**Въведение в софтуерните технологии**

**Софтуерен продукт** = програми + данни + документация

**Роля на софтуер:**

Продукт- информационен преобразувател.

Средство за доставяне на продукт- отговаря за извършване на трансфер на информация, създаване на друг софтуере и контрол на компютъра.

**Характеристики на софтуера:**

Разработва се

Абстрактност

Мултидисциплинираност- познания в различни области

Надеждност

Рискове

Не се износва

Компоноентно базирано преработване

**Дефиниции:**

Software Engineering е дисциплина, която предоставя методи и средства за разработване на качествен софтуер с ограничен бюджет, поставени срокове и с постоянно изменящи се изисквания. “

*“Software engineering* - приложение на систематичен, дисциплиниран и определен подход за разработването, изпълнението и поддръжката на качествен софтуер.”

**Области включени в СТ:**

Управление на проекти

Потребителска документация

Обсъждане на хардуера

Тестване

Програмиране

Проектирне на потребителския интерфейс

Спецификация на софтуера

Проектиране на софтуер

**Предразположеност към грешки:**

Работи по друг начин различен от очаквания

Обикновено е по-скъп от планираното

Разработката му приключва твърде късно

Не е приложим

**Жизнен цикъл на програмния продукт**- целият период на неговото създаване и използване

Начало- възникване на идеята за създаване на продукта

Край- моментът, в който се преустановяван използването му

**Може да бъде характеризиран в 2 аспекта:**

Статичен- фазите на жизнения цикъл са ограниени във времето интервали със сравнително самостоятелни ресурси

Динамичен- фазите са свързани помежду си

**Фази на жизнения цикъл-** анализ, проектиране, тестване, поддръжка

**Софтуерен процес**- цялостният процес на поставяне на задачата, планиране, реализиране и оценка на едно софтуерно и хардуерно приложение, включително и използваните помощни средства, методи и необходимия персонал.

**Фаза на дефиниция-** оценка на онформацията, планирне на софтуерния проект, анализ на изискванията

**Фаза на разработване-** софтуерно проектиране, генериране на кода, тестване на софтуера

**Фаза на поддръжка**

**Модели на софтуерния процес-** Стратегия за разработване, която включва процес, методи, средства и основните фази за разработка

**Защо са нужни модели?-** правят процеса методологичен, спомагат за определянето на организацията на екипа, дават технологични насоки

**Характеристики на модел**- естествена форма за разработка на проекти, оценките ръководят проектните решения, разработва се прототип още в реална фаза

**Последователни модели**- спазва се сравнително строга последователност на дейностите

**Инеративни**- доускат връщане назад към предишни фази. Проблеми- връщането назад е скъпа дейност; спецификацията на изискванията може да се окаже грешна.

**Линейни последователни модели:**

Водопаден модел(класически жизнен цикъл)

V-модел

Cleanroom-engineering

**Непоследователни модели:**

Еволюционен развой

Прототипиране- вертикално(избират се само някои ф-ии, които се реализират напълно) и хоризонтално(разглежда се цялата функционалност, релаизира се обаче само в общи щрихи)

Инкрементален развой

Спирален модел- абстрактен модел, пгъркав

Whirlpool модел

Фонтанен модел

**Основни концепции, описващи софтуерната разработка**

***Функция*** *=*Задача на софтуера, функция на продукта

### Функционални дървета-Функциите обикновено описват определени дейности или ясно очертани задачи в рамките на една проблемна област.; Декомпозирането на общи комплексни задачи на по-прости функционални идентичности; Оценка- по-добри от линеен списък; фаза на проектиране, фаза на дефиницяи

### Data-Flow Diagrams- идея: разработванта система е един информационен поток от данни м/у ф-ии, памет, интерфейси към външни обекти(могат да участват веднъж); Оценка- лесни по устройство и четаемост, съдържат повече инофрмация от функционалните дървета, НО- стават големи и нечетаеми за цялата система; фаза на дефиницията;

Основните **предимства** на диаграмите на потока от данни са следните:

* Могат да бъдат лесно създавани и имат добра четаемост
* Лесни са за разбиране и от непрофесионалисти, което улеснява значително контактите между партньорите (клиенти, доставчици)
* Съдържат повече информация от функционалните дървета

**Недостатъци** на тези диаграми:

* Ако искаме да представим една цяла система диаграмите могат да станат много големи, трудно обозрими и нечетаеми – възможно решение в такъв случай е изграждане на йерархични нива (метод на структурния анализ)
* Трудно се поддържа единно абстракционно ниво за данните и функциите
* В определени случаи означаването на потоците данни само с имена не е достатъчно.

**Use- Case Diagrams**-Описва всички бизнес процеси, връзките им един с друг и с актьорите. Описват функционалните изисквания; фаза на дефиницията

### Основните предимства на използването на сценарии са следните:

* Фокусират се върху основния работен поток, а не върху елементарни функции
* Концентрират се върху стандартните работни потоци
* Лесни са за разбиране и от клиентите (не само от професионалисти)

**Недостатък** е, че диаграмите лесно могат да се претоварят и утежнят с твърде големи и излишни детайли.

**Основни концепции на ориентирания към данни изглед на системата**

**Речник от данни(Data dictionary)-** предшества ЕР-диаграмите; текстово представяне, структурни езици; фаза на дефиниция и проектиране и разработка; **Цел-** синтактична структура на данните от потребителска гледна точка; **Употреба**- да направи информацията, която се съдържа в диаграми на поток на данни по-прецизна; **Оценка**- **Предимство**-разработката на структурите от данни е формално и сбито описание, **Недостатък**- трудно се чете, отнема много време;

**Обектни връзки(Entity- Relationship)-** фаза на дефиниция и анализ; графично представяне; **използва** **се** в моделирането на БД; **Цел**- описване на постоянните записи на данни и техните връзки, множество от индентичности на табл; във фаза на анализ се определят множествата от иденти и асоциации; основните елементи са идентичности; асоциация- семантична връзка м/у множествата от идентичности; кардиналност- сложност на връзката; роля- функция на идентичност във връзка

**Сравнение**- DD са синтактични стриктури от данни(външни и вътрешни), ER за моделиране на външни данни.

**Основни концепции на ориентирания към правила изглед на системата- фаза на реализацията**

**Цели-** описание на зависими от условия действия, потоци, ф-ии, задачи;

ФАД- да се направят по- прецизни словесни описания в спецификаицята на изискванията;

**Правила-**предикатна логика, ясен синтаксис и семантика; **условия-** описват определена ситуация, при която могат да се извършат операциите; **операции-** могат да се изпълняват при наличието на описана в условието ситуация, **2 типа**- **импликации (**извеждат се логически стойности на определени твърдения**) и действия (**с тях се променят определени състояния**)**; правилата правят спецификациите на изискванията по-прецизни; **структура** **на** **правилата**- да определи елементарните условия и действия; **Проблем-** трудно се постига яснота( възможност за общ изглед) и пълнота(дали ще бъде възможно да се обхванат всички условия); фаза на разработка, фаза на дефиниция;

**Таблици на решенията-**  дефинират се кратко, ясно и прегледно действия или операции, които зависят от изпълнението или неизпълнението на определени условия; **оптимизация на таблиците-** търсят се правила сидентични действия и се обединяват; **оценка-** с над 5 условия стават неясни -> затруднява четаемостта и създаването на пълни таблици с решения -> възможност за решение- декомпозиране; ); , възможност за пропускане на условия; фаза на разработка, фаза на дефиниция;

**Дървета на решенията-** всички алтернариви се представят явно; хоризонтално представяне на таблицата на решенията;

**Основни концепции на ориентирания към състояния изглед на системата**

**Крайни автомати-** фаза на дефиниция, фаза на проектиране; **цел-** моделиране на поведението на системата; **употреба-** динамичен изглед на обектите в класа (жизнен цикъл на обекта), спецификация на операциите на класа (промяна на състояните, докато операцията се изпълни) и на Use Cases (промяна на състоянието по време на вазимодействие между потребител и система); **дефиниция-** адекватно средство за моделиране на вътрешните състояния на системата или от история на обработката до определен момент.; може да се предтави като граф;

**Activity diagrams-** фаза на проектиране; вариант на крайните автомати при който:

**-**Алгоритмите и бизнес-процесите се описват посредством състояния на действията (action states)

*-Action states* са стъпки (дейности) по време на изпълнението на един алгоритъм или бизнес-процес

*-Аction state* е напуснато, когато дейността свързана с него  е приключила (implicit event)

Подобни са на  другите диаграми, разгледани в предишните глави.

**Основни концепции на ориентирания към сценарии изглед на системата**

И двата вида диаграми **се използват за** моделиране на сценариите на системата при обектно-ориентиран развой. Могат да бъдат **разработени по време** на фазата на анализ.; фаза на проектиране; и 2те са Interaction diagrams

**Съобщение-**  моделираме заявки на клиенти за  изпълнение на определени услуги;

**Сценарии-** описват динамични последователности от операции в една система, в която съществува дефинирана подредбата на съобщенията

**Sequence diagram-**.показват сценарии за някои от процеите в един UseCase, участващите обекти, разменяните съобщения, подредени последователно във времето;

**Collaboration diagram-**  използват за поетапно моделиране на информационните потоци в един use case.

**различни изгледи** на софтуерния продукт трябва да бъдат възможни

**независими от приложението**

типа на приложението определя само избора на подходяща основна концепция

**Независими от метода на развой**

Основните концепции могат да се комбинират за общи методи/

**Фаза на дефиниция**

**Фаза на проектиране**

**Фаза на разработка**

Резултати от фазата на “анализ и дефиниция”

**Фази от процес на разработка**

Анализ и дефиниция

Проектиране

Разработване

Тестване

**Цели на „Анализ и дефиниция”**- анализ на проблема; дефиниция на изискванията към софтуера. Двете изпълняват една задача- описание на външното поведение на софтуерната система

**2 части на АД:**

**Фаза на планиране**

Разучаване (речник, предваритенла спецификация на изискванията, определяне на цената, план на проекта)

**Фаза на дефиниция:**

Дефиниране на продукта(спецификация на изискванита, модел на продукта, UI, ръководство за потребителя)

***Разделя се на 2 фази, защото може да се стигне до прекъсване на взаимоотношенията с клиента(неразбирателство?), което води до прекратяване на проекта и работата върху него.***

**Резултат от фазата на дефиниця**- дейност във фазата на планиране.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Влизат: | Участват: | Излизат: |
| ФАД | Указания на клиента  Шаблон(предварителна спцификация, речник) | Клиент  Ръководител клиент  Приложен специалист | предварителна спцификация, речник,  определена цена,  план на проекта |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

***функционални* изисквания**: функционалност, данни (логически изглед), потребителски интерфейс

**изисквания на средата на приложението**: например потребителски профили

***технически изисквания***: език за разработка, операционна система, хардуер ***изисквания за производителност***: производителност, количество данни

**изисквания за валидност**: разработване на тестове, тестови случаи

**изисквания за качество**: удобен за използване интерфейс, надеждност, ...

**изисквания за реализацията**: модел на процеса, документация, срокове, цена

Предварителната спецификация на изискванита включва:

основни функции , основни данни , обща производителност , Важни аспекти на потребителския интерфейс ,Важни критерии за качество

**Фаза на планиране:**