Dayananda Sagar Academy of Technology and Management
(Affiliated to VTU, Belagavi & Approved by AICTE, New Delhi)
OPP. ART OF LIVING, UDAYAPURA, KANAKAPURA Road, BENGALURU- 560082

Department of Computer Science & Engineering

Operating Systems

(21CSE34)

OPERATING SYSTEM STRUCTURES

Dr. Nagaraj M. Lutimath
Associate Professor
Dept. of CS&E

Contents

- Operating System Design and Implementation
- Implementation
- Operating System Structure
- Traditional Unix Structure
- Microkernel
- Layered Structure
- Module Structure

Operating System Design and Implementation

- Design and Implementation of OS not “solvable”, but some approaches have proven successful.
- Internal structure of different Operating Systems can vary widely.
- Start the design by defining goals and specifications .
- Highest level: affected by choice of hardware, type of system.
- The requirements can be divided into User and System goals,
  - User goals – operating system should be convenient to use, easy to learn, reliable, safe, and fast.
  - System goals – operating system should be easy to design, implement, and maintain, as well as flexible, reliable, error-free, and efficient.

Operating System Design and Implementation

- Important principle to separate
- Policy: What will be done?
- Mechanism: How to do it?
- Mechanisms determine how to do something, policies decide what will be done
- The separation of policy from mechanism is a very important principle, it allows maximum flexibility if policy decisions are to be changed later (example – timer).
- Specifying and designing an OS is highly creative task of software engineering

Implementation

- Much variation.
- Early OSes in assembly language.
- Then system programming languages like Algol, PL/1.
- Now C, C++.
- Actually usually a mix of languages.
- Lowest levels in assembly.
- Main body in C.
- Systems programs in C, C++, scripting languages like PERL, Python, shell scripts.
- More high-level language easier to port to other hardware.
- But slower.
- Emulation can allow an OS to run on non-native hardware

Operating System Structure

- General-purpose OS is very large program.
- Various ways to structure ones.
- Simple structure – MS-DOS.
- More complex – UNIX.
- Layered – an abstraction.
- Microkernel –Mach
- Module Structure

Operating System Structure

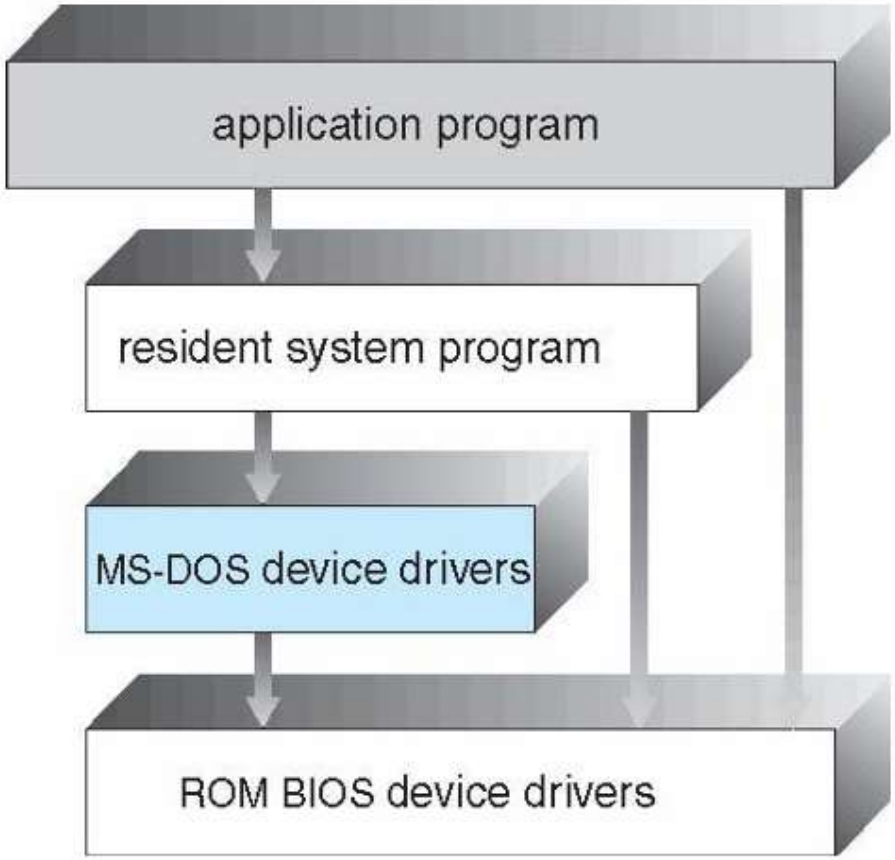

Fig 1: MS-DOS Layer Structure

Traditional Unix Structure

- UNIX – limited by hardware functionality, the original UNIX operating system had limited structuring.
- The UNIX OS consists of two separable parts .
- Systems programs The kernel
  - Consists of everything below the system-call interface and above the physical hardware
  - Provides the file system, CPU scheduling, memory management, and other operating-system functions; a large number of functions for one level

Traditional Unix Structure

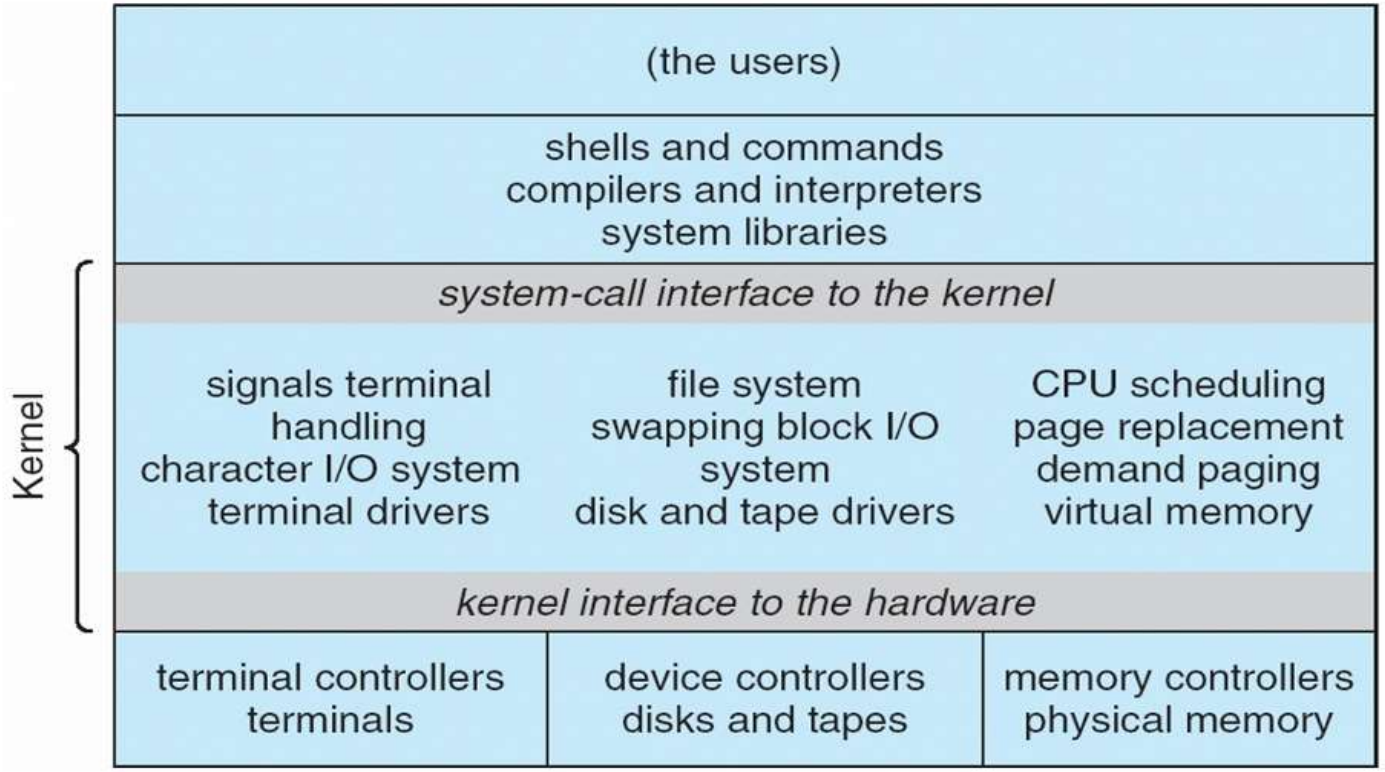

Fig 2: Traditional Unix Structure

Layered Structure

- The operating system is divided into a number of layers (levels), each built on top of lower layers. The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.
- With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers
- Simplifies debugging and system verification

Layered Structure

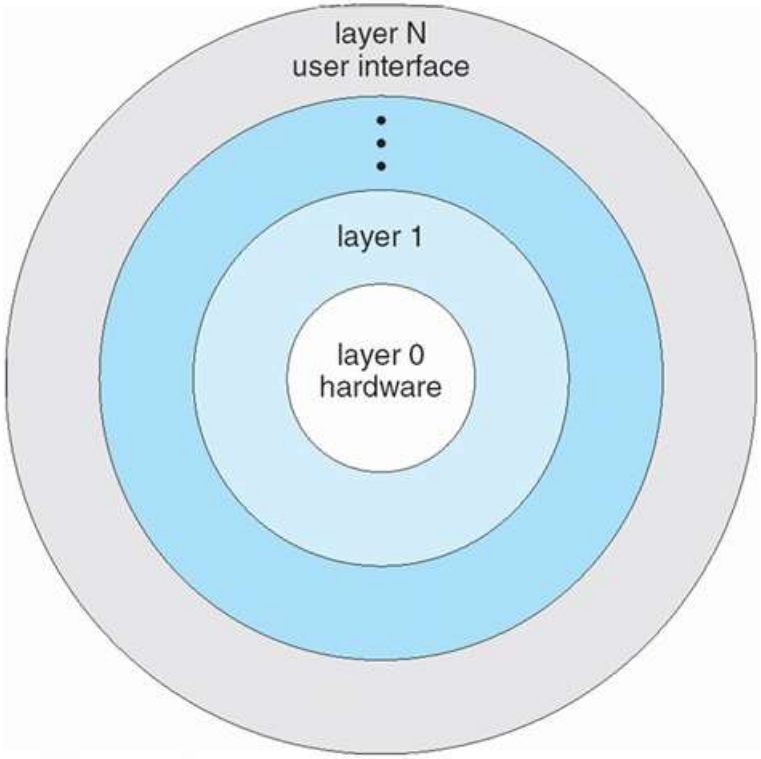

Fig 3: Layered Structure

Microkernel

- Moves as much from the kernel into user space Mach example of microkernel
- Mac OS X kernel (Darwin) partly based on Mach
- Communication takes place between user modules using message passing
- Benefits:
- Easier to extend a microkernel
- Easier to port the operating system to new architectures
- More reliable (less code is running in kernel mode)
- More secure
- Detriments:
- Performance overhead of user space to kernel space communication

Microkernel

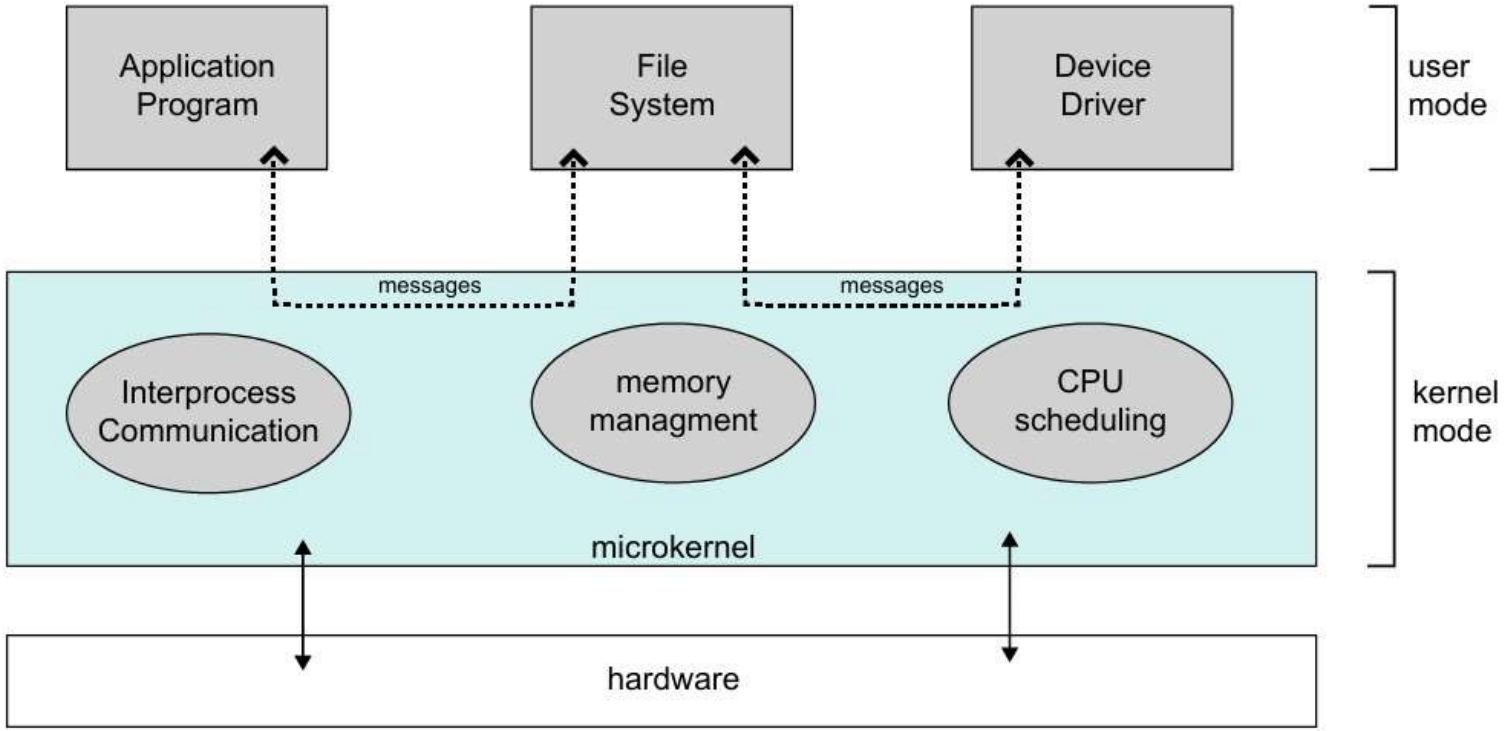

Fig 4: Microkernel

Module Structure

- Many modern operating systems implement loadable kernel modules
- Uses object-oriented approach
- Each core component is separate
- Each talks to the others over known interfaces
- Each is loadable as needed within the kernel
- Overall, similar to layers but with more flexible Linux, Solaris, etc

Module Structure

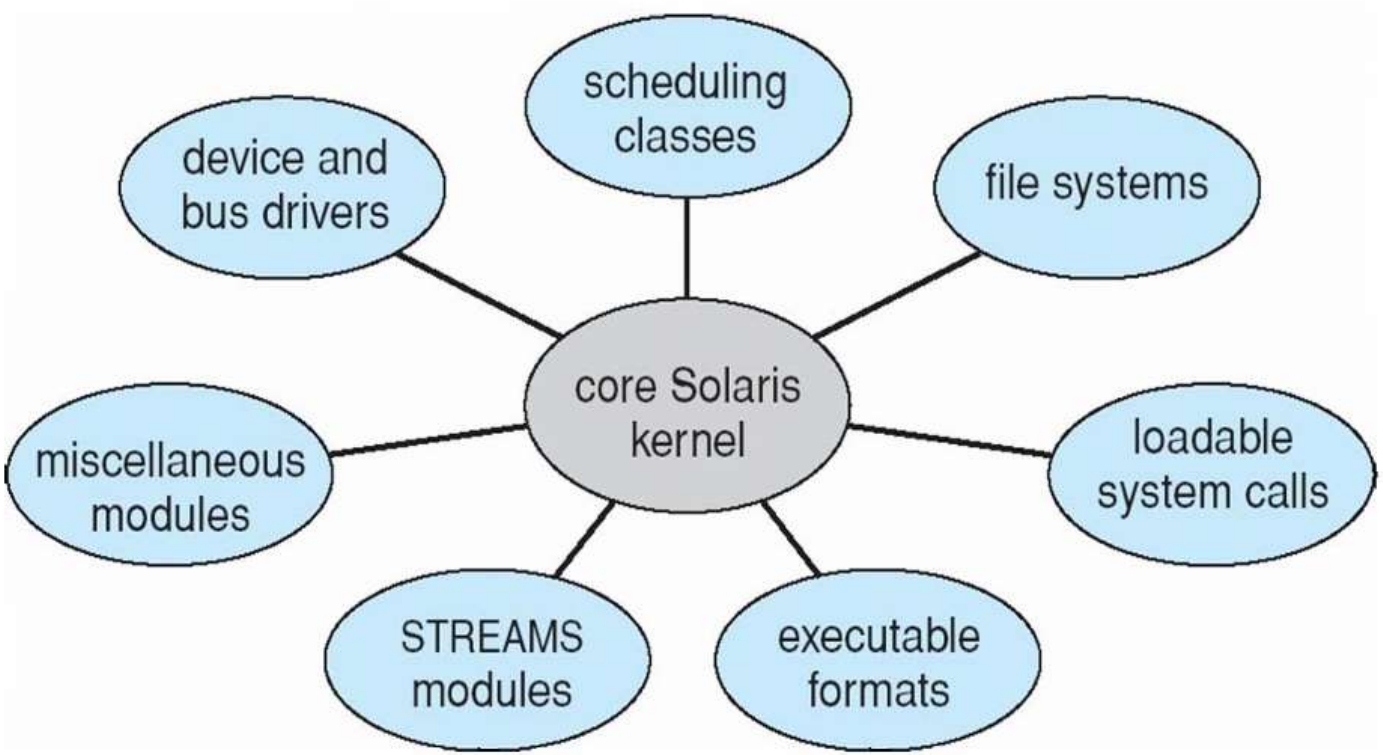

Fig 5: Module Structure