Тестирование ПО

Тестирование ПО

- Что такое тестирование?
- Чем придется заниматься?
- Что нужно знать?
- К чему нужно быть готовым?

Тестирование ПО

Тестирование ПО - процесс проверки соответствия заявленных к продукту требований и реально реализованной функциональности, осуществляемый путем наблюдения за его работой в искусственно созданных ситуациях и на ограниченном наборе тестов, выбранных определенным образом

QC ≠ QA

Quality Control (контроль качества) - контроль соответсвия определенных артефактов ожиданиям.

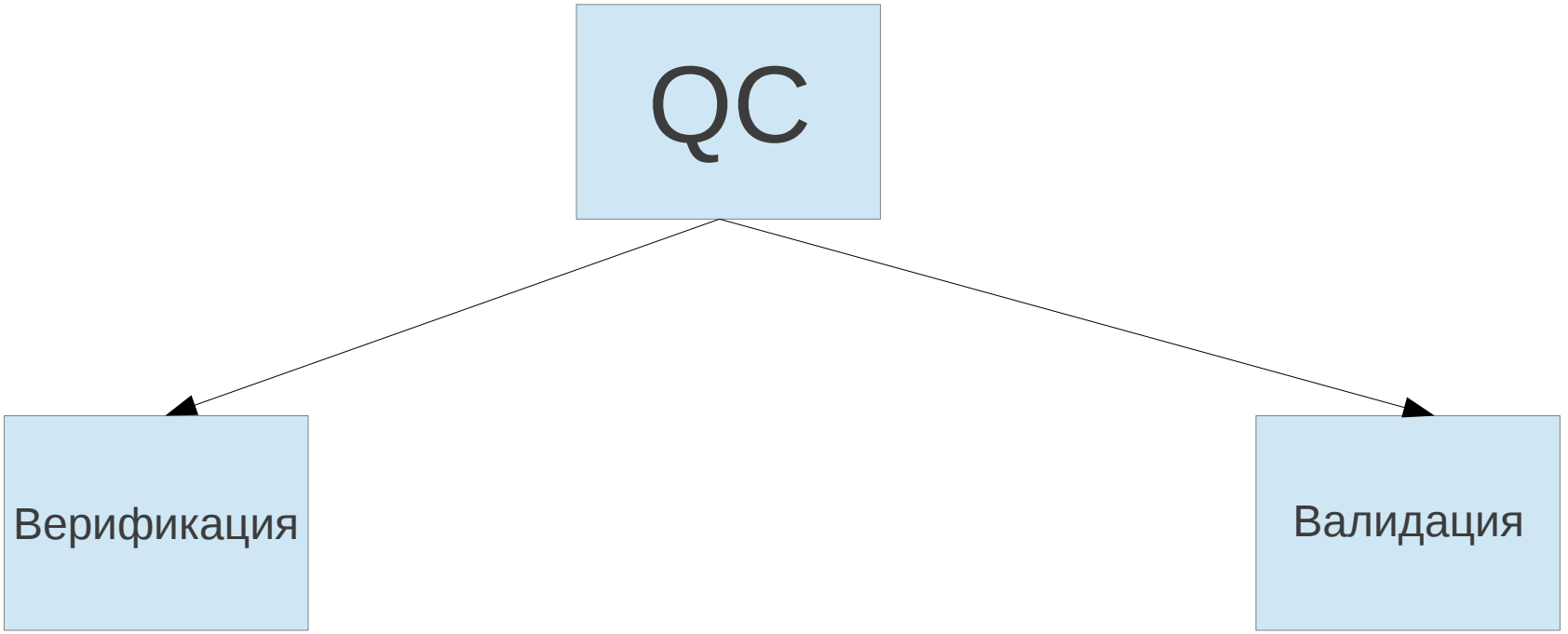
Пример:

•Проверка соответствия функционала спецификации.

•Инспекция документов относительно требований к их написанию, содержанию и формату.

•Инспекция кода относительно стандарта кодирования, архитектурной документации, требованиям безопасности и т.п.

QC ≠ QA



QC ≠ QA

Задача QC – оценка качества продукта

Ответственность специалиста по контролю качества – быстро и качественно предоставить информацию о качестве продукта. Предоставленная информаци может быть использованна для изменения продукта.

QC ≠ QA

Quality Assurance (обечпечение качества) – элемент Quality Management (управление качеством), который отвечает за качество процессов, которые в свою очередь применяются для создания артефактов.

Чем я буду заниматься?

•Изучать требования к продукту

•Составлять тестовую документацию, описывающую искусственные ситуации для верификации и валидации продукта

•Моделировать искусственные ситуации в соотвествии с тестовой документацией

•Управлять выполнением программы в искусственных ситуациях

•Наблюдать за поведением программы и сравнивать наблюдаемое поведение с ожидаемым

•Предоставлять информацию о найденных несоответсвиях в форме описанных дефектов

•Предоставлять оценку качества продукта в форме отчета о тестировании

Как мне делать это хорошо?

Тестирование не соревнование по поиску ошибок. Не надо стараться найти как можно больше ошибок, нужно **пропустить как можно меньше**.

Типичные ошибки тестировщика

Моя задача? Завести как можно больше багов.

Где их найти? Отыскать в дебрях продукта низкоприоритетный модуль и устроить ему полную проверку.

Я нашел труднопроизводимый баг и три бага на опечатки, что делать? Три бага лучше, чем один, статистика – наше всё.

Что тестировать в первую очередь? Самые нестандартные ситуации, точно где-то свалится.

Правила хорошего тестировщика

Моя задача? Пропустить как можно меньше приоритетных для пользователя багов.

Где их найти? В самых приоритетных для пользователя местах. Даже если они стабильно и успешно работают, я всё равно буду проверять основные пользовательские сценарии.

Я нашел трудновоспроизводимый баг, что делать? Если это важный функционал, то выяснять «что не так», «как правильно», даже если на это уйдёт много времени и сил.

Что тестировать в первую очередь? Самые стандартные ситуации. Выполнение самого основного сценария в самых основных условиях, чтобы убедиться, что самый важный функционал работает.

Как быть эффективным?

- Анализируйте и документируйте
- Оценивайте свою работу
- Обсуждайте и эскалируйте
- Знайте пользователей
- Повышайте техническую квалификацию

Что нужно знать?

- Уровни тестирования
- Виды тестирования
- Типы тестирования
- Основные документы

Уровни тестирования

Модульное (unit-testing) — проверка на корректность отдельного модуля исходного кода программы.

Интеграционное (integration testing) — проверка связи между разными модулями.

Системное — поиск несоответсвие общим стандартам и спецификации в целом

Приёмочное — проходит как правило уже с клиентом при передаче проекта и показывает готовность работы с клиентами. Именно на этой стадии появляются альфа-версии и бета-версии.

Типы тестирования

- Функциональное
- Нефункциональное
- Тестирование изменений

Функциональное тестирование

Функциональные тесты базируются на функциях и особенностях, а также взаимодействии с другими системами, и могут быть представлены на всех уровнях тестирования: модульном , интеграционном , системном и приёмочном.

Виды функциональных тестов:

•*Функциональное тестирование* (Functional testing) — проверка реализуемости функциональных требований, то есть способности ПО в определённых условиях решать поставленные задачи.

•*Тестирование безопасности* (Security and Access Control Testing)

•*Тестирование взаимодействия* (Interoperability Testing) — тестирование, проверяющее способность приложения взаимодействовать с одним и более компонентами или системами и включающее в себя тестирование совместимости.

Нефункциональное тестирование

Нефункциональное тестирование описывает тесты, необходимые для определения характеристик программного обеспечения, которые могут быть измерены различными величинами.

Виды нефункциональных тестов:

- Тестирования производительности
  - нагрузочное тестирование (Performance and Load Testing)
  - стрессовое тестирование (Stress Testing)
  - тестирование стабильности (Stability / Reliability Testing)
  - объемное тестирование (Volume Testing)
- Тестирование установки (Installation testing)
- Тестирование удобства пользования (Usability Testing)
- Тестирование отказоустойчивости и восстановления (Failover and Recovery Testing)
- Конфигурационное тестирование (Configuration Testing)

Тестирование изменений

После проведения необходимых изменений, таких как исправление бага/дефекта, программное обеспечение должно быть перетестировано для подтверждения того факта, что проблема была действительно решена.

Виды тестирования изменений:

•Дымовое тестирование (Smoke Testing)

•Регрессионное тестирование (Regression Testing)

•Тестирование сборки (Build Verification Test)

•Проверка исправности (Sanity Testing)

Основные документы

- План тестирования
- Тестовая спецификация
- Отчет о тестировании
- Отчет о дефекте

План тестирования

План тестирования (Test plan) описывает задачи, объём работ, стратегию и организацию тестирования проекта.

Как правило, в план тестирования включаются следующие **разделы**:

- цели и задачи тестирования;
- объем тестирования;
- ресурсы;
- тестовая платформа;
- стратегия тестирования, этапы выполнения работ;
- график и ожидаемые трудозатраты;
- управление изменениями;
- доставка новых версий ПО;
- критерии начала и остановки тестирования;
- доставка результатов тестирования;
- оценка рисков.

Тестовая спецификация

Тестовая спецификация (Test Specification) должна содержать перечень тестовых наборов для проверки всех зарегистрированных требований к системе.

Как правило, в тестовую спецификацию включаются следующие **разделы**:

•наименование объекта тестирования;
•определение тестовой задачи;
•предварительные условия-данные для выполнения задачи (среда испытаний);
•описание входных данных с четким указанием реакции системы на их ввод;
•описание последовательности действий пользователя;
•описание требуемых результатов выполнения теста.

Отчет о тестировании

Отчет о тестировании базируется на тестовой спецификации и предоставлят объективные сведения о тестируемом продукте.

Как правило, в отчет о тестировании включаются следующие **разделы**:

•наименование объекта тестирования;
•предварительные условия-данные для выполнения задачи (среда испытаний);
•найденные ошибки (с описанием критических ошибок);
•выводы о соответствии качества продукта ожидаемому.

Отчет о дефекте

Отчет о дефекте (bug report) — документ, содержащий отчет о любом недостатке в компоненте или системе, который может привести компонент или систему к невозможности выполнить требуемую функцию.

Отчет о дефекте

Обязательные элементы

- ID
- Краткое описание
- Компонент
- Окружение
- Приоритет
- Шаги
- Фактический результат
- Ожидаемый результат
- Приложения (скриншоты, логи и т.д)

Отчет о дефекте

ID: PROD-0986

Module: UploadBox

Summary: RadioButton is incorrectly enabled

PRIO 3

Build: Test_V24203_2009.12.09
Environment: IE7, SLES10/RHEL4.8.

_ _
After Frequency RadioButton is switched from ‘Once’ to ‘Daily’ and back to ‘Once’ in ‘Task settings…’ dialog window, all RadioButtons in ‘Keep Log’ frame are disabled
=>

Actual result: All RadioButtons in ‘Keep Log’ frame are disabled only after switching RadioButtons
Expected result: All RadioButtons are disabled just on opening ‘Task settings…’ dialog window as RadioButton is set to “Once” by default.

_ _
How to reproduce the problem:

Pre-conditions: Time controls are enabled, there is at least 1 manageable cluster node

Steps:
1. Run QManager -> go to ‘UploadBox’ module
2. Go to ‘Task Management’ Tab -> press PushButton ‘Add…’
3. Switch Frequency RadioButton to ‘Daily’
4. Switch Frequency RadioButton to ‘Once’
=>
Actual result: All RadioButtons in ‘Keep Log’ frame are disabled (see screenshots attached)
Expected result: All RadioButtons in ‘Keep Log’ frame are disabled after step 2 is done

К чему нужно быть готовым?

- Монотонность
- Ответственность
- Конфликт интересов
- Давление

Задание 1

Протестировать и описать один дефект этой белой чайной фарворовой кружки

Задание 2

Указать и обосновать, какие виды и типы тестирования можно применить при тестировании чугунного чапельника с кленовой ручкой

Задание 3

Составить чеклист для проверки меню “File → Open...” Windows Notepad