Лабораторная работа № 1 «Тестирование программного обеспечения: разработка тестов»
(8 часов)

Цель работы: Изучение принципов и формирование практических навыков разработки тестовой документации, разработка чек-листов и тест-кейсов.

1 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Рассмотрим наиболее распространённый способ тестирования – *тестирование на основе тест-кейсов* – формализованный подход, в котором тестирование производится на основе заранее подготовленных тест-кейсов, наборов тест-кейсов и иной документации.

Тест-кейс (тестовый случай) – набор тестовых данных, условий выполнения теста и последовательность действий тестировщика, а также ожидаемый результат, которые разрабатывается с целью проверки тех или иных аспектов работы программы.

В зависимости от инструмента управления тест-кейсами внешний вид их записи может немного отличаться, могут быть добавлены или убраны отдельные поля, но концепция остаётся неизменной.

Общий вид всей структуры тест-кейса представлен в таблице 1.

Таблица 2.1. Общий вид структуры тест-кейса

Идентификатор	Приоритет	Связанное с тест-кейсом требование	Модуль приложения	Подмодуль приложения	Заглавие (суть) тест-кейса и его шаги	Ожидаемый результат по каждому шагу тест-кейса
UG_U1.12	A	R97	Галерея	Загрузка файла	**Галерея, загрузка файла, имя со спецсимволами** Приготовления: создать непустой файл с именем #$%^&.jpg. 1. Нажать кнопку «Загрузить картинку». 2. Нажать кнопку «Выбрать». 3. Выбрать из списка приготовленный файл. 4. Нажать кнопку «OK». 5. Нажать кнопку «Доба-вить в галерею».	1. Появляется окно загрузки картинки. 2. Появляется диалоговое окно браузера выбора файла для загрузки. 3. Имя выбранного файла появляется в поле «Файл». 4. Диалоговое окно файла закрывается, в поле «Файл» появляется полное имя файла. 5. Выбранный файл появляется в списке файлов галереи.

Рассмотрим каждый атрибут подробно.

Идентификатор (identifier) представляет собой уникальное значение, позволяющее однозначно отличить один тест-кейс от другого и используемое во всевозможных ссылках. В общем случае идентификатор тест-кейса может представлять собой просто уникальный номер, но (если позволяет инструментальное средство управления тест-кейсами) может быть и куда сложнее: включать префиксы, суффиксы и иные осмысленные компоненты, позволяющие быстро определить цель тесткейса и часть приложения (или требований), к которой он относится (например: UR216_S12_DB_Neg).

Приоритет (priority) показывает важность тест-кейса. Он может быть выражен буквами (A, B, C, D, E), цифрами (1, 2, 3, 4, 5), словами («крайне высокий», «высокий», «средний», «низкий», «крайне низкий») или иным удобным способом. Количество градаций также не фиксировано, но чаще всего лежит в диапазоне от трёх до пяти.

Приоритет тест-кейса может коррелировать с:

- важностью требования, пользовательского сценария или функции, с которыми связан тест-кейс;
- потенциальной важностью дефекта, на поиск которого направлен тест-кейс;
- степенью риска, связанного с проверяемым тест-кейсом требованием, сценарием или функцией.

Основная задача этого атрибута — упрощение распределения внимания и усилий команды (более высокоприоритетные тест-кейсы получают их больше), а также упрощение планирования и принятия решения о том, чем можно пожертвовать в некоей форс-мажорной ситуации, не позволяющей выполнить все запланированные тест-кейсы.

Связанное с тест-кейсом требование (requirement) показывает то основное требование, проверке выполнения которого посвящён тест-кейс (основное — потому, что один тест-кейс может затрагивать несколько требований). Наличие этого поля улучшает такое свойство тест-кейса, как прослеживаемость.

Частые вопросы, связанные с заполнением этого поля, таковы:

- можно ли его оставить пустым? Да. Тест-кейс вполне мог разрабатываться вне прямой привязки к требованиям, и (пока?) значение этого поля определить сложно. Хоть такой вариант и не считается хорошим, он достаточно распространён.
- можно ли в этом поле указывать несколько требований? Да, но чаще всего стараются выбрать одно самое главное или «более высокоуровневое» (например, вместо того, чтобы перечислять R56.1, R56.2, R56.3 и т.д., можно просто написать R56). Чаще всего в инструментах управления тестами это поле представляет

собой выпадающий список, где можно выбрать только одно значение, и этот вопрос становится неактуальным. К тому же многие тест-кейсы всё же направлены на проверку строго одного требования, и для них этот вопрос также неактуален.

Модуль и подмодуль приложения (module and submodule) указывают на части приложения, к которым относится тест-кейс, и позволяют лучше понять его цель.

Идея деления приложения на модули и подмодули проистекает из того, что в сложных системах практически невозможно охватить взглядом весь проект целиком, и вопрос «как протестировать это приложение» становится недопустимо сложным. Тогда приложение логически разделяется на компоненты (модули), а те, в свою очередь, на более мелкие компоненты (подмодули). И вот уже для таких небольших частей приложения создать хорошие тест-кейсы становится намного проще.

Как правило, иерархия модулей и подмодулей создаётся как единый набор для всей проектной команды, чтобы исключить путаницу из-за того, что разные люди будут использовать разные подходы к такому разделению или даже просто разные названия одних и тех же частей приложения.

Теперь – самое сложное: как выбираются модули и подмодули. В реальности проще всего отталкиваться от архитектуры и дизайна приложения.

Но что делать, если мы не знаем «внутренностей» приложения (или не очень разбираемся в программировании)? Модули и подмодули можно выделять на основе графического интерфейса пользователя (крупные области и элементы внутри них), на основе решаемых приложением задач и подзадач и т.д. Главное, чтобы эта логика была одинаковым образом применена ко всему приложению. Наличие полей «Модуль» и «Подмодуль» улучшает такое свойство тест-кейса, как прослеживаемость

Заглавие (суть) тест-кейса (title) призвано упростить и ускорить понимание основной идеи (цели) тест-кейса без обращения к его остальным атрибутам. Именно это поле является наиболее информативным при просмотре списка тест-кейсов.

Заглавие тест-кейса может быть полноценным предложением, фразой, набором словосочетаний – главное, чтобы выполнялись следующие условия:

- информативность;
- хотя бы относительная уникальность (чтобы не путать разные тест-кейсы).

Исходные данные, необходимые для выполнения тест-кейса (precondition, preparation, initial data, setup), позволяют описать всё то, что должно быть подготовлено до начала выполнения тест-кейса, например:

- состояние базы данных;

- состояние файловой системы и её объектов;
- состояние серверов и сетевой инфраструктуры.

Шаги тест-кейса (steps) описывают последовательность действий, которые необходимо реализовать в процессе выполнения тест-кейса. Общие рекомендации по написанию шагов таковы:

- начинайте с понятного и очевидного места, не пишите лишних начальных шагов (запуск приложения, очевидные операции с интерфейсом и т.п.);
- даже если в тест-кейсе всего один шаг, нумеруйте его (иначе возрастает вероятность в будущем случайно «приклеить» описание этого шага к новому тексту);
- если вы пишете на русском языке, используйте безличную форму (например, «открыть», «ввести», «добавить» вместо «откройте», «введите», «добавьте»), в английском языке не надо использовать частицу «to» (т.е. «запустить приложение» будет «start application», не «to start application»);
- соотносите степень детализации шагов и их параметров с целью тест-кейса, его сложностью, уровнем и т.д. – в зависимости от этих и многих других факторов степень детализации может варьироваться от общих идей до предельно чётко прописанных значений и указаний;
- ссылайтесь на предыдущие шаги и их диапазоны для сокращения объёма текста (например, «повторить шаги 3–5 со значением…»);
- пишите шаги последовательно, без условных конструкций вида «если… то…».

Ожидаемые результаты (expected results) по каждому шагу тест-кейса описывают реакцию приложения на действия, описанные в поле «шаги тест-кейса». Номер шага соответствует номеру результата.

По написанию ожидаемых результатов можно порекомендовать следующее:

- описывайте поведение системы так, чтобы исключить субъективное толкование (например, «приложение работает верно» – плохо, «появляется окно с надписью…» – хорошо);
- пишите ожидаемый результат по всем шагам без исключения, если у вас есть хоть малейшие сомнения в том, что результат некоего шага будет совершенно тривиальным и очевидным (если вы всё же пропускаете ожидаемый результат для какого-то тривиального действия, лучше оставить в списке ожидаемых результатов пустую строку – это облегчает восприятие);
- пишите кратко, но не в ущерб информативности;
- избегайте условных конструкций вида «если… то…».

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 формированию тест-кейсов.
3. Не менее 40 тест-кейсов для тестируемого приложения согласно форме, приведенной в таблице 1.
4. Выводы.

4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое функциональное тестирование, его цели?
2. Перечислите уровни функциональное тестирования.
3. Перечислите виды тестирования.
4. Что такое тест-кейс?
5. Перечислите основные элементы структуры тест-кейса.