17. Управление на качеството на софтуерни приложения. Тестване на софтуер. (УК)

Анотация: Изложението по въпроса трябва да включва следните по-съществени елементи:
1. Осигуряване на качеството – качество на софтуера, алтернативи за осигуряване на качество.
2. Тестови дейности, управление и автоматизация.
3. Видове тестване – тестване с контролни списъци, тестване с класове на еквивалентност, разделяне на входния домейн и тестване на границите.
4. Нива на тестване и приложение на техниките за тестване.
5. Измерване. Метрики – характеристики и класификация. Метрики за качество на софтуерен продукт и софтуерен процес.

1. Осигуряване на качеството

Осигуряването на качество изисква да се изпълнят набор от дейности (например тестване), проектирани да оценят процеса, спрямо който се създават и поддържат продуктите. Целта е да се докаже качеството, т.е. правилното поведение, както и да се установят и отстранят проблеми.

1.1. **Качество на софтуера** се дефинира като степента, до която система, компонент или софтуерен процес отговаря на специфицираните изисквания и удовлетворява нуждите или очакванията на заинтересованите лица.

1.2. **Алтернативи за осигуряване на качеството** са **софтуерна разработка (ориентирана към процеса)** и **софтуерна поддръжка (ориентирана към продукта).**

2. Тестови дейности, управление и автоматизация

2.1. Основните **тестови дейности** са:
1. **Планиране на тестовете** – дефиниране на тестова цел, планиране на ресурси, персонал и избор на формални модели и техники за тестване.
2. **Конструиране на тестов модел** – идентифициране на източник на информация и събиране на данни, както и анализ и създаване на модела.
3. **Генериране на тестови сценарии от модела**, където:
   - **Тестови сценарий (Test case)** дефинираме като колекция от елементи и свързана с тях информация, осигуряващи изпълнението на тест или тестова серия.
   - **Тестова серия (Test run)** е динамична единица от специфични тестови дейности в общата тестова последователност върху избран тестов обект. На прост език изпълнение на няколко теста.
4. **Създаване и управление на тестови пакети (test suites)**, където:
   - **Тестов пакет (Test suite)** наричаме колекция от отделни тестови сценарии, които се стартират в тестова последователност докато не се удовлетвори даден критерий за спиране.
5. **Подготовка на тестова процедура**, която определя последователността и превключването на тестовите серии.
6. **Изпълнение на тестовете и наблюдение**, което се състои от:
   - Заделяне на време и ресурси;
   - Стартиране и изпълнение на тестове и събиране на резултати от изпълнението им;
   - Проверка на тестовите резултати чрез тестови оракули.
7. **Анализ и проследяване** – анализиране на индивидуални тестови сценарии и проверка на резултатите от тях с цел идентифициране на повреди.

2.2. **Управление**

Съществуват следните видове организация и управление на екипа за тестване:
- **Вертикален модел на организация** – организацията е около продукта, като една или повече тестови задачи се асоциират с определен изпълнител.
- **Хоризонтален модел на организация** – за големи организации, като един тестов екип изпълнява един тип тестване за всички продукти.
- **Смесен модел на организация**, където:

– ниските нива на тестване се изпълняват от екипите отговарящи за съответните проекти;
– системното тестване се споделя между подобни проекти;
– съществува обща поддръжка на проекта от осигурена централна единица.

2.3. Автоматизация

Целта на автоматизацията е да освободи тестерите от досадни и повтарящи се задачи и да повиши тестовата производителност. Важни въпроси късаещи автоматизацията са възможността за автоматизация на специфични тестови сценарии, изборът на достъпни софтуерни инструменти за автоматизация и крайната цена на автоматизацията.

Тестова дейност	Възможност за автоматизация
Изпълнение	
Изпълнение на тестове	Висока
Планиране и подготовка	
Генериране на тестови сценарии	Висока
Създаване на формални модели	Средна
Подготовка на тестови сценарии	Средна
Цялостно планиране на тестовия процес	Ниска
Планиране на тестовата процедура	Ниска
Анализ и проследяване	
Анализ на надеждността	Висока
Анализ на тестовото покритие	Висока
Действия за подобряване на продукта	Ниска

3. Видове тестване

3.1. Тестване с контролни списъци

Ad-hoc тестването представлява стартиране на софтуера, наблюдение на поведението и идентифициране на проблемите. Тестването с **контролни списъци** представлява създаване на неформални **TODO списъци** за проследяване на изтестваните по ad-hoc начин елементи.

Съществуват следните **типове контролни списъци**:

- **Базови** – прости списъци от елементи, които трябва да се изтестват;
- **Йерархични** – списъците са разделени на нива, като елементите от по-горните нива съдържат списъци от по-ниски нива;
- **Комбинирани (многомерни)** – списъците са многомерни, като всеки контролен списък се обхожда за всички елементи от останалите контролни списъци;
- **Смесени** – съчетание на йерархичните и комбинираните контролни списъци.

При контролните списъци се срещат следните проблеми и ограничения:

- **Трудности при покриване на всички функционалности** от различни гледни точки и нива на грануларност;
- **Припокриване на елементи** в различни контролни списъци;
- **Трудност при описване на сложни взаимодействия** между различни компоненти на системата.

3.2. Тестване с класове на еквивалентност

От дискретната математика знаем, че класовете на еквивалентност породени от дадена релация $\equiv_A$ на еквивалентност над множество A, представляват разбиване на това множество. С други думи, ако $C_1, \ldots, C_n$ е разбиване на множеството A, за някое $n \in \mathbb{N}$, то:

- $\forall i, j, i \neq j$ е изпълнено, че $C_i \cap C_j = \emptyset$;
- $\bigcup_{i=1}^{n} C_i = A$;
- $C_i = [x]_A = \{a \in A \mid x \equiv_A a\}$.

Практическата импликация при тестването е, че тестовите сценарии (или входа им) се разбиват на класове на еквиваленстност спрямо очаквания тип на взаимодействие. Може да се раздели на следните стъпки:

1. Дефиниране на класове на еквивалентност (или още по абстрактно – на релация на еквивалентност върху множеството от входни данни);
2. Избор на един тестов сценарии за всеки клас на еквивалентност;
3. Постигане на пълно покритие на класовете на еквивалентност.

Типовете разделяне на класове на еквивалентноаст са:
- Базирано на **софтуерни елементи**;
- Базирано на **определени свойства, релации, логически условия**;
- Базирано на комбинация от горните две.

Дърво и таблици за взимане на решения – може да се разгледат като йерархичен контролен списък. Прилагат се за тестване базирано на решения и тестване базирано на предикати.

3.3. Разделяне на входния домейн и тестване на границите

Генериране на тестови сценарии посредством присвояване на специфични стойности на входните променливи въз основа на **анализ на входния домейн**.

Тестване с разделяне на входния домейн наричаме покриване на малък брой входни ситуации посредством систематичен избор на определени входни стойности. Характеризира се с:
- Тестване на входно/изходните зависимости – осигуряване на стойности за всички входни променливи;
- Изходните променливи не се специфицират експлицитно – проверяваме дали изход се получава от вход;
- Прилага се главно за функционално тестване;
- Детайлите по реализацията могат да се използват за анализ на входните променливи, което е предпоставка за извършване на структурно тестване.

Разделя се на следните **стъпки**:
1. Идентифициране на входното пространство и дефиниране на входен домейн.
2. Разделяне на входния домейн на поддомейни;
3. Анализ на поддомейните с цел определяне на границите им по всички измерения;
4. Избор на всички тестови точки (сценарии) покриващи поддомейните;
5. Тестване с избраните тестови точки, решаване на проблеми и анализиране на разултатите.

Проблемите при разделянето на входния домейн са:
- **неопределеност на даден вход** – тестваната програма не обработва някои входни стойности;
- **противоречивост за даден вход** – повреда в системата или производство на различни изходи при един и същ вход;
- **противоречивост за даден вход** – повреда в системата или производство на различни изходи при един и същ вход.

Проблеми при определянето на границите на поддомейните са:
- **Проблем със затвореността на границите** – отворена граница се специфицира като затворена;
- **Изместване на границата**;
- **Липсваща граница**;
- **Излишна граница**.

3.4. Други
Други начини за тестване са чрез:
- тестване с **покритие на класове**;
- тестване с **машина на крайните състояния (автомат)**;
- тестване с **граф на данновия поток**.

4. Нива на тестване и приложение на техниките за тестване

4.1. Тестови подфази

Откъм йерархична гледна точка тестовете са:
1. **Тестване на ниво програмна единица (Unit testing)** – тестване на малки програмни единици, като функция, метод или най-много един клас, по метода на бялата кутия. Представлява най-ниското ниво на абстракция в йерархията на тестовете.
2. **Компонентно тестване (Component testing)** – тестване на софтуерните компоненти, като например библиотечни пакети, като се допуска тестване, както по метода на бялата, така и по метода на черната кутия.
3. **Интеграционно тестване (Integration testing)** – тестване на интерфейси и взаимодействието между компоненти при интегрирането им, както по метода на бялата, така и на черната кутия. Управлението на изпълнението на тестовете става чрез машина на крайни състояния.
4. **Системно тестване (System testing)** – тестване на външните системни операции като цяло, от гледна точка на клиента, т.е. по метода на черната кутия. Тества се чрез машини на крайните състояния и модел на Марков.
5. **Тестване за приемане на системата (Acceptance testng)** – тестване готовността на продукта за доставяне на крайните потребители. Извършва се срещу среда където са разгърнати всички компоненти на системата като настъпва в края на системното тестване. При процеси като scrum/kanban се извършва поне веднъж в края на всеки sprint, като част от тестовия цикъл, преди да се направи release. Техниките за тестване, които се използват за статистическо тестване, базирано на употреба и профили на Муса и Марков.

Някои други видове тестове са:
- **Тестване, базирано на дефекти** – тестване при което се използват открити или потенциални дефекти. Стратегии за тестване са инжектиране на дефекти и тестване на мутации.
- **Бета тестване** – тестване при контролирана и ограничена доставка на продукта до крайните потребители.
- **Регресионно тестване** – използва се за проверка дали съществуващите софтуерни функции са засегнати от новите версии на продукта. Има фокус над интеграционното тестване между новите и старите компоненти.
- **Диагностично тестване** – пресъздаване и диагностициране на проблеми, възникнали при клиента. Реализира се чрез изпълнение на последователност от свързани тестове като се прилага при инспекция на качеството.

5. Измерване и метрики

5.1. Дефиниции
Измерване наричаме процес, при който в съответствие с определени правила, на характеристиките на изследвания обект се съпоставят стойности.

Мярка наричаме стойност съпоставена на някое измерване. Следователно тя показва състоянието на измерваната характеристика.

Метрика на дадена характеристика, наричаме функцията съпоставяща нейните състояния на мерките им.

5.2. Измерване

Нивата на измерване са:

- **Номинална (Nominal) скала** – изброимо множество от категории/стойности, които се присвояват, без да се взимат предвид количествени съотношения;
- **Порядкова (Ordinal) скала** – наредено множество от категории;
- **Интервална (Interval) скала** – числови стойности, като разликата между всеки две последователни е една и съща. Могат да се прилагат математически операции;
- **Относителна (Ratio) скала** – интервална скала, където съществува ясно дефинирана мярка 0.

Важни свойства на измерването са:

- **Обективност** – получените мерки не зависят от субекта, извършващ измерването;
- **Надеждност (Еднозначност)** – еднакви резултати при еднакви условия;
- **Валидност** – отразяват реално свойствата на измервания обект;
- **Точност (Accuracy)** – мярката има необходимата различаваща способност.

Дейности при измерването са:

1. Формулиране на метрична система;
2. Събиране на данни;
3. Анализиране;
4. Интерпретиране;
5. Връщане на обратна връзка.

5.3. Характеристики на метрики

Характеристики

- Общи изисквания
  - **Надеждна (Еднозначна)** – еднакви резултати при еднакви условия;
  - **Валидна** – измерва искания атрибут;
  - **Уместна** – измерва значим (нетривиален) атрибут;
  - **Взаимно изключваща** – не измерва вече измерен атрибут.
- Оперативни изисквания:
  - Неподатлива на предубедени намеси от заинтересовани страни;
  - Не изисква независимо събиране на данни.

5.4. Класификация на метрики

Класификация в зависимост от:

- **предназначението** – оценяване на необходими ресурси, производителността на разработчиците, прогнозиране на надеждност на продукта и други;
- **целта на прилагане на метриката** – за оценяване, за прогнозиране, за debug-ване и други;
- **типа на изследвания обект** – метрики на софтуерен продукт, процес, проект;
- **начина на получаване на информацията** – регистрационен подход, измервателен подход.

5.5. Метрики за качество на софтуерен процес

- Метрики за **честота на грешките** – размер на софтуера, брой грешки;
- Метрики за **сериозност на грешката** – средна сериозност на грешките в кода, средна сериозност на грешките в разработката.

5.6. Метрики за качество на софтуерен продукт

- Средно време до отказ (**MTTF**);
- Процент грешки (**Defect rate**);

- Брой проблеми идентифицирани от потребителите (**Customer problems**);
- Удовлетвореност на потребителите (**Customer satisfaction**).