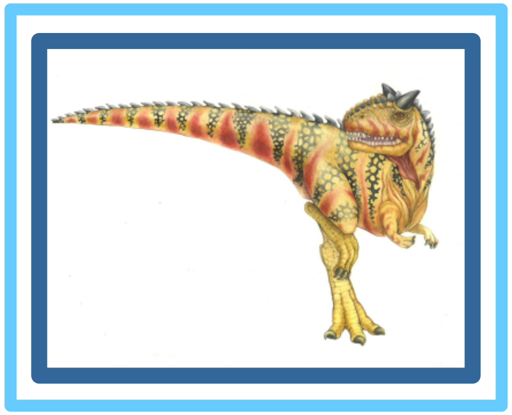
Chapter 2: Operating-System Structures

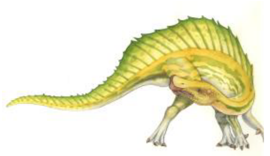


Review

- OS services
- OS interfaces
- System calls

Next Class

- System Programs
- OS Design and Implementation
- OS Structure
- OS Debugging



System Programs

- System programs provide a convenient environment for program development and execution. They can be divided into:
  - **File manipulation**
  - **Status information sometimes stored in a File modification**
  - **Programming language support**
  - **Program loading and execution**
  - **Communications**
  - **Application programs**
- Most users' view of the operation system is defined by system programs, not the actual system calls

System Programs

- Provide a convenient environment for program development and execution
  - Some of them are simply **user interfaces to system calls**; others are considerably **more complex**
- **File management** - Create, delete, copy, rename, print, dump, list, and generally manipulate files and directories
- **Status information**
  - Some ask the system for info - **date, time, amount of available memory, disk space, number of users**
  - Others provide detailed performance, logging, and debugging information
  - Typically, these programs format and print the output to the terminal or other output devices
  - Some systems implement a **registry** - used to store and retrieve configuration information

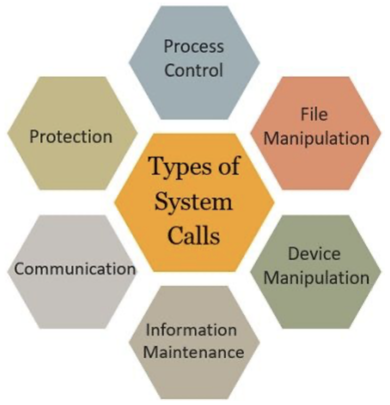
System Programs (Cont.)

- **File modification**
  - Text editors to create and modify files
  - Special commands to search contents of files or perform transformations of the text
- **Programming-language support** - Compilers, assemblers, debuggers and interpreters sometimes provided
- **Program loading and execution**
- **Communications** - Provide the mechanism for creating virtual connections among processes, users, and computer systems
- **Application programs**
  - Don’t pertain to system
  - Run by users
  - Not typically considered part of OS
  - Launched by command line, mouse click, finger poke

System Calls Vs System Programs

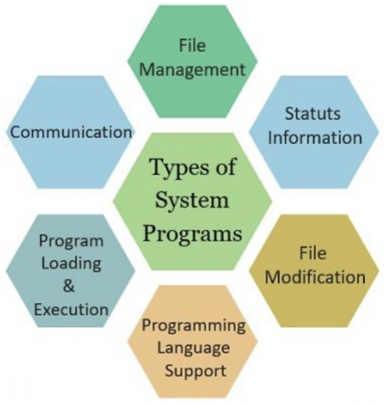
System Calls

- Allow user process to request the services of OS
- Defines interface to the services of OS
- It satisfies the low-level request of user program.



System Programs

- Creates an environment for program to develop and execute.
- Defines a user interface of operating system.
- It satisfies the high-level request of the user program.



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
- Affected by choice of hardware, type of system
- The requirements are of two types
  - **User** goals and **System** goals

Operating System Design and Implementation (Cont.)

- **Policy**: ***What*** will be done?
- **Mechanism**: ***How*** to do it?

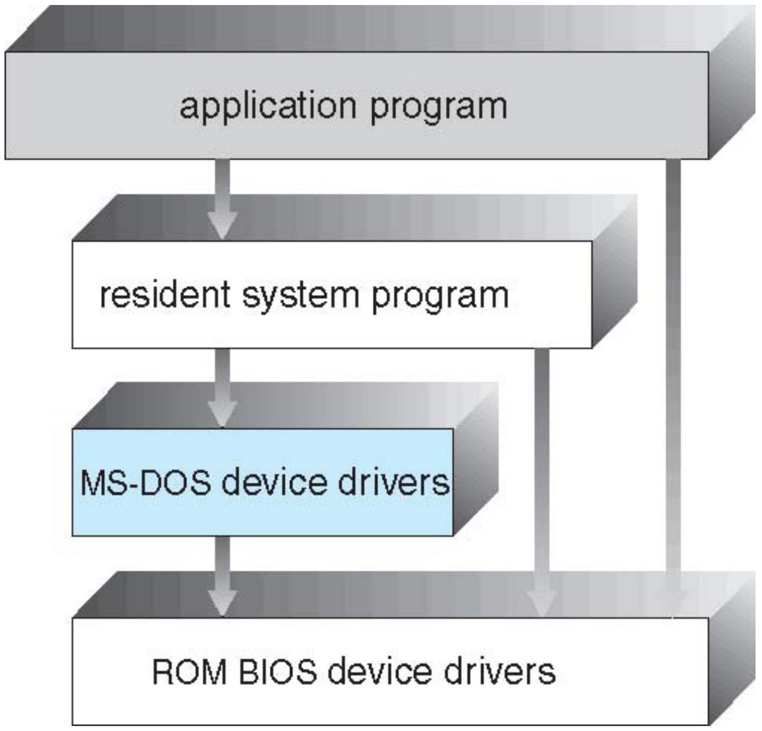
- The separation of policy from mechanism allows maximum flexibility
- Specifying and designing an OS is highly creative task of **software engineering**
- Actually usually a mix of languages
  - Lowest levels in assembly
  - Main body in C
  - Systems programs in C, C++, scripting languages like PERL, Python, shell scripts

Operating System Structure

- General-purpose OS is very large program
- Types of structures are,
  - Simple — MS-DOS
  - Monolithic — UNIX
  - Layered – an abstraction
  - Microkernel — Mach
  - Modules

Simple Structure -- MS-DOS

- MS-DOS – written to provide the most functionality in the least space
  - Not divided into modules
  - Although MS-DOS has some structure, its interfaces and levels of functionality are not well separated

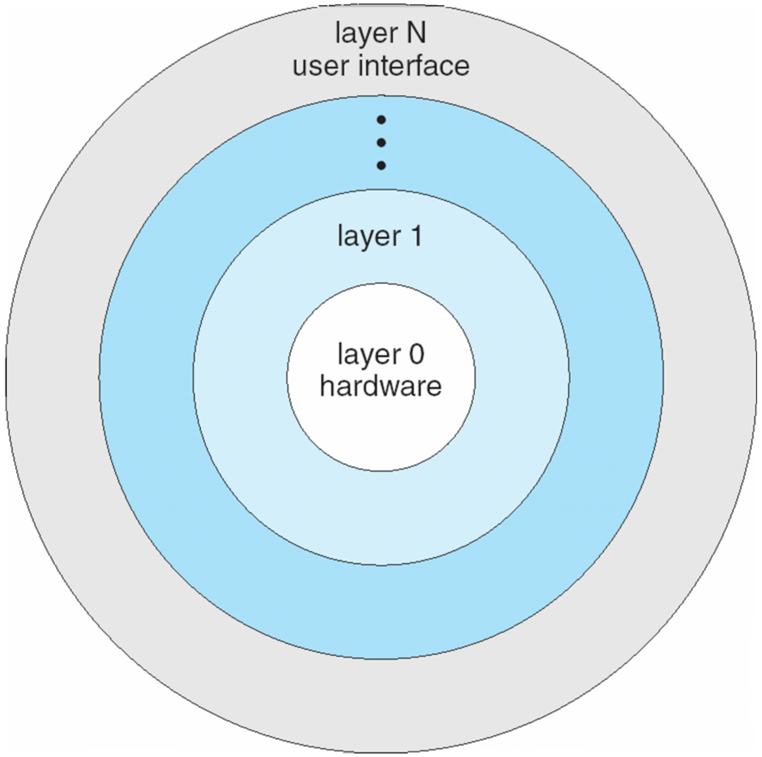


Monolithic Structure -- UNIX

- 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

Layered Approach

- 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



Task

- A user is using the windows interface with a command-line UNIX Shell. The user has to navigate to a certain folder in a particular directory.

- What are the system calls and system programs in this case?

Task

- A user is using the windows interface with a command-line UNIX Shell. The user has to navigate to a certain folder in a particular directory.

- What are the system calls and system programs in this case?

- System Calls — get the folder name, locate the folder folder, move to that folder
- System programs — Either by using the mouse (because it is a windows interface) or by using the command-line UNIX Shell.