CHAPTER 5: SYSTEM SOFTWARE

5.1 OPERATING SYSTEMS

5.1.1 Need for Operating System

Definition: A set of programs designed to run in the background on a computer system.

Functions:

- Controls operation of computer system
- Provides user interface
- Controls how computer responds to requests
- Controls how hardware communicates
- Provides environment for application software

Why OS is Essential:

- Hardware is unusable without OS
- Acts as interface between user and hardware
- Manages communication between components

5.1.2 Key Management Tasks

Memory Management:

- Memory protection (ensures programs don't use same memory)
- Paging (uses virtual memory)
- Memory allocation

File Management:

- Provides file naming conventions
- Maintains directory structure

- Allocates space to files
- File access control

Security Management:

- Provides usernames & passwords
- Ensures data privacy
- Prevents unauthorized access
- Carries out automatic backup

Hardware Management:

- Installation of driver software
- Controls access to peripherals
- Handles interrupts from devices

Process Management:

- Enables multiprogramming and multitasking
- Resolves conflicts when processes need same resource
- Methods: Round-robin, priority scheduling

5.1.3 Utility Software

Disk Formatter:

- Prepares hard disk for data storage
- Deletes existing data
- Creates sectors and tracks

Virus Checker:

- Checks for viruses
- Removes viruses found
- Monitors incoming/outgoing files

Defragmentation Software:

- Reorganizes files to contiguous sectors
- Reduces read/write head movements
- Improves performance

Disk Repair Software:

- Visualizes disk space usage
- Reports errors (bad sectors)
- Attempts to fix issues

File Compression:

- Reduces file size
- Removes redundant data
- Improves storage efficiency

Backup Software:

- Makes copies of files
- Stores on different medium
- Provides synchronization between devices

5.1.4 Program Libraries

Definition: Pre-written code that can be linked to software under development.

Benefits:

- Saves time (less code to write)
- Smaller testing time (pre-tested)
- Complex algorithms available without understanding implementation

DLL (Dynamic Link Library):

- Shared library file containing code and data
- Loaded to memory only when required
- Available to several applications simultaneously
- Reduces .EXE file size

5.2 LANGUAGE TRANSLATORS

5.2.1 Assembler

Purpose: Translates assembly language to machine code (binary).

Characteristics:

- One-to-one relationship with machine code
- Simple translation process

5.2.2 Compiler vs Interpreter

Feature	Compiler	Interpreter
Translation	Translates entire program before execution	Translates line-by-line
Output	Creates .exe file	No .exe created
Execution	Faster (already compiled)	Slower (translates each time)
Error Reporting	All errors at end	Stops at first error
Development	Used when development complete	Used during development
Debugging	Difficult (all errors at end)	Easier (stops at error)

5.2.3 Two-Step Translation (Java)

Process:

1. Java compiler translates source code to bytecode
2. Java Virtual Machine (JVM) interprets bytecode to machine code

Benefits:

- Platform independence (write once, run anywhere)

5.2.4 IDE Features

Coding Features:

- Context-sensitive prompts
- Variable highlighting (undeclared/unassigned)
- Autocomplete

Error Detection:

- Dynamic syntax checking
- Type checking
- Parameter checking

Presentation:

- Prettyprint (automatic indentation, colour-coding)
- Expand and collapse code blocks

Debugging:

- Single stepping (execute line-by-line)
- Breakpoints (pause at specific line)
- Variables/expressions report window

Revision #1
Created 2026-03-16 12:01:32 UTC by Samuel Lee
Updated 2026-03-16 12:01:47 UTC by Samuel Lee