Лабораторная работа №1

### Тест-дизайн. Техники тест-дизайна

*Цель работы:* научиться основным техникам тест-дизайна

*Отчет по лабораторной работе:* спецификация тестируемой формы, результирующая таблица данных, 3 позитивных тест-кейса, 3 негативных тест-кейса.

## Теоретическая часть

**Тест-дизайн** – это этап процесса тестирования ПО, на котором проектируются и создаются тестовые случаи (тест-кейсы), в соответствии с определёнными ранее критериями качества и целями тестирования.

**Тестовый случай** (Test Case) - это артефакт, описывающий совокупность шагов, конкретных условий и параметров, необходимых для проверки реализации тестируемой функции или её части.

Под тест-кейсом понимается структура вида:

Action > Expected Result > Test Result

Пример:

| **Action** | **Expected Result** | **Test Result (passed/failed/blocked)** |
| --- | --- | --- |
| Open page "login" | Login page is opened | Passed |

**Виды тест-кейсов**

Тест-кейсы разделяются по ожидаемому результату на позитивные и негативные.

*Позитивный* тест-кейс использует только корректные данные и проверяет, что приложение правильно выполнило вызываемую функцию.

*Негативный* тест-кейс оперирует как корректными, так и некорректными данными (минимум 1 некорректный параметр) и ставит целью проверку исключительных ситуаций (срабатывание валидаторов), а также проверяет, что вызываемая приложением функция не выполняется при срабатывании валидатора.

**Структура тест-кейсов** (Test Case Structure)

Каждый тест-кейс должен иметь 3 части:

| **PreConditions** | Список действий, которые приводят систему к состоянию пригодному для проведения основной проверки. Либо список условий, выполнение которых говорит о том, что система находится в пригодном для проведения основного теста состояния. |
| --- | --- |
| **Test Case Description** | Список действий, переводящих систему из одного состояния в другое, для получения результата, на основании которого можно сделать вывод о удовлетворении реализации, поставленным требованиям |
| **PostConditions** | Список действий, переводящих систему в первоначальное состояние (состояние до проведения теста - initial state) |

Примечание: Post Conditions не является обязательной частью. Это скорее всего - правило хорошего тона: "намусорил - убери за собой". Это особенно актуально при автоматизированном тестировании, когда за один прогон можно наполнить базу данных сотней или даже тысячей некорректных документов.

**Пример** тест-кейса:

*do A1, verify B1*

*do A2, verify B2*

*do A3, verify B3*

В приведенном примере конечная проверка - В3. Это значит, что именно она является ключевой. Значит, A1 и А2 - это действия приводящие систему в тестопригодное состояние. А В1 и В2 - условия того, что система находится в состоянии пригодном для тестирования. Таким образом, имеем:

| **Action** | **Expected Result** | **Test Result (passed/failed/blocked)** |
| --- | --- | --- |
| **PreConditions** |  |  |
| do A1 | verify B1 |  |
| do A2 | verify B2 |  |
| **Test Case Description** |  |  |
| do A3 | verify B3 |  |
| **PostConditions** |  |  |

PostConditions в данном примере не были описаны, но по логике вещей надо выполнить шаги, которые бы вернули систему в первоначальное состояние. (например, удалили созданную запись, или отменили бы изменения сделанные в документе)

Теперь ответим на вопрос: "Почему данное разбиение удобно использовать?"

Ответ: конечная проверка одна, т.е. в случае если тест провален (test failed) будет сразу ясно из-за чего. Т.к. если провальными окажутся проверки В1 и/или В2, то тест-кейс будет заблокирован (test blocked), из-за того, что функцию невозможно привести в тестопригодное состояние (состояние пригодное для проведения тестирования), но это не значит, что тестируемая функция не работает.

| **Action** | **Expected Result** | **Test Result (passed/failed/blocked)** |
| --- | --- | --- |
| **PreConditions** |  |  |
| do A1 | verify B1 | passed |
| do A2 | verify B2 | failed |
| **Test Case Description** |  |  |
| do A3 | verify B3 | blocked |
| **PostConditions** |  |  |

**Детализация описания тест-кейсов** (Test Case Specification)

Бытует много разных мнений об уровне детализации при написании тест кейсов, а также количестве проверок в одном тест кейсе. Все они по-своему правильные. Уровень детализации тест кейсов должен быть таков, чтобы обеспечивать разумное соотношение времени прохождения к тестовому покрытию. Т.е. до тех пор, пока покрытие тестами определенного функционала не меняется, можно уменьшать детализацию тест кейсов.

**Пример** тест-кейса 1:

| Проверка отображения страницы | | |
| --- | --- | --- |
| **Действие** | **Ожидаемый результат** | **Результат теста** |
| Открыть страницу "Вход в систему" | - Окно «Вход в систему» открыто  - Название окна - «Вход в систему»  - Логотип компании отображается в правом верхнем углу  - На форме 2 поля - Имя и Пароль  - Кнопка Вход доступна  - Ссылка "забыл пароль" - доступна | ... |

**Пример** тест-кейса 2:

| Проверка отображения страницы | | |
| --- | --- | --- |
| **Действие** | **Ожидаемый результат** | **Результат теста** |
| Открыть страницу "Вход в систему" | Отображаемая страница соответствует странице на картинке (и прилагаем изображение страницы "Вход в систему") | ... |

В примерах 1 и 2 покрытие будет одинаковым, но вот время, которое потребуется для прохождения, будет разным. Второй пример будет нагляднее.

Пример оформления тест-кейса можно скачать по адресу: http://www.protesting.ru/documentation/test\_case\_example.zip

**План** работы над тест-дизайном:

1. анализ имеющихся проектных артефактов: документация (спецификации, требования, планы), модели, исполняемый код и т.д.
2. написание спецификации по тест-дизайну (Test Design Specification)
3. проектирование и создание тестовых случаев (Test Cases)

Роли, ответственные за тест дизайн:

* Тест-аналитик - определяет "ЧТО тестировать?"
* Тест-дизайнер - определяет "КАК тестировать?"

Попросту говоря, задача тест-аналитиков и дизайнеров сводится к тому, чтобы используя различные стратегии и техники тест-дизайна, создать набор тестовых случаев, обеспечивающий оптимальное тестовое покрытие тестируемого приложения. Однако, на большинстве проектов эти роли не выделяется, а доверяется обычным тестировщикам, что не всегда положительно сказывается на качестве тестов, тестировании и, как из этого следует, на качестве программного обеспечения (конечного продукта).

**Техники тест-дизайна (Test Design Technics)**

Многие люди тестируют и пишут тестовые случаи (test cases), но не многие пользуются специальными техниками тест-дизайна. Постепенно, набираясь опыта они осознают, что постоянно делают одну и ту же работу, поддающуюся конкретным правилам. И тогда они находят, что все эти правила уже описаны.

Ниже приведено краткое описание наиболее распространенных техник тест-дизайна:

**Эквивалентное Разделение** (Equivalence Partitioning - EP). Как пример, у вас есть диапазон допустимых значений от 1 до 10, вы должны выбрать одно верное значение внутри интервала, скажем, 5, и одно неверное значение вне интервала - 0.

**Анализ Граничных Значений** (Boundary Value Analysis - BVA). Если взять пример выше, в качестве значений для позитивного тестирования выберем минимальную и максимальную границы (1 и 10), и значения больше и меньше границ (0 и 11). Анализ граничных значений может быть применен к полям, записям, файлам, или к любого рода сущностям, имеющим ограничения.

**Причина / Следствие** (Cause/Effect - CE). Это, как правило, ввод комбинаций условий (причин), для получения ответа от системы (Следствие). Например, вы проверяете возможность добавлять клиента, используя определенную экранную форму. Для этого вам необходимо будет ввести несколько полей, таких как "Имя", "Адрес", "Номер Телефона" а затем, нажать кнопку "Добавить" - эта "Причина". После нажатия кнопки "Добавить", система добавляет клиента в базу данных и показывает его номер на экране - это "Следствие".

**Предугадывание ошибки** (Error Guessing - EG). Это когда тест-аналитик использует свои знания системы и способность к интерпретации спецификации на предмет того, чтобы "предугадать" при каких входных условиях система может выдать ошибку. Например, спецификация говорит: "пользователь должен ввести код". Тест аналитик, будет думать: "Что, если я не введу код?", "Что, если я введу неправильный код? ", и так далее. Это и есть предугадывание ошибки.

**Исчерпывающее тестирование** (Exhaustive Testing - ET) - это крайний случай. В пределах этой техники вы должны проверить все возможные комбинации входных значений, и в принципе, это должно найти все проблемы. На практике применение этого метода не представляется возможным, из-за огромного количества входных значений.

**Применение техник тест дизайна при разработке тест-кейсов**

Многие знают что такое тест дизайн, но не все умеют его применять. Ниже изложен последовательный подход к разработке тестовых случаев (тест кейсов), с использованием техник тест-дизайна:

* Эквивалентное Разделение (Equivalence Partitioning), далее в тексте - EP
* Анализ Граничных Значений (Boundary Value Analysis), далее в тексте - BVA
* Предугадывание ошибки (Error Guessing), далее в тексте - EG
* Причина / Следствие (Cause/Effect), далее в тексте - CE

**План** разработки тест кейсов предлагается следующий:

1. Анализ требований.
2. Определение набора тестовых данных на основании EP, BVA, EG.
3. Разработка шаблона теста на основании CE.
4. Написание тест кейсов на основании первоначальных требований, тестовых данных и шагов теста.

Далее на примере, рассмотрим предложенный подход.

**Пример**:

Протестировать функциональность формы приема заявок, требования к которой представлены в следующей таблице:

| **Элемент** | **Тип элемента** | **Требования** |
| --- | --- | --- |
| Тип обращения | combobox | Набор данных:   1. Консультация 2. Проведение тестирования 3. Размещение рекламы 4. Ошибка на сайте |
| Контактное лицо | editbox | 1. Обязательное для заполнения 2. Максимально 25 символов 3. Использование цифр и спецсимволов не допускается |
| Контактный телефон | editbox | 1. Обязательное для заполнения 2. Допустимые символы "+" и цифры 3. "+" можно использовать только в начале номера 4. Допустимые форматы:    * начинается с плюса - 11-15 цифр +31612361264 +375291438884    * без плюса - 5-10 цифр, например: 0613261264 2925167 |
| Сообщение | text area | 1. Обязательное для заполнения 2. Максимальная длина 1024 символа |
| Отправить | button | Состояние:  По умолчанию - не активна (Disabled)  После заполнения обязательных полей становится активна (Enabled)  Действия после нажатия:   1. Если введенные данные корректны - отправка сообщения 2. Если введенные данные НЕ корректны - валидационное сообщение |

1. **Анализ требований**

Читаем, анализируем требования и выделяем для себя следующие нюансы:

* какие из полей обязательные для заполнения?
* имеют ли поля ограничения по длине или по размерности (границы)?
* какие из полей имеют специальные форматы?

1. **Определение набора тестовых данных**

Отталкиваясь от требований к полям, используя техники тест-дизайна начинаем определение набора тестовых данных:

* в зависимости от того, обязательное поле или нет, определим какие поля необходимо проверить на пустое значение, так как оно может вызывать ошибку (В результирующей таблице оранжевый цвет)
* т.к. исчерпывающее тестирование не представляется возможным из-за огромного числа всевозможных комбинаций значений, в первую очередь необходимо определить минимальный набор данных. Это можно сделать, используя такие техники, как EP и BVA. (В результирующей таблице голубой цвет)
* На форме присутствует поле, имеющее составной тип (цифры используются совместно с символами), обладает специальным форматом данных и поэтому выделение тестовых данных для него - это достаточно трудоемкая задача. В пределах данной статьи ограничимся только простой проверкой форматов и основных требований описанных в форме приема заявок.
* По завершению генерации данных, используя стандартные техники, можно добавить некоторое количество значений на основании личного опыта (техника EG) - это будет использование спец. символов, очень длинных строк, разных форматов данных, регистров в строках (Upper, Lower, Mixed cases), отрицательные и нулевые значения, ключевые слова Null - NaN - Infinity и т.д. Сюда можно включить все, что вы полагаете, может вывести приложение из строя (В результирующей таблице фиолетовый цвет)

Примечание:

Отметим, что количество тестовых данных после окончательной генерации будет достаточно большим, даже при использовании специальных техник тест дизайна. Поэтому ограничимся лишь несколькими значениями для каждого поля, так как цель данной лабораторной работы показать именно процесс создания тест кейсов, а не процесс получения конкретных тестовых данных.

* 1. **Выбор тестовых данных для каждого отдельно взятого поля**
* Поле Тип обращения. Так как все данные входят в 1 класс эквивалентности, то есть не изменяют сам процесс выполнения приема заявки, берем любую (1-ю) позицию в листе с ожидаемым результатом ОК. Но т.к. реализовано поле как лист, имеет также смысл рассмотреть и граничные условия (техника BVA), т.е. берем первый и последний элементы. Итого: 1-я и последняя позиции в листе. Ожидаемый результат при использовании - ОК.
* Поле Контактное лицо. Это обязательное поле размером от 1 до 25 символов (включая границы). Проверка на обязательность добавляет к тестовым данным пустое значение. Проведем анализ граничных условий (BVA), получим набор: 0, 1, 2, 24, 25 и 26 символов. Пустое значение (0 символов) уже было добавлено при анализе обязательности поля для ввода, поэтому при BVA мы не будем добавлять его еще раз. (если его добавить второй раз, произойдет дублирование тестовых данных, которое не приведет к нахождению новых дефектов, а значит повторное добавление в домен не имеет смысла). В связи с тем, что значения 2 и 24 символа являются, с нашей точки зрения, некритичными, их можно не добавлять. В итоге получаем, что минимальный набор данных для тестирования поля - это строки 1 и 25 - ОК, и 0 (пустое значение), 26 символов - NOK.
* поле Контактный телефон состоит из нескольких частей: код страны, код оператора, номер телефон (который может быть составной и разделенный дефисами). Для определения правильного набора тестовых данных необходимо рассматривать каждую составную часть по-отдельности. Применяя BVA и EP, получим:
* для номеров с плюсом

По BVA получим номера с 10, 11, 12 и 14, 15, 16 цифрами, где 10 и 16 - NOK, а 11, 12, 14, 15 - OK

Рассматривая полученные данные с позиции EP выделим, что 11, 12, 14, 15 входят в один класс эквивалентности. Поэтому при тестировании мы можем использовать любое из них, но так как 11 и 15 - это границы интервала, то на наш взгляд их пропускать нельзя. Следовательно мы можем уменьшить набор значений до двух, исключив 12 и 14, а оставив 11 и 15 для проверки граничных условий.

Итого имеем:

11 и 15 цифр - OK, (+12345678901, +123456789012345)

10 и 16 цифр - NOK; (+1234567890, +1234567890123456)

* для номеров без плюса:

По BVA получим номера с 4, 5, 6 и 9, 10, 11 цифрами.

Действуя аналогично примеру для номеров телефонов с плюсом, исключим значения 6 и 9, оставив 5 и 10.

Итого имеем:

5 и 10 цифр - OK, (12345, 1234567890)

4 и 11 цифр - NOK; (1234, 12345678901)

* поле Сообщение. подбор данных проводим по аналогии с полем Контактное лицо. На выходе получаем значения: строки 1 и 1024 - ОК, и 1025 символов - NOK.

**Результирующая таблица данных, для использования при последующем составлении тест кейсов**

| **Поле** | **OK/NOK** | **Значение** | **Комментарий** |
| --- | --- | --- | --- |
| Тип обращения | OK | Консультация | первый в списке |
| Ошибка на сайте | последний в списке |
| NOK |  |  |
| Контактное лицо | OK | йцукенгшщзйцукенгшщзйцуке | 25 символов нижний регистр |
| a | 1 символ |
| ЙЦУКЕНГШЩЗФЫВАПРОЛДЖЯЧСМИ | 25 символов ВЕРХНИЙ регистр |
| ЙЦУКЕНГШЩЗфывапролджЯЧСМИ | 25 символов СМеШаННыЙ регистр |
| NOK |  | пустое значение |
| йцукенгшщзйцукенгшщзйцукей | длина больше максимальной(26 символов |
| @#$%^&;.?,>|\/№"!()\_{}[<~ | спец. символы (ASCII) |
| 1234567890123456789012345 | только цифры |
| adsadasdasdas dasdasd asasdsads(...)sas | очень длинная строка (~1Mb) |
| Контактный телефон | OK | +12345678901 | с плюсом - минимальная длина |
| +123456789012345 | с плюсом - максимальная длина |
| 12345 | без плюса - минимальная длина |
| 1234567890 | без плюса - максимальная длина |
| NOK |  | пустое значение |
| +1234567890 | с плюсом - < минимальной длины |
| +1234567890123456 | с плюсом - > максимальной длины |
| 1234 | без плюса - < минимальной длины |
| 12345678901 | без плюса - > максимальной длины |
| +YYYXXXyyyxxzz | с плюсом - буквы вместо цифр |
| yyyxxxxzz | без плюса - буквы вместо цифр |
| +###-$$$-%^-&^-&! | спец. символы (ASCII) |
| 1232312323123213231232(...)99 | очень длинная строка (~1Mb) |
| Сообщение | OK | йццуйцуйц(...)йцу | максимальная длина (1024 символа) |
| NOK |  | пустое значение |
| йццуйцуйц(...)йцуц | длина больше максимальной (1025 символов) |
| adsadasdasdas dasdasd asasdsads(...)sas | очень длинная строка (~1Mb) |
| @##$$$%^&^& | только спец. символы (ASCII) |

**3. Разрабатываем шаблон теста**

На основании техники CE и, по возможности, имеющихся вариантов использования (Use case) создадим шаблон планируемого теста. Данный документ будет представлять собой шаги и ожидаемые результаты теста, но без конкретных данных, которые подставляются на следующем этапе разработки тест-кейсов.

Пример шаблона тест кейса

| **Действие** | **Ожидаемый результат** |
| --- | --- |
| 1. Открываем форму отправки сообщения | * Форма открыта * Все поля по умолчанию пусты * Обязательные поля помечены - \* * Кнопка "Отправить" не активна |
| 2. Заполняем поля формы:   * Тип обращения * Контактное лицо * Контактный телефон * Сообщение | * Поля заполнены * Кнопка "Отправить" - активна (Enabled) |
| 3. Нажимаем кнопку "Отправить" | * Если введенные данные корректны -   + Сообщение "Заявка отправлена"выведено на экран.   + Новая заявка появилась в списке на странице "Заявки". * Если введенные данные НЕ корректны -;   + Валидационное сообщение со всеми ошибками выведено на экран.   + Заявка НЕ появилась в списке на странице "Заявки". |

**4. Написание тест-кейсов на основании первоначальных требований, тестовых данных и шаблона теста**

После того, как тестовые данные и шаги теста готовы приступаем непосредственно к разработке тест кейсов. Здесь нам помогут такие методы комбинирования, как:

* Последовательный перебор. Представляет собой перебор всех возможных комбинаций имеющихся значений. Таким образом получается, что количество тест кейсов будет равно произведению количества вариантов тестовых данных для каждого поля. Для нашего конкретного примера мы получим 1170 тест-кейсов.
* Попарный перебор (Pairwise Testing). Зачастую, сбои вызывают не сложное сочетание всех параметров, а сочетание лишь пары параметров. Техника попарного перебора, позволяет создать тестовые наборы, комбинирующие данные из двух полей. Благодаря этому, количество полученных на выходе тест кейсов в разы меньше, чем при комбинировании того же набора данных при последовательном переборе. Отметим также, что в данный момент существует несколько алгоритмов генерации комбинаций для попарного тестирования: Orthogonal Arrays Testing, All pairs, IPO (In-Parameter Order). Так например, при использовании техники All Pairs в нашем конкретном случае мы получим всего 118 тест-кейсов.

По завершению подготовки комбинаций данных, подставляем их в шаблон тест-кейса, и в результате имеем набор тестовых случаев, покрывающий тестируемые нами требования к форме приема заявок.

Примечание:

Тест-кейсы разделяются по ожидаемому результату на позитивные и негативные тест кейсы.

**Пример позитивного тест кейса (все поля OK):**

| **Действие** | **Ожидаемый результат** |
| --- | --- |
| 1. Открываем форму отправки сообщения | * Форма открыта * Все поля по умолчанию пусты * Обязательные поля помечены - \* * Кнопка "Отправить" не активна |
| 2. Заполняем поля формы:   * Тип обращения = Консультация * Контактное лицо = йцукенгшщзйцукенгшщзйцуке * Контактный телефон = +7-916-111-11-11 * Сообщение | * Поля заполнены * Кнопка "Отправить" - активна (Enabled) |
| 3. Нажимаем кнопку "Отправить" | * Сообщение "Заявка отправлена"выведено на экран. * Новая заявка появилась в списке на странице "Заявки". |

**Пример негативного тест кейса (поле Контактное лицо - NOK):**

| **Действие** | **Ожидаемый результат** |
| --- | --- |
| 1. Открываем форму отправки сообщения | * Форма открыта * Все поля по умолчанию пусты * Обязательные поля помечены - \* * Кнопка "Отправить" не активна |
| 2. Заполняем поля формы:   * Тип обращения = Консультация * Контактное лицо = @#$%^&;.?,>|\/№"!()\_{}[<~ * Контактный телефон = (916)333-33-33 * Сообщение = йццуйцуйц(...)йцу - 1024 символа | * Поля заполнены * Кнопка "Отправить" - активна (Enabled) |
| 3. Нажимаем кнопку "Отправить" | * Валидационное сообщение со всеми ошибками выведено на экран: "В поле "Контактное лицо" запрещено использование цифр и спец. символов." * Заявка НЕ появилась в списке на странице "Заявки". |

## Задание

Разработать требования, результирующую таблицу данных и тест-кейсы для поисковой формы на веб-сайте согласно своему варианту.

Вариант 1, 11, 21

<https://www.google.com/>

Вариант 2, 12, 22

<http://www.yandex.ru/>

Вариант 3, 13, 23

<http://www.rambler.ru/>

Вариант 4, 14, 24

<https://mail.ru/>

Вариант 5, 15, 25

<http://www.bing.com/>

Вариант 6, 16, 26

<https://duckduckgo.com/>

Вариант 7, 17, 27

<https://www.ask.com/>

Вариант 8, 18, 28

<https://www.yahoo.com/>

Вариант 9, 19, 29S

<https://swisscows.com/ru>

Вариант 10, 20, 30

<https://gibiru.com/>