Introduction to Software Types

- Software refers to a collection of programs and data that instruct a computer to perform tasks.

Major classifications include:

- System Software
- Application Software
- Utility and Middleware Software

These categories help in understanding the role and functionality of software in computing systems.

System Software vs Application Software

System Software:

- Manages hardware and system resources
- Examples: Operating Systems, Device Drivers

Application Software:

- Designed for end-users
- Examples: Word Processors, Web Browsers

Flow Representation:

- User->Application Software->System Software->Hardware

Software Categories and Platforms

Licensing Models:

- Proprietary – Owned and restricted
- Open-Source – Source code publicly available
- Freeware – Free to use, limited rights

Application Platforms:

- Web Applications (Browser-based)
- Desktop Applications (Installed locally)
- Mobile Applications (Smartphones/Tablets)

Software Architecture and Design Patterns

Software Architecture defines the high-level structure of a system.

Design Patterns:

- Reusable solutions to common problems
- Based on proven best practices

Principles:

- Modularity
- Reusability
- Maintainability

Benefits:

- Improved code quality
- Faster development
- Reduced complexity

Common Design Patterns and Anti-Patterns

Common Design Patterns:

- Singleton
- Factory
- Observer
- Strategy

Anti-Patterns:

- Poor or ineffective solutions
- Lead to maintainability issues

Example:

- Quick Fix ---> Long-Term Problem

MVC (Model-View-Controller) Architecture

Core Components:

- Model – Data and business logic
- View – User interface
- Controller – Handles input and control flow

Flow Diagram:

- User → Controller → Model → View

MVC Benefits, Variants, and Challenges

Benefits:

- Separation of concerns
- Easier maintenance and testing

Variants:

- MVP (Model-View-Presenter)
- MVVM (Model-View-ViewModel)

Challenges:

- Increased complexity
- Coordination between components

Solution:

- Clear interface definitions
- Proper documentation

Service-Oriented Architecture (SOA)

SOA Definition:

- Architectural style using reusable services

Core Principles:

- Loose coupling
- Reusability
- Interoperability

SOA Components:

- Service Provider ... Service Registry ... Service Consumer

SOA vs Microservices:

- SOA – Enterprise-level, shared services
- Microservices – Fine-grained, independent services