Практический курс тестирования программного обеспечения

Урок 2

Test Club 2014

http://www.testclub.com.ua

План занятия:

1. Классификация видов тестирования
2. Уровни тестирования

1. Классификация видов тестирования

1.1. По знанию системы

1.2. По объекту тестирования

1.3. По субъекту тестирования

1.4. По позитивности сценариев

1.5. По степени автоматизации

1.6. По статичности

1.7. По времени проведения тестирования

1.8. По степени изолированности компонентов

1. Классификация видов тестирования

1.1. По знанию системы

Выделяют три вида тестирования по знанию системы:

- Тестирование Чёрного ящика (Black – Box testing) – тестирование ПО в том виде, в котором его будет использовать конечный пользователь
- Тестирование Белого ящика (White – Box testing) – тестирование исходного кода ПО и его внутренней структуры
- Тестирование Серого ящика (Grey – Box testing) – тестирование ВВ с учетом знаний внутренней структуры и кода

1. Классификация видов тестирования
1.1. По знанию системы

При тестировании чёрного ящика, тестировщик имеет доступ к ПО только через те же интерфейсы, что и заказчик или пользователь (Front – End, Browser, Desktop form), либо через внешние интерфейсы (Web services, специализированные службы), позволяющие другому компьютеру либо другому процессу подключиться к системе для тестирования.

При данном подходе не используется знание о внутреннем устройстве тестируемого объекта.

1. Классификация видов тестирования
1.1. По знанию системы

Как правило, тестирование чёрного ящика ведётся:

- с использованием Functional Specification, SRS, TD, описывающих требования к системе
- если документация отсутствует – то на основании Product Backlog, который содержит пожелания заказчика к создаваемой системе.
- Если нет требований ни в каком виде, то:
  - они должны быть созданы на основе коммуникаций с заказчиком (звонки, встречи, письма, визиты)
  - или же написаны бизнес – аналитиком или тест – дизайнером и утверждены с заказчиком

1. Классификация видов тестирования
1.1. По знанию системы
Приёмы при тестировании чёрного ящика

Эквивалентное разбиение

Суть метода подхода эквивалентного разбиения состоит в том, что выделяются множества значений, для которых система применяет одинаковую логику. Совокупность таких множеств называют **классами эквивалентности.**

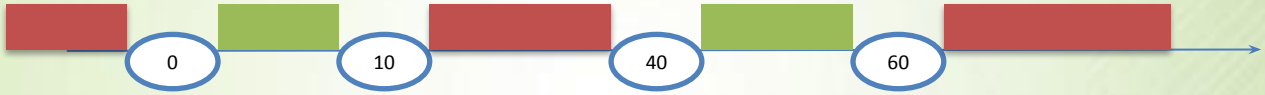
Пример: В поле можно ввести значения от 0 до 10 включительно, и от 40 до 60 включительно

Вопрос: Сколько классов эквивалентности? Что тестируем?

Ответ: 5 классов

1. Классификация видов тестирования
1.1. По знанию системы
Приёмы при тестировании чёрного ящика

Есть два множества корректных значений, и три множества некорректных значений



Тестируем несколько значений из диапазонов < 0, от 10 до 40, > 60

Тестируем несколько значений из диапазонов (0, 10) и (40, 60) не включительно

1. Классификация видов тестирования
1.1. По знанию системы
Приёмы при тестировании чёрного ящика

Анализ причинно-следственных связей

Шаги, которые необходимо предпринять для анализа:

- Спецификация разбивается на рабочие участки (различные функциональности, отдельные модули, отдельные компоненты или группы компонент).
  Пример: Интернет магазин состоит из модуля регистрации, каталога, поиска, личного кабинета, корзины, страницы оплаты.

1. Классификация видов тестирования
1.1. По знанию системы
Приёмы при тестировании чёрного ящика

- В спецификации определяются множество причин и следствий. Под причиной понимается отдельное входное условие или класс эквивалентности. Следствие представляет собой выходное условие или преобразование системы.
  Пример: при тестировании формы логина пользователя есть ***три входа:***
  - корректный логин / корректный пароль
  - некорректный логин / любой пароль
  - корректный логин / некорректный пароль

  два выхода:
  - пользователь зашёл в систему
  - пользователь остался на странице ввода логина /пароля

1. Классификация видов тестирования
1.1. По знанию системы
Приёмы при тестировании чёрного ящика

- На основе анализа семантического (смыслового) содержания спецификации строится таблица истинности, в которой последовательно перебираются всевозможные комбинации причин и определяются следствия для каждой комбинации причин. Таблица снабжается примечаниями, задающими ограничения и описывающими комбинации, которые невозможны.
  Пример: при кросс – платформенном тестировании выделяют список браузеров и список операционных систем

1. Классификация видов тестирования
1.1. По знанию системы
Приёмы при тестировании чёрного ящика

Предугадывание ошибки, позитивные/негативные тесты

Необходимо составить список, который перечисляет возможные ошибки и ситуации, в которых эти ошибки могли проявиться. На основе списка составляются негативные тесты.

Негативные тесты предполагают корректное поведение (correct handling) при обработке (processing) заранее некорректно введённых данных – предупреждения, сообщение об ошибках, невозможность перехода на следующий шаг.

Correct handling of incorrect data with warnings, errors and no possibility to proceed further.

1. Классификация видов тестирования
1.1. По знанию системы
Приёмы при тестировании чёрного ящика

Предугадывание ошибки, позитивные/негативные тесты

Позитивные тесты выполняются с использованием данных или сценариев, которые соответствуют нормальному (штатному, ожидаемому) поведению системы.

Основной целью позитивного тестирования является проверка того, что при помощи системы можно делать то, для чего она создавалась.

1. Классификация видов тестирования
1.1. По знанию системы

При тестировании (white-box testing), разработчик теста (Tester or Developer) имеет доступ к:

- исходному коду программ
- базе данных (таблицам, хранимым процедурам, триггерам, индексам)

Техники белого ящика:

- **покрытие операторов** — каждая ли строка исходного кода была выполнена и протестирована
- **покрытие условий** — каждая ли точка решения (вычисления истинно ли или ложно выражение) была выполнена и протестирована

1. Классификация видов тестирования
1.1. По знанию системы
Техники тестирования белого ящика

- **покрытие путей** — все ли возможные пути через заданную часть кода были выполнены и протестированы
- **покрытие функций** — каждая ли функция программы была выполнена
- **покрытие вход/выход** — все ли вызовы функций и возвраты из них были выполнены.

Примеры: http://bit.ly/1guXIVh

1. Классификация видов тестирования
1.1. По знанию системы

Метод белого ящика используется при написании Unit Tests.

Юнит тест содержит в себе булевую функцию (http://bit.ly/1ve5iri) **assert**, которая возвращает значения TRUE и FALSE. Одним из критериев качества программного обеспечения является то, что:

- реализованные функциональности покрыты юнит тестами
- все юнит тесты проходят (на выходе результат – TRUE).

Таких юнит тестов за частую в системе есть несколько сотен.

1. Классификация видов тестирования
1.1. По знанию системы

Пример:

Допустим, в системе реализована функция двух переменных myFunction(a1,a2). Согласно функциональной спецификации, она должна выполнять сложение чисел a1 и a2.

```
function testSum()
{ a1=4;
  a2=5;
  c = myFunction(a1,a2);
  assertTrue(c=9); }
```

Пример:

http://bit.ly/QPF4uR

1. Классификация видов тестирования
1.1. По знанию системы

При тестировании **Серого ящика** (Grey-box testing) совмещаются приёмы, используемые при тестировании чёрного и белого ящиков. Сочетание происходит таким образом: снаружи на продукт смотрим как на черный ящик, но выбор тестов основываем на знании внутреннего устройства программы, знании ее кода. Этот метод часто используется для тестирования Web приложений, тестировании базы данных, тестировании любых других приложений, где есть доступ к исходному коду.

1. Классификация видов тестирования
1.1. По знанию системы

Пример:

- Используется просмотр кода (Пункт View page source контекстного меню)
- Используется sniffer (анализатор траффика)
- Используется Firebug (надстройка для Firefox)

1. Классификация видов тестирования
1.1. По знанию системы

Метод серого ящика используется при CRUD tests. Название CRUD - это аббревиатура по первым буквам слов **Create – Read – Update – Delete**.

Пример: Тестирование функциональности регистрации пользователя и входа в систему

Create:

Через интерфейс (десктоп или веб) регистрируется новый пользователь. Далее путём специального запроса в базу (обычно такие запросы выглядят примерно так
SELECT * from users where login='<userlogin>';
проводится проверка успешности регистрации и занесения нового пользователя в базу

1. Классификация видов тестирования
1.1. По знанию системы

Read:

Через интерфейс (десктоп или веб) новый пользователь входит в систему. Далее путём специального запроса в базу (обычно такие запросы выглядят примерно так
SELECT * from userSessions u where u.login='userlogin'
проводится проверка того, что пользователь зашёл в систему

1. Классификация видов тестирования
1.1. По знанию системы

Update:

Через интерфейс (десктоп или веб) новый пользователь входит в систему. Через функциональность Пользовательский Аккаунт или Личный Кабинет пользователь меняет свои данные, например, e-mail. Далее путём специальных запросов в базу (обычно такие запросы выглядят примерно так:
SELECT * from where email='<oldemail>';
SELECT * from where email='<newemail>';

проводится проверка того, что пользователь поменял свой e-mail – первый запрос вернёт нулевой результат, а второй – вернёт строку с новым значением поля e-mail

1. Классификация видов тестирования
1.1. По знанию системы

Delete:

Через административную часть сайта (back – end) или через соответствующую функциональность на Front – End администратор удаляет пользователя. Далее путём специального запроса в базу (обычно такие запросы выглядят примерно так

SELECT * from users u where login='<userlogin>';

проводится проверка отсутствия пользователя в базе. При этом в таблице userSessions записи активности данного пользователя могут сохраниться, а могут быть удалены.

?

1. Классификация видов тестирования
1.2. По объекту тестирования

При рассмотрении в плоскости объекта тестирования выделяют **функциональное** и **нефункциональное** тестирование.

1. Классификация видов тестирования
1.2. По объекту тестирования

Функциональное тестирование — это тестирование ПО в целях проверки реализуемости функциональных требований, то есть способности ПО в определённых условиях решать задачи, нужные пользователям. Функциональные требования определяют, что именно делает ПО, какие задачи оно решает.

Функциональные тесты базируются на функциях и особенностях, а также взаимодействии с другими системами, и могут быть представлены на всех уровнях тестирования. Функциональные виды тестирования рассматривают внешнее поведение системы.

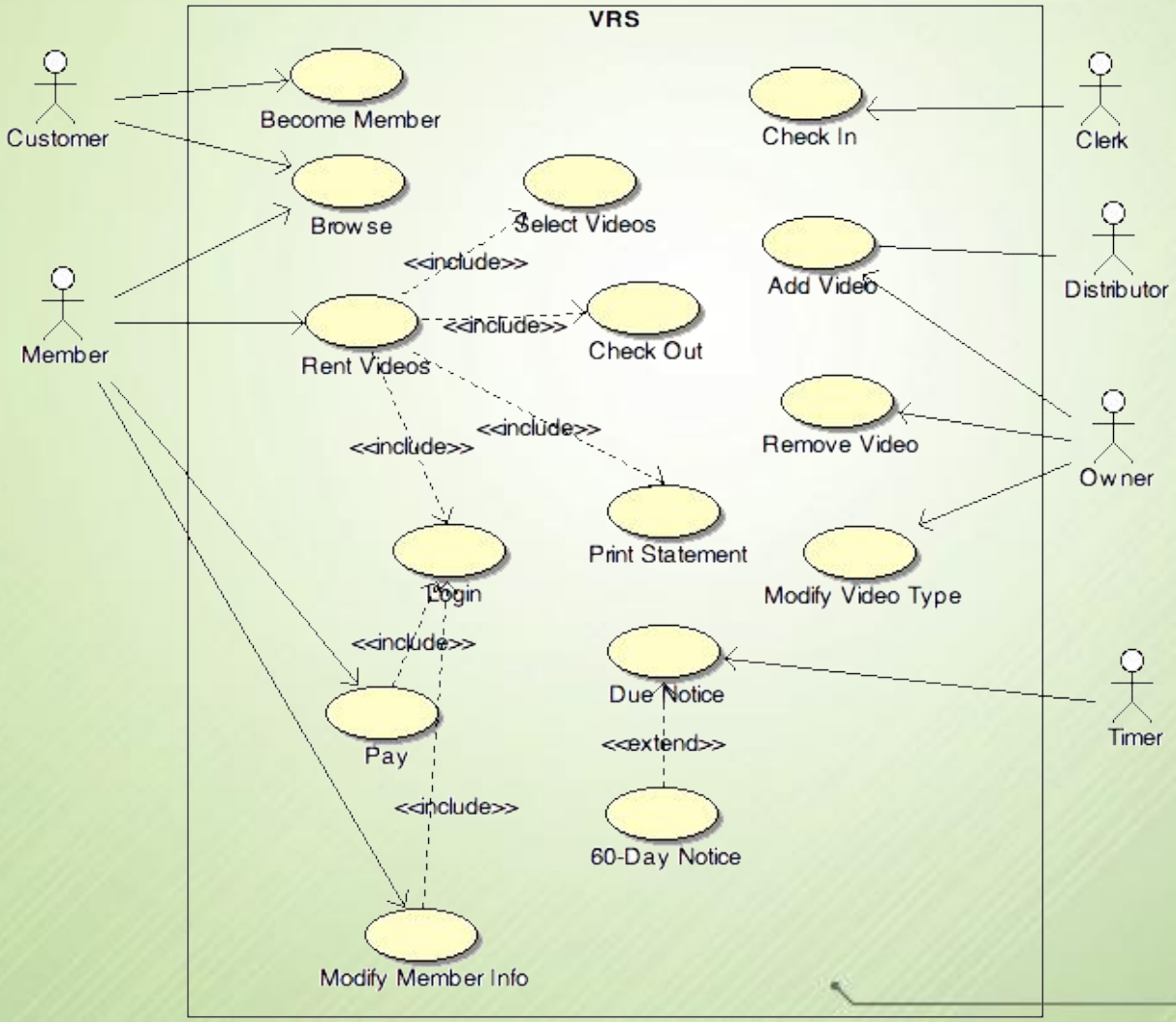
1. Классификация видов тестирования
1.2. По объекту тестирования

Тестирование функциональности может проводится в двух аспектах:

- требования – requirements
- бизнес-процессы – use cases

Пример: Высокоуровневые требования при тестировании магазина по прокату видео - дисков

1. Классификация видов тестирования
1.2. По объекту тестирования



1. Классификация видов тестирования
1.2. По объекту тестирования

Нефункциональное тестирование описывает тесты, необходимые для определения характеристик программного обеспечения, которые могут быть измерены различными величинами. В целом, это тестирование того, "Как" система работает.

1. Классификация видов тестирования
1.2. По объекту тестирования

Виды нефункционального тестирования:

- Тестирование производительности(Performance Testing)
- Объемное тестирование (Volume Testing)
- Тестирование установки (Installation testing)
- Тестирование удобства пользования (Usability Testing)
- Тестирование на отказ и восстановление (Failover and Recovery Testing)
- Конфигурационное тестирование (Configuration Testing)
- Тестирование локализации (Localization testing)

1. Классификация видов тестирования
1.2. По объекту тестирования

Тестирование производительности (Performance testing) включает:

1. Нагрузочное тестирование (Load testing)
2. Стресс тестирование (Stress testing)
3. Тестирование стабильности (Stability testing)

1. Классификация видов тестирования
1.2. По объекту тестирования

Пример: В системе могут одновременно находится 500 пользователей (банковская система)

Нагрузочное тестирование (Load testing)

В систему входят постепенно 1, 2, 5, 10, 20, 50, 100, 200, 300, 400, 500 пользователей. Идёт постепенная нагрузка до максимума

Снимаемые показатели: Время отклика системы, загрузка процессора и оперативной памяти.

1. Классификация видов тестирования
1.2. По объекту тестирования

Стресс тестирование (Stress testing)

В систему входят 1000, 2000, 4000 пользователей. Увеличение максимума в 2, 4, 8 раз

Снимаемые показатели:

Время отклика системы, загрузка процессора и оперативной памяти, проверка на устойчивость

1. Классификация видов тестирования
1.2. По объекту тестирования

Тестирование стабильности (Stability testing)

В систему входят 250 пользователей и работают 8 часов

1. Классификация видов тестирования
1.2. По объекту тестирования

Usability testing предполагает проверку удобности пользования интерфейсом desktop или web приложения.
***Пример**:* необходимо купить товар через Интернет сайт.
Критерии: Удобен ли интерфейс, навигация, понятность, не рябит ли сайт в глазах, нет ли миллиона шагов для выполнения простой операции

GUI testing - проверка всех элементов (controls) приложения (страницы, ссылки, кнопки, формы, radio-buttons, check-boxes). Также проверяется фирменный стиль, цветовая гамма и логика присутствия / отсутствия элементов.

***Пример**:* файл со спецификацией на user interface для сайта iplay.com – «GUI Testing requirements Example 1.pdf»

1. Классификация видов тестирования
1.2. По объекту тестирования

При **Security testing** (тестировании безопасности) тестируются конфиденциальность, целостность и доступность данных. Более детально - http://bit.ly/1szGMwL

Пример: тестирование логина, прав и ограничений пользователя, безопасности протокола передачи данных, Cache, Cookies

1. Классификация видов тестирования
1.2. По объекту тестирования

При **Localization testing** (тестирование локализации) тестируются интерфейс пользователя и файлы с данными.

Основные объекты:

- Operating System – операционная система
- Keyboards – раскладки клавиатуры
- Text Filters – текстовые фильтры
- Hot keys – горячие клавиши
- Spelling Rules – правила написания, Sorting Rules – правила сортировка

1. Классификация видов тестирования
1.2. По объекту тестирования

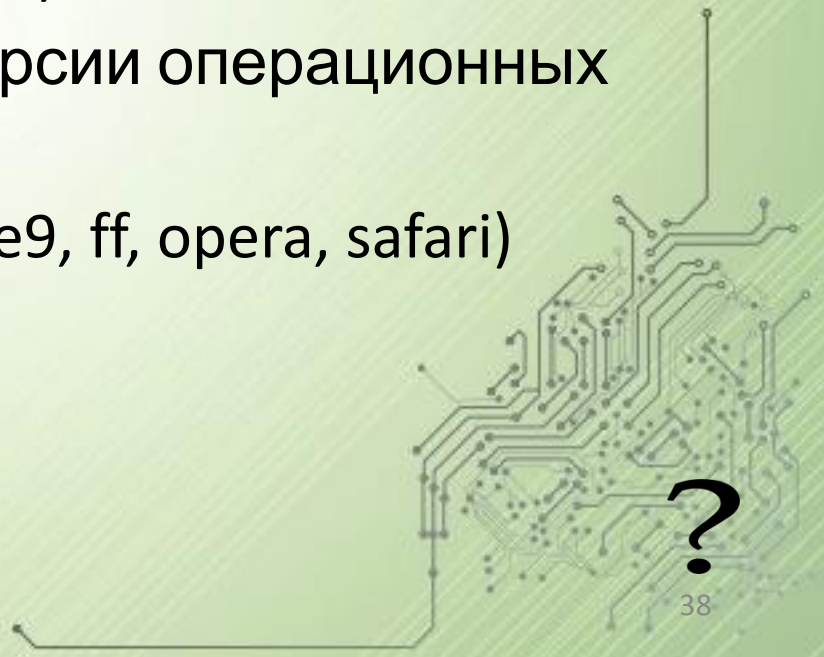
Основные объекты Localization testing:

- Upper and Lower case conversions – правила использования заглавных букв
- Printers – печать на принтере, Size of Papers – размеры бумаги
- Mouse – настройки мыши, Date formats – форматы дат
- Restricted content – доступность данных в разных странах

1. Классификация видов тестирования
1.2. По объекту тестирования

При **Compatibility testing** (тестирование совместимости) тестируются следующие области:

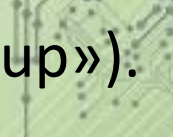
1. Различные операционные системы (xp, 7, 8, mac, ubuntu)
2. Базы данных (MS SQL, Oracle)
3. x86 и x64 (32 и 64 битные версии операционных систем)
4. Разные браузеры (ie7, ie8, ie9, ff, opera, safari)



1. Классификация видов тестирования
1.3. По субъекту тестирования

При рассмотрении в плоскости субъекта тестирования выделяют **Альфа тестировщик**(Alpha tester) и **Бета тестировщик**(Beta tester) тестирование.

- **Альфа тестировщик**(Alpha tester) – люди, принимающие участие в тестировании и работающие внутри компании (тестировщики, программисты, бизнес аналитики, HR,…). Выполняют свои тесты до релиза.
- **Бета тестировщик**(Beta tester) - люди, принимающие участие в тестировании но не работающие в компании. Обычно это пользователи ПО (заказчики или выбранная «target group»). Выполняют свои тесты до релиза.



1. Классификация видов тестирования
1.4. По позитивности сценариев

При рассмотрении в плоскости позитивности сценариев тестирования выделяют **Позитивное тестирование**(Positive testing) и **Негативное тестирование**(Negative testing) тестирование.

- **Позитивное тестирование**(Positive testing) – проверяет сценарии, предполагающие нормальное («правильное») использование и/или работу системы.
- **Негативное тестирование**(Negative testing) – противоположность позитивному, проверяет сценарии связанные с потенциальной ошибкой или с потенциальным дефектом в системе.

?

1. Классификация видов тестирования
1.5. По степени автоматизации

При рассмотрении в плоскости степени автоматизации тестирования выделяют **Ручное тестирование**(Manual testing), **Автоматизированное тестирование**(Automated testing) и **Полуавтоматизированное тестирование**(Semi automated testing)

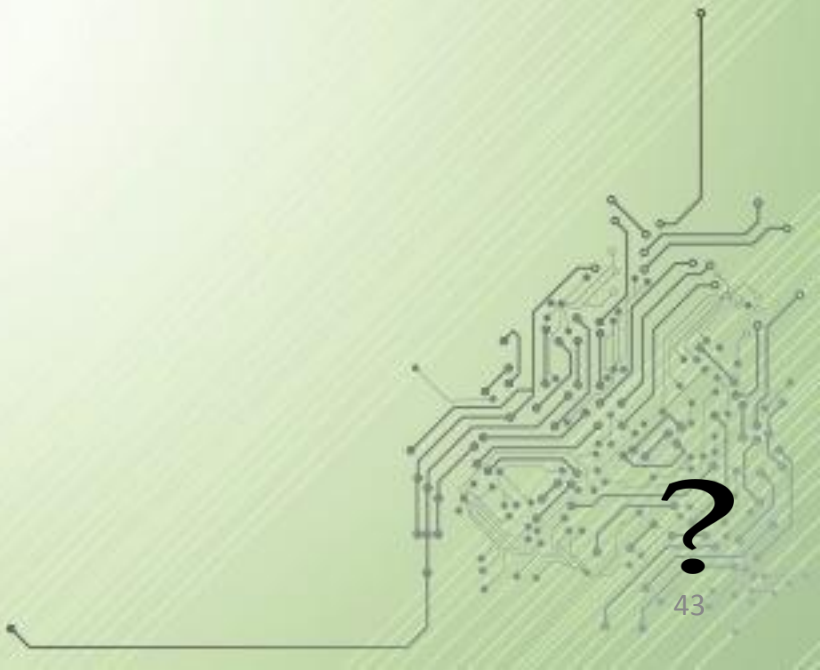
- **Ручное тестирование**(Manual testing) – выполняется человеком без использования дополнительных инструментов, автоматизирующих процесс тестирования.

1. Классификация видов тестирования
1.5. По степени автоматизации

- **Автоматизированное тестирование**(Automated testing) – выполняется без участия человека с помощью специальных инструментов (программы для тестирования скорости и надежности системы, программы для регрессивного тестирования…)

1. Классификация видов тестирования
1.5. По степени автоматизации

- **Полуавтоматизированное тестирование**(Semi automated testing) – Смесь предыдущих двух видов (например с помощью инструмента автоматизации создаем аккаунт пользователя в системе а потом вручную создаем покупки в интернет магазине)



1. Классификация видов тестирования
1.6. По статичности

При рассмотрении в плоскости статичности тестирования выделяют **Статическое тестирование** (Static testing) и **Динамическое тестирование**(Dynamic testing).

- **Статическое тестирование**(Static testing) не требует запуска программного кода (тестирование документации, требований запуск анализаторов кода)
- **Динамическое тестирование**(Dynamic testing) подразумевается во всех видах тестирования при которых программный продукт работает

?

1. Классификация видов тестирования
1.7. По времени проведения тестирования

При рассмотрении в плоскости времени проведения тестирования выделяют:

- **«Дымовое тестирование»** (Smoke testing) - Тестируются основные сценарии и воспроизводимость известных критических багов (не должны воспроизводиться) в течении 5 – 15 минут, чтобы понять, не повреждён ли основной функционал (логин, кабинет, добавление в корзину, поиск – для сайта интернет магазина)
- **Тестирование новой функциональности** (New feature testing)

1. Классификация видов тестирования
1.7. По времени проведения тестирования

- **Регрессионное тестирование** (Regression testing) - выполняется при выходе новой версии программного продукта с целью проверки, не был ли повреждён старый работающий функционал новыми изменениями. В него входит проверка основных бизнес транзакций и проверка ранее найденный наиболее критичных багов. Зачастую подлежит автоматизации.
- **Тестирование приемки** (Acceptance testing) - тестирование по заранее подготовленным сценариям со стороны заказчика. Зачастую проводится перед выходом в Production (использование продукта широкими массами или конкретными пользователями)

1. Классификация видов тестирования
1.8. По степени изолированности компонентов

Компонентное (модульное) тестирование (component/unit testing)

Пример: Тестирование одной операции или одного модуля (регистрация пользователя, кабинет пользователя, выбор товара)

1. Классификация видов тестирования
1.8. По степени изолированности компонентов

Интеграционное тестирование (integration testing)

Пример: Тестирование взаимодействия двух компонентов (пользователь зарегистрировался и увидел свои данные в личном кабинете

1. Классификация видов тестирования
1.8. По степени изолированности компонентов

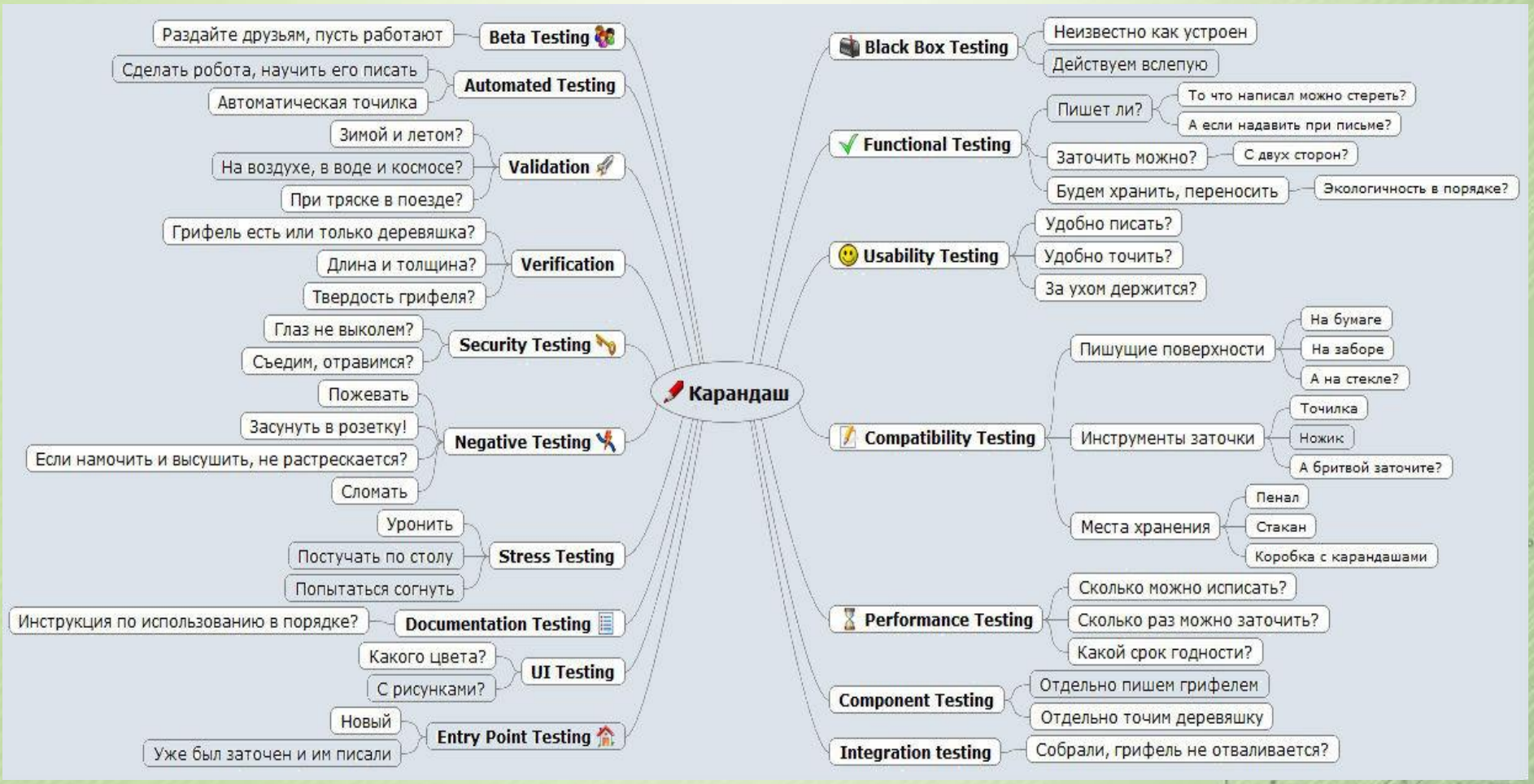
Системное тестирование (system/end-to-end testing)

Тестирование 3 и более компонент или систем, которые при связке позволяют провести бизнес транзакцию.

Пример: Пользователь зарегистрировался, залогинился, положил товары в корзину, добавил кредитную карту, оплатил товары кредитной картой, получил подтверждение на электронную почту

?

1. Классификация видов тестирования
Пример тестирования карандаша



1. Классификация видов тестирования Тестирование двери

План проверки двери

1. Функциональные проверки.

1.1. Проверить, что дверь открывается.
1.2. Проверить, что дверь закрывается.
1.3. Попытаться закрыть уже закрытую дверь.
1.4. Попытаться открыть уже открытую дверь.

2. GUI (интерфейс пользователя)

2.1. Проверить табличку на двери.
2.2. Проверить покраску двери.
2.3. Проверить наличие дверной ручки.

1. Классификация видов тестирования Тестирование двери

3. Permissions

3.1. Проверить, что правильным ключом дверь открывается.
3.2. Проверить, что неправильным ключом дверь не открывается.
3.3. Проверить, что закрытую на ключ дверь нельзя открыть.
3.4. Проверить, что не закрытую на ключ дверь можно открыть без ключа.
3.4. Позвонить в дверь. Если там никого нет, дверь не должна открыться сама.
3.5. Постучать в дверь. Если там кто-то есть и он спросит “кто?”, ответить “Полиция”. Дверь должна открыться.

1. Классификация видов тестирования Тестирование двери

4. Stress/Loading

4.1. Открывайте и закрывайте дверь со скоростью 120 циклов в минуту

4.2. Открывайте и закрывайте дверь со скоростью 6 раз в минуту на протяжении 48 часов.

4.3. Стучите в дверь с частотой 1200 стуков в минуту.

4.4. Стучите в дверь с частотой 10 раз в минуту на протяжении 24 часов.

4.5. Открывайте и закрывайте дверь ключом на протяжении 12 часов.

1. Классификация видов тестирования Тестирование двери

5. End to end

5.1. Постучать в дверь. Позвонить в звонок. Открыть ключом. Открыть дверь. Закрыть дверь. Закрыть ключом. Прочитать табличку на двери. Ничего не отвалилось, не звякает, не взрывается?

6. Usability

6.1. Проверить, что ручка двери помещается в ладонь.
6.2. Проверить, что ручка находится именно на двери, а не на соседней стене на высоте 20 см.
6.3. Проверить, что высота двери больше человеческого роста

1. Классификация видов тестирования Тестирование двери

Добавка

1. Начать с использования двери одним человеком. Увеличивать количество пользователей с шагом 5 человек в 5 сек. Увеличивать нагрузку, пока дверь не сломается.

2. Проверка документации к двери – инструкции пользователя, технического паспорта..

3. Проверка сердцебиения и давления открывающего. Действия по открыванию-закрыванию не должны пожирать все ресурсы пользователя.

1. Классификация видов тестирования Тестирование двери

Добавка

4. Проверить влияние функционирования двери на появление трещин в стене.

5. White box tests: проверить волокна древесного полотна на параллельность. Проверить отдельные элементы (классы) на предмет избыточности (а может там 6 замочных скважин). Проверка алгоритма запирания двери.

1. Классификация видов тестирования Тестирование двери

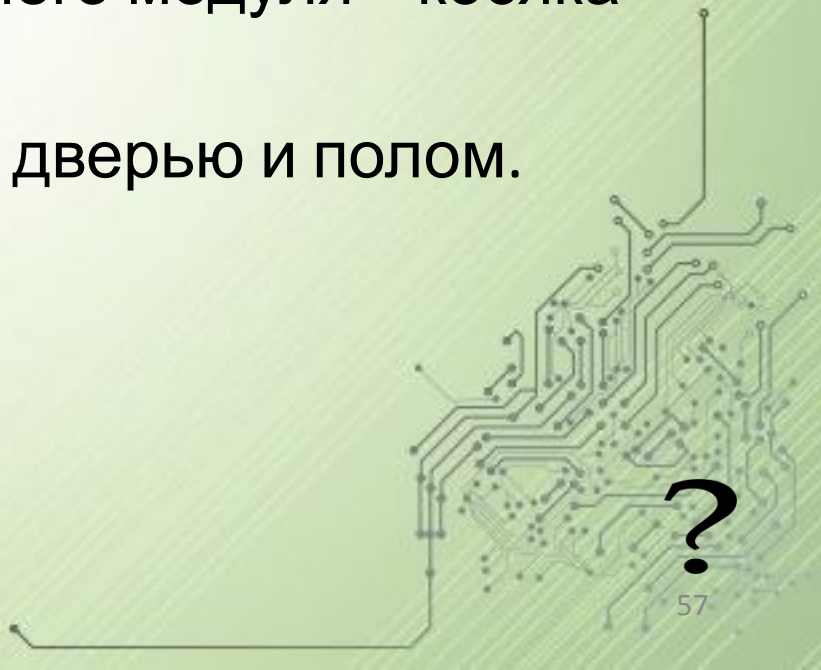
В релизе не проверено

Отсутствуют проверки на стрессовость (удар ногой или головой)

Отсутствуют проверки на крепеж двери к косяку

Отсутствуют проверки соседнего модуля – косяка (зазоры между ним и дверью)

Отсутствуют проверки между дверью и полом.



2. Уровни тестирования

Тестирование на разных уровнях производится на протяжении всего жизненного цикла разработки и сопровождения программного обеспечения. Уровень тестирования определяет то, над чем производятся тесты: над отдельным модулем, группой модулей или системой, в целом. Проведение тестирования на всех уровнях системы - это залог успешной реализации и сдачи проекта.

2. Уровни тестирования

Уровни тестирования:

2.1. Компонентное или Модульное тестирование (Component or Unit Testing)

2.2. Интеграционное тестирование (Integration Testing)

2.3. Системное тестирование (System Testing)

2.4. Приемочное тестирование (Acceptance Testing)

2. Уровни тестирования
2.1. Компонентное или Модульное тестирование

Компонентное или Модульное тестирование предназначено для проверки функционирования отдельно взятого компонента или модуля системы (одна функция программы)

Этот вид тестирования относится к методу белого ящика и обычно выполняется программистами. Все найденные дефекты, как правило исправляются в коде без формального их описания в Bug Tracking System.

Используя этот вид тестирования легко изменять и улучшать код программы, т.к. протестировать отдельный модуль после изменения достаточно просто. В одном модуле получается достаточно маленький набор вариантов развития событий, и достаточно легко

2. Уровни тестирования
2.2. Интеграционное тестирование

Интеграционное тестирование предназначено для проверки связи между компонентами, а также взаимодействия с различными частями системы (операционной системой, оборудованием либо связи между различными системами).

Снизу вверх (Bottom Up Integration): Все низкоуровневые модули, процедуры или функции собираются воедино и затем тестируются. После чего собирается следующий уровень модулей для проведения интеграционного тестирования. Данный подход считается полезным, если все или практически все модули, разрабатываемого уровня, готовы. Также данный подход помогает определить по результатам тестирования уровень готовности приложения

2. Уровни тестирования
2.2. Интеграционное тестирование

Сверху вниз (Top Down Integration): Вначале тестируются все высокоуровневые модули, и постепенно добавляются низкоуровневые. Все модули более низкого уровня симулируются заглушками с аналогичной функциональностью, затем по мере готовности они заменяются реальными активными компонентами.

Большой взрыв ("Big Bang" Integration): Все или практически все разработанные модули собираются вместе в виде законченной системы или ее основной части, и затем проводится интеграционное тестирование. Такой подход очень хорош для сохранения времени. Однако если тест кейсы и их результаты записаны не верно, то сам процесс

?

2. Уровни тестирования
2.2. Системное тестирование

Основная задача системного тестирования – проверка функциональных и не функциональных требований в системе в целом. При этом выявляются дефекты, такие как неверное использование ресурсов системы, непредусмотренные комбинации данных пользовательского уровня, несовместимость с окружением, непредусмотренные сценарии использования, отсутствующая или неверная функциональность, неудобство использования и т.д.

Во время системного тестирования рекомендуется использовать окружение максимально приближенное к тому, на которое будет установлен продукт после выдачи.

?

2. Уровни тестирования
2.2. Системное тестирование

Можно выделить два подхода к системному тестированию:

- на базе требований (requirements based): для каждого требования пишутся тестовые случаи (test cases), проверяющие выполнение данного требования.
- на базе случаев использования (use case based): на основе представления о способах использования продукта создаются случаи использования системы (Use Cases). По конкретному случаю использования можно определить один или более сценариев. На проверку каждого сценария пишутся тест кейсы (test cases), которые должны быть протестированы.

2. Уровни тестирования
2.4. Приемочное тестирование

Формальный процесс тестирования, который проверяет соответствие системы требованиям и проводится с целью:

- определения удовлетворяет ли система приемочным критериям;
- вынесения решения заказчиком или другим уполномоченным лицом принимается приложение или нет.

Приемочное тестирование выполняется на основании набора типичных тестовых случаев и сценариев, разработанных на основании требований к данному приложению.

2. Уровни тестирования
2.4. Приемочное тестирование

Решение о проведении приемочного тестирования принимается, когда:

- продукт достиг необходимого уровня качества;
- заказчик ознакомлен с Планом Приемочных Работ (Product Acceptance Plan) или иным документом, где описан набор действий, связанных с проведением приемочного тестирования, дата проведения, ответственные и т.д.

Фаза приемочного тестирования длится до тех пор, пока заказчик не выносит решение об отправлении приложения на доработку или выдаче приложения.

?

Домашнее задание

1. Написать test case headers для всех видов и типов по аналогии с карандашом для:
   Смартфона. Формат – Microsoft Excel, имя файла – SmartPhone_Excel_Test_Case_Headers_[Name]_[Surname].xls
2. Используя требования к смартфону из урока 1, дополнив количество требований до 10, оформить traceability matrix
   формат – template made in Microsoft Excel, имя файла – TM_Phone_[Name]_[Surname].xls
3. Придумать свои примеры для каждого вида и типа тестирования, оформить в виде слайдов презентации по аналогии с теми примерами, которые были приведены на занятии
   формат – Presentation in Xmind, имя файла – Testing_Type_Examples_[Name]_[Surname].xmind*
4. Перевод на русский http://bit.ly/1ozbJ6F, http://bit.ly/1nHyUKb
5. Перевод на английский - http://bit.ly/1oOJm28
   Оба перевода – в один файл Transaltions_Lesson2_[Name]_[Surname].doc

*Xmind can be download from: http://www.xmind.net

Домашнее задание

Четыре файла, которые будут сделаны в процессе выполнения домашнего задания, необходимо вложить в ОДНО письмо!

Тема письма: Homework_Lesson2_[Name]_[Surname]

Тело письма по аналогии с приведённым в презентации к уроку 1.

Также необходимо зайти в свою папку (“User*”) и скопировать туда файлы, которые будут сделаны в процессе выполнения домашнего задания.

Нажать на вашей папке (“User*”) правой кнопкой и выполнить операцию **SVN commit**