Тестування та верифікація програмного забезпечення

Лекція №5

Тема: Проєктування програмного забезпечення

ЖИТОМИРСЬКА ПОЛІТЕХНІКА
ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Лектор: асистент кафедри комп'ютерних наук Українець Микола Олександрович

Питання лекції

1. Види проєктування ПЗ
2. Парадигми проєктування ПЗ
3. Ідентифікація класів предметної області. UML-діаграми ієрархії класів.

Види проєктування ПЗ

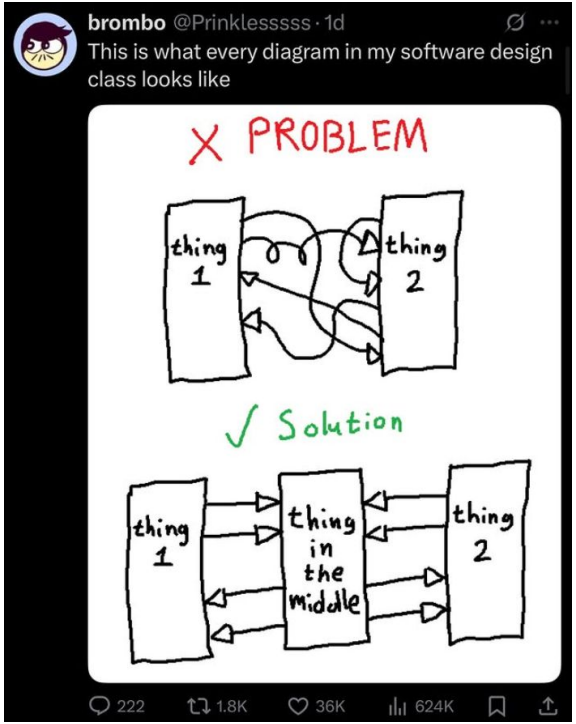
Проєктування — процес створення прототипу, прообразу майбутнього об'єкта, стану та способів його виготовлення. У проєктуванні застосовують системний підхід, який полягає у встановлені структури системи, типу зв'язків, визначенні атрибутів, аналізуванні впливів зовнішнього середовища.

Проєктування програмного забезпечення (англ. Software design) – це процес визначення архітектури, компонентів, інтерфейсів та інших атрибутів (структур даних, алгоритмів і т.д.) системи або компонента програмного забезпечення. Результатом цього процесу є проєкт програмного забезпечення.

Види проєктування ПЗ

Розглянемо наступні види проєктування ПЗ:

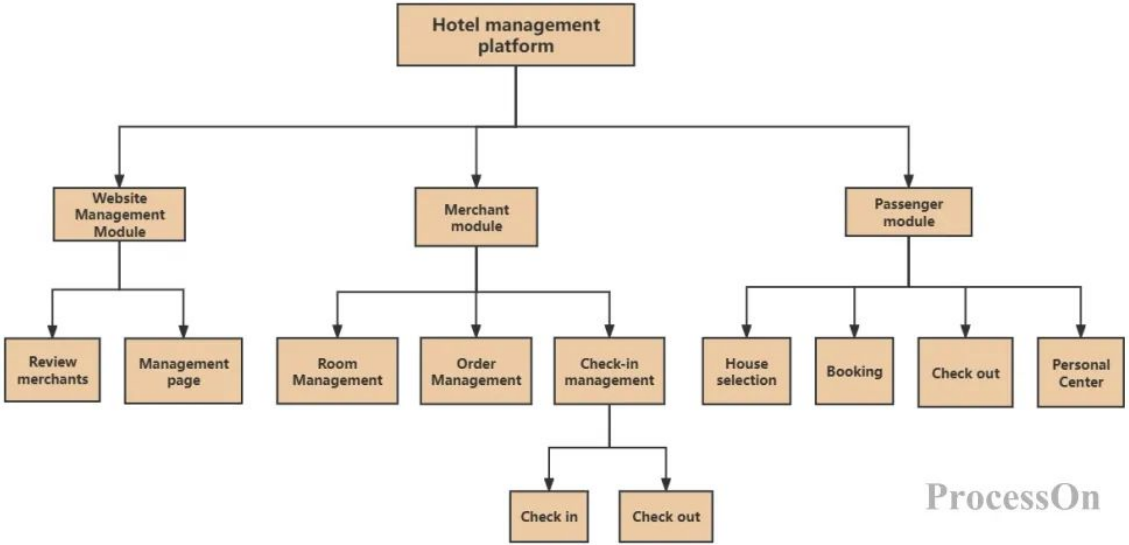
- Структурне проектування (Structural Design)
- Об'єктно-орієнтоване проектування (Object-Oriented Design)
- Функціональне проектування (Functional Design)
- Архітектурне проектування (Architectural Design)
- Інтерфейсне проектування (Interface Design)



Види проєктування ПЗ

Структурне проектування (Structural Design)

Структурний підхід до проєктування ПЗ — це методологія розробки, що базується на принципі функціональної декомпозиції системи: складна система послідовно розбивається на підсистеми, модулі та підмодулі до рівня елементарних процедур.



Види проєктування ПЗ

Структурне проектування (Structural Design)

Система проєктується зверху вниз (top-down):

- спочатку визначається загальна функція системи;
- далі вона розбивається на підфункції;
- кожна підфункція деталізується до рівня процедур або алгоритмів.

Кожен модуль:

- виконує одну логічно завершену функцію;
- має чітко визначений інтерфейс;
- мінімізує залежності від інших модулів.

Принцип модульності

Модулі повинні:

- мати високу **зв’язність (cohesion)** — виконувати одну логічну задачу;
- мати низьку **зв’язаність (coupling)** — мінімально залежати від інших модулів.

Види проєктування ПЗ

Структурне проектування (Structural Design)

SADT (Structured Analysis and Design Technique)

Методологія SADT є сукупністю методів, правил і процедур, призначених для побудови функціональної моделі предметної області. Функціональна модель SADT відображує функціональну структуру об'єкту, тобто дії, що ним виконуються, і зв'язки між цими діями.

Графічна модель

- Функція зображується блоком
- Входи / виходи — дуги, що входять і виходять з блоку
- Інтерфейсні дуги описують взаємодію блоків

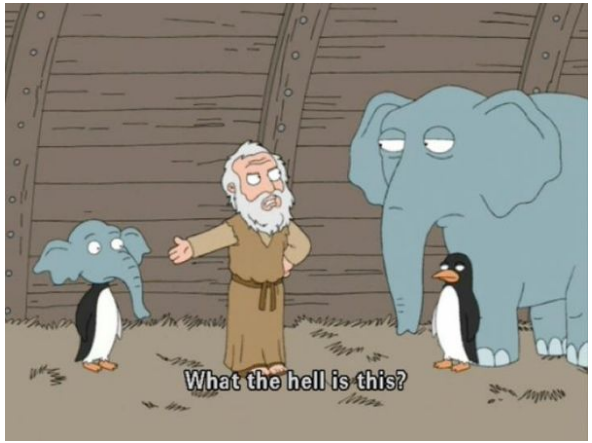
Дуги можуть відображати:

- обмеження
- керування
- умови виконання функцій

Види проєктування ПЗ

Об'єктно-орієнтоване проектування (Object-Oriented Design)

Об'єктно-орієнтоване проектування - це методологія проектування, що поєднує процес об'єктної декомпозиції і прийоми подання логічної і фізичної, а також статичної і динамічної моделей проектованої системи.



Об’єкт — це сутність, що:

- має стан (дані),
- має поведінку (методи),
- взаємодіє з іншими об’єктами через повідомлення.

Головна ідея:

Система проєктується навколо сутностей предметної області, а не навколо функцій.

Види проєктування ПЗ

Об'єктно-орієнтоване проектування (Object-Oriented Design)

Абстракція – виділення суттєвих характеристик об’єкта та відокремлення їх від несуттєвих деталей.

Інкапсуляція – приховування внутрішньої реалізації об’єкта та надання доступу до нього через публічний інтерфейс.

Наслідування – механізм створення нових класів на основі вже існуючих з можливістю розширення або модифікації поведінки.

Поліморфізм – можливість об’єктів різних класів реагувати на однакові повідомлення по-різному.

Види проєктування ПЗ

Об'єктно-орієнтоване проектування (Object-Oriented Design)

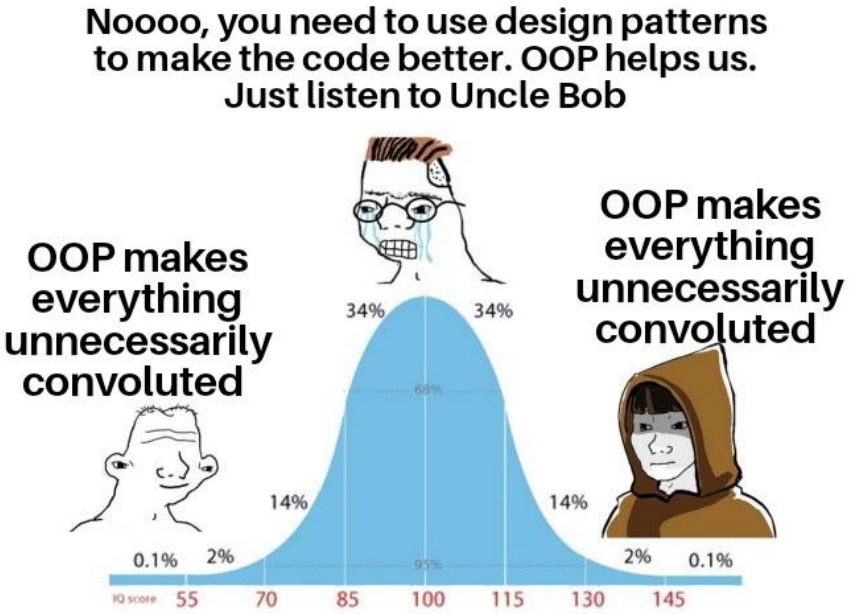
Клас — шаблон для створення об’єктів.

Об’єкт — екземпляр класу.

Асоціація — зв’язок між класами.

Агрегація / композиція — відношення “частина–ціле”.

Інтерфейс — контракт поведінки без реалізації.



Види проєктування ПЗ

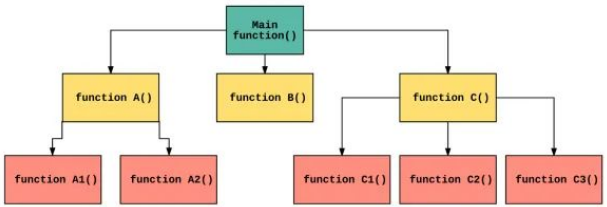
Функціональне проектування (Functional Design)

Функціональне проєктування (Functional Design) — це підхід до проєктування програмного забезпечення, у центрі якого знаходяться функції системи та потоки даних між ними.

Система розглядається як сукупність функціональних перетворень:

вхідні дані ➜ обробка ➜ вихідні дані.

На відміну від об’єктно-орієнтованого підходу, де основною одиницею є об’єкт, у функціональному проєктуванні основною одиницею є **функція**.



Види проєктування ПЗ

Функціональне проектування (Functional Design)

Функціональне проєктування базується на такій послідовності кроків:

- визначити, що повинна робити система;
- розбити загальну функцію на підфункції;
- описати алгоритми їх виконання;
- визначити потоки даних між функціями.

Складна система декомпонується на прості функціональні модулі, кожен з яких виконує одну завершену операцію.

Види проєктування ПЗ

Функціональне проектування (Functional Design)

Основні принципи:

- Функціональна декомпозиція
- Ієрархічність
- Чіткість потоків даних
- Мінімізація залежностей

Види проєктування ПЗ

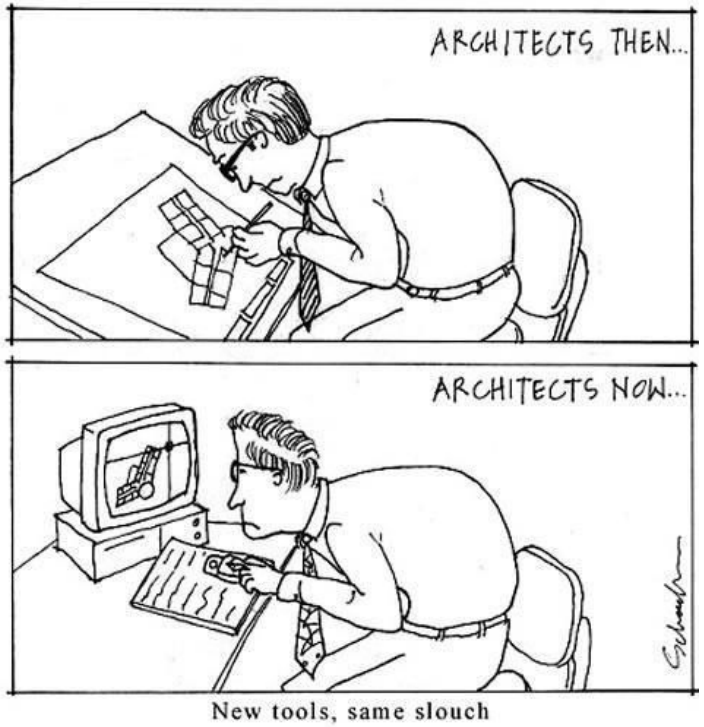
Архітектурне проєктування (Architectural Design)

Архітектурне проєктування ПЗ — це вид проєктування, який визначає високорівневу структуру системи: її компоненти, їхні відповідальності, спосіб взаємодії та принципи організації.

Архітектура відповідає на запитання:

- З яких частин складається система?
- Як ці частини взаємодіють?
- Які технологічні та структурні рішення лежать в основі?
- Як система забезпечує нефункціональні вимоги?

Архітектура — це концептуальна модель системи, що визначає її довгострокову життєздатність.



New tools, same slouch

Види проєктування ПЗ

Інтерфейсне проєктування (Interface Design)

Інтерфейсне проєктування (Interface Design) — це процес розробки способів взаємодії між користувачем і програмною системою або між компонентами системи.

Інтерфейс визначає:

- як користувач взаємодіє з системою;
- як система відображає інформацію;
- як компоненти обмінюються даними;
- як забезпечується зручність і зрозумілість роботи.

Парадигми проєктування

4.5.2.	Парадигми проєктування: функціональна декомпозиція згори донизу, архітектура, орієнтована на дані, об'єктно-орієнтований аналіз та проєктування, подієво-керована архітектура.		В

Парадигма (з грец. зразок, приклад) — це система фундаментальних уявлень, принципів, цінностей та методів, що приймаються науковою або суспільною спільнотою за основу і визначають стиль мислення, підходи до вирішення завдань та сприйняття світу у певний період.

?

Парадигми проєктування

Функціональна декомпозиція (Functional Decomposition) допомагає управляти складністю та зменшити невизначеність, розбиваючи процеси, системи, функціональні області або результати на простіші складові частини і дозволяючи аналізувати кожну частину незалежно.

Розбиття великих компонентів на підкомпоненти дозволяє масштабувати, відстежувати і вимірювати робочі зусилля для кожного з них. Це також полегшує оцінку успішності кожного підкомпонента в порівнянні з іншими більшими або меншими компонентами.

Архітектура, орієнтована на дані, (data-oriented architecture, DOA) була вперше описана Раджівом Джоші у звіті RTI 2007 року, а потім у 2017 році Крістіаном Ворхемом та Еріхом Шикутою з Віденського університету у статті iiWAS. **DOA** — це інверсія традиційної дихотомії між монолітним кодом та сховищем даних (монолітна архітектура) з одного боку, та невеликими розподіленими незалежними компонентами з власними сховищами (мікросервіси та сервіс-орієнтована архітектура) з іншого.

Парадигми проєктування

Об’єктно-орієнтований аналіз (OOA)

Мета: зрозуміти предметну область через об’єкти.

Що виконується на етапі OOA:

- Виявлення сутностей (класи предметної області)
- Визначення їхніх атрибутів
- Встановлення зв’язків між ними
- Виділення основних сценаріїв використання
- Побудова концептуальної моделі

Основні артефакти:

- Діаграма варіантів використання (Use Case)
- Концептуальна діаграма класів
- Опис бізнес-об’єктів

Об’єктно-орієнтоване проєктування (OOD)

Мета: перетворити аналітичну модель у технічну архітектуру.

Що додається на етапі OOD:

- Технічні класи
- Інтерфейси
- Патерни проєктування
- Розподіл відповідальностей
- Архітектурні рішення

Основні артефакти:

- Детальна діаграма класів
- Діаграми послідовностей
- Діаграми компонентів
- Архітектурна схема

Парадигми проєктування

Подієво-орієнтована архітектура (англ. Event-driven architecture; надалі EDA) — шаблон архітектури програмного забезпечення, який призначений для створення подій, їх виявлення, споживання і реагування на них.

Подія може бути визначена як значна зміна стану. Наприклад, коли споживач купує автомобіль, стан автомобіля змінюється з «на продаж» до «продано». Архітектура системи дилера автомобілів може трактувати цю зміну стану як подію, поява якої може стати відомою іншим застосункам даної архітектури.

Подієв-орієнтована система як правило складається з емітерів подій і споживачів подій . Споживачі несуть відповідальність за здійснення реагування на появу події. Реакція не завжди може бути повністю забезпечена самим стоком. Наприклад, стік, може бути відповідальним лише за фільтрацію, трансформацію і відправку події до іншого компонента або він може забезпечити повністю самостійну реакцію на таку подію.

Ідентифікація класів предметної області. діаграми ієрархії класів.

В об’єктно-орієнтованому аналізі важливо виділити сутності (класи), які відображають предметну область системи.

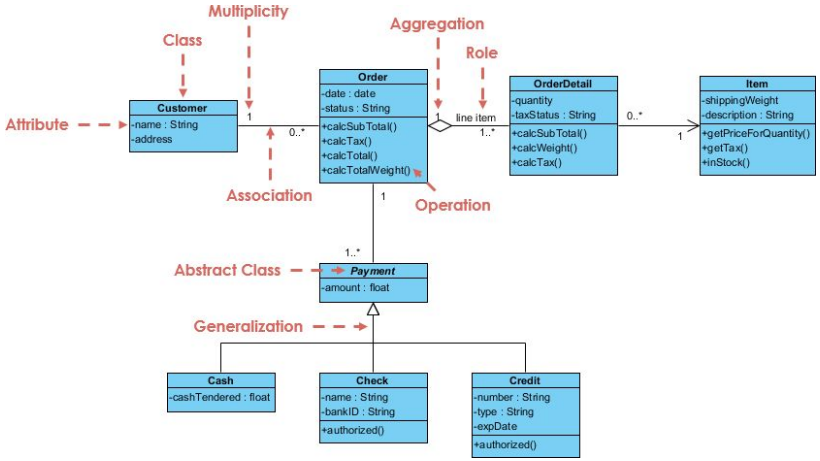
Ідентифікація класів пов’язана з класифікацією об’єктів за спільними ознаками чи поведінкою.

Основні підходи до класифікації:

- **Класична категоризація** — групування об’єктів за чіткими властивостями (не завжди працює для складних систем).
- **Концептуальна кластеризація** — формування груп на основі загальних описів та моделей.
- **Теорія прототипів** — визначення класу за подібністю до типового представника (прототипу).

Ідентифікація класів предметної області. діаграми ієрархії класів.

Діаграма класів (англ. Class diagram) — статичне представлення структури моделі в UML. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення. Діаграма класів може містити позначення для пакетів та може містити позначення для вкладених пакетів. Також, діаграма класів може містити позначення деяких елементів поведінки, однак їх динаміка розкривається в інших типах діаграм.



Дякую за увагу!