**Техники тест-дизайна**

**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.

**Роли в тест дизайне:**

* 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. На большинстве проектов эти роли выполняет QA инженер.

**Тестовое покрытие**— это одна из метрик оценки качества тестирования, представляющая из себя плотность покрытия тестами требований либо исполняемого кода.

Существуют следущее подходы к оценке и измерению тестового покрытия:

* **Покрытие требований (Requirements Coverage)** — оценка покрытия тестами функциональных и нефункциональных требований к продукту, путем построения матриц трассировки (traceability matrix).
* **Покрытие кода (Code Coverage)** — оценка покрытия исполняемого кода тестами, путем отслеживания непроверенных в процессе тестирования частей программного обеспечения.
* **Тестовое покрытие на базе анализа потока управления** — оценка покрытия основанная на определении путей выполнения кода программного модуля и создания выполняемых тест кейсов для покрытия этих путей.

**Техники 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» — это «Причина». После нажатия кнопки «Добавить», система добавляет клиента в базу данных и показывает его номер на экране — это «Следствие».
* **Предугадывание ошибки (Error Guessing — EG).** 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.
* **Парное тестирование (Pairwise Testing — PT)** — это техника формирования наборов тестовых данных. Сформулировать суть можно, например, таким образом: формирование таких наборов данных, в которых каждое тестируемое значение каждого из проверяемых параметров хотя бы единожды сочетается с каждым тестируемым значением всех остальных проверяемых параметров.

**Пример использования техники анализа классов эквивалентности:**

Эквивалентное разделение, алгоритм использования техники:

1. Необходимо определить класс эквивалентности. Это главный шаг техники. От него во многом зависит эффективность её применения.
2. Затем нужно выбрать одного представителя от каждого класса. На этом шаге из каждого эквивалентного набора тестов мы выбираем один тест.
3. Нужно выполнить тесты. На этом шаге мы выполняем тесты от каждого класса эквивалентности.

Можно разбивать тесты на классы эквивалентности по разным принципам. Например, если мы тестируем поле ввода, которое принимает максимум 5 символов, то можем выбрать разные принципы разбиения на классы эквивалентности:

* По количеству символов.
* По типу символов (цифры, буквы, спец символы).

**Пример:** функцию подсчета комиссии при отмене бронирования авиабилетов. Размер комиссии зависит от времени до вылета, когда совершена отмена:

* За 5 суток до вылета комиссия составляет 0%.
* Меньше 5 суток, но больше 24 часов – 50%.
* Меньше 24 часов, но до вылета – 75%.
* После вылета – 100%.

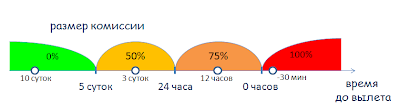


Рис. 5.1. Пример: функция подсчета комиссии при отмене бронирования авиабилетов

**Шаги:**

**1. Определим классы эквивалентности:**

(для каждого теста из этих классов мы ожидаем получить одинаковый результат):

* 1 класс: время до вылета > 5 суток.
* 2 класс: 24 часа < время до вылета < 5 суток.
* 3 класс: 0 часов < время до вылета < 24 часа.
* 4 класс: время до вылета < 0 часов (вылет уже состоялся).

**2. Выберем представителя от каждого класса:**

Здесь мы можем поступить так, как нам хочется и выбрать любые значения из класса. Ведь, если предположить, что мы правильно разбили на классы эквивалентности, то нет разницы, какое значение из диапазона мы выберем.

* время до вылета = 10 суток (тест из 1-го класса).
* время до вылета = 3 суток (тест из 2-го класса).
* время до вылета = 12 часов (тест из 3-го класса).
* время до вылета = -30 мин (тест из 4-го класса).

**3. Выполним тесты:**

* Отменим бронь за 10 суток до вылета и проверим, что комиссия составила 0%.
* Отменим бронь за 3 суток до вылета и проверим, что комиссия составила 50%.
* Отменим бронь за 12 часов до вылета и проверим, что комиссия составила 75%.
* Отменим бронь через 30 мин после вылета и проверим, что комиссия составила 100%.

Фактически, осталось всего 4 теста. А сколько возможных тестов существует? Даже если мы введем ограничение, что отмена бронирования может произойти в рамках 10 суток до вылета и 1 суток после вылета, то у нас будет около 950400 возможных тестов (количество секунд в 11 сутках).

**Плюсы и минусы техники анализа классов эквивалентности**

* К плюсам можно отнести заметное сокращение времени и улучшение структурированности тестирования.
* К минусам можно отнести то, что, при неправильном использовании техники, мы рискуем потерять баги.

**Пример использования техники анализа граничных значений**

Примерный алгоритм использования техники анализа граничных значений:

1. Во-первых, нужно выделить классы эквивалентности. Опять же, это очень важный шаг и от правильности разбиения на классы эквивалентности зависит эффективность тестов граничных значений.
2. Далее нужно определить граничные значения этих классов.
3. Нам нужно понять, к какому классу будет относиться каждая граница.
4. Для каждой границы нам нужно провести тесты по проверке значения до границы, на границе, и сразу после границы.

**Пример:** функцию подсчета комиссии при отмене бронирования авиабилетов. Размер комиссии зависит от времени до вылета, когда совершена отмена:

* За 5 суток до вылета комиссия составляет 0%.
* Меньше 5 суток, но больше 24 часов – 50%.
* Меньше 24 часов, но до вылета – 75%..
* После вылета – 100%

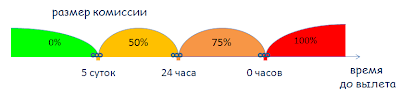


Рис. 5.2. Пример: функция подсчета комиссии при отмене бронирования авиабилетов

**Шаги:**

**1. Выделим классы эквивалентности:**

* 1 класс: время до вылета > 5 суток.
* 2 класс: 24 часа < время до вылета < 5 суток.
* 3 класс: 0 часов < время до вылета < 24 час.
* 4 класс: время до вылета < 0 часов (вылет уже состоялся).

**2. Определим границы:**

* 5 суток.
* 24 часа.
* 0 часов.

**3. Определим, к какому классу относятся границы:**

Примечание: На этом шаге был спорный момент, на который нужно обратить внимание. При составлении задачи я не подумал, какая логика должна быть на самих границах. Это типичный пример того, как составители требований не задумываются о границах. И поэтому программист может трактовать их по-своему.

1. 5 суток – ко 2-му классу.
2. 24 часа – к о2-му классу.
3. 0 часов – к 4-му классу.

**4. Протестируем значения на границах, до и после них:**

1. Отменим бронь за 5 суток + 1 секунду до вылета (или просто постараемся выполнить бронь как можно ближе к границе, но слева от нее) и проверим, что комиссия равна 0%.
2. Отменим бронь ровно за 5 суток до вылета и проверим, что комиссия равна 50%.
3. Отменим бронь за 5 суток – 1 секунду до вылета и проверим, что комиссия равна 50%.
4. Отменим бронь за 24 часа + 1 секунду до вылета и проверим, что комиссия равна 50%.
5. Отменим бронь ровно за 24 часа до вылета и проверим, что комиссия равна 50%.
6. Отменим бронь за 24 часа — 1 секунду до вылета и проверим, что комиссия равна 75%.
7. Отменим бронь за 1 секунду до вылета и проверим, что комиссия равна 75%.
8. Отменим бронь ровно во время вылета и проверим, что комиссия равна 100%.
9. Отменим бронь спустя 1 секунду после вылета и проверим, что комиссия равна 100%.

Мы получили 9 тестов, по 3 теста на каждую границу.

Если суммировать тесты, необходимые для проверки классов эквивалентности и граничных значений, получим 4 + 9 =13 тестов.

**Плюсы и минусы техники анализа граничных значений**

Эта техника добавляет в технику анализа классов эквивалентности ориентированность на конкретный тип ошибок.

То есть, техника анализа классов эквивалентности просто говорит нам о том, что нужно разбить все тесты на классы и провести тестирование всех классов. А техника граничных значений ориентирована на обнаружение конкретной проблемы – возникновения ошибок на границах классов эквивалентности.

Но, как и для техники анализа классов эквивалентности, эффективность техники анализа граничных значений зависит от правильности ее использования. Мы должны приложить усилия, чтобы правильно определить классы эквивалентности и их границы. Если мы отнесемся к этому поверхностно, то рискуем пропустить ошибки.