Агрегация и Композиция. Наследяване – Резюме

Структурни връзки в OOA// и OOD:

Асоциация: Най-простата форма на връзка между класове:

-Слаба – метод на клас A включва като параметър инстанция на ClassB или връща инстанция на ClassB.

-Силна - ClassA съдържа препратка към истанция на ClassB

Агрегация: Спец. форма на асоциация, която моделира връзка "част-цяло" между агрегат (цялото) и неговите части, където частите могат да съществуват независимо от цялото. Пр: библиотека с книги

-Shared Aggregation- силна връзка между два класа, така че една и съща инстанция участва в две различни агрегации. Напр. Човек и бизнес имат един и същ адрес

-Self-Aggregations - класът е съвкупност от др. екземпляри на същия клас. Тук имената на ролите са от значение за разграничаване. пр. Един **продукт** се състои от др. **подпродукти**;

-Nested classes - Клас / интерфейс съдържа др.класове

Композиция: Агрегация с по-силна собственост и съвпадащ живот на частите с агрегата. Частите не могат да съществуват без цялото.

Наследяване = Generalization || Specialization:

-Обща връзка: връзка между по-общ и по-специфичен елемент. По-спец. елемент е съвместим с по-общия и има допълн. инф.

-Абстрактни класове: Класове, които не могат да бъдат инстанцирани и съществуват само за да бъдат наследявани.

-Полиморфизъм: динамично се избира метод за операция по време на изпълнение.

-Single and Multiple Inheritance - Клас наследява няколко други класа - **множествено** наследяване, но обикновено наследява един клас.

Интерфейс: Дефинира набор от поведения на клас или компонент.

Реализация: Клас наследява интерфейс без да е подклас на интерфейса.

SOLID Принципи на дизайн:

SRP (Single Responsibility): Всеки клас трябва да има една единствена отговорност.

OCP (Open/Closed): Софт. единици трябва да бъдат отворени за разширение, но затворени за модификация.

LSP (Принцип на заместване на Лисков): Обекти - подтипове да заменят обекти от базовия тип без да нарушават функционалността.

ISP (Принцип на разделяне на интерфейса): Клиентите не трябва да бъдат принуждавани да зависят от интерфейси, които не използват.

DIP (Принцип на инверсия на зависимостите): Високо ниво модули да не зависят от ниско ниво модули. И двете зависят от абстракции.

Composite Structure: Показва вътр. структура на класа и връзки

-Именуване частите: partName:partType[multiplicity] пр. part1: Type[0..2]

- Aggregated класове са част от класа
- Връзките с външни обекти се показват като част – правоъгълник, очертан с прекъсната линия

Package диаграма- показва структура и зависимости между подсистеми или модули, например multi-layered application model.