8.Процес за проектиране на софтуерната архитектура

ADD (Attribute Driven Design): Подход за проектиране, в който основна роля играят **качествените свойства (атрибути)**. Това е рекурсивен процес на дефиниране на архитектурата, като на всяка стъпка се използват **тактики и архитектурни модели** за постигане на желаните **качествени свойства**. В следствие на приложението на ADD се получават първите няколко нива на модулната декомпозиция - достатъчно високо ниво, без излишни детайли

-Цикличен процес на създаване на архитектурата

Стъпки на ADD

1/Избор на модул за декомпозиция: Първоначално това е цялата система, която се разлага на подсистеми, модули и под-модули.

2/Детайлизиране на модула:

2.1.Избор на архитектурни драйвери: Най-важните изисквания за този етап.

2.2.Избор на архитектурен модел/ конфигурацията от избрани тактики/– идентифициране на модули: Модел, който удовлетворява драйверите, базиран на тактики за постигане на избраните свойства.

2.3.Създаване на под-модули, Декомпозиция, Приписване на функционалност: Идентифициране на типовете под-модули и приписване на функционалност съгласно сценариите за употреба.

3.Създаване на други структури

4.Дефиниране на интерфейсите: Документиране на интерфейсите към и от под-модулите.

5.Проверка и детайлизиране на изискванията, Проверява се декомпозицията: Проверка дали всичко съществено е налично и

подготовка на под-модулите за по-нататъшна декомпозиция, а, като същевременно се поставят ограничения върху под-модулите

6/Рекурсивен ADD: Повторение на процеса за всички модули, които се нуждаят от по-нататъшна декомпозиция.

7. Формиране на екипи

8. Създаване на скелетна система

Входни данни на ADD

- **Функционални изисквания**: Сценарии за употр. (use-cases).
- **Функционални ограничения**: Constraints.
- **Качествени свойства**: Специфични сценарии за проявление.

Пример:

Функционални изисквания: *Потребителят трябва да може да се регистрира с имейл и парола.*

Функционални ограничения: *С трябва да обработва максимум 1000 заявки в секунда.*

Качествени свойства: *С трябва да се възстановява автоматично в рамките на 5 минути след срив.*

Детайлизиране на модула

-Архитектурни драйвери: Избор на най-важните изисквания.

-Архитектурен модел: Избор на тактики за постигане на лесна промяна и бързодействие. Например:

- ✓ **Тактики за лесна промяна**: Локализиране на промените, скриване на информация
- ✓ **Тактики за бързодействие**: Увеличаване на ефективността на алгоритмите, управл. на ресурсите.

Създаване на под-модули

- **Идентифициране на под-модули**
- **Приписване на функционалност**: Описание на отговорностите на под-модулите съгл. сценарии за употреба.
- **Създаване на други структури**: Разглеждане на процесите и разполож. за покриване на изискваната функционалност.

Дефиниране на интерфейсите

- **Интерфейси**: Съвкупност от услуги и свойства, които модулът предлага/изисква. Документиране на всички свойства и услуги от всички структури.

Проверка на декомпозицията - дали декомпозицията е коректна и покрива всички изисквания. Подготовка на под-модулите за следваща декомпозиция, ако е необходимо.

Рекурсивен ADD - рекурсивно извикване на ADD за някои от модулите, които се нуждаят от детайлизация

Формиране на екипи, които да работят по съответните модули. Структурата на екипите отговаря на структурата на декомпозицията.

Създаване на скелетна система

-Скелетна система: Започване на работа по системата, използвайки стъбове за разработка и тестване на модулите поотделно.

-Последователност на създаването: Първо се създават компонентите, свързани с изпълнението и взаимодействието между архитектурните компоненти (middleware), след това прости функционалности и накрая функционалности, диктувани от намаляване на риска, наличния персонал и бързото създаване на продаваем продукт.

-След това на база структурата на употребата се определят следващите функционалности