13 Lecture - CS504

Important Subjective

Q: What is Object-Oriented Analysis and Design (OOAD)? A: Object-Oriented Analysis and Design is a software engineering approach that focuses on modeling a system using objects, classes, and their interactions to develop robust and maintainable software solutions. **Q: Explain the key concepts of Object-Oriented Analysis and Design. A**: The key concepts include abstraction, encapsulation, inheritance, and polymorphism. Abstraction hides unnecessary details, encapsulation bundles data and behavior together, inheritance enables code reuse, and polymorphism allows objects of different classes to be treated as instances of the same class. **Q: Describe the purpose of UML diagrams in OOAD. A**: UML (Unified Modeling Language) diagrams are used to visually represent different aspects of a system, such as its structure, behavior, and interactions. They aid in understanding, communicating, and documenting the system's design. **Q: What is the difference between a class diagram and an object diagram in UML? A**: A class diagram depicts the static structure of a system, showing classes, their attributes, and relationships. An object diagram, on the other hand, shows specific instances of classes and their relationships at a particular moment in time. **Q: Explain the importance of the "Single Responsibility Principle" in OOAD. A**: The Single Responsibility Principle states that a class should have only one reason to change. It promotes maintainability by ensuring that each class has a focused responsibility, making the code easier to understand and modify. **Q: What is the purpose of the "Factory Method" design pattern in OOAD? A**: The Factory Method pattern provides an interface for creating objects, allowing subclasses to decide which class to instantiate. It promotes loose coupling and allows the system to be more flexible when creating objects. **Q: How does the "Observer" design pattern work, and where is it commonly used in OOAD? A** : The Observer pattern defines a one-to-many dependency between objects, so when one object changes state, all its dependents are notified and updated automatically. It is commonly used in GUI frameworks and event-driven systems. **Q: Describe the difference between composition and aggregation relationships in OOAD. A**: Composition represents a strong "whole-part" relationship where the child cannot exist independently of the parent. Aggregation represents a weaker relationship where the child can exist independently. **Q: What is "Use Case Analysis" in OOAD, and how is it used during system development? A**: Use Case Analysis involves identifying and defining the functional requirements of a system from the perspective of its users. Use cases describe the interactions between actors (users) and the system to understand the system's behavior. **Q: How does OOAD facilitate software development in large-scale projects? A**: OOAD promotes modularity, reusability, and maintainability through its principles and design patterns. It allows developers to break down complex systems into smaller manageable components, making it easier to collaborate and develop large-scale software projects.