Chapter 1
Basic Concepts of Operating Systems
Software

• A program is a **sequence of instructions** that enables the computer to carry out some specific task.

• Before a program executes, it has to be translated from its original text form (source program) into a **machine language** program. Then, the program needs to be linked and loaded into memory.
Software Components

• The software components are the collection of programs that execute in the computer.
• These programs perform computations, control, manage, and carry out other important tasks.
• Two general types of software components are:
  – System software
  – Application software
System Software

• The system software is the set of programs that control the activities and functions of the various hardware components, programming tools and abstractions, and other utilities to monitor the state of the computer system.

• The system software forms an environment for the programmers to develop and execute their programs (collectively known as application software).

• Three types of users can be identified: system programmers, application programmers and end-users.
Application Software

• Application software are the user programs and consist of those programs that solve specific problems for the users and execute under the control of the operating system.

• Application programs are developed by individuals and organizations for solving specific problems.
Types of Software Systems

- System software - Operating System, Assemblers, Loaders, Linkers, Compilers, Editors, ...
- Application software - All User-Oriented Programs.
What is an Operating System?

• A large and complex software component for the operation and control of the computer system.
• It acts as an intermediary between a user and the computer system.
• Examples: Unix, MS Windows, MacOS, Linux, Sun Solaris, DEC VMS, etc.
The Operating System

• A provider of **services** to user programs

• A huge **resource** manager
Design 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.
External View of a Computer
Operating Systems User Interfaces

Three levels of interface:

1. Graphics GUI (windows oriented)
2. Command level (also known as the shell). At login time, the shell starts computing
3. System calls invoked from user programs
The Shell

• The program that handles user interaction with the system is called:
  – Shell
  – Command-line interpreter

• Two types of Shells
  – Graphical
  – Character oriented
System Calls

• This is also known as the Application Programming Interface (API)
• Programs use the API to request the OS to perform some function
Basic Structure of an OS

- GUI
- Command line interpreter (shell)
- System call interface
- Process, memory, file, and device management
- Hardware control

Kernel
Multi-Level Views

• The overall structure of an operating system is divided into the various software components using a top-down (layered) approach.

• The top layer provides the easiest interface to the human operators and users interacting with the system.

• Any layer uses the services or functions provided by the next lower layer.
Operating Systems Abstract Views

• External views
  – Set of interface of the computer system
  – A layer of software on top of the hardware

• Internal view
  – Resource manager - It controls and manages CPU, memory, I/O devices, etc.
Abstract Views of an OS
Layered Structure of an OS

- Users (top layer)
- Application User Interface (AUI): shell, commands, application programs
- Application program Interface (API): libraries, system calls
- OS kernel
System Programs

• The Operating System media will include programs that are not part of the operating system kernel.

• Examples
  – Web Browser
  – Email program

• Most users’ view of the Operating System is defined by System Programs, not the OS itself
Internal View of an Operating System

• The system call interface separates the kernel from the application layer and the kernel is located above the hardware.

• The kernel is the core and most critical part of the operating system and needs to be always resident in memory.

• A detailed knowledge about the different components, including these lower-level components of the operating system, correspond to an internal view of the system.
Functional Components of an OS

The most important components of an operating system are:

• Process manager
• Memory manager
• Resource manager
• File manager
• Device manager
Services Provided by the OS

• Process Control, execution, scheduling, etc.
• Communication between processes
• File Manipulation
• Device Manipulation
• Information Maintenance
• Memory Management
Jobs and Processes

A job is a unit of work submitted by a user to the operating system. A typical job consists of the parts listed below:

• A sequence of commands to the operating system
• A program either in a source language or in binary form
• A set of input data used by the program when it executes

A process basically refers to an execution instance of a program.
Categories of Operating Systems

• **Batch** systems, in which a set of jobs are submitted in sequence for processing.

• **Interactive** systems, which support computing for on-line users. The most common type of operating systems that support interactive computing is time-sharing, which are multi-user systems.

• **Real-time** systems, which support application programs with very tight timing constraints.

• **Hybrid systems**, which support batch and interactive computing.
A Time-Sharing System
Small and Specialized OS

• A mobile OS controls a mobile device
• Are relatively simpler and smaller OS
• Focus on wireless broadband and local connectivity
• Found on smart phones and tablet PCs
Embedded OS

- For embedded computer systems
- Very compact and efficient
- Are very specialized
- Most are real-time OS
History of Operating Systems

• First generation - No operating system, bare hardware, machine language.

• Second generation
  
  – Batch systems, assemblers, linkers, loaders, compilers

  – Batch systems with Automatic Job Sequencing
History of Operating Systems(2)

• Third generation -- O.S. for complete families of computers (OS/360)
  – Batch with Multiprogramming
  – Spooling
  – Timesharing (MULTICS, UNIX, ...)

• Fourth generation
  – Network and distributed operating systems
Modern Operating Systems

- Windows (Microsoft Corporation) these include a family of systems: 98, Me, CE, 2000, XP, Vista, Windows 7, and others
- Linux (Linus Torvalds, OSF GNU)
- MacOS (Apple)
- Solaris (Sun Microsystems)
- OSF-1 (OSF, DEC)
- IRIX (Silicon Graphics)
- OS2 (IBM)
- OS/390 (IBM)
- VMS (Dec/Compaq/HP)
64-bit OS

- Developed for 64-bit processors and systems with 64-bit architectures
- MS 64-bit Windows 7
- Mac OS X
Mechanisms and Policies

• Mechanisms determine the implementation of some technique, policies decide what type of service is provided.

• The separation of policy from mechanism is a very important principle, it allows maximum flexibility if policy decisions are to be changed later.
System Implementation

• Traditionally written in assembly language, operating systems can now be written in higher-level languages.

• Code written in a high-level language:
  – can be written faster.
  – is more compact.
  – is easier to understand and debug.

• An operating system is far easier to *port* (move to some other hardware) if it is written in a high-level language.