SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107

Accredited by NAAC-UGC with 'A' Grade

Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of Artificial Intelligence and Data Science

Design Concepts

Design Concepts

- Design concepts provide the foundation for creating **efficient, scalable, and maintainable** object-oriented software.
- Below are key design concepts with explanations and real-world analogies.
- Abstraction
- Encapsulation
- Modularity
- Cohesion
- Coupling
- Inheritance
- Polymorphism

Design Concepts

1. Abstraction

- **Definition**: Abstraction focuses on exposing only the essential features of an object while hiding the underlying complexity.
- **Example**: A **car's dashboard** allows the driver to start, stop, and accelerate without knowing the internal mechanics of the engine.

Design Concepts

2. Encapsulation

- **Definition**: Encapsulation is the bundling of data (variables) and methods (functions) within a class, restricting direct access to some details.
- **Example:** A bank account system hides the account balance details from direct modification and only allows transactions through specific methods like deposit() and withdraw().

Design Concepts

3. Modularity

- **Definition**: Modularity refers to dividing a system into smaller, independent modules, making it easier to develop, maintain, and reuse.
- **Example**: A **library management system** can be divided into modules like **User Management, Book Catalog, and Borrowing System**, each handling a specific function.

Design Concepts

4. Cohesion

- **Definition:** Cohesion measures how closely related the functionalities within a single module or class are. High cohesion improves maintainability.
- **Example:** A Student class that only contains student-related attributes and methods (such as name, age, calculateGrade()) has high cohesion, whereas adding unrelated functions (e.g., printReport()) reduces cohesion.

Design Concepts

5. Coupling

- **Definition**: Coupling refers to the degree of dependency between different modules. Low coupling ensures that changes in one module do not heavily impact others.
- **Example**: A **shopping cart module** should interact with the **payment system** but not depend on its internal logic. Instead, it should use an interface for communication, reducing dependency.

Design Concepts

6. Inheritance

- **Definition:** Inheritance allows a new class (subclass) to acquire properties and behaviors from an existing class (superclass), promoting reusability.
- **Example:** A Vehicle class with attributes like speed and fuel can be inherited by subclasses like Car, Bike, and Truck, reducing code duplication.

Design Concepts

7. Polymorphism

- **Definition:** Polymorphism allows the same interface or method to be used for different types, increasing flexibility.
- **Example:** A "Print()" function in a Document class can behave differently when used with a PDF or Word file, ensuring the correct format is printed.


thank
you