

20 Софтуерна архитектура. Проектиране и документиране на софтуерната архитектура (Обобщение) да е ясно и точно нама тук са да е ясно

1. Дефиниция на софтуерната архитектура. Структури и изрази на архитектурата.

Дефиниция: Архитектура на дадена софтуерна система е общиност от структури, показващи различни софтуерни елементи на системата, включно видимите и невидимите между тях.
- абстракция, която разглежда елементите като цели групи, и се интересува от взаимодействията между тях.

Структура - общиност от софт. елементи, техните видими и невидими не засягат конкретните техни детали елементи и връзките между тях.

Изглед ^(view) - конкретно документирано представление на дадена структура.
- пакет за ЗА

Видове архитектурни структури:

1. Модулни - дават един по-дълъг изглед на модулите (единици работи за изпълнение) на системата и функционалността, която те реализират

- декомпозиция на модулите - X е подмодул на Y
- репрезентация както X не е макс. пром
- обособяват функционалността

- употреба на модулите - X използва X

- структура на аловете - алове не се пренасят Направи само N-1

- иерархия на модулите X наследства Y

2. Структура на процесите - пренасят връзката между софтуерните елементи от еволюцията на приложението с която се налага системата по време на разработката на приложението

- елементите са процеси, изпълнявани в системата и операции между тях

- използва се при структ. на потока на данните

3. Структура на разпределението - показват връзката между софт. и елементите на средата, в която се налага имплементация по време на разработката на приложението. В нея участват:

- структура на въндряването
- физична структура
- разпределение на работата.

Да и опишат какъв
видове структури

2. Изисванията и как качеството (функционални изисвания) на системата

Два типа изисвания:

- функционални - какво да прави системата
- качествени (нфункционални) - как трябва да работи системата
 - технически изисвания
 - бизнес ограничения
 - технологични

Бизнес изисвания определят качеството, които системата трябва да притежава.

Качеството е субективно понятие \Rightarrow формализирано с чрез "сценарии за качество"

Компоненти на сценарии за качество:

- Въздействие - събитие, което да се обработи
- ~~Действие~~ Чутощник - обект, който генерира въздействие
- Обект - системата или нейна част, която поема и изпълнява въздействия
- Контекст - условията, в които се налага обектът
- Резултат - действието, предприети от обекта
- Качествени параметри - метрики, на база които се проверява дали изпълнява изисванията

изисване за поведение
на системата в дадена
ситуация

Основни групи качества:

- Технологични - надеждност, изменчивост, производителност, сигурност, изпитаемост, достъпност и др.
- Бизнес качества - време за пускане на пазара, себестойност и наклада, предвидено време за плаване
- Архитектурни - присъщи на самата архитектура
 - логична целост, урадливост, коректност

структурно изисване

3. Граденето на СА

Градене на проектиране

Граденето на проектиране на последователност етапи, в следствие, че поимо се използва СА. Стартира се при пакети на основните функции и тех. изисвания.

За driver на СА се подготват пакети при не повече от 10 изисвания (архитектурни фрагменти), които се превръщат в бисес и качествени сценарии.

Избор на подразделка ~~архитектурни~~ - зависи от driver-ите

Декомпозиция на модулите - налага се при разделяне на функционалностите на отдалечени модули.

Структура на проектиране - за системи нуждае се от близост и надеждност

2

3

Годуването на създаване на архитектура

Attribute-Driven Development (ADD)!

- подход за проектиране
- главна роля играят качествените изисквания
- репрезентативният модел е декомпозиция на модулите

Входни данни ???

- функции, изисквания като use-cases
- функции, ограничения
- качествени сценарии

Стъпки:

1. Извират се модули за декомпозиция
2. Извират се арх. драйвери отнасящи се за модула
3. Извират се арх. модели на модула (стремеж тачки)
4. Разбъркват се подмодули
5. Групират се функции на подмодулите
6. Създават се други структури за подмодулите
7. Дефинират се интерфейсите от/към подмодулите
8. Верификация на декомпозицията
9. Задълбаване рекурсивно, ако е нужно

Платформа (арх. решението) за постигане на желаните качествени изисквания

Платформата е арх. решение, чрез което се контролира резултата на даден сценарий за качество. Наподобява от конкретни тачки се нарича арх. стремеж. Конфигурираща от тяхните съорганизации арх. модел.

① Платформи за управление на качеството

- отговаряне на откази
 - Echo (Ping/echo) - един комп. проверява друг и отваря отг. в даден интервал
 - Heartbeat, keepalive - периодичен сигнал за конпонентни данни е жив, даден във времето
 - Сънктогония - обратност на изключението
- отстраняване на откази
 - Активен резерв - активно дублиране и резултатът се връща само от актив.
 - Гласовен резерв - един отговор и инф. останалиши за синхронизиране
 - Резерва - поддръжка на резервна изчислителна мощ
 - Убеждение от употреба - премахване на кашп., за издаване на отказ
 - Съдене на провалите - процес, който следи останалиши

- Гловтарто обаждане в употреба
- Гарантия радом (shadow mode) - приготвя съвместно със същата команда
- Ресинхронизация на изчисленията
- Контролни точки и rollback - при сриване с времето последно записаното

② Техники за производителност

- Изкачване на изчислението
- Увеличение на производителността на изчислението - на-додрийки, кеш
- Пренасаждане на overhead от ненужни изчисления
- Правила на периода на периодични обединения
- Правила на тахтовата честота (rate limiting)
- Оптимика с чист резултат
- Управление на ресурсите
 - Гарантия за достъп
 - Излишък на данни - сache, load-balancing
 - Възпроизвеждане на допълнителни ресурси
 - Адаптивни на ресурсите
 - Scheduling - приоритизиране на обединения
- Оптимизация на изчислението
- Годеждане на премени - за създаването на ограничения за използването
- Годеждане на ефекта на балансиране (от SOLID)
- Определяне на свързането - контрол на свързането
 - ping and play
 - config files

③ Техники за сигурност

- Установяване на атаки
 - auth
 - нива на достъп
 - криптиране
 - ограничаване на достъпа
- Откриване на атаки (alarm)
- Възстановяване след атака

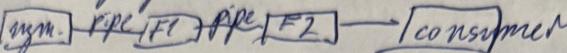
④ Техники за изпитаемост

- Запис и възпроизвеждане - logs
- Разделение на интерфейс от реализация
- Специален интерфейс за тестване
- Вградени модули за мониторинг

4. Арх. стилове

Арх. стилът дефинира начинът от системи чрез използването на шаблон за структурна организација.

① Pipe-and-Filter



- модулите се дават на източник, фильтри, консуматор
- pipe - връзка между фильтри
- фильтри - преобразват и прибавят трансформират данни
- въвervo - всеки фильтър не знае за следите
- позволява паралелизъм
- минус - постоянно parse-ване на данни

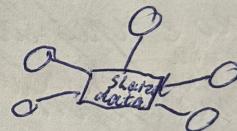
② Layered

- компонентите са вертикално разделини по слоеве
- един слой използва слоя под него и предлага услуги на този над него
- абстракционни и логични изрази



③ Client-Server

- серверите предлагат услуги и клиентите ги използват
- праща се заявка, така се отговор
- клиент идентифицира и обработва заявката *кога е имплементиран*
- центризирана данни и сигурност *организационното*
- риск от притоварвания
- нужда от резерви



④ Shared data

- всички компоненти получават общите данни
- лого се добавява нови компоненти

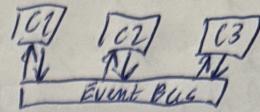
Data bus

- Blackboard - при този тип, всички components са уведомени
- Repository - при този тип, има уведомление
- проблем за имплементация при разпределени системи

⑤ MVC

- модел - управлява данните, дава данни на изгледа, приема заявки за пращане
- изглед - презентира данните
- контролер - отговаря за взаимодействието с потребител
- издава и разширява

- ⑥ Implicit invocation / Message passing / Pub-sub / Event based
- компонентите си комуникират чрез събития.
 - събитията са общи за всички / обобщени
 - събитията са придават по шина
 - компонентите работят паралелно
 - loose coupling на компонентите
 - компоненти в надеждността



⑦ Cloud инови

- Circuit breaker - спира приложението да изврши операции, когато се очаква да не е успешна
- Queue
- Caching
- Sharding - разделяне на данните

5. Документиране на СА

- как са основният начин за док.
- как се използват структурите - на база на съзб. ЗД

5.1 Групнодокументи

- важна за мяна поддръжка, докладване и изследование
- яко раз才是真正 на мястото
- трябва да е достатъчно абстрактна (за да се раздели от нови служби)
- но и достатъчно детайлна (за да е пътна основа за преместване)
- най-често - надзор от документите с обогатено съдържание, която улеснява

5.2 Основен принцип

- отстои съотв. описание на различни структури
- структурите могат да се използват по различни начини в зависимост от това е предвидяното

5.3 Съдържание

- мяна установен стандарт за съдържанието, трябва да е последователна
- пример за доказано в практиката съдържание:

① Первичен измер - графично, UML

② Описание на елементи и връзки - детайлно, описва и поведение и интерфейс

③ Описание на обкръжението - как са взаимодействват с други системи, интерфейси, протоколи и др.

④ Описание на върховни варианти - описание на варианти на някои детали (с условия и ограничения)

⑤ Арг. основное - аргументация на взети решения

⑥ Гарантиращия ръчник

⑦ Допълнителна информация - Всичко останало

5.4 Структура - не е ли гафто???