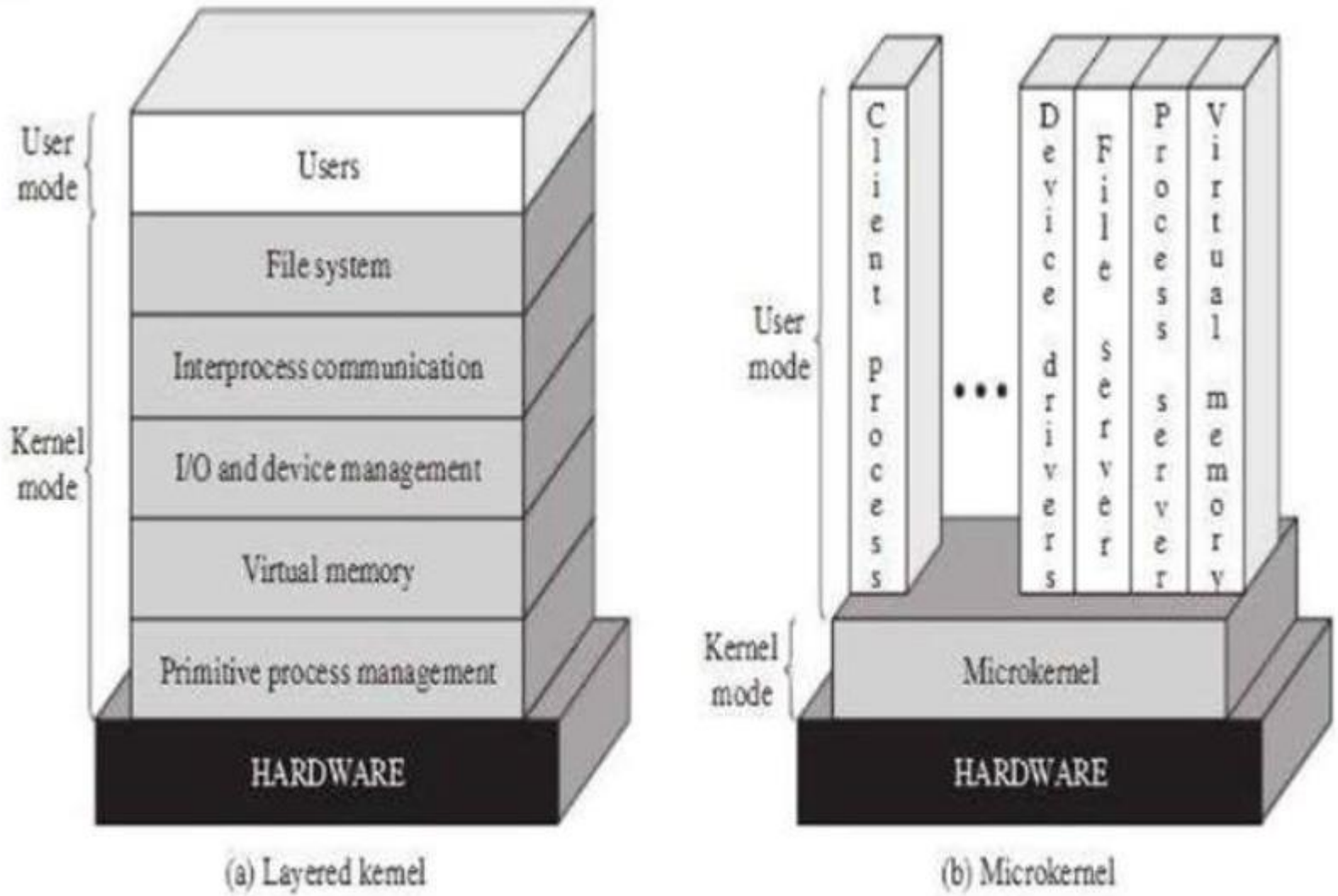


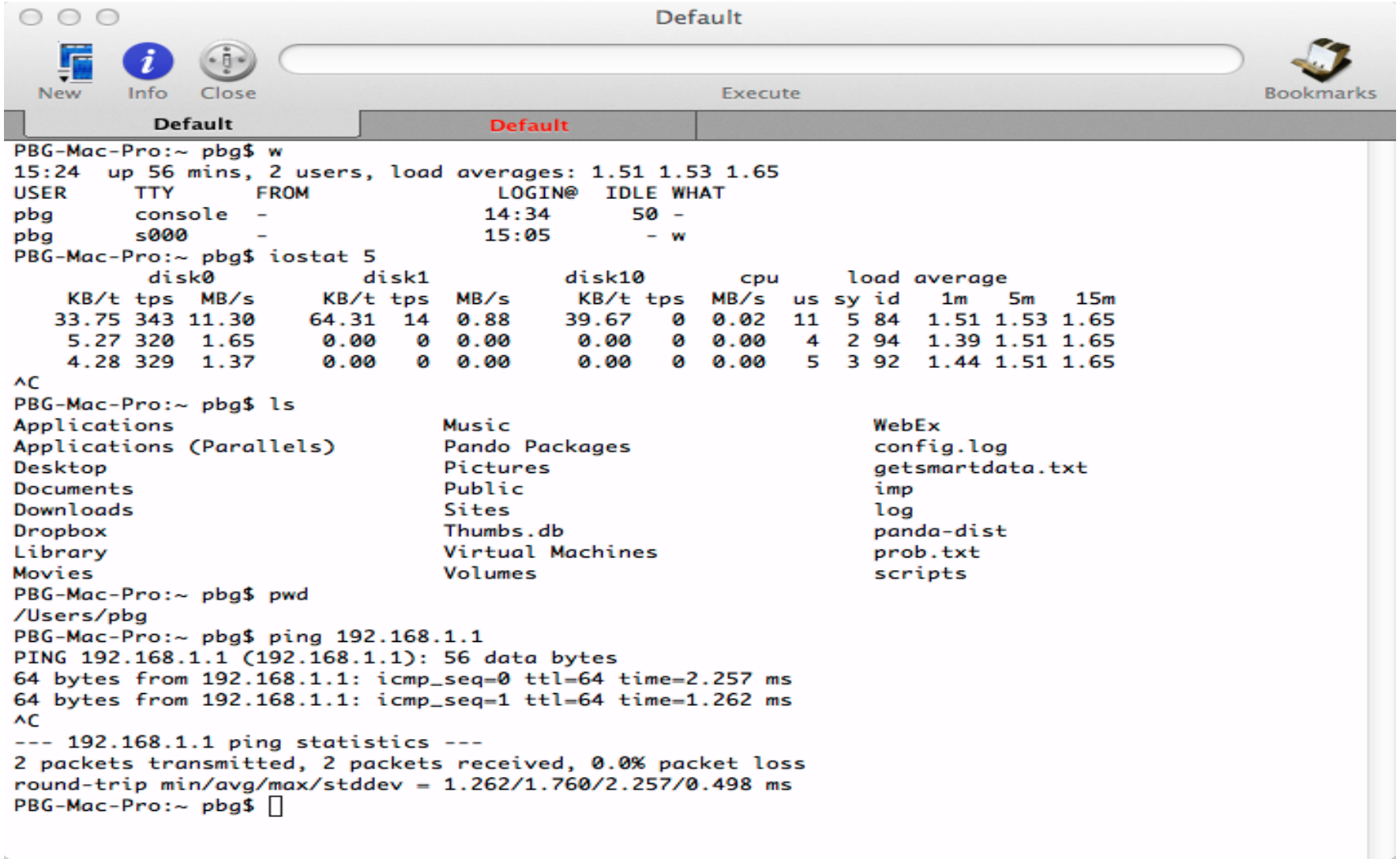
Figure 4.10 Kernel Architecture

CLI is the User OS Interface

CLI allows direct command entry:

- Sometimes implemented in kernel, sometimes by systems program.
- Sometimes multiple flavors implemented – shells.
- Primarily fetches a command from user and executes it.
- Sometimes commands built-in, sometimes just names of programs; if the latter, adding new features doesn’t require shell modification.

Bourne Shell (bsh)




```
PBG-Mac-Pro:~ pbg$ w
15:24  up 56 mins, 2 users, load averages: 1.51 1.53 1.65
USER     TTY      FROM              LOGIN@  IDLE WHAT
pbg      console  -                 14:34      50 -
pbg      s000     -                 15:05       - w
PBG-Mac-Pro:~ pbg$ iostat 5
          disk0           disk1           disk10       cpu      load average
    KB/t tps  MB/s     KB/t tps  MB/s     KB/t tps  MB/s  us sy id   1m   5m   15m
   33.75 343 11.30    64.31  14  0.88    39.67   0  0.02  11  5 84  1.51 1.53 1.65
    5.27 320  1.65     0.00   0  0.00     0.00   0  0.00   4  2 94  1.39 1.51 1.65
    4.28 329  1.37     0.00   0  0.00     0.00   0  0.00   5  3 92  1.44 1.51 1.65
^C
PBG-Mac-Pro:~ pbg$ ls
Applications                    Music                           WebEx
Applications (Parallels)        Pando Packages                  config.log
Desktop                         Pictures                        getsmartdata.txt
Documents                       Public                          imp
Downloads                       Sites                           log
Dropbox                         Thumbs.db                       panda-dist
Library                         Virtual Machines                prob.txt
Movies                          Volumes                         scripts
PBG-Mac-Pro:~ pbg$ pwd
/Users/pbg
PBG-Mac-Pro:~ pbg$ ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=64 time=2.257 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=1.262 ms
^C
--- 192.168.1.1 ping statistics ---
2 packets transmitted, 2 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 1.262/1.760/2.257/0.498 ms
PBG-Mac-Pro:~ pbg$
```

A very simplified Shell

```
#define TRUE 1

while (TRUE) {                                  /* repeat forever */
        type_prompt( );                         /* display prompt on the screen */
        read_command(command, parameters);      /* read input from terminal */

        if (fork( ) != 0) {                     /* fork off child process */
            /* Parent code. */
            waitpid(-1, &status, 0);            /* wait for child to exit */
        } else {
            /* Child code. */
            execve(command, parameters, 0);     /* execute command */
        }
}
```

SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY