Operating System Structures

Chapter 2

Different Perspectives

Three facets

- Operating systems
  - Services
  - Interface
  - Components
- “An operating system provides an environment for the execution of programs.”

Services

An introduction

- User interface
- Program execution
- I/O operations
- File system services
- Communications
- Resource allocation
- Accounting
- Protection and security
- Error detection

User Interface

Computer controls for humans

- Command line interface (CLI)
  - Command interpreter contains command code
  - Command interpreter executes a system command
  - Multiple, selectable CLI often called shells
- Graphical user interface (GUI)
- Batch

Early Object-Oriented History

The “salad days”

- 1965
  - Simula programming language
  - Kristen Nygaard and Ole-Johan Dahl in Norway
- 1966-1969
  - Alan Kay is a PhD student at the University of Utah
  - Early work in object-orientation
- Early 1970s
  - Alan Kay is a leader at Xerox Palo Alto Research Center (PARC)
  - Kay becomes known as the “Father of the personal computer”
    - Conceived the Dynabook, which became the Alto workstation, which became the Apple Macintosh
    - Created Smalltalk – coined many contemporary object-oriented terms
    - Pioneered the use of icons as keys to actions

System Calls

“Standardized system calls”

- Interface to operating system services
  - C, C++, assembly
- APIs
  - Wrap system calls, often object-oriented
  - POSIX
  - Java
  - Win32

return value

BOOL ReadFile c HANDLE file, LPVOID buffer, DWORD bytes To Read, LPDWORD bytesRead, lpVOID lpOverlapped;
Types of System Calls

- Process control
- File management
- Device management
- UI with OS / simple information
- Interprocess communication (IPC)
  - Message passing
  - Shared-memory

System Call Example

Mechanisms and Policies

- Mechanism
  - How to do something
- Policy
  - What will be done
- Separating mechanism from policy allows for a flexible, configurable system
- Policies often kept in tables or files, which are read by the implementing mechanism

Ad Hoc Architecture

- Evolve over time
- Little design, no modularity
- Interface and functionality not well separated
- Little protection or security
- Difficult to maintain, update, validate, debug, and understand

Layered Architecture

- Improved control
- Easier to maintain/modify
  - Debugging and validation
  - Easier to understand
- Layers only use services of lower layers
- Difficult to order layers
- Less efficient / too slow

Microkernel Architecture

- A minimalist approach
- Nonessential parts removed from kernel
  - Provided as programs running in user space
  - No consensus about kernel services
- Message communication
  - User application to kernel
  - Kernel to service program
- Improved security
- Easily extended
- High overhead / low performance
Module Architecture

- Modularized & minimalist approach – best current practice
- Based on object-oriented paradigm
- Kernel provides core services, dynamically links additional services
- Benefits of layered and microkernel but w/o drawbacks

Virtual Machine Architecture

- A “smoke and mirrors” approach
- Simulate hardware in software
- Hard to implement
- Powerful
  - Debugging, tracing, profiling, verifying
  - Machines isolated
- Typically slower

Popular Virtual Machines

- More “smoke and mirrors”
- Compilers produce architecture-neutral virtual machine code
  - Common language runtime
  - Java virtual machine

System Boot

- Pulling itself up by the bootstraps
- Simple OS in ROM
  - Expensive
  - Slower than RAM
- Simple loader in ROM
- Loads OS or more sophisticated loader
  - Loader stored in boot block/sector
  - Loads OS or final loader