OPERATING SYSTEM STRUCTURE:

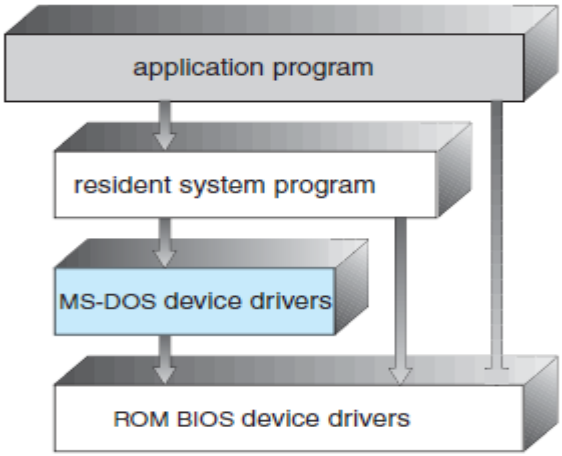
- The operating systems are large and complex. A common approach is to partition the task into small components, or modules, rather than have one **monolithic** system.
- The structure of an operating system can be defined the following structures.
    - Simple structure
    - Layered approach
    - Microkernels
    - Modules
    - Hybrid systems

Simple structure:

- The Simple structured operating systems do not have a well defined structure. These systems will be simple, small and limited systems.

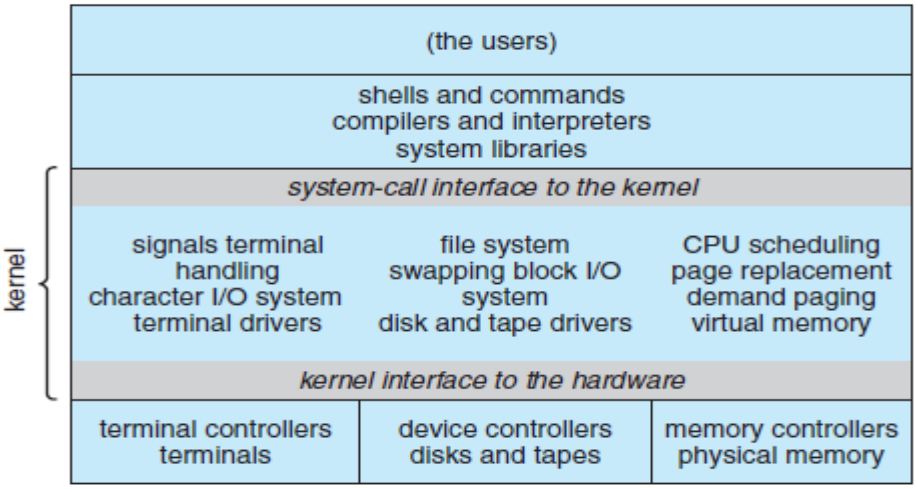
Example: MS-DOS.

- In MS-DOS, the interfaces and levels of functionality are not well separated.
- In MS-DOS application programs are able to access the basic I/O routines. This causes the entire systems to be crashed when user programs fail.



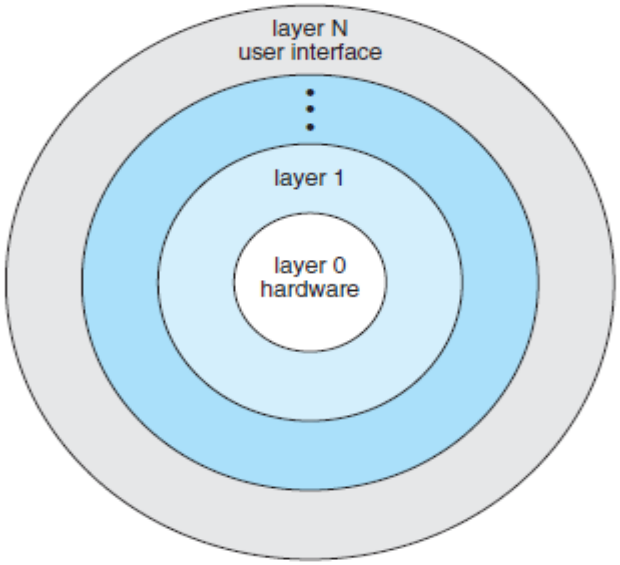
Example: Traditional UNIX OS

- It consists of two separable parts: the kernel and the system programs.
- The kernel is further separated into a series of interfaces and device drivers
- The kernel provides the file system, CPU scheduling, memory management, and other operating-system functions through system calls.
- This monolithic structure was difficult to implement and maintain.



Layered approach:

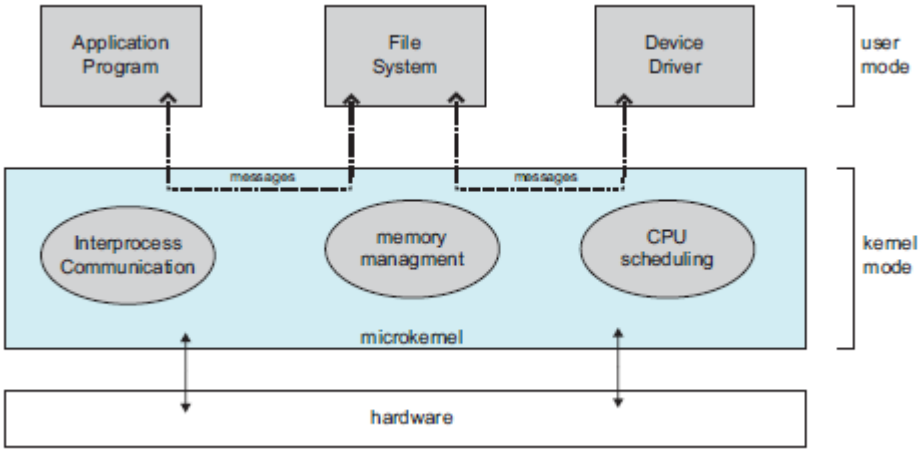
- A system can be made modular in many ways. One method is the **layered approach**, in which the operating system is broken into a number of layers (levels). The bottom layer (layer 0) is the hardware; the highest (layer N) is the user interface.



- An operating-system layer is an implementation of an abstract object made up of data and the operations that can manipulate those data.
- The main advantage of the layered approach is simplicity of construction and debugging. The layers are selected so that each uses functions (operations) and services of only lower-level layers.
- Each layer is implemented only with operations provided by lower-level layers. A layer does not need to know how these operations are implemented; it needs to know only what these operations do.
- The major difficulty with the layered approach involves appropriately defining the various layers because a layer can use only lower-level layers.
- A problem with layered implementations is that they tend to be less efficient than other types.

Microkernels:

- In the mid-1980s, researchers at Carnegie Mellon University developed an operating system called **Mach** that modularized the kernel using the **microkernel** approach.
- This method structures the operating system by removing all nonessential components from the kernel and implementing them as system and user-level programs.

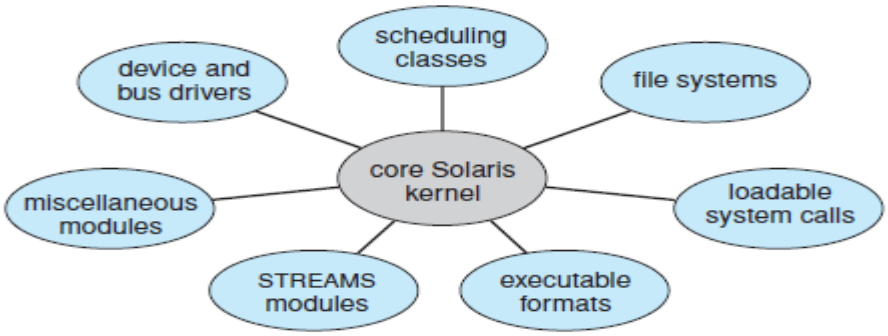


- Microkernel provide minimal process and memory management, in addition to a communication facility.
- The main function of the microkernel is to provide communication between the client program and the various services that are also running in user space.
- The client program and service never interact directly. Rather, they communicate indirectly by exchanging messages with the microkernel.
- One benefit of the microkernel approach is that it makes extending the operating system easier. All new services are added to user space and consequently do not require modification of the kernel.
- The performance of microkernel can suffer due to increased system-function overhead.

Modules:

- The best current methodology for operating-system design involves using **loadable kernel modules**
- The kernel has a set of core components and links in additional services via modules, either at boot time or during run time.
- The kernel provides core services while other services are implemented dynamically, as the kernel is running.
- Linking services dynamically is more comfortable than adding new features directly to the kernel, which would require recompiling the kernel every time a change was made.

Example: Solaris OS



- The Solaris operating system structure is organized around a core kernel with seven types of loadable kernel modules:
    - Scheduling classes
    - File systems
    - Loadable system calls
    - Executable formats
    - STREAMS modules
    - Miscellaneous
    - Device and bus drivers

Hybrid Systems:

- The Operating System combines different structures, resulting in hybrid systems that address performance, security, and usability issues.
- They are monolithic, because having the operating system in a single address space provides very efficient performance.
- However, they are also modular, so that new functionality can be dynamically added to the kernel.
- Example: Linux and Solaris are monolithic (simple) and also modular, IOS.
- Apple IOS Structure