Operating System Overview

Chapter 2

Operating System

- A program that controls the execution of application programs
- An interface between applications and hardware

Operating System Objectives

- Convenience
  - Makes the computer more convenient to use
- Efficiency
  - Allows computer system resources to be used in an efficient manner
- Ability to evolve
  - Permit effective development, testing, and introduction of new system functions without interfering with service

Layers of Computer System

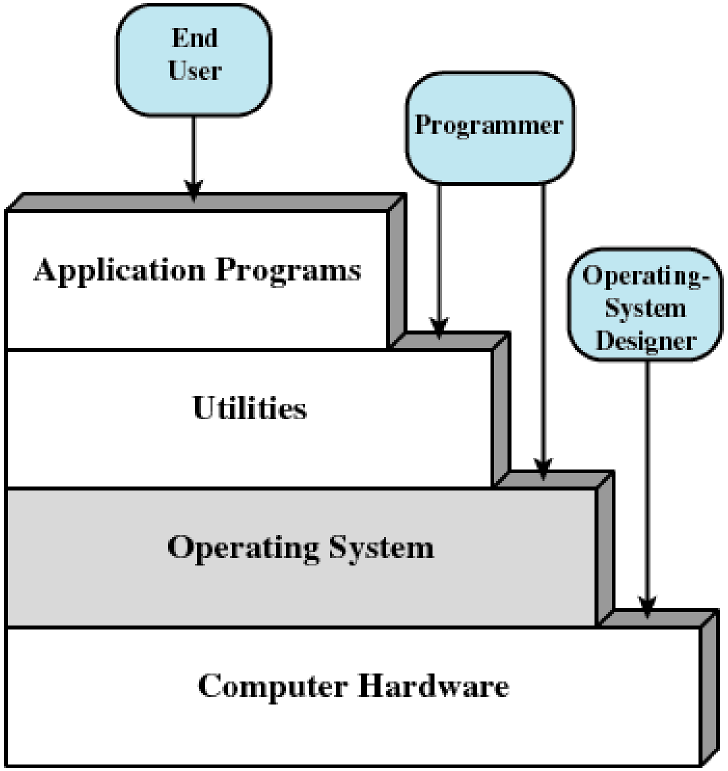


Figure 2.1 Layers and Views of a Computer System

Services Provided by the Operating System

- Program development
  - Editors and debuggers
- Program execution
- Access to I/O devices
- Controlled access to files
- System access

Services Provided by the Operating System

- Error detection and response
  - Internal and external hardware errors
    - Memory error
    - Device failure
  - Software errors
    - Arithmetic overflow
    - Access forbidden memory locations
  - Operating system cannot grant request of application

Services Provided by the Operating System

- Accounting
  - Collect usage statistics
  - Monitor performance
  - Used to anticipate future enhancements
  - Used for billing purposes

Operating System

- Responsible for managing resources
- Functions same way as ordinary computer software
  - It is program that is executed
- Operating system relinquishes control of the processor

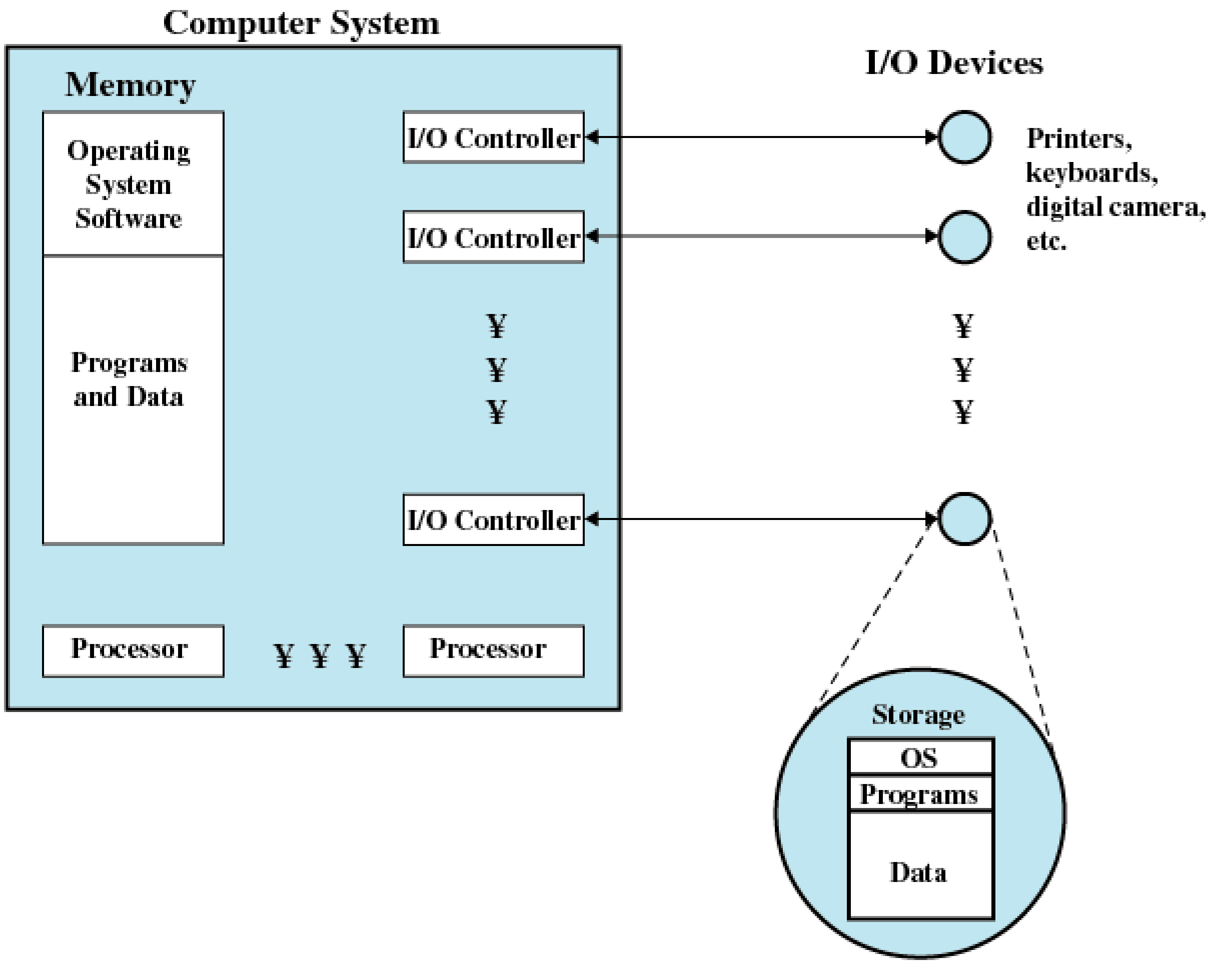


Figure 2.2 The Operating System as Resource Manager

Kernel

- Portion of operating system that is in main memory
- Contains most frequently used functions
- Also called the nucleus

Evolution of an Operating System

- Hardware upgrades plus new types of hardware
- New services
- Fixes

Evolution of Operating Systems

- Serial Processing
  - No operating system
  - Machines run from a console with display lights, toggle switches, input device, and printer
  - Schedule time
  - Setup included loading the compiler, source program, saving compiled program, and loading and linking

Evolution of Operating Systems

- Simple Batch Systems
  - Monitors
    - Software that controls the sequence of events
    - Batch jobs together
    - Program branches back to monitor when finished

Job Control Language (JCL)

- Special type of programming language
- Provides instruction to the monitor
  - What compiler to use
  - What data to use

Hardware Features

- Memory protection
  - Do not allow the memory area containing the monitor to be altered
- Timer
  - Prevents a job from monopolizing the system

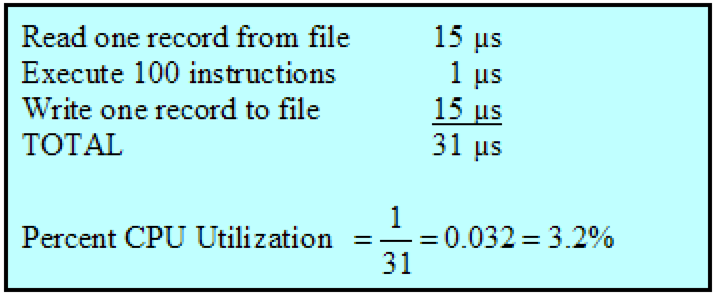
Hardware Features

- Privileged instructions
  - Certain machine level instructions can only be executed by the monitor
- Interrupts
  - Early computer models did not have this capability

Memory Protection

- User program executes in user mode
  - Certain instructions may not be executed
- Monitor executes in system mode
  - Kernel mode
  - Privileged instructions are executed
  - Protected areas of memory may be accessed

I/O Devices Slow



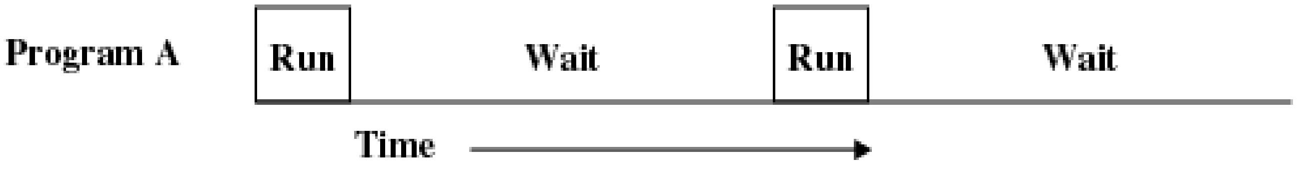
Read one record from file	15 µs
Execute 100 instructions	1 µs
Write one record to file	15 µs
TOTAL	31 µs

$$\text{Percent CPU Utilization } = \frac{1}{31} = 0.032 = 3.2\%$$

Figure 2.4 System Utilization Example

Uniprogramming

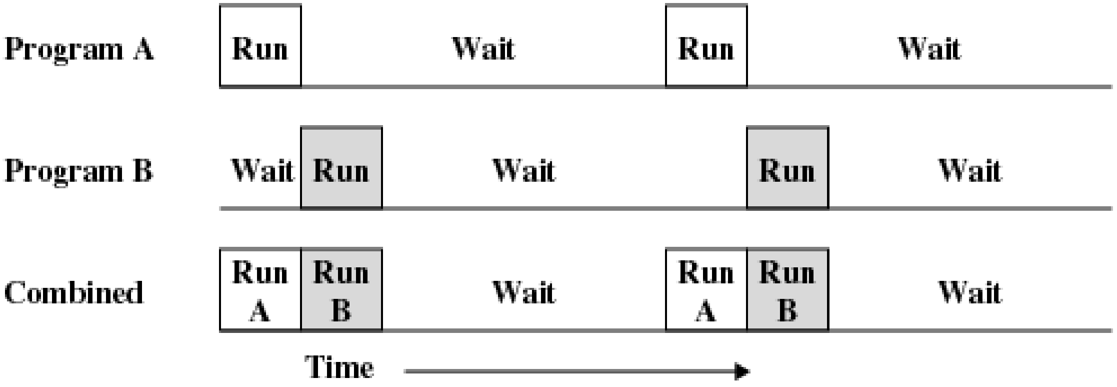
- Processor must wait for I/O instruction to complete before preceding



(a) Uniprogramming

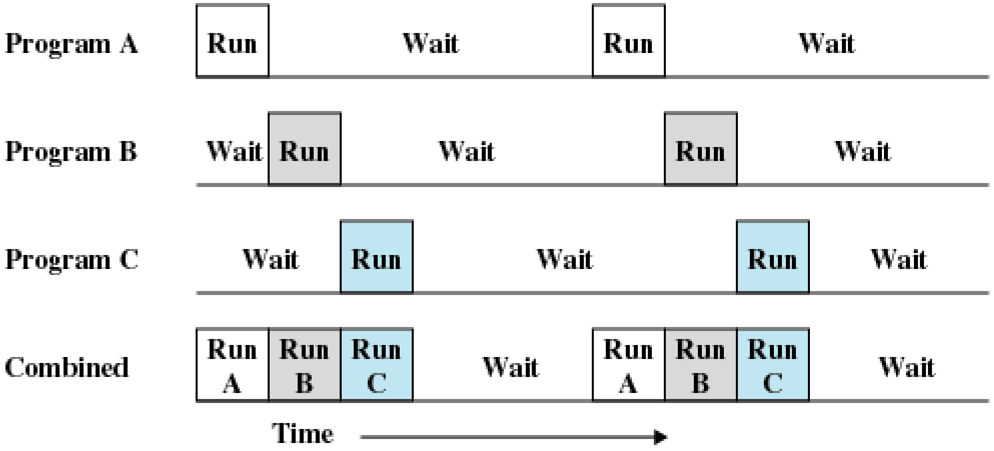
Multiprogramming

- When one job needs to wait for I/O, the processor can switch to the other job



(b) Multiprogramming with two programs

Multiprogramming



(c) Multiprogramming with three programs

Utilization Histograms

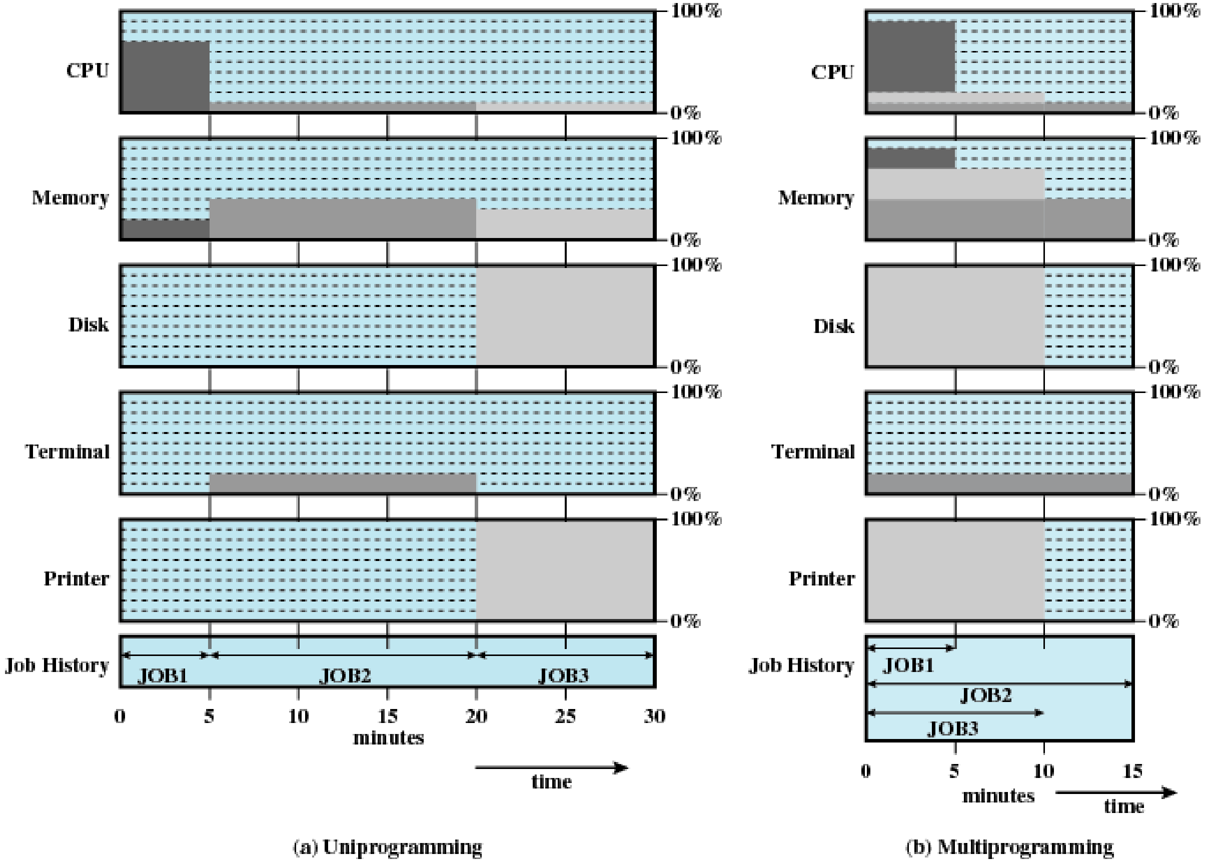


Figure 2.6 Utilization Histograms

Example

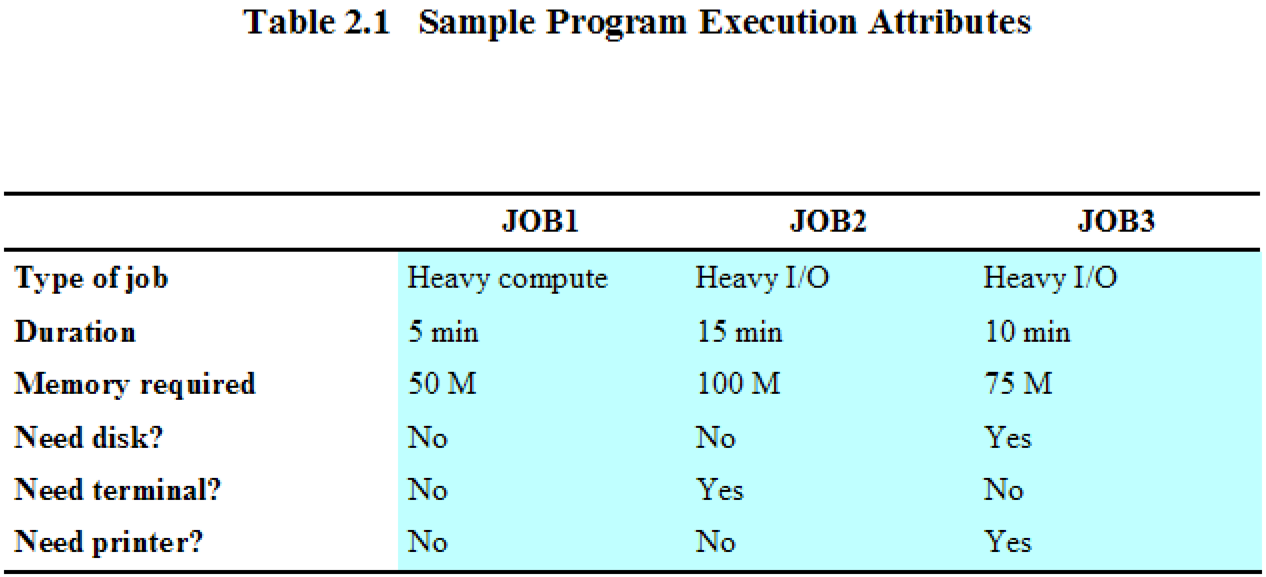


Table 2.1 Sample Program Execution Attributes

	JOB1	JOB2	JOB3
Type of job	Heavy compute	Heavy I/O	Heavy I/O
Duration	5 min	15 min	10 min
Memory required	50 M	100 M	75 M
Need disk?	No	No	Yes
Need terminal?	No	Yes	No
Need printer?	No	No	Yes

Time Sharing

- Using multiprogramming to handle multiple interactive jobs
- Processor’s time is shared among multiple users
- Multiple users simultaneously access the system through terminals

Compatible Time-Sharing System (CTSS)

- First time-sharing system developed at MIT

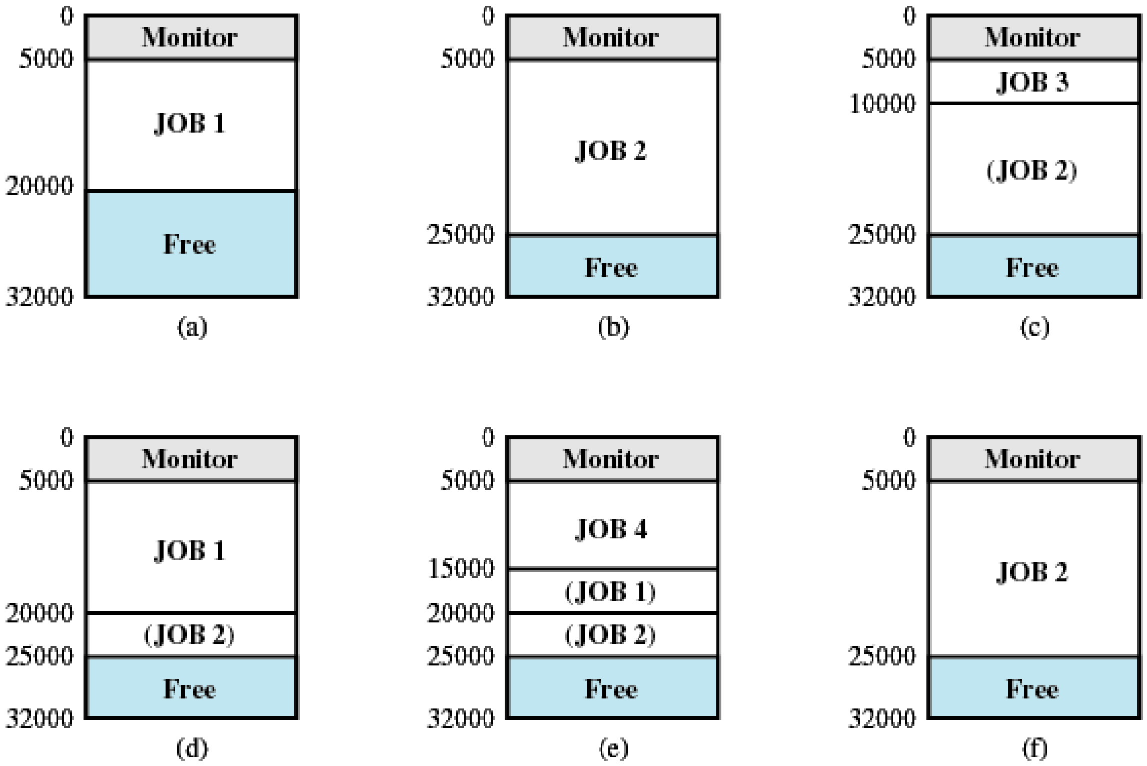


Figure 2.7 CTSS Operation

Major Achievements

- Processes
- Memory Management
- Information protection and security
- Scheduling and resource management
- System structure

Processes

- A program in execution
- An instance of a program running on a computer
- The entity that can be assigned to and executed on a processor
- A unit of activity characterized by a single sequential thread of execution, a current state, and an associated set of system resources

Difficulties with Designing System Software

- Improper synchronization
  - Ensure a process waiting for an I/O device receives the signal
- Failed mutual exclusion
- Nondeterminate program operation
  - Program should only depend on input to it, not on the activities of other programs
- Deadlocks

Process

- Consists of three components
  - An executable program
  - Associated data needed by the program
  - Execution context of the program
    - All information the operating system needs to manage the process

Process

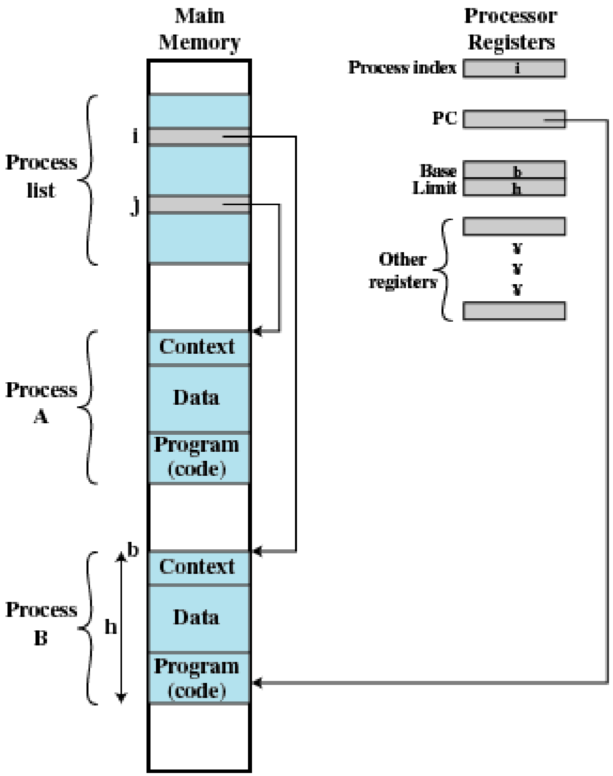


Figure 2.8 Typical Process Implementation

Memory Management

- Process isolation
- Automatic allocation and management
- Support of modular programming
- Protection and access control
- Long-term storage

Virtual Memory

- Allows programmers to address memory from a logical point of view
- No hiatus between the execution of successive processes while one process was written out to secondary store and the successor proceeess was read in

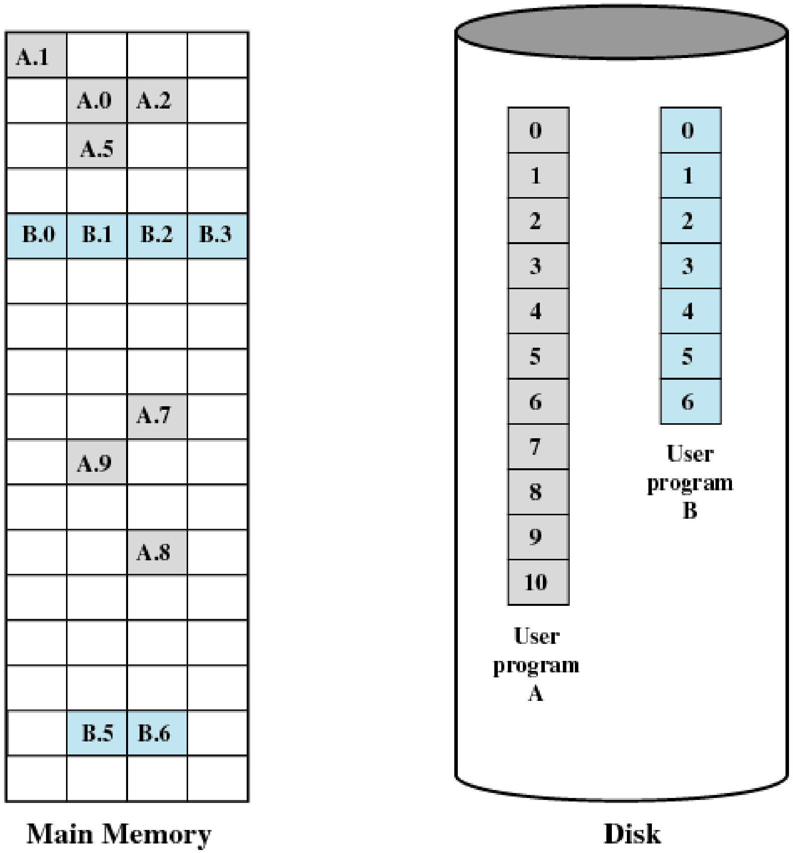
Virtual Memory and File System

- Implements long-term store
- Information stored in named objects called files

Paging

- Allows process to be comprised of a number of fixed-size blocks, called pages
- Virtual address is a page number and an offset within the page
- Each page may be located any where in main memory
- Real address or physical address in main memory

Virtual Memory



Main memory consists of a number of fixed-length frames, each equal to the size of a page. For a program to execute, some or all of its pages must be in main memory.

Secondary memory (disk) can hold many fixed-length pages. A user program consists of some number of pages. Pages for all programs plus the operating system are on disk, as are files.

Figure 2.9 Virtual Memory Concepts

Virtual Memory Addressing

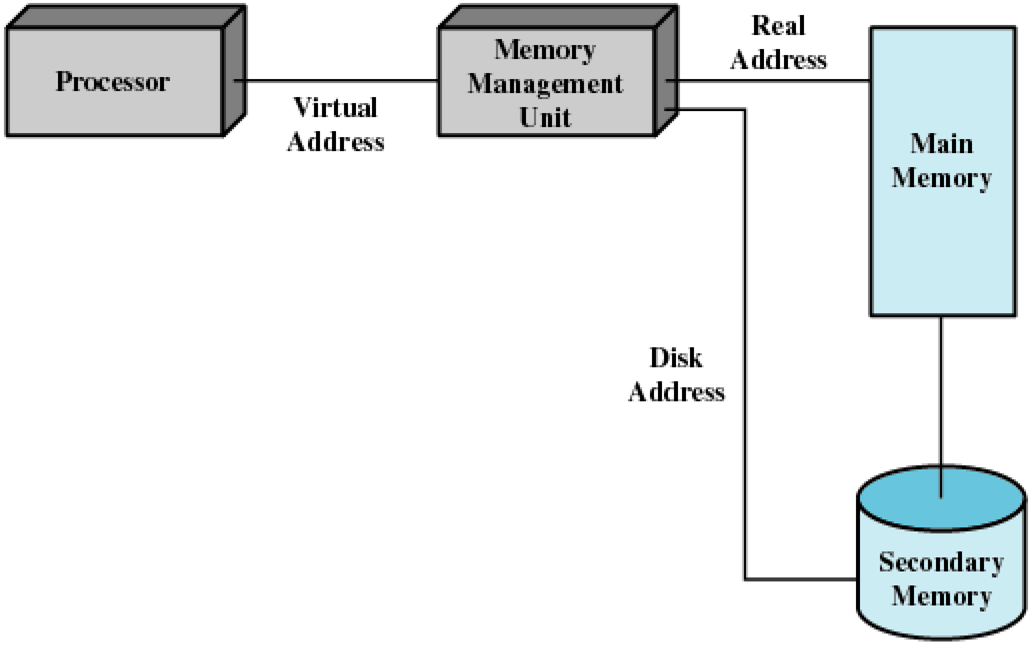


Figure 2.10 Virtual Memory Addressing

Information Protection and Security

- Availability
  - Concerned with protecting the system against interruption
- Confidentiality
  - Assuring that users cannot read data for which access is unauthorized

Information Protection and Security

- Data integrity
  - Protection of data from unauthorized modification
- Authenticity
  - Concerned with the proper verification of the identity of users and the validity of messages or data

Scheduling and Resource Management

- Fairness
  - Give equal and fair access to resources
- Differential responsiveness
  - Discriminate among different classes of jobs
- Efficiency
  - Maximize throughput, minimize response time, and accommodate as many uses as possible

Key Elements of Operating System

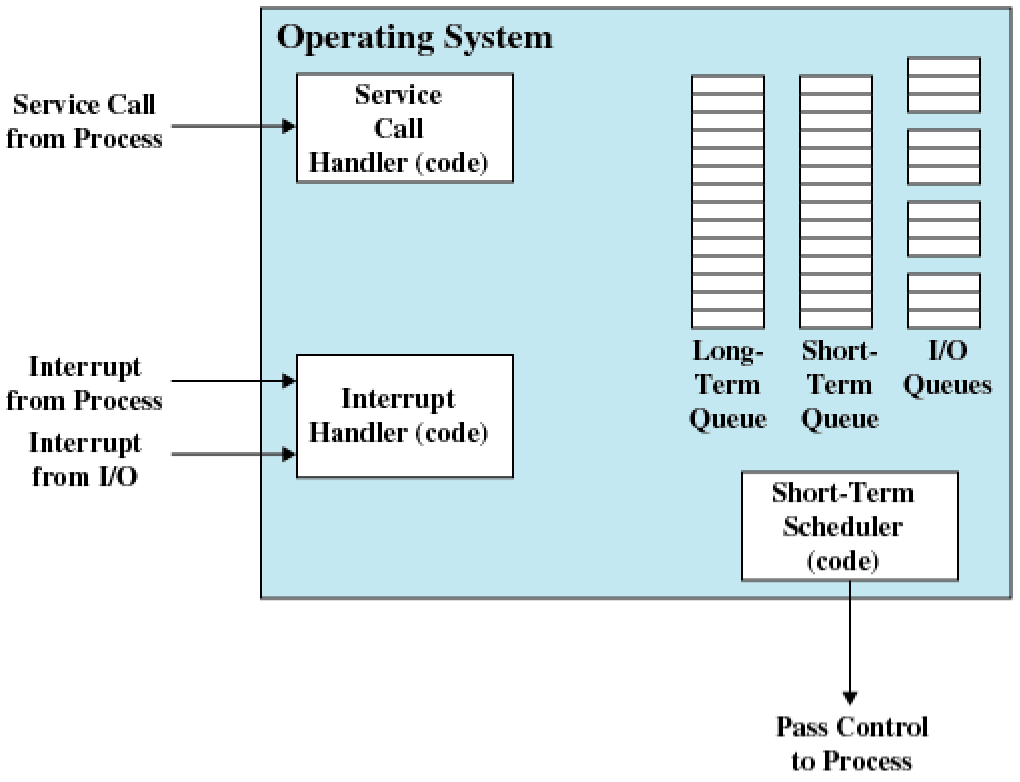


Figure 2.11 Key Elements of an Operating System for Multiprogramming

System Structure

- View the system as a series of levels
- Each level performs a related subset of functions
- Each level relies on the next lower level to perform more primitive functions
- This decomposes a problem into a number of more manageable subproblems

Process Hardware Levels

- Level 1
  - Electronic circuits
  - Objects are registers, memory cells, and logic gates
  - Operations are clearing a register or reading a memory location
- Level 2
  - Processor’s instruction set
  - Operations such as add, subtract, load, and store

Process Hardware Levels

- Level 3
  - Adds the concept of a procedure or subroutine, plus call/return operations
- Level 4
  - Interrupts

Concepts with Multiprogramming

- Level 5
  - Process as a program in execution
  - Suspend and resume processes
- Level 6
  - Secondary storage devices
  - Transfer of blocks of data
- Level 7
  - Creates logical address space for processes
  - Organizes virtual address space into blocks

Deal with External Objects

- Level 8
  - Communication of information and messages between processes
- Level 9
  - Supports long-term storage of named files
- Level 10
  - Provides access to external devices using standardized interfaces

Deal with External Objects

- Level 11
  - Responsible for maintaining the association between the external and internal identifiers
- Level 12
  - Provides full-featured facility for the support of processes
- Level 13
  - Provides an interface to the operating system for the user

Modern Operating Systems

- Microkernel architecture
  - Assigns only a few essential functions to the kernel
    - Address spaces
    - Interprocess communication (IPC)
    - Basic scheduling

Modern Operating Systems

- Multithreading
  - Process is divided into threads that can run concurrently
    - Thread
      - Dispatchable unit of work
      - executes sequentially and is interruptable
    - Process is a collection of one or more threads

Modern Operating Systems

- Symmetric multiprocessing (SMP)
  - There are multiple processors
  - These processors share same main memory and I/O facilities
  - All processors can perform the same functions

Multiprogramming and Multiprocessing

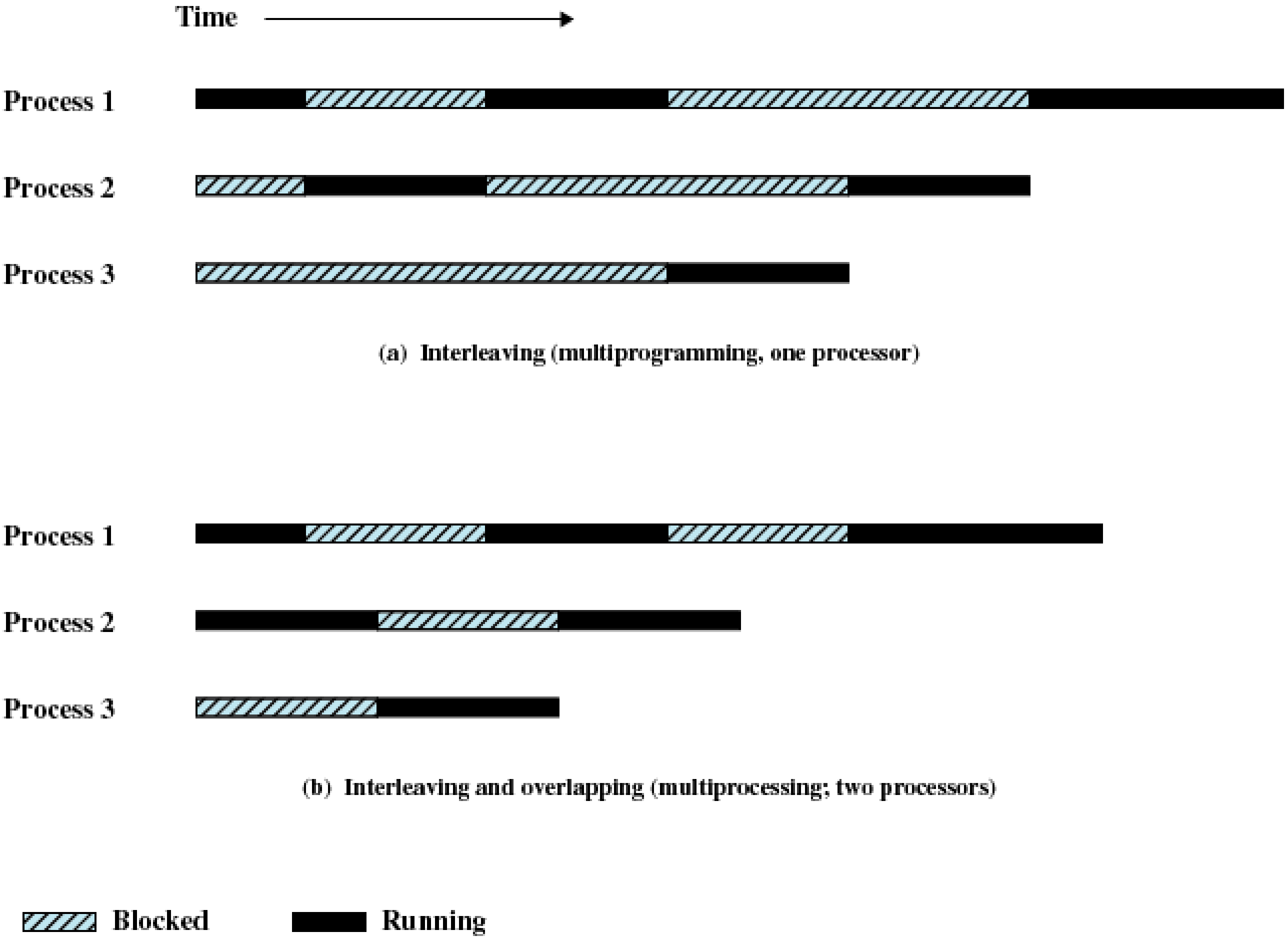

Time
Process 1
Process 2
Process 3
(a) Interleaving (multiprogramming, one processor)
Process 1
Process 2
Process 3
(b) Interleaving and overlapping (multiprocessing; two processors)
Blocked
Running

Figure 2.12 Multiprogramming and Multiprocessing

Modern Operating Systems

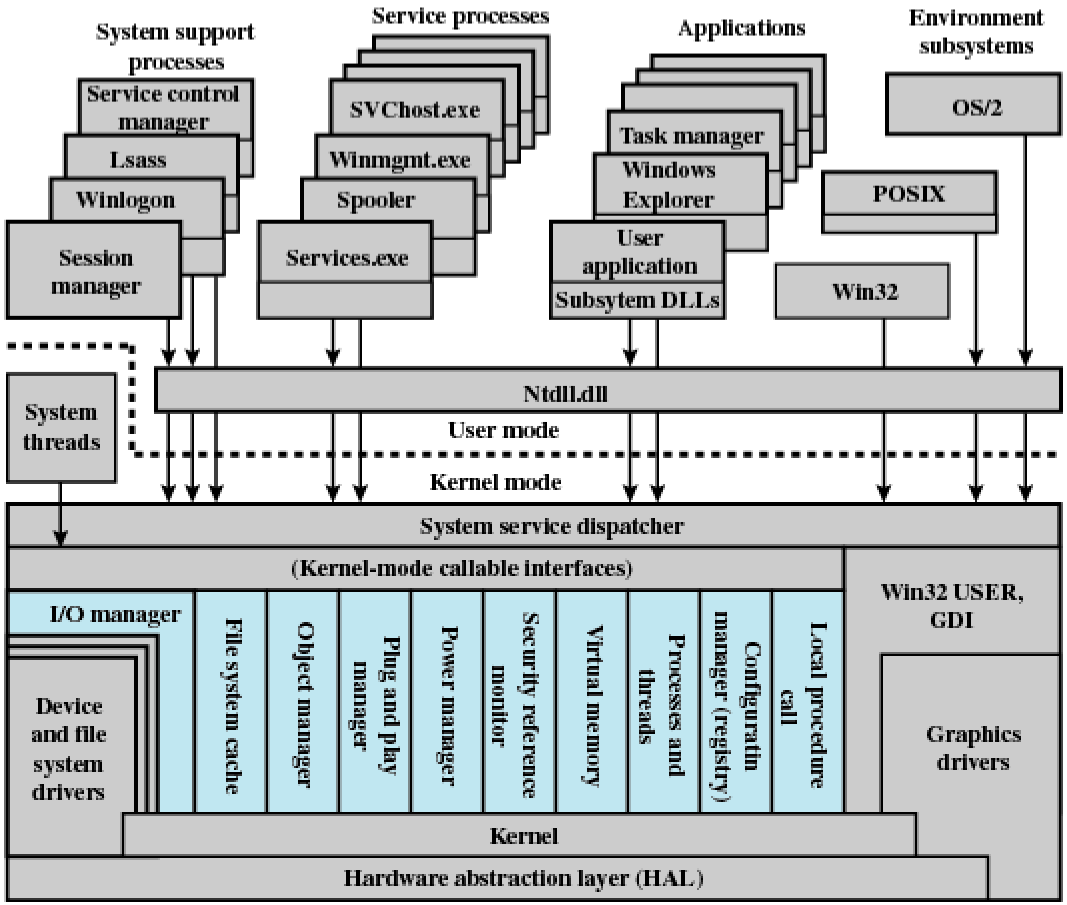
- Distributed operating systems
  - Provides the illusion of a single main memory space and single secondary memory space

Modern Operating Systems

- Object-oriented design
  - Used for adding modular extensions to a small kernel
  - Enables programmers to customize an operating system without disrupting system integrity

Windows Architecture

- Modular structure for flexibility
- Executes on a variety of hardware platforms
- Supports application written for other operating system



Lsass = local security authentication server
POSIX = portable operating system interface
GDI = graphics device interface
DLL = dynamic link libraries

Colored area indicates Executive

Figure 2.13 Windows 2000 Architecture [SOLO00]

Operating System Organization

- Modified microkernel architecture
  - Not a pure microkernel
  - Many system functions outside of the microkernel run in kernel mode
- Any module can be removed, upgraded, or replaced without rewriting the entire system

Kernel-Mode Components

- Executive
  - Contains base operating system services
    - Memory management
    - Process and thread management
    - Security
    - I/O
    - Interprocess communication
- Kernel
  - Consists of the most used components

Kernel-Mode Components

- Hardware abstraction layer (HAL)
  - Isolates the operating system from platform-specific hardware differences
- Device drivers
  - Translate user I/O function calls into specific hardware device I/O requests
- Windowing and graphics systems
  - Implements the graphical user interface (GUI)

Windows Executive

- I/O manager
- Cache manager
- Object manager
- Plug and play manager
- Power manager
- Security reference monitor
- Virtual memory manager
- Process/thread manager
- Configuration manager
- Local procedure call (LPC) facility

User-Mode Processes

- Special system support processes
  - Ex: logon process and the session manager
- Service processes
- Environment subsystems
- User applications

Client/Server Model

- Simplifies the Executive
  - Possible to construct a variety of APIs
- Improves reliability
  - Each service runs on a separate process with its own partition of memory
  - Clients cannot not directly access hardware
- Provides a uniform means for applications to communicate via LPC
- Provides base for distributed computing

Threads and SMP

- Operating system routines can run on any available processor
- Different routines can execute simultaneously on different processors
- Multiple threads of execution within a single process may execute on different processors simultaneously
- Server processes may use multiple threads
- Share data and resources between process

Windows Objects

- Encapsulation
  - Object consists of one or more data items and one or more procedures
- Object class or instance
  - Create specified instances of an object
- Inheritance
  - Support to some extent in the Executive
- Polymorphism

UNIX

- Hardware is surrounded by the operating system software
- Operating system is called the system kernel
- Comes with a number of user services and interfaces
  - Shell
  - Components of the C compiler

UNIX

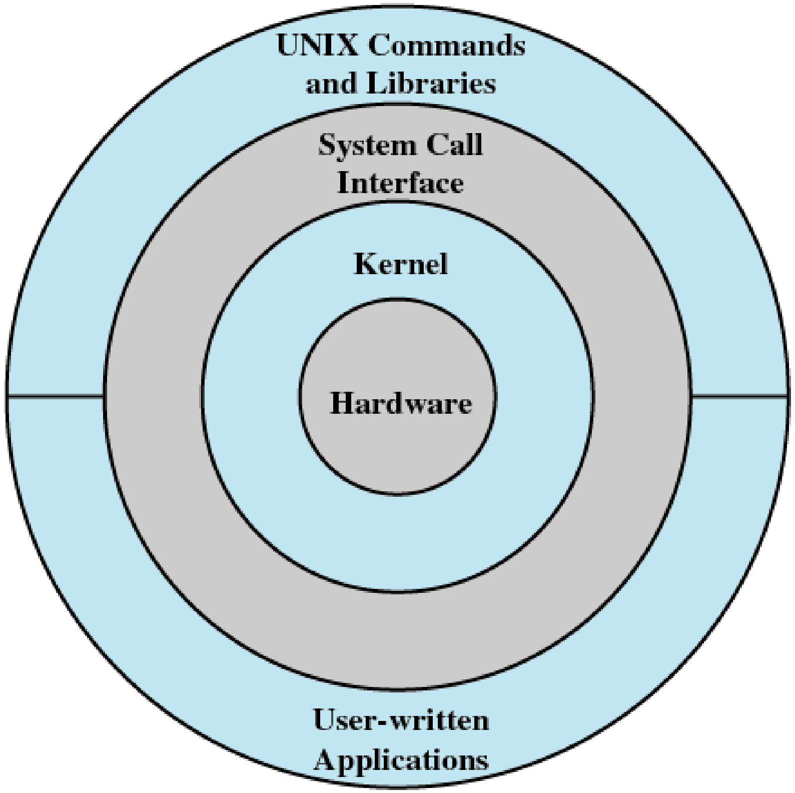


Figure 2.14 General UNIX Architecture

UNIX Kernel

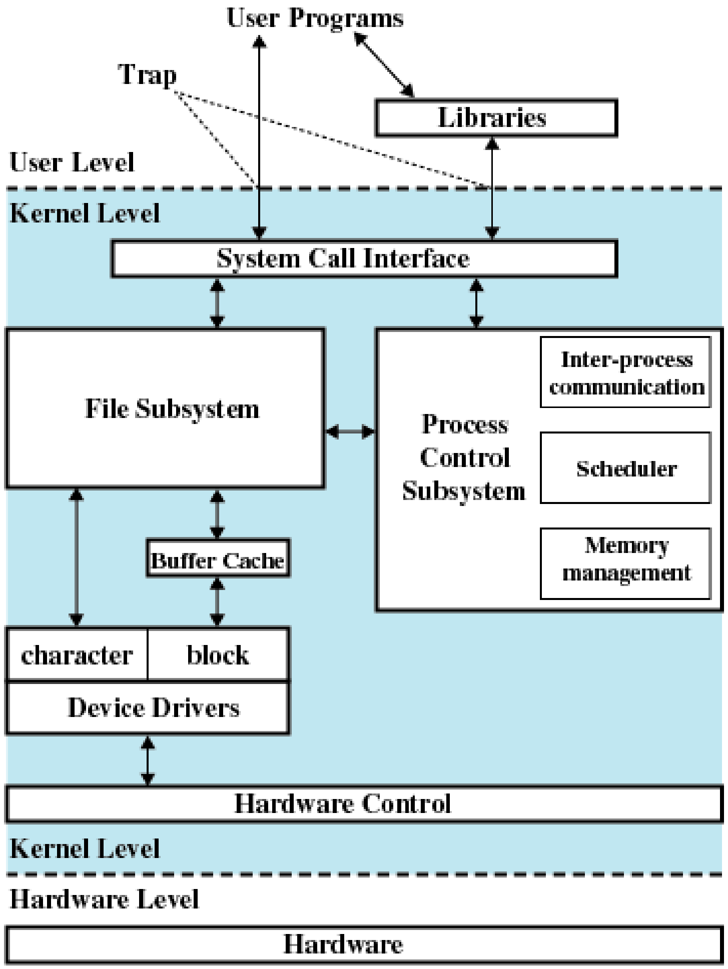


Figure 2.15 Traditional UNIX Kernel [BACH86]

Modern UNIX Kernel

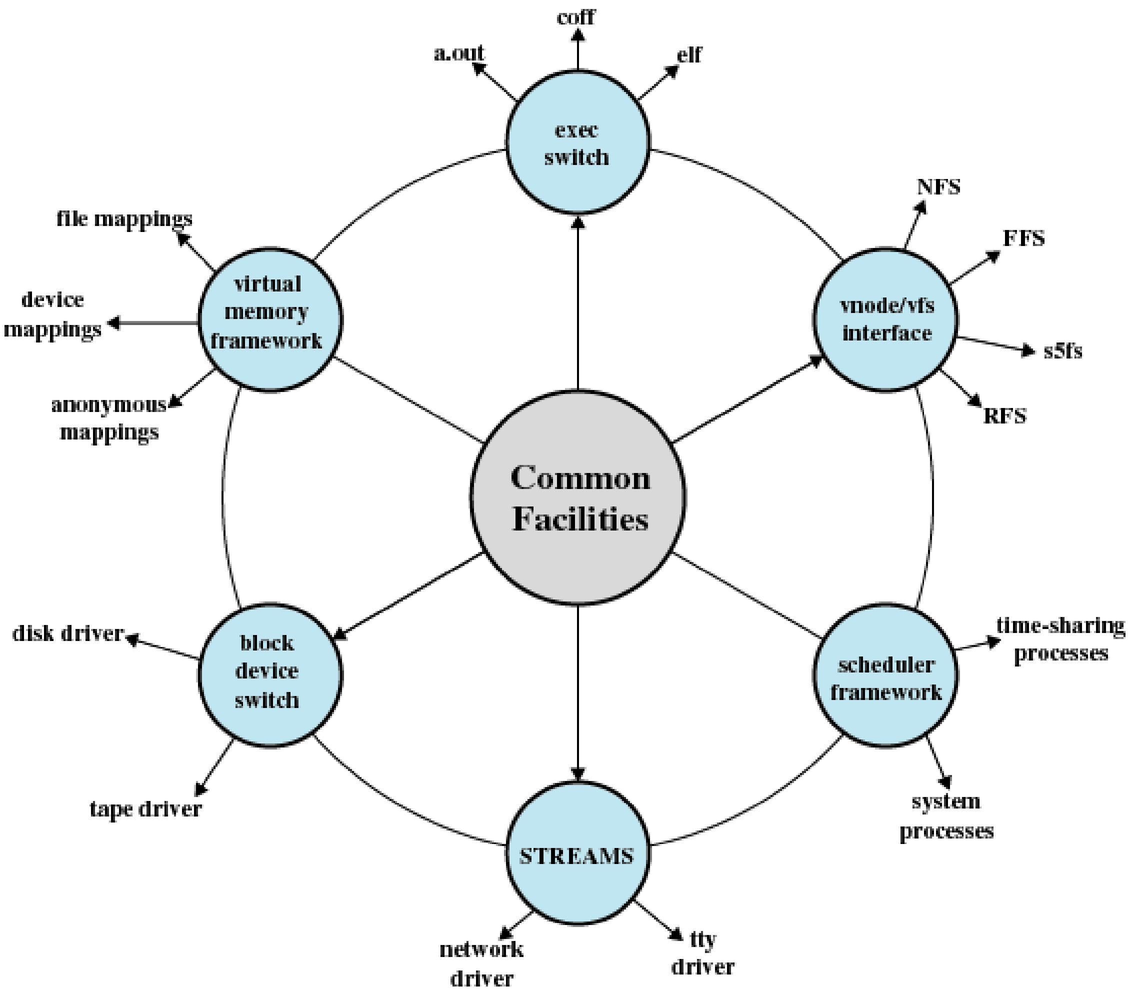


Figure 2.16 Modern UNIX Kernel [VAHA96]

Modern UNIX Systems

- System V Release 4 (SVR4)
- Solaris 9
- 4.4BSD
- Linux