Design of Software Systems (CS612) Class 21

Dr. Nilotpal Chakraborty, IIIT Guwahati

October 3, 2024

In the course, till Mid-Semester

- Software – Importance of Software Design
- Software Development Life Cycle
- Software Design Process – Requirements Analysis
- Software Design Process – System Design, Detailed Design

In the course, till Mid-Semester

Principles of Software Design – Fundamental Principles, Design Principles

- Modularity
- Encapsulation
- Abstraction
- Cohesion
- Coupling
- Don’t Repeat Yourself (DRY)
- You Aren’t Gonna Need It (YAGNI)
- Separation of Concerns

In the course, till Mid-Semester

Software Design Paradigms:

- Procedural Design
- Object-Oriented Design
- Component Based Design
- Model Driven Design

In the course, till Mid-Semester

- Software – Software Architecture
- Key Components of Software Architecture
  - Views: 4 + 1 views (Logical, Process, Development, Physical, Use-Case)
  - Structures
  - Elements
  - Relationship
- Software Architecture Styles
- Software Architecture Patterns
- Software Design Anti-Patterns
- Refactoring
- SOLID principles

Real-World Deployments of Object Oriented Design

- On **Facebook**, core concepts such as User, Post, Comment, Like, and Group are modeled as objects. These objects encapsulate data (e.g., a user's profile information) and behavior (e.g., liking or commenting on a post).
- In **Uber**'s mobile app, various entities like Driver, Rider, Trip, and Payment are modeled as objects. Each object has its own data (driver's location, rider's destination, trip details) and actions (start trip, end trip, calculate fare).
- In **WordPress**, core objects like Post, Page, User, and Comment are modeled as classes. These objects manage all the content and user interactions in the CMS.
- In **Tesla**'s autopilot system, entities such as Car, Sensor, Obstacle, Route, and Driver are modeled as objects. Each object handles specific functions such as processing sensor data, controlling the car's motion, and interacting with the driver.

Component Based Design

- A software design approach that emphasizes the *separation of concerns* through independent, reusable components.
- **Characteristics**: *Reusability, Modularity, Separation of Concerns, Scalability, and Maintainability*.
- Facilitates the creation of large systems by integrating pre-developed, independently deployable components.
- Enhances flexibility and reduces development time by reusing components across multiple projects.

Real-World Deployments of CBD

Microservices Architecture in Web Applications:

- **Netflix** uses a microservices architecture, where each service is an independent component responsible for a specific task (e.g., user authentication, content delivery, recommendations). Each service is loosely coupled and communicates through APIs (typically *RESTful APIs*).
- Microservices align with CBD principles by being modular, independently deployable components that handle distinct business functions.
- Microservices interact via lightweight protocols like HTTP, or message queues, facilitating communication between services. This decoupling of services allows Netflix to scale, maintain, and upgrade individual components without affecting the entire system.