BCA Fourth Semester

Operating Systems

Unit -2 Operating System Structures [2 Hrs]

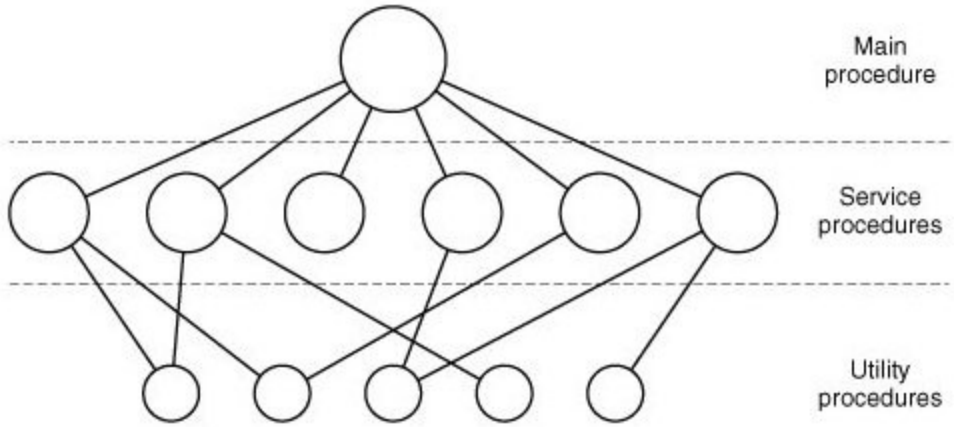
Operating System Structure:

- **The structure of an operating system is dictated by the model employed in building them.**
- **An operating system model is a broad framework that unifies the many features and services the operating system provides and tasks it performs.**

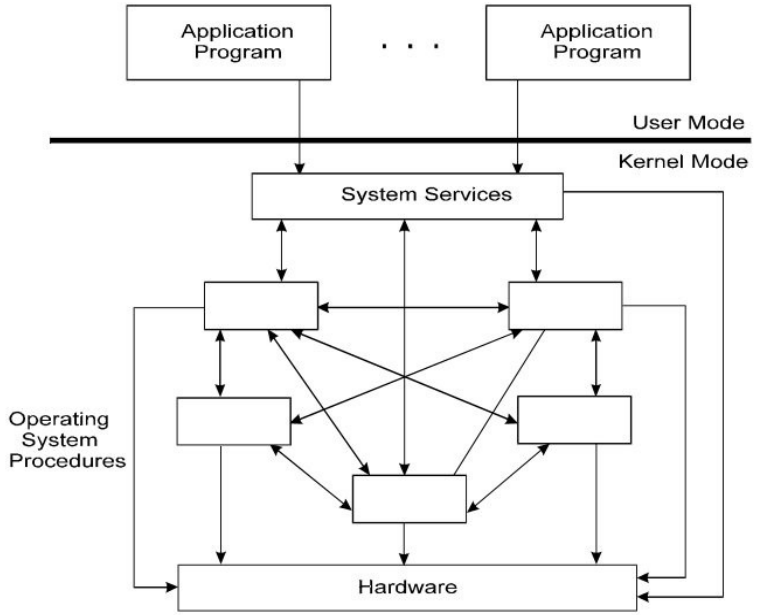
- Operating systems are broadly classified into following categories, based on the their structuring mechanism as follows:
  a. Monolithic System
  b. Layered System
  c. Virtual Machine
  d. Kernels
  e. Client-Server Model

Monolithic System

- **The components of monolithic operating system are organized haphazardly and any module can call any other module without any reservation.**
- **Similar to the other operating systems, applications in monolithic OS are separated from the operating system itself.** That is, the operating system code runs in a privileged processor mode (referred to as kernel mode), with access to system data and to the hardware; applications run in a non-privileged processor mode (called the user mode), with a limited set of interfaces available and with limited access to system data.

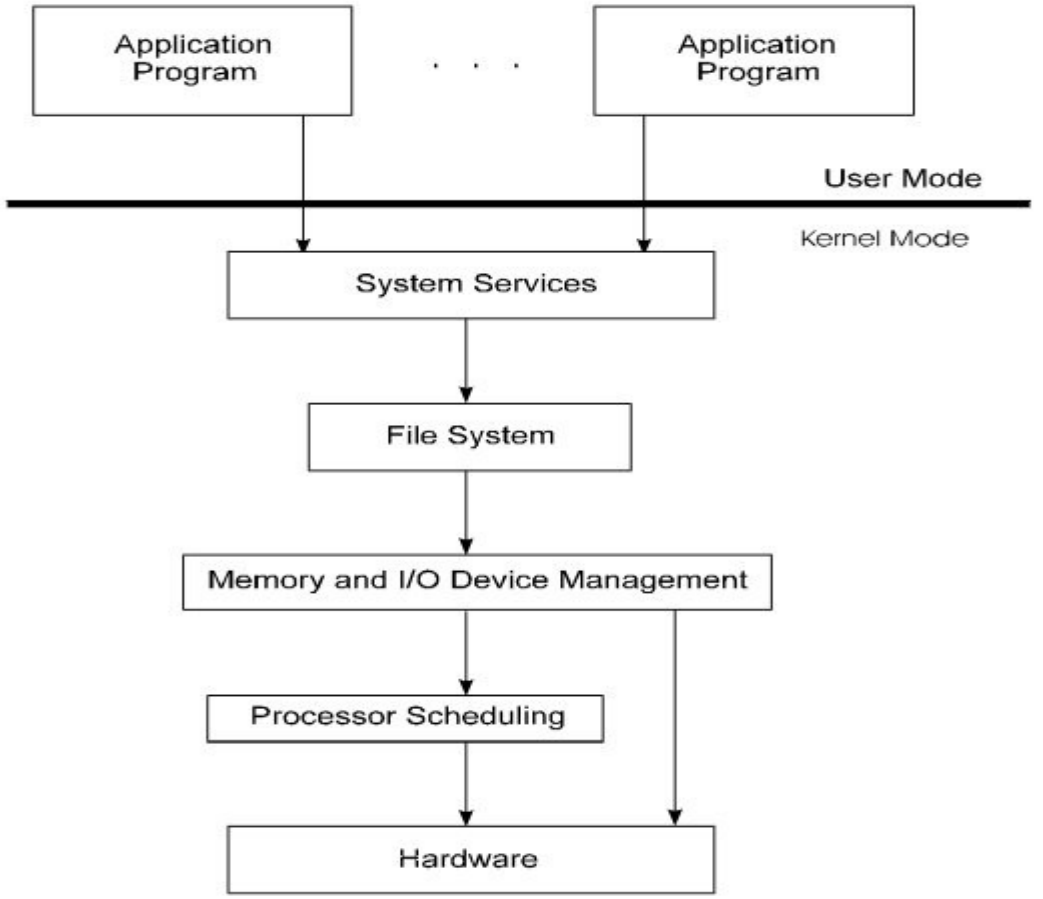


- **This approach might well be subtitled "The Big Mess." The structure is that there is no structure.** The operating system is written as a collection of procedures, each of which can call any of the other ones whenever it needs to. When this technique is used, each procedure in the system has a well-defined interface in terms of parameters and results, and each one is free to call any other one, if the latter provides some useful computation that the former needs.
- **Example Systems:** CP/M and MS-DOS



Layered System

- **The layered approach consists of breaking the operating system into the number of layers(level), each built on the top of lower layers.**
- The bottom layer (layer 0) is the hardware layer; the highest layer is the user interface.
- **The main advantages of the layered approach is modularity.** The layers are selected such that each uses functions (operations) and services of only lower-level layers.
- This approach simplifies debugging and system verifications. That is in this approach, the Nth layer can access services provided by the (N- 1)th layer and provide services to the (N+1)th layer.
- This structure also allows the operating system to be debugged starting at the lowest layer, adding one layer at a time until the whole system works correctly. Layering also makes it easier to enhance the operating system; one entire layer can be replaced without affecting other parts of the system.



Layered System (contd..)

- The layer approach to design was first used in the THE operating system at the Technische Hogeschool Eindhoven.
- The THE system was defined in the six layers , as shown in the fig below.

Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and drum management
0	Processor allocation and multiprogramming

- **Example Systems:** VAX/VMS, Multics, UNIX

Virtual Machines:

- **Virtual machine approach provides an interface that is identical to the underlying bare hardware.** Each process is provided with a (virtual) copy of the underlying computer as shown in the fig.

- The resources of the physical computer are shared to create the virtual machine. CPU scheduling can be used to share the CPU and to create the appearance that users have their own processors.

- Although the virtual machine concept is useful, it is difficult to implement. Much effort is required to provide an exact duplicate of the underlying machine.
- **Example**. Java

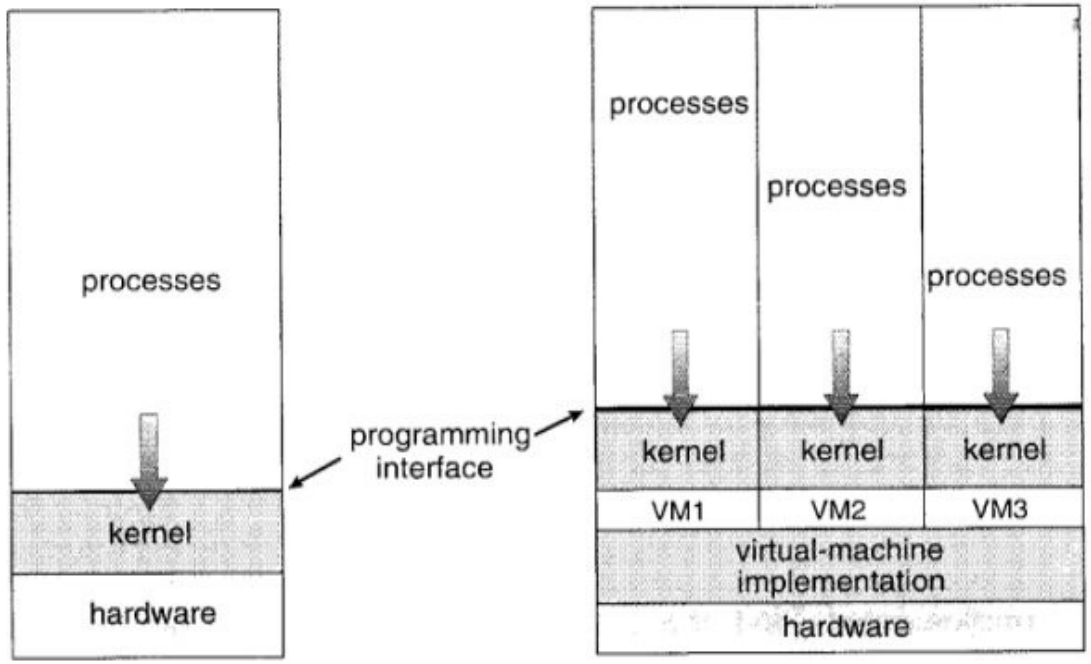
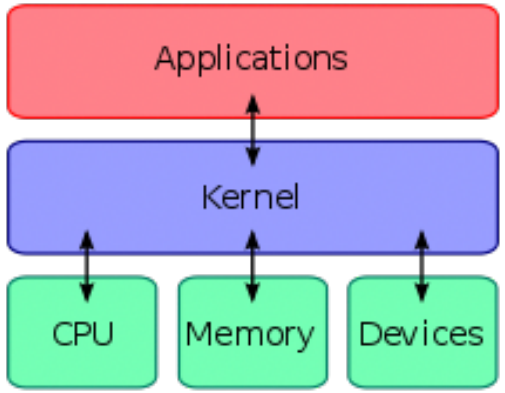


Fig: a). Non Virtual Machine b). Virtual Machine.

Kernel:

- **In computing, the kernel is the main component of most computer operating systems; it is a bridge between applications and the actual data processing done at the hardware level.**
- The kernel's responsibilities include managing the system's resources (the communication between hardware and software components).
- Usually as a basic component of an operating system, a kernel can provide the lowest-level abstraction layer for the resources (especially processors and I/O devices) that application software must control to perform its function.
- It typically makes these facilities available to application processes through inter-process communication mechanisms and system calls.



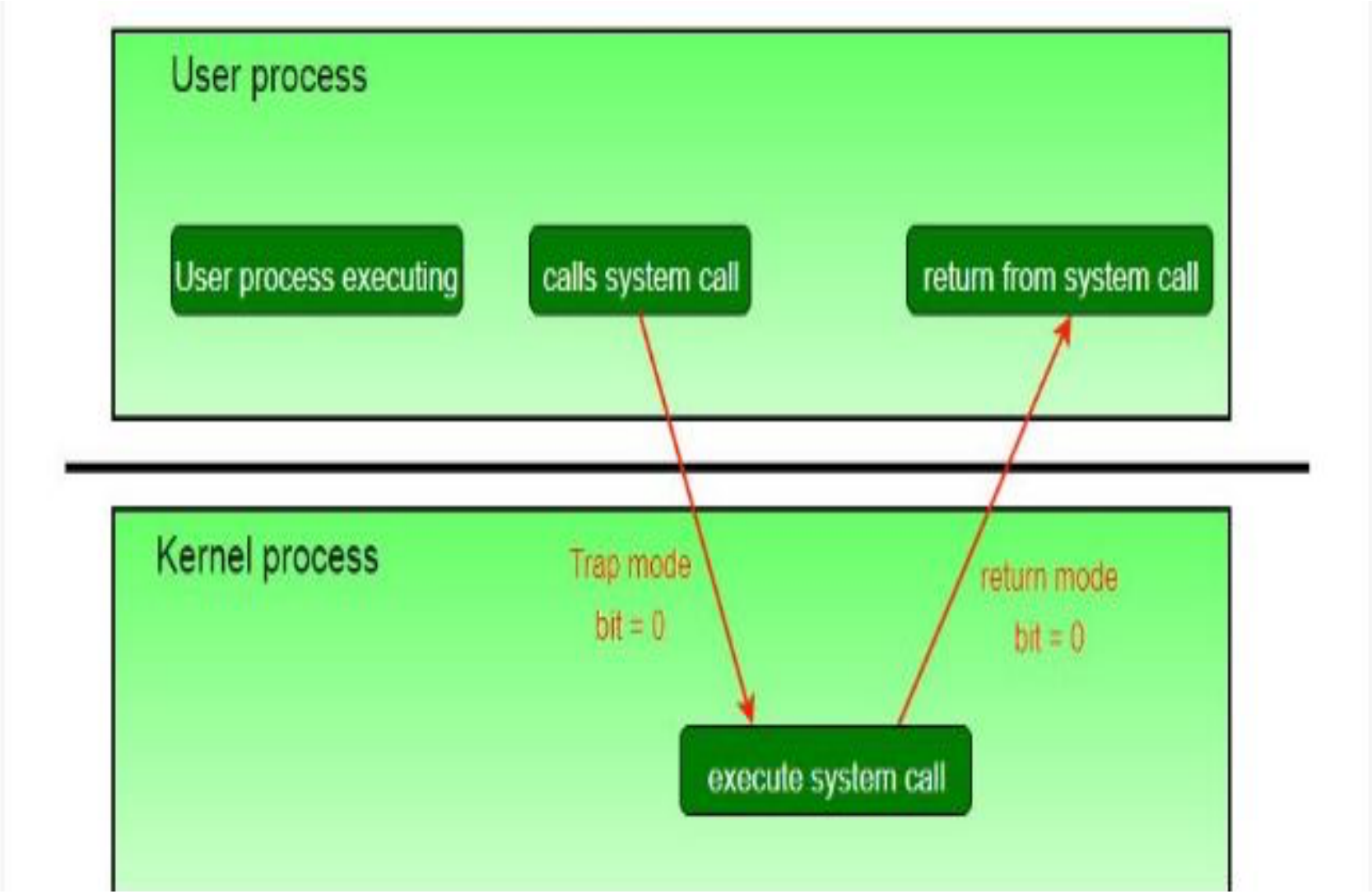
- Operating system tasks are done differently by different kernels, depending on their design and implementation. While monolithic kernels execute all the operating system code in the same address space to increase the performance of the system, microkernels run most of the operating system services in user space as servers, aiming to improve maintainability and modularity of the operating system. A range of possibilities exists between these two extremes.

Kernel mode and User mode of CPU operation

The CPU can execute certain instruction only when it is in the kernel mode. These instruction are called privilege instruction. They allow implementation of special operation whose execution by the user program could interface with the functioning of operating system or activity of another user program. For example, instruction for managing memory protection.

- The operating system puts the CPU in kernel mode when it is executing in the kernel so, that kernel can execute some special operation.
- The operating system puts the CPU in user mode when a user program is in execution so, that user program cannot interface with the operating system program.
- User-level instruction does not require special privilege. Example are ADD,PUSH,etc.

Kernel mode and User mode of CPU operation (contd..)



a) Microkernel

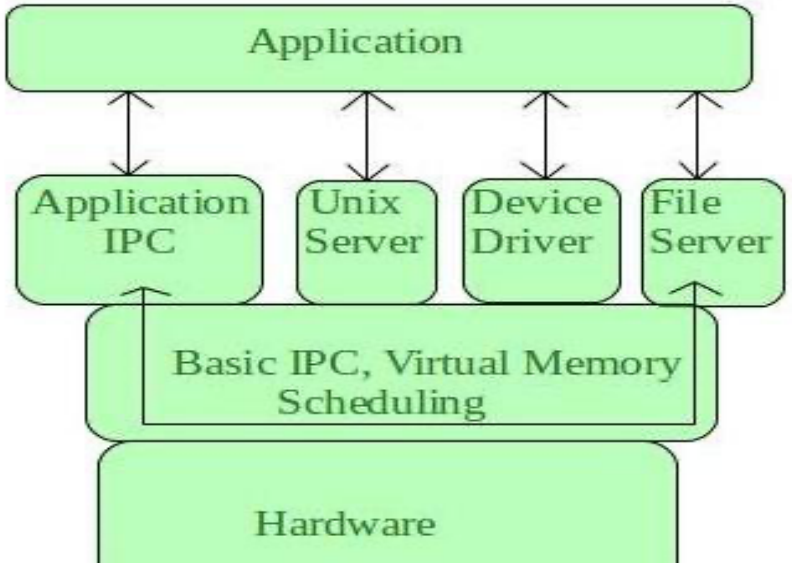
What is Microkernel?

Microkernel is one of the classification of the kernel. Being a kernel it manages all system resources. But in a microkernel, the **user services** and **kernel services** are implemented in different address space. The user services are kept in **user address space**, and kernel services are kept under **kernel address space**, thus also reduces the size of kernel and size of operating system as well.

Advantages of Microkernel –

- The architecture of this kernel is small and isolated hence it can function better.
- Expansion of the system is easier, it is simply added in the system application without disturbing the kernel.

Eclipse IDE is a good example of Microkernel Architecture.



b) Monolithic Kernel

Apart from microkernel, **Monolithic Kernel** is another classification of Kernel. Like microkernel this one also manages system resources between application and hardware, but **user services** and **kernel services** are implemented under same address space. It increases the size of the kernel, thus increases size of operating system as well.

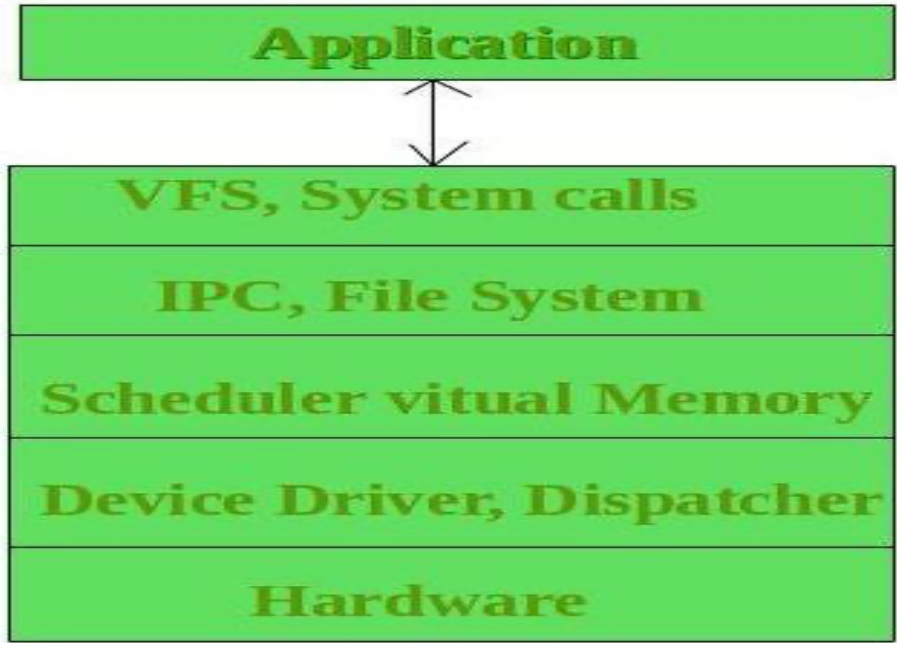
This kernel provides CPU scheduling, memory management, file management and other operating system functions through system calls. As both services are implemented under same address space, this makes operating system execution faster.

Advantages of Monolithic Kernel –

- One of the major advantage of having monolithic kernel is that it provides CPU scheduling, memory management, file management and other operating system functions through system calls.
- The other one is that it is a single large process running entirely in a single address space.
- It is a single static binary file. Example of some Monolithic Kernel based OSs are: Unix, Linux, Open VMS, XTS-400, z/TPF.

Disadvantages of Monolithic Kernel –

- One of the major disadvantage of monolithic kernel is that, if anyone service fails it leads to entire system failure.
- If user has to add any new service. User needs to modify entire operating system.



Monolithic Kernel Vs Micro Kernel

Basis for Comparison	Microkernel	Monolithic Kernel
Size	Microkernel is smaller in size	It is larger than microkernel
Execution	Slow Execution	Fast Execution
Extendible	It is easily extendible	It is hard to extend
Security	If a service crashes, it does effects on working on the microkernel	If a service crashes, the whole system crashes in monolithic kernel.
Code	To write a microkernel more code is required	To write a monolithic kernel less code is required
Example	QNX, Symbian, L4Linux etc.	Linux,BSDs(FreeBSD,OpenBSD,NetBSD)etc.

c) Exokernels

- Exo-kernel is a type of operating system that provides application-level management of hardware resources.
- The exokernel architecture is designed to separate resource protection from management to facilitate application-specific customization.
- Exokernels are typically small in size because of their limited operability.
- The main goal of an exokernel is to ensure that there is no forced abstraction, which is what makes an exokernel different from micro- and monolithic kernels.

Benefits of the exokernel operating system

- Improved performance of applications
- More efficient use of hardware resources through precise resource allocation and revocation
- Easier development and testing of new operating systems
- Each user-space application is allowed to apply its own optimized memory management

Drawbacks of the exokernel operating system

- Reduced consistency
- Complex design of exokernel interfaces

Client-Server Model

- The advent of new concepts in operating system design, microkernel, is aimed at migrating traditional services of an operating system out of the monolithic kernel into the user-level process.
- **The idea is to divide the operating system into several processes, each of which implements a single set of services - for example, I/O servers, memory server, process server, threads interface system.**
- **Each server runs in user mode, provides services to the requested client**. The client, which can be either another operating system component or application program, requests a service by sending a message to the server.
- An OS kernel (or microkernel) running in kernel mode delivers the message to the appropriate server; the server performs the operation; and microkernel delivers the results to the client in another message, as illustrated in Figure.

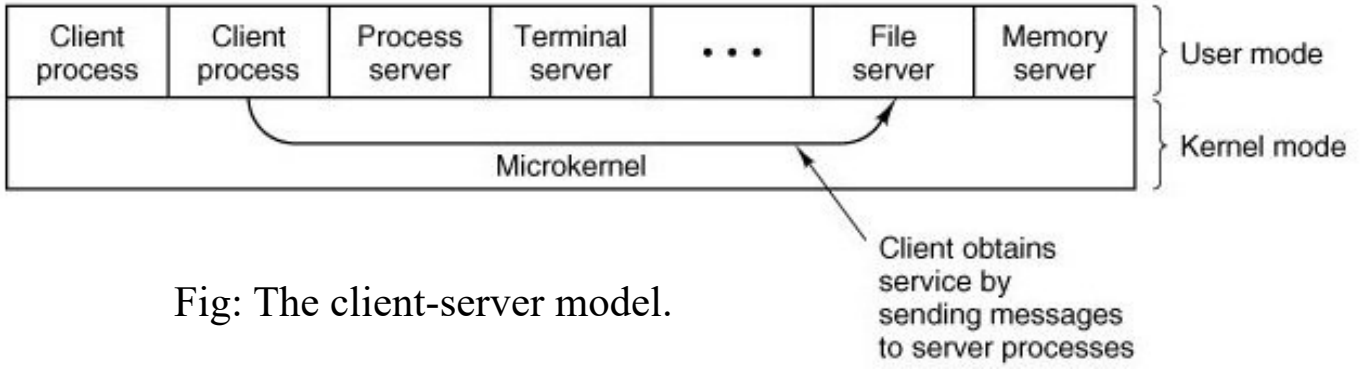


Fig: The client-server model.

Client-Server Model (contd..)

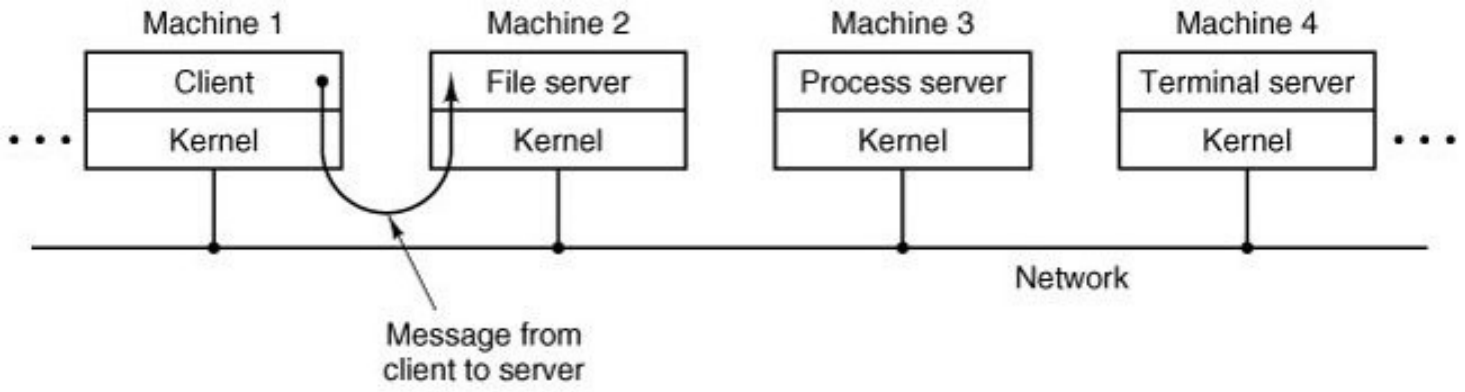


Fig: The client-server model in a distributed system.

The Shell

- **A shell is a program that provides the traditional text only user interface for Linux and other Unix operating system. Its primary function is to read commands typed into a console or terminal window and then execute it.**
- **A shell is software that provides an interface for an operating system's users to provide access to the kernel's services.**
- **An OS starts a shell for each user when the user logs in or opens a terminal or console window.**
- **It is named a shell because it is the outermost layer around the operating system kernel.**
- The term shell derives its name form the fact that it is an outer layer of the OS. A shell is an interface between the user and the internal part of the operating system.
- A user is in shell(i.e. interacting with the shell) as soon as the user has logged into the system. A shell is the most fundamental way that user can interact with the system and the shell hides the detail of the underlying system from the user.

- **Example:**

  Bourne Shell

  Bash shell

  Korn Shell

  C shell