Тестирование программного обеспечения

Выполнил: Дорофеев Сергей ПИ-13

История развития

- В 1960-х много внимания уделялось «исчерпывающему» тестированию, которое должно проводиться с использованием всех путей в коде или всех возможных входных данных. Было отмечено, что в этих условиях полное тестирование программного обеспечения невозможно, потому что:
  - количество возможных входных данных очень велико
  - существует множество путей
  - сложно найти проблемы в архитектуре и спецификациях
- В начале 1970-х годов тестирование программного обеспечения обозначалось как «процесс, направленный на демонстрацию корректности продукта» или как «деятельность по подтверждению правильности работы программного обеспечения»
- Во второй половине 1970-х тестирование представлялось как выполнение программы с намерением найти ошибки, а не доказать, что она работает.

История развития

- В 1980-е годы тестирование расширилось таким понятием, как предупреждение дефектов. Проектирование тестов — наиболее эффективный из известных методов предупреждения ошибок. В это же время стали высказываться мысли, что необходима методология тестирования, в частности, что тестирование должно включать проверки на всем протяжении цикла разработки, и это должен быть управляемый процесс.
- В середине 1980-х появились первые инструменты для автоматизированного тестирования. Предполагалось, что компьютер сможет выполнить больше тестов, чем человек, и сделает это более надёжно.

История развития

- В начале 1990-х годов в понятие «тестирование» стали включать планирование, проектирование, создание, поддержку и выполнение тестов и тестовых окружений, и это означало переход от тестирования к обеспечению качества, охватывающего весь цикл разработки программного обеспечения. В это время начинают появляться различные программные инструменты для поддержки процесса тестирования
- В 2000-х появилось ещё более широкое определение тестирования, когда в него было добавлено понятие «оптимизация бизнес-технологий»

  Основной подход заключается в оценке и максимизации значимости всех этапов жизненного цикла разработки программного обеспечения для достижения необходимого уровня качества, производительности, доступности.

Что такое тестирование?

- **Тестирование программного обеспечения** - процесс анализа программного средства и сопутствующей документации с целью выявления дефектов и повышения качества продукта.

Уровни тестирования

- **Компонентное или Модульное тестирование**
  - Компонентное (модульное) тестирование проверяет функциональность и ищет дефекты в частях приложения, которые доступны и могут быть протестированы по отдельности (модули программ, объекты, классы, функции и т. д.).
  - Обычно компонентное (модульное) тестирование проводится вызывая код, который необходимо проверить и при поддержке сред разработки, таких как фреймворки для модульного тестирования или инструменты для отладки. Все найденные дефекты, как правило исправляются в коде без формального их описания в системе менеджмента багов (Bug Tracking System).

Уровни тестирования

- **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).
- **Уровни интеграционного тестирования:**
  - **Компонентный интеграционный уровень**

    Проверяется взаимодействие между компонентами системы после проведения компонентного тестирования.
  - **Системный интеграционный уровень**

    Проверяется взаимодействие между разными системами после проведения системного тестирования.

Уровни тестирования

- **Системное тестирование**
  - Основной задачей системного тестирования является проверка как функциональных, так и не функциональных требований в системе в целом и оценка характеристик качества системы - ее устойчивости, надежности, безопасности и производительности.
  - При этом выявляются дефекты, такие как неверное использование ресурсов системы, непредусмотренные комбинации данных пользовательского уровня, непредусмотренные сценарии использования, отсутствующая или неверная функциональность, неудобство использования и т. д.

Уровни тестирования

- **Приемочное тестирование**
  - Формальный процесс тестирования по отношению к потребностям, требованиям и бизнес процессам пользователя, проводимое с целью определения соответствия системы критериям приёмки (критерии выхода, которым должны соответствовать компонент или система, для того, чтобы быть принятыми пользователем, заказчиком или другим уполномоченным лицом)

Виды тестирования

- **Функциональные виды тестирования**
  - Функциональные тесты базируются на функциях и особенностях, а также взаимодействии с другими системами, и могут быть представлены на всех уровнях тестирования они рассматривают внешнее поведение системы.
- **Основные виды функционального тестирования:**
- **Функциональное тестирование**
  - Функциональное тестирование рассматривает заранее указанное поведение и основывается на анализе спецификаций функциональности компонента или системы в целом.
  - Функциональные тесты основываются на функциях, выполняемых системой, и могут проводиться на всех уровнях тестирования (компонентном, интеграционном, системном, приемочном). Как правило, эти функции описываются в требованиях, функциональных спецификациях

Виды тестирования

- **Преимущества функционального тестирования:**
  - имитирует фактическое использование системы;
- **Недостатки функционального тестирования:**
  - возможность упущения логических ошибок в программном обеспечении;
  - вероятность избыточного тестирования.

Виды тестирования

- **Тестирование безопасности**
  - Тестирование безопасности - это стратегия тестирования, используемая для проверки безопасности системы, а также для анализа рисков, связанных с обеспечением целостного подхода к защите приложения, атак хакеров, вирусов, несанкционированного доступа к конфиденциальным данным.
- **Тестирование взаимодействия**
  - Функциональное тестирование, проверяющее способность приложения взаимодействовать с одним и более компонентами или системами и включающее в себя тестирование совместимости и интеграционное тестирование.

Виды тестирования

- **Нефункциональные виды 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.
- **Основные виды нефункционального тестирования:**
- **Тестирования производительности**
  - Автоматизированное тестирование, имитирующее работу определенного количества бизнес пользователей на каком-либо общем ресурсе.

Виды тестирования

- **Тестирование Установки**
  - Тестирование установки направленно на проверку успешной инсталляции и настройки, а также обновления или удаления программного обеспечения.
- **Тестирование удобства пользования**
  - Метод тестирования, направленный на установление степени удобства использования, обучаемости, понятности и привлекательности для пользователей разрабатываемого продукта в контексте заданных условий.

Виды тестирования

- **Тестирование на отказ и восстановление**
  - Проверяет тестируемый продукт с точки зрения способности противостоять и успешно восстанавливаться после возможных сбоев, возникших в связи с ошибками программного обеспечения, отказами оборудования или проблемами связи.
  - Целью данного вида тестирования является проверка систем восстановления (или дублирующих основной функционал систем), которые, в случае возникновения сбоев, обеспечат сохранность и целостность данных тестируемого продукта.

Виды тестирования

- **Конфигурационное тестирование**
  - Специальный вид тестирования, направленный на проверку работы программного обеспечения при различных конфигурациях системы (заявленных платформах, поддерживаемых драйверах, при различных конфигурациях компьютеров и т.д.)

Виды тестирования

- В зависимости от типа проекта конфигурационное тестирование может иметь разные цели:
- **Проект по профилированию работы системы**
  - Цель Тестирования: определить оптимальную конфигурацию оборудования, обеспечивающую требуемые характеристики производительности и времени реакции тестируемой системы.
- **Проект по миграции системы с одной платформы на другую**
  - Цель Тестирования: Проверить объект тестирования на совместимость с объявленным в спецификации оборудованием, операционными системами и программными продуктами третьих фирм.

Методы тестирования

- **Тестирование методом черного ящика**
- При тестировании по стратегии чёрного ящика руководствуются спецификацией системы, и оценивается её функциональность.
- Система, которую представляют как «черный ящик», рассматривается как имеющая некий «вход» для ввода информации и «выход» для отображения результатов работы, при этом происходящие в ходе работы системы процессы наблюдателю неизвестны.

Методы тестирования

- **Метод чёрного ящика включает в себя следующие методы тестирования:**
- **Метод эквивалентного разбиения:**
  - Разбиение тестов на такие классы эквивалентности, что если один тест из него не выполняется, то другие также не будут выполнены, и наоборот
  - Каждый тест должен входить в максимальное число классов эквивалентности
- **Метод анализа граничных условий:**
  - Выбор любого элемента в классе эквивалентности в качестве представительного осуществляется таким образом, чтобы проверить границы этого класса

Методы тестирования

- **Метод черного ящика имеет следующие недостатки:**
  - Невозможно найти взаимоуничтожающихся ошибок
  - Некоторые ошибки возникают достаточно редко (ошибки работы с памятью) и потому их трудно найти и воспроизвести.

Методы тестирования

- **Тестирование методом белого ящика**
  - Это техника тестирования, которая позволяет проверить внутреннюю структуру программы, ее логику и корректность работы.
  - Техника тестирования белого ящика подразумевает под собой тестирование программного обеспечения, анализируя логику работы программы для получения тестовых данных.

Методы тестирования

- **Метод белого ящика включает в себя следующие методы тестирования:**
- **покрытие операторов** - выполнение каждого оператора программы, по крайней мере, один раз
- **покрытие решений** - необходимо составить такое число тестов, при которых каждое условие в программе примет как истинное значение, так и ложное значение
- **покрытие условий** - если после составления тестов у нас останутся не покрытые операторы, то мы должны дополнить свой набор тестов таким образом чтобы каждый оператор выполняется не менее одного раза
- **покрытие решений и условий -** необходимо составить тесты так, чтобы результаты каждого условия выполнялись хотя бы один раз, результаты каждого решения так же выполнялись хотя бы один раз, и каждый оператор должен быть выполнен хотя бы один раз
- **комбинаторное покрытие условий -** все возможные комбинации результатов условий в каждом решении, а также каждый оператор выполнились по крайней мере один раз

Методы тестирования

- **Преимущества тестирования Белого ящика:**
  - Обнаружение дефектов в "скрытом" коде
  - Оптимизация
  - Необходимость разработчика тщательно обдумывать реализацию
- **Недостатки методики тестирования Белого ящика:**
  - Метод обладает недостаточной чувствительностью к ошибкам, упущенным в коде
  - Дорогостоимость

Спасибо за внимание