GeekBrains

Основы ручного тестирования

Работа с дефектами

На этом уроке

1. Узнаем, что такое ошибка, дефект и отказ.
2. Выясним, как «живёт» дефект.
3. Изучим атрибуты баг-репорта.
4. Научимся составлять отчёт о дефектах.

Оглавление

Виды дефектов программного обеспечения

По определению качественное ПО обладает двумя характеристиками:

1. Соответствует требованиям — спецификации.
2. Удовлетворяет потребность пользователя.

При работе с ПО пользователь или тестировщик ожидает определённого поведения системы, согласно требованиям, спецификации, опыту или здравому смыслу. Дефект — это отклонение фактического результата от ожидаемого.

В примере с интернет-магазином ожидаемый результат — списание средств с карточки и сообщение «Ваш заказ оплачен». Он сформировался на основе опыта использования аналогичных программ. Фактический результат — невозможность оплатить и сообщение об ошибке. Он противоречит ожиданиям, следовательно, это дефект.

Не все проекты имеют документацию: требования, спецификации и технические задания. Поэтому тестировщику приходится опираться:

- на предыдущий опыт тестирования подобных приложений;
- на критерии качества, работающие на рынке;
- на логику и здравый смысл.

Дефект — общее название для отклонений в работе ПО. Виды дефектов:

1. **Ошибка** — действие человека, приводящее к некорректным результатам.
    a. Ошибка разработчика — код написан с ошибками.
    b. Ошибка пользователя — введены неверные данные. Например, для оплаты заказа в магазине используется карточка с недостатком средств.
2. **Сбой** — кратковременное нарушение в работе системы, которое проходит самостоятельно или чинится незначительным вмешательством. Например, во время работы программа «подвисает» и не отвечает на действия пользователя, но спустя несколько секунд восстанавливается и продолжает работу без потери данных.
3. **Отказ** — событие, заключающееся в нарушении работоспособного состояния приложения. В этом случае требуется серьёзное вмешательство, и возможна потеря данных. Самый распространённый пример, с которым сталкиваются пользователи Windows, — программа перестаёт отвечать, и её приходится закрывать через диспетчер задач.

4. **Инцидент** — дефект, обнаруженный пользователями приложения, о котором поступило сообщение в поддержку. Инциденты инициируются:
    - пользователями — сообщение в техподдержку;
    - системами мониторинга — создаются автоматически.

Важно! *Иногда эти признаки отсутствуют. Например, пользователь хочет оплатить заказ в интернет-магазине картой, но получает сообщение об ошибке. В этом случае он столкнулся с дефектом, т. е. с отклоняющимся поведением программы.*

Причины появления дефектов

Проблемы в коммуникациях между членами команды

Если команда не работает слаженно, то разработка тормозит или идёт с ошибками. Разработчик не понял требований, задал вопрос аналитику, но не получил ответ или получил его слишком поздно — вероятно, в системе появились дефекты. Тестировщик обнаружил отклоняющееся поведение, но не нашёл документов с описанием «как надо», не дождался ответа разработчика или заказчика — есть риск получить инцидент.

Сложность программного обеспечения

Чем сложнее программа, тем больше вероятность появления ошибок. Сложная логика работы, множество взаимодействующих систем, нетривиальные алгоритмы — всё это элементы повышенного риска появления дефектов.

Изменение требований

ПО подстраивается под меняющийся мир и потребности пользователя. Изменения требований приводят к модификации кода, и чем позже изменились требования, тем больше работ выполняется для трансформации системы. Правки влекут за собой новые дефекты или возврат уже исправленных.

Ошибки программистов

Разработчики должны писать и документировать код по установленным правилам. Но нехватка времени вынуждает программистов работать быстро, поэтому срабатывает фактор невнимательности и забывчивости, что снижает качество программы.

Ошибки тестировщиков

Тестировщики не замечают ошибки из-за недостатка опыта, упускают проблемы из-за сложности программного обеспечения и невозможности исчерпывающего тестирования.

Архитектура программного обеспечения

Непродуманный выбор структурных элементов, интерфейсов и взаимодействия компонентов ведут к дефектам в функционировании ПО.

Недостаток финансирования

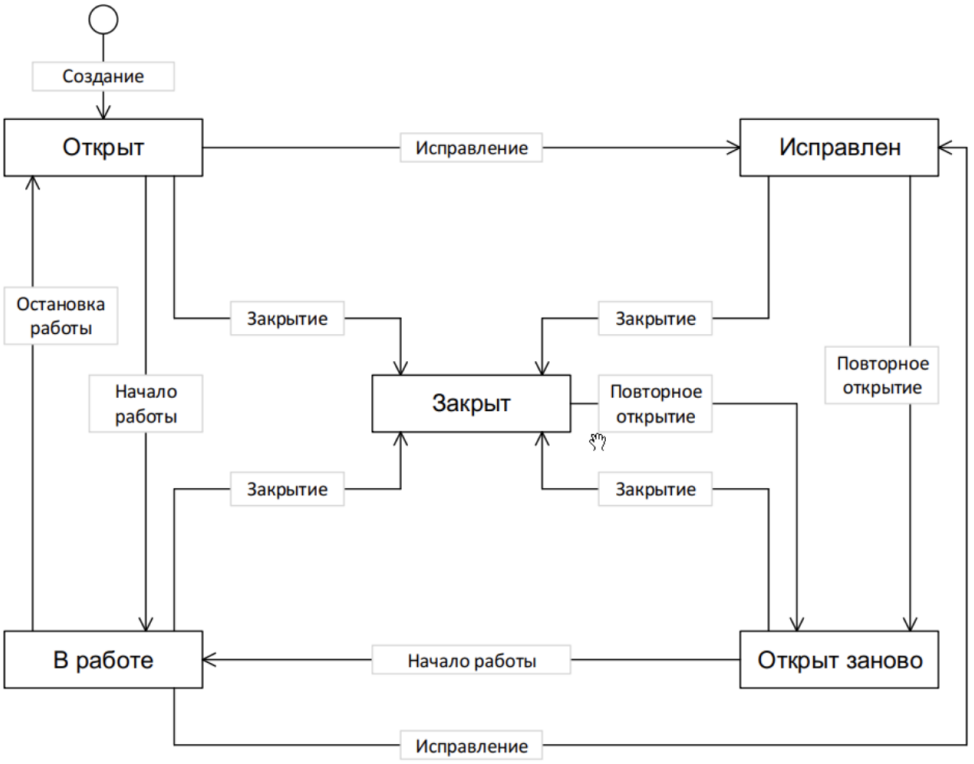
Программный продукт тестируется ровно настолько, насколько предоставлено финансов на процесс тестирования. Если бюджета не хватает, тестирование остановится.

Важно! *Основные причины ошибок в программном обеспечении: невнимательность, временные и финансовые ограничения, дефекты в инструментах для разработки, а также невозможность проведения исчерпывающего тестирования.*

Жизненный цикл дефекта

Дефект создаётся в системе, затем его чинит разработчик и, если работоспособность ПО восстановилась, тестировщик закрывает дефект. Эти этапы составляют **жизненный цикл дефекта**.

1. Открыт: тестировщик обнаружил отклоняющееся поведение и создал отчёт.
2. В работе: назначен на разработчика, и разработчик начал исправление.
3. Исправлен: разработчик внёс изменения в программный код.
4. Закрыт: тестировщик проверил, что всё работает в соответствии с ТЗ.
5. Открыт заново: ранее исправленный дефект обнаружен снова.



Отчёт о дефекте

Чтобы сообщить о дефекте, тестировщик описывает его установленным образом. В результате получается отчёт о дефекте, или баг-репорт (bug report). Для краткости его называют «баг».

Отчёт о дефекте — это документ, который описывает шаги воспроизведения дефекта, фактический и ожидаемый результат, серьёзность дефекта и приоритет устранения.

Отчёты о дефектах — инструменты для сбора статистики на проекте. Они помогают определить:

- в каких областях приложения;
- при каких условиях концентрируются дефекты.

Важная функция отчётов о дефектах — приоритизация проблем, обнаруженных в приложении. При большом числе найденных дефектов и ограниченных временных резервах разработчику требуется понимать, какие дефекты хуже всего влияют на работу приложения и должны быть исправлены в первую очередь.

Отчёт о дефекте не только предоставляет важные подробности для понимания сути случившегося, но и анализирует причины возникновения проблемы, а также даёт рекомендации по исправлению ситуации.

Атрибуты отчёта о дефекте

1. **Уникальный идентификатор** (ID) — присваивается автоматически, может содержать в себе данные о требовании, на которое ссылается дефект.

2. **Тема** (краткое описание, Summary) — кратко сформулированная суть дефекта по правилу «Что? Где? Когда?»

3. **Подробное описание** (Description) — более широкое описание сути дефекта (при необходимости).

4. **Шаги для воспроизведения** (Steps To Reproduce) — последовательное описание действий, которые привели к выявлению дефекта (которые нужно выполнить для воспроизведения дефекта). Описываются максимально подробно, с указанием конкретных вводимых значений.

5. **Фактический результат** (Actual result) — указывается, что не так работает, в каком месте продукта и при каких условиях. Описывая фактический результат, необходимо ответить на три вопроса: что? где? когда?

6. **Ожидаемый результат** (Expected result) — указывается, как именно должна работать система по мнению тестировщика, основанному на требованиях и прочей проектной документации.

7. **Серьёзность дефекта** (важность, Severity) — характеризует влияние дефекта на работоспособность приложения.

8. **Приоритет дефекта** (срочность, Priority) — указывает на очерёдность выполнения задачи или устранения дефекта. Чем выше приоритет, тем быстрее нужно исправить дефект.

9. **Статус** (Status) — определяет текущее состояние дефекта. Отражает жизненный цикл дефекта от начального состояния до завершения. Названия статусов дефектов могут быть разными в разных баг-трекинговых системах.
10. **Тестовое окружение** — указывает, на какой платформе (и в каком браузере, в случае с веб-приложениями) этот дефект воспроизводится (iOS, Android, Windows, Mac);

Дополнительно могут встречаться такие атрибуты, как:

- **вложения** (Attachments) — скриншоты, видео или лог-файлы.
- **версия** — указывает, на каком этапе разработки программного продукта был обнаружен дефект;
- **назначение** — указывается личность, на которую данный баг-репорт назначается для дальнейшей проверки или исправления;
- **номер сборки** — указывается номер билда, в котором был обнаружен дефект.

Что важно при написании названия отчёта о дефекте:

Название должно содержать ответы на вопросы:

1. Что происходит или не происходит согласно спецификации или представлению тестировщика о нормальной работе продукта?

2. В каком месте интерфейса пользователя или архитектуры программного продукта находится проблема?

3. В какой момент работы программного продукта, при наступлении какого события или при каких условиях проявляется проблема?

Это называется правилом «Что? Где? Когда?»

Название содержит предельно краткую, но достаточную для понимания сути проблемы информацию о дефекте. Оптимальная длина — до 10 слов и 80 символов. В названии не употребляются слова: «некорректно», «неправильно», «ошибка» и т. д. Тестировщик описывает, что именно некорректно, и в чём именно ошибка.

Неудачное название: «Возникает ошибка при загрузке большого файла».

Какая ошибка? Насколько большого файла?

Удачное название: «Код ошибки DB1001 при загрузке файла 10 Гб».

Шаги по воспроизведению — это руководство к действию для тех, кто будет решать проблему. Шаги составляются по правилам:

1. Вернуться к началу. Первый шаг — указание, по какой ссылке перейти, какое окно открыть и т. д.
2. Шаг отвечает на вопрос «Что сделать?».
    - нажать кнопку «Найти»;
    - ввести валидный email и пароль;
    - заполнить требуемые поля валидными данными.
3. Количество шагов — от 2 до 8. Если больше — подумать, какие шаги лишние.
4. Последний шаг — указание, на что обратить внимание.
    - осмотреть текст в выпадающем списке (подменю);
    - обратить внимание на Profile-форму.

Фактический и ожидаемый результаты рекомендуется описывать следующим образом:

1. Один дефект — один фактический и ожидаемый результат.
2. В ожидаемом результате дать ссылку на документацию.
    a. Фактический результат. Элементы подменю «Курсы» отображаются за границами выпадающего списка в главном меню после наведения курсора на блок «Курсы».
    b. Ожидаемый результат. Элементы подменю «Курсы» находятся в пределах выпадающего списка в главном меню после наведения курсора на блок «Курсы».

Серьёзность и приоритет

Серьёзность показывает степень ущерба, который наносится проекту дефектом.

- Критический (Critical) — функция работает, но с ограничениями
- Значительный (Major) — функция работает, но неправильно
- Незначительный (Minor) — функция работает, но неудобно
- Блокирующий (Blocker) — функция не работает
- Тривиальный (Trivial) — грамматические ошибки в пользовательской документации

Приоритет (срочность) показывает, как быстро нужно устранить дефект.

Срочность: высокая (High), средняя (Medium), низкая (Low). Порядок исправления ошибок по их приоритетам:

1. High.
2. Medium.
3. Low.

Правила хорошего баг-репорта

1. Следовать правилу «Что? Где? Когда?».
2. Одна ошибка — один отчёт о дефекте.
3. Краткость — сестра баг-репорта.
4. Писать техническим языком с применением терминологии, принятой на проекте.
5. Прикреплять дополнительные файлы: логи, скриншоты, видео.
6. Прикреплять ссылки к требованиям, это позволит избежать споров.
7. Избегать дубликатов дефектов — прежде чем сохранить отчёт о дефекте, надо проверить в баг-трекере, есть ли уже такой дефект.
8. Указывать версию ПО и тестовый стенд (окружение), на котором обнаружился дефект.
9. Воспроизводить дефект, следуя собственным шагам.

Багтрекинговая система

Система отслеживания ошибок (от англ. bug tracking system) — прикладная программа, созданная, чтобы разработчикам программного обеспечения (программистам, тестировщикам и др.) было проще:

- учитывать и контролировать ошибки и неполадки, найденные в программах, а также пожелания пользователей;
- следить за устранением этих ошибок и выполнением или невыполнением пожеланий.

Багтрекинговые системы используются для создания отчётов о дефектах и управления ими. Помимо создания отчётов, они применяются в создании задач команде разработки.

Бесплатные багтрекинговые системы устанавливаются, чтобы:

- отрабатывать навыки работы с багтрекинговой системой;

- получить более глубокое понимание работы с ней (её администрирование).

Например, Redmine или Mantis.

Контрольные вопросы

1. Какие виды ошибок возникают в ПО?
2. Почему возникают дефекты?
3. Какие этапы проходит дефект за время своего существования?
4. Какие атрибуты есть у дефекта?

Практическое задание

1. Откройте документ, созданный в практическом задании №2. На вкладке «баг-репорты» заведите 3 дефекта по результатам выполнения чек-листа.

   Создайте аккаунт Atlassian. Создайте проект - багтрекинговую систему в Jira. Продублируйте один баг в Jira и приложите скриншот.

2. Тест для самопроверки - https://coreapp.ai/app/player/lesson/614344b640a70c676714c752 (сдавать не нужно)

Глоссарий

Дефект — отклонение фактического результата от ожиданий наблюдателя, сформированных на основе требований, спецификаций, иной документации или опыта и здравого смысла.

Жизненный цикл дефекта — это этапы, которые он проходит за время своего существования.

Инцидент — дефект, обнаруженный пользователями приложения.

Отказ — событие, заключающееся в нарушении работоспособного состояния приложения.

Отчёт о дефекте — это документ, который описывает шаги воспроизведения дефекта, фактический и ожидаемый результат, серьёзность дефекта и приоритет устранения.

Ошибка — действие человека, приводящее к некорректным результатам.

Приоритет (срочность) — как быстро нужно устранить дефект.

Сбой — кратковременный сбой в работе системы, который проходит самостоятельно или чинится незначительным вмешательством.

Серьёзность — степень ущерба, который наносится проекту дефектом.

Система отслеживания ошибок (от англ. bug tracking system) — прикладная программа, созданная, чтобы разработчикам программного обеспечения (программистам, тестировщикам и др.) было проще:

- учитывать и контролировать ошибки и неполадки, найденные в программах, а также пожелания пользователей;
- следить за устранением этих ошибок и выполнением или невыполнением пожеланий.

Дополнительные материалы

1. Статья «Как тестировать на удалёнке, чтобы не запороть продукт и свою жизнь».
2. Статья «Краудтестинг: где взять опыт для первой работы в тестировании».
3. Статья «Путь развития тестировщика: как найти компанию по душе».
4. Блог про будни тестировщика.
5. Блог Brainual Tester.
6. Статья «Образ современного тестировщика. Что нужно знать и уметь».

Используемые источники

1. Статья «RSTQB FAQ. Общие вопросы».
2. ISTQB Glossary.
3. Статья ISO / IEC 9126.
4. Статья ISO 25010.