Лабораторная работа № 2 «Поиск и документирование дефектов» (4 часа)

Цель работы: Изучение принципов и формирование практических навыков по поиску и описанию дефектов, составление баг-репорта.

1 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

В зависимости от инструментального средства управления отчётами о дефектах внешний вид их записи может немного отличаться, могут быть добавлены или убраны отдельные поля, но концепция остаётся неизменной.

Общий вид всей структуры отчёта о дефекте представлен в таблице 2.2.

Идентификатор	Краткое описание	Подробное описание	Шаги по воспроизведению	Воспроизводимость	Важность	Срочность	Симптом	Возможность обойти	Комментарий	Приложения
19	Бесконечный цикл обработки входного файла с атрибутом «только для чтения»	Если у входного файла выставлен атрибут «только для чтения», после обработки приложению не удаётся переместить его в каталог-приёмник: создаётся копия файла, но оригинал не удаляется, и приложение снова и снова обрабатывает этот файл и безуспешно пытается переместить его в каталог-приёмник. **Ожидаемый результат:** после обработки файл перемещён из каталога-источника в каталог-приёмник. **Фактический результат:** обработанный файл копируется в каталог-приёмник, но его оригинал остаётся в каталоге-источнике. **Требование:** ДС-2.1.	1. Поместить в каталог-источник файл допустимого типа и размера. 2. Установить данному файлу атрибут «только для чтения». 3. Запустить приложение. Дефект: обработанный файл появляется в каталоге-приёмнике, но не удаляется из каталога-источника, файл в каталоге-приёмнике непрерывно обновляется (видно по значению времени последнего изменения).	Всегда	Средняя	Обычная	Некорректная операция	Нет	Если заказчик не планирует использовать установку атрибута «только для чтения» файлам в каталоге-источнике для достижения неких своих целей, можно просто снимать этот атрибут и спокойно перемещать файл.	–

Идентификатор (identifier) представляет собой уникальное значение, позволяющее однозначно отличить один отчёт о дефекте от другого и используемое во всевозможных ссылках. В общем случае идентификатор отчёта о дефекте может представлять собой просто уникальный номер, но (если позволяет инструментальное средство управления отчётами) может быть и куда сложнее: включать префиксы, суффиксы и иные осмысленные компоненты, позволяющие быстро определить суть дефекта и часть приложения (или требований), к которой он относится.

Краткое описание (summary) должно в предельно лаконичной форме давать исчерпывающий ответ на вопросы «Что произошло?» «Где это произошло»? «При каких условиях это произошло?». Например: «Отсутствует логотип на странице приветствия, если пользователь является администратором».

- Что произошло? Отсутствует логотип.
- Где это произошло? На странице приветствия.
- При каких условиях это произошло? Если пользователь является администратором.

Одной из самых больших проблем для начинающих тестировщиков является именно заполнение поля «краткое описание», которое одновременно должно:

- содержать предельно краткую, но в то же время достаточную для понимания сути проблемы информацию о дефекте;
- отвечать на только что упомянутые вопросы («что, где и при каких условиях случилось») или как минимум на те 1–2 вопроса, которые применимы к конкретной ситуации;
- быть достаточно коротким, чтобы полностью помещаться на экране (в тех системах управления отчётами о дефектах, где конец этого поля обрезается или приводит к появлению скроллинга);
- при необходимости содержать информацию об окружении, под которым был обнаружен дефект;
- по возможности не дублировать краткие описания других дефектов (и даже не быть похожими на них), чтобы дефекты было сложно перепутать или посчитать дубликатами друг друга;
- быть законченным предложением русского или английского (или иного) языка, построенным по соответствующим правилам грамматики.

Подробное описание (description) представляет в развёрнутом виде необ-ходимую информацию о дефекте, а также (обязательно!) описание фактического результата, ожидаемого результата и ссылку на требование (если это возможно).

В отличие от краткого описания, которое, как правило, является одним предложением, здесь можно и нужно давать подробную информацию. Если одна и та же проблема (вызванная одним источником)

проявляется в нескольких местах приложения, можно в подробном описании перечислить эти места.

Шаги по воспроизведению (steps to reproduce, STR) описывают действия, которые необходимо выполнить для воспроизведения дефекта. Это поле похоже на шаги тест-кейса, за исключением одного важного отличия: здесь действия прописываются максимально подробно, с указанием конкретных вводимых значений и самых мелких деталей, т.к. отсутствие этой информации в сложных случаях может привести к невозможности воспроизведения дефекта.

Воспроизводимость (reproducibility) показывает, при каждом ли прохождении по шагам воспроизведения дефекта удаётся вызвать его проявление. Это поле принимает всего два значения: всегда (always) или иногда (sometimes).

Можно сказать, что воспроизводимость «иногда» означает, что тестировщик не нашёл настоящую причину возникновения дефекта. Это приводит к серьёзным дополнительным сложностям в работе с дефектом.

Как легко догадаться, такая ситуация является крайне неприятной, а потому рекомендуется один раз потратить время на тщательную диагностику проблемы, найти её причину и перевести дефект в разряд воспроизводимых всегда.

Важность (severity) показывает степень ущерба, который наносится проекту существованием дефекта. В общем случае выделяют следующие градации важности:

- *критическая (critical)* – существование дефекта приводит к масштабным последствиям катастрофического характера, например: потеря данных, раскрытие конфиденциальной информации, нарушение ключевой функциональности приложения и т.д.;
- *высокая (major)* – существование дефекта приносит ощутимые неудобства многим пользователям в рамках их типичной деятельности, например: недоступность вставки из буфера обмена, неработоспособность общепринятых клавиатурных комбинаций, необходимость перезапуска приложения при выполнении типичных сценариев работы;
- *средняя (medium)* – существование дефекта слабо влияет на типичные сценарии работы пользователей, и/или существует обходной путь достижения цели, например: диалоговое окно не закрывается автоматически после нажатия кнопок «OK»/«Cancel», при распечатке нескольких документов подряд не сохраняется значение поля «Двусторонняя печать», перепутаны направления сортировок по некоему полю таблицы;
- *низкая (minor)* – существование дефекта редко обнаруживается незначительным процентом пользователей и (почти) не влияет на их работу, например: опечатка в глубоко вложенном пункте меню настроек, некое окно сразу при отображении расположено

неудобно (нужно перетянуть его в удобное место), неточно отображается время до завершения операции копирования файлов.

Срочность (priority) показывает, как быстро дефект должен быть устранён. В общем случае выделяют следующие градации срочности:

- *наивысшая (ASAP, as soon as possible)* срочность указывает на необходимость устранить дефект настолько быстро, насколько это возможно. В зависимости от контекста «настолько быстро, насколько возможно» может варьироваться от «в ближайшем билде» до единиц минут;
- *высокая (high)* срочность означает, что дефект следует исправить вне очереди, т.к. его существование или уже объективно мешает работе, или начнёт создавать такие помехи в самом ближайшем будущем;
- *обычная (normal)* срочность означает, что дефект следует исправить в порядке общей очерёдности. Такое значение срочности получает большинство дефектов;
- *низкая (low)* срочность означает, что в обозримом будущем исправление данного дефекта не окажет существенного влияния на повышение качества продукта.

Несколько дополнительных рассуждений о важности и срочности стоит рассмотреть отдельно.

Один из самых частых вопросов относится к тому, какая между ними связь. Никакой. Для лучшего понимания этого факта можно сравнить важность и срочность с координатами X и Y точки на плоскости. Хоть «на бытовом уровне» и кажется, что дефект с высокой важностью следует исправить в первую очередь, в реальности ситуация может выглядеть совсем иначе.

Чтобы проиллюстрировать эту мысль подробнее, вернёмся к перечню градаций: заметили ли вы, что для разных степеней важности примеры приведены, а для разных степеней срочности — нет? И это не случайно.

Зная суть проекта и суть дефекта, его важность определить достаточно легко, т.к. мы можем проследить влияние дефекта на критерии качества, степень выполнения требований той или иной важности и т.д. Но срочность исправления дефекта можно определить только в конкретной ситуации.

Симптом (symptom) — позволяет классифицировать дефекты по их типичному проявлению. Не существует никакого общепринятого списка симптомов. Более того, далеко не в каждом инструментальном средстве управления отчётами о дефектах есть такое поле, а там, где оно есть, его можно настроить. В качестве примера рассмотрим следующие значения симптомов дефекта:

- *косметический дефект (cosmetic flaw)* — визуально заметный недостаток интерфейса, не влияющий на функциональность

приложения (например, надпись на кнопке выполнена шрифтом не той гарнитуры);

- *повреждение/потеря данных (data corruption/loss)* — в результате возникновения дефекта искажаются, уничтожаются (или не сохраняются) некоторые данные (например, при копировании файлов копии оказываются повреждёнными);
- *проблема в документации (documentation issue)* — дефект относится не к приложению, а к документации (например, отсутствует раздел руководства по эксплуатации);
- *некорректная операция (incorrect operation)* — некоторая операция выполняется некорректно (например, калькулятор показывает ответ 17 при умножении 2 на 3);
- *проблема инсталляции (installation problem)* — дефект проявляется на стадии установки и/или конфигурирования приложения;
- *ошибка локализации (localization issue)* — что-то в приложении не переведено или переведено неверно на выбранный язык интерфейса;
- *нереализованная функциональность (missing feature)* — некая функция приложения не выполняется или не может быть вызвана (например, в списке форматов для экспорта документа отсутствует несколько пунктов, которые там должны быть);
- *проблема масштабируемости (scalability)* — при увеличении количества доступных приложению ресурсов не происходит ожидаемого прироста производительности приложения;
- *низкая производительность (low performance)* — выполнение неких операций занимает недопустимо большое время;
- *крах системы (system crash)* — приложение прекращает работу или теряет способность выполнять свои ключевые функции (также может сопровождаться крахом операционной системы, веб-сервера и т.д.);
- *неожиданное поведение (unexpected behavior)* — в процессе выполнения некоторой типичной операции приложение ведёт себя необычным (отличным от общепринятого) образом (например, после добавления в список новой записи активной становится не новая запись, а первая в списке);
- *недружественное поведение (unfriendly behavior)* — поведение приложения создаёт пользователю неудобства в работе (например, на разных диалоговых окнах в разном порядке расположены кнопки «OK» и «Cancel»);
- *расхождение с требованиями (variance from specs)* — этот симптом указывают, если дефект сложно соотнести с другими симптомами, но тем не менее приложение ведёт себя не так, как описано в требованиях;

- *предложение по улучшению (enhancement)* — во многих инструментальных средствах управления отчётами о дефектах для этого случая есть отдельный вид отчёта, т.к. предложение по улучшению формально нельзя считать дефектом: приложение ведёт себя согласно требованиям, но у тестировщика есть обоснованное мнение о том, как ту или иную функциональность можно улучшить.

Часто встречается вопрос о том, может ли у одного дефекта быть сразу несколько симптомов. Да, может. Например, крах системы очень часто ведёт к потере или повреждению данных. Но в большинстве инструментальных средств управления отчётами о дефектах значение поля «Симптом» выбирается из списка, и потому нет возможности указать два и более симптома одного дефекта. В такой ситуации рекомендуется выбирать либо симптом, который лучше всего описывает суть ситуации, либо «наиболее опасный» симптом (например, недружественное поведение, состоящее в том, что приложение не запрашивает подтверждения перезаписи существующего файла, приводит к потере данных; здесь «потеря данных» куда уместнее, чем «недружественное поведение»).

Возможность обойти (workaround) — показывает, существует ли альтернативная последовательность действий, выполнение которой позволило бы пользователю достичь поставленной цели (например, клавиатурная комбинация Ctrl+P не работает, но распечатать документ можно, выбрав соответствующие пункты в меню). В некоторых инструментальных средствах управления отчётами о дефектах это поле может просто принимать значения «Да» и «Нет», в некоторых при выборе «Да» появляется возможность описать обходной путь. Традиционно считается, что дефектам без возможности обхода стоит повысить срочность исправления.

Комментарий (comments, additional info) – может содержать любые полезные для понимания и исправления дефекта данные. Иными словами, сюда можно писать всё то, что нельзя писать в остальные поля.

Приложения (attachments) – представляет собой не столько поле, сколько список прикреплённых к отчёту о дефекте приложений (копий экрана, вызывающих сбой файлов и т.д.)

Общие рекомендации по формированию приложений таковы:

- если вы сомневаетесь, делать или не делать приложение, лучше сделайте;
- обязательно прилагайте т.н. «проблемные артефакты» (например, файлы, которые приложение обрабатывает некорректно).

Если вы прилагаете видеоролик с записью происходящего на экране, обязательно оставляйте только тот фрагмент, который относится к описываемому дефекту (это будет буквально несколько секунд или минут из возможных многих часов записи). Старайтесь подобрать настройки кодеков так, чтобы получить минимальный размер ролика при сохранении достаточного качества изображения.

2 ЗАДАНИЕ

1. Изучить теоретические сведения
2. Подготовить отчет, который должен содержать цель, задание, краткие теоретические сведения, выводы по работе.

3 ТРЕБОВАНИЕ К ОТЧЕТУ

В отчете должны быть отображены следующие пункты:

1. Задание.
2. Краткие теоретические сведения по поиску и документированию дефектов.
3. Заполненный баг-репорт с не менее 10 багами программного обеспечения.
4. Выводы.

4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Перечислите главные этапы функционального тестирования.
2. Перечислите основные элементы баг-репорта.
3. Охарактеризуйте два элемента баг-репорта по выбору преподавателя.