**Тестирование**

*Основные понятия*

**Требования** – совокупность утверждений относительно атрибутов, свойств или качеств программной системы, подлежащей реализации

Спецификация – законченное описание поведения программы, которую требуется разработать.

**Функциональные требования** – требуемые характеристики системы (функциональность).

**Нефункциональные требования** – требования, которые не влияют на основную функциональность системы.

**Тестовый случай(тест, test case)** – это набор условий, при которых тестировщик будет определять, удовлетворяется ли заранее определенное требование.

Отчёт об ошибке (bug report) – документ, описывающий ситуацию, которая привела к обнаружению ошибки, фактический и ожидаемый результат.

*Процесс тестирования*

**Тестирование ПО** – проверка и оценка соответствия между реальными и ожидаемым поведением программы.

Цель тестирования – убедиться в том, что ПО отвечает заявленным требованиям.

Выявить ситуации, в которых поведение программы является неправильным, нежелательным или не соответствующим требованиям.

Задачи тестирования

* Система работает согласно требованиям
* Проверить по установленным приоритетам
* Пропустить как можно меньше дефектов
* Проверить, что известные дефекты устранены
* Проверить, что при устранении известных дефектов не были внесены новые
* Информировать всех остальных о качестве системы

Цикл тестирования ПО

Начало – Планирование тестирования – Разработка тестов – Выполнение тестов – Анализ и отчёт о результатах тестирования – Завершение

Принципы тестирования

1. Тестирование демонстрирует наличие дефектов (Если баги не найдены – это не означает, что их нет)
2. Исчерпывающее тестирование недостижимо
3. Ранее тестирование
4. Скопление дефектов
5. Парадокс пестицида (Если повторять тесты снова и снова, то они перестанут находить новые дефекты)
6. Тестирование зависит от контекста
7. Заблуждение об отсутствии ошибок

Классификация видов тестирования

Функциональное

* Функциональное (Рассматривается заранее указанное поведение и основывается на анализе функциональных требований)
* Тестирование безопасности (Проверяет безопасность системы, а также выполняется для анализа рисков связанных с защитой приложения)
* Тестирование взаимодействия (Нацелено на оценку качетсва взаимодействия системы или приложения с другими приложениями или системами)

Нефункциональное тестирование

* Конфигурационное (Проверяет работу ПО при различных конфигурациях системы)
* Тестирование на отказ и восстановление (проверяет способность противостоять и успешно восстанавливаться после возможных сбоев)
* Требование установки (Проверка успешной инсталляции, а также обновление или удаление ПО )
* Тестирование производительности (Производиться с целью определения, как быстро работает система или её часть под определенной нагрузкой)
* Тестирование удобства использования (Исследование, выполняемое с целью определения, удобна ли “Веб-страница”, пользовательский интерфейс или устройство для его применения)

Тестирования связанные с изменениями

* Тестирование сборки (Проверка, что приложение можно тестировать)
* Регрессионное тестирование (Проверка работоспособности существующей ранее функциональности)
* Дымовое тестирование (Минимальный набор тестов, проверяющих базовую функциональность)
* Санитарное (Проверка работоспособности определенной части ПО)

Дополнительная классификация

* Позитивное тестирование
* Негативное тестирование
* По степени подготовленности (Тестирование по документации, Интуитивное тестирование, Исследовательское тестирование)
* По степени автоматизации (Ручное тестирование, Полуавтоматизированное, Автоматизированное)
* По знанию системы (Тестирование черного ящика, Тестирование белого ящика, Тестирование серого ящика)

**Тестирование**

**Тестовая документация**

**План тестирования** – документ описывающий весь объём работ по тестированию.

План

* Что тестируем (Пример: Проект «Интернет магазин» Цель – создание сайта для реализации товара).
* Какая функциональность главная?
* Где тестируем? (Chrome, OS Windows)
* Как тестируем? ( Функциональное. Конфигурационное, Нагрузочное, Тестирование удобства использования, тестирование доступности )
* Когда должны закончить?
* Какие риски? (Изменение требований, больничный сотрудников)

**Тест-кейс**

1. Название тестового случая
2. Описание тестового случая (Проверка фильтрации товаров в каталоге по брендам)
3. Приоритет тестового случая
4. Предусловия
5. Шаги теста
6. Тестовые данные
7. Ожидаемый результат
8. Постусловия

**Набор тест-кейсов** – совокупность тестовых случаев предназначенных для проверки отдельной функции системы

**Матрица покрытия** – документ, отражающий связь требований в тестовых сценариях

**Чек-лист** – документ, содержащий в себе краткое описание функциональности всего приложения, для его проверки

*Что указывать в чек-листе?*

1. Список проверок с требуемой степенью детализации
2. Приоритет
3. Ожидаемый результат
4. Тестовое окружение
5. Дата тестирования
6. Результат проверок
7. Поле для комментариев

**Отчёт об ошибке** – документ, содержащий отчёт о любом недостатке в компоненте или системе, который он может привести компонент или систему к невозможности выполнить требуемую функцию

Что указывать в описании дефектов?

1. Заголовок
2. Шаги по воспроизведению
3. Фактический результат
4. Ожидаемый результат
5. Приоритет
6. Серьёзность
7. Тестовое окружение
8. Предварительные условия
9. Шаги по восстановлению системы
10. Приложения

**Отчёт о результат тестирования** – документ, описывающий текущее качество продукта, содержащий информацию о выполненных действиях и результатах проведенной работы

Что указывают в отчёте?

1. Название проекта, дата, кто составил
2. Период тестирования
3. Состав тестовой команды
4. Список новых функций
5. Оценка текущего качества проекта
6. Сводка тестов
7. Дефекты

**Тестирование**

Дефект – не соответствие ожидаемого результата фактическому.

Ожидаемый результат – это то, как система должна себя вести,

* Логически понятная функциональность
* Требования

Отчёт от дефекте – документ, содержащий отчёт о любом недостатке в компоненте или системе, который может привести к невозможности выполнять требуемую функцию.

Обязательные поля отчёта о дефекте

* Заголовок
* Шаги воспроизведения
* Фактический результат
* Ожидаемый результат

Необязательные поля отчёта о дефекте

* Приоритет
* Серьёзность
* Тестовое окружение
* Предварительные условия
* Шаги по восстановлению системы
* Приложения

Жизненный цикл дефекта

1. Найден дефект
2. Назначен исполнитель
3. Воспроизводится?
4. Исправить
5. Закрыть дефект

Причины появления дефектов

* Сложность ПО
* Сжатые сроки разработки
* Проблемы с коммуникацией команды
* Изменение требований в процессе разработки
* Человеческий фактор
* Не так может пойти всё

Виды дефектов

1. Функциональные
2. Логические
3. Графического интерфейса
4. Дефекты документации
5. Дефекты производительности
6. Безопасности
7. Сторонних сервисов
8. Удобства использования
9. Локализации
10. Контента

Тестирование

test design позволяет выбрать основные проверки и сделать тестирование действительно качественно.

test design это один из процессов этапа тестирования на этом этапе мы занимаемся тем что планируем проверки который позволит убедиться в том что функциональность работает как нужно и документируем тест кейсов в соответствии с теми требованиями которых были предъявлены к задаче на этапе ее формирования

Зачем нужен тест-дизайн?

* Критические проверки и покрытие тестами
* Меньше проверок
* Проверки зафиксированы
* Планирование и статус
* Параллельная работа
* Чёткая последовательность шагов

Как сделать тест-дизайн?

1. Анализ требований
2. Разработка
3. Тестирование

Популярные техники

1. Тестирование классов эквивалентности и граничных значений
2. Тестирование переходов между состояниями и таблица перехода
3. Диаграмма причина-следствие и таблица принятия решений
4. Папарное тестирование
5. Тестирование по сценариям использования

Эквивалентное разделение – техника тест-дизайна, при которой множество возможных наборов тестовых данных разбивается на классы

Особенности:

* Все представители обрабатываются одним модулем программы
* Все представители находят одни и те же дефекты
* Для тестирования достаточно одного представителя

Анализ граничных значений – техника поиска ошибок на границах классов эквивалентности

Преимущества

* Значительно меньше тестов
* Хорошее покрытие тестами

Ограничения

* Входное значение можно разделить на группы, классы
* Группы немногочисленны или сфера связана с риском для жизни людей

Тестирование переходов между состояниями – техника тест-дизайна, основанная на анализе состояний системы и переходов между состояними

Диаграмма состояний

* Состояние – система ждёт события
* Переходы – изменение состояний в результате события
* События – импульс
* Действия – операция внутри системы в результате события
* Точка входа/ точка выхода

Преимущества

* Отслеживаем переходы состояний
* Диаграмма: переходы наглядны
* Таблица: все переходы, в том числе – невалидные

Ограничения

Применение необоснованно

* Если не предполагается моментальное реагирование на событие
* Если нет акцента на состояния

Диаграмма причина-следствие – это графический метод анализа причинно-следственных связей, инструмент для определения причин проблемы и графического представления

Decision table – таблица, которая содержит комбинации входных параметров и зависящий от них результат, чётко описывает сложную систему логики системы

Преимущества

* Демонстрирует зависимости между причинами и следствиями
* Четко документирует сложную систему логики

Ограничения

* Использование возможности тогда, когда бизнес-логика может быть представлена в виде комбинаций условий, и когда эти условия приводят к четко определенным результатам

Попарное тестирование – метод генерации тестовых данных, при котором проверки делаются для каждой пары всех возможных водных значений

Как сократить количество проверок?

Большинство дефектов может быть выявлено

* При тестировании модуля самого по себе
* При тестировании модуля в паре с другим модулем

Преимущества

* Значительно сокращает время тестирования
* Находит большинство дефектов

Ограничения

* Результат работы требует ручной корректировки:

(не учитывает логически возможные пары, не учитывает опасные кейсы)

* Не используется в сферах с большим уровнем риска

Сценарий использования – последовательность действий пользователя и реакций системы на эти действия.

Структура

* Имя сценария
* Актор
* Критерии успешного завершения
* Критерии неуспешного завершения
* Основные сценарии
* Альтернативный сценарий 1
* Альтернативный сценарий 2
* Доп.условия

Преимущества

* Покрывают основные функциональные требования с точки зрения пользователя
* Позволяет определить структуру, элементы системы, основные отношения
* Является основной для разработки приемочных тестов

Ограничения

* Не проверяют элементы интерфейса и другие детали реализации
* Не дают достаточного тестирования входных парамтеров