**ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНТЕЛЕКТУАЛЬНИХ**

**ТЕХНОЛОГІЙ І ЗВ’ЯЗКУ**

**Звіт**

**з дисципліни Архітектура та Проектування Програмного Забезпечення**

**Конспект №1**

Виконав: студент 4 курсу, групи ІПЗ-4.04 спеціальності

121 Інженерія програмного забезпечення

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Бухта М.М.

**Одеса  2024**

**Компонентна архітектура**

* **Інкапсуляція**: Компоненти приховують внутрішні процеси, надаючи лише необхідні інтерфейси.
* **Незалежність**: Компоненти мінімально залежні від інших, що дозволяє їх розгортати у будь-якому середовищі.
* **Переваги**:
  + Простота розгортання: Можливість заміни компонентів без впливу на інші частини системи.
  + Найменша вартість: Використання сторонніх компонентів знижує витрати на розробку.
  + Простота розробки: Використання відомих інтерфейсів.
  + Можливість повторного використання: Компоненти можуть бути використані в різних додатках.
  + Спрощення технічної підтримки: Компоненти полегшують систему через контейнер і його сервіси.

**Проектування на основі предметної галузі (DDD)**

* **Основна концепція**: Модель предметної області використовується як основа для створення програмних систем, з акцентом на співпрацю між розробниками та фахівцями в галузі.
* **Переваги**:
  + Обмін інформацією: Використання спільної мови серед усіх учасників розробки.
  + Розширюваність: Модель предметної області є модульною та гнучкою.
  + Зручність тестування: Об'єкти моделі характеризуються слабкою пов'язаністю та високою зв'язністю.

**Багатошарова архітектура**

* **Основні принципи**:
  + Анотація: Система представляється як єдине ціле з деталізацією окремих шарів.
  + Інкапсуляція: Шари приховують внутрішні деталі.
  + Чітко визначені функціональні верстви: Поділ функцій між шарами.
  + Висока зв'язність: Межі відповідальності для кожного шару чітко визначені.
  + Можливість повторного використання: Шари можуть бути повторно використані в інших сценаріях.
* **Переваги**:
  + Анотація: Шари дозволяють змінювати систему на абстрактному рівні.
  + Ізоляція: Оновлення можуть бути ізольовані в окремих шарах.
  + Керованість: Полегшує управління та організацію коду.
  + Продуктивність: Масштабованість через розподіл шарів.
  + Тестованість: Певні інтерфейси шарів полегшують тестування.

**Архітектура, заснована на шині повідомлень**

* **Основна концепція**: Програми взаємодіють шляхом обміну повідомленнями через спільну шину.
* **Основні характеристики**:
  + Взаємодія через повідомлення: Використання відомих схем повідомлень.
  + Зміна логіки обробки: Логіка змінюється шляхом вставки або видалення додатків на шині.
  + Інтеграція з різними інфраструктурами: Взаємодія між додатками на різних платформах.
* **Переваги**:
  + Розширюваність: Додавання/видалення програм без впливу на інші.
  + Гнучкість: Зміни у конфігурації або маршрутизації.
  + Слабке зв'язування: Відсутність залежностей між програмами.
  + Масштабованість: Підключення до шини множини екземплярів програми.
  + Простота програми: Підтримка лише одного підключення до шини.

**N-рівнева/3-рівнева архітектура**

* **Основні принципи**:
  + Функціональна декомпозиція додатку, розподілене розгортання для підвищення масштабованості, доступності та керованості.
  + Кожен рівень відповідає за певну функціональність і може розміщуватися на різних серверах.
* **Переваги**:
  + Зручність підтримки: Рівні незалежні один від одного.
  + Масштабованість: Організація рівнів дозволяє легко масштабувати систему.
  + Гнучкість: Управління кожним рівнем незалежно підвищує гнучкість.
  + Доступність: Використання модульної архітектури підвищує доступність.

**Об'єктно-орієнтована архітектура**

* **Основні принципи**:
  + Анотація: Узагальнення складних операцій для збереження їх основних характеристик.
  + Композиція: Об'єкти можуть бути утворені іншими об'єктами.
  + Інкапсуляція: Об'єкти приховують свої дані та методи від інших об'єктів.
  + Поліморфізм: Можливість об'єктів обробляти різні типи даних через однакові інтерфейси.
* **Переваги**:
  + Повторне використання: Об'єкти можуть бути повторно використані в інших програмах.
  + Гнучкість: Об'єкти можна легко модифікувати або розширювати.
  + Масштабованість: Система може бути легко масштабована шляхом додавання нових об'єктів.