

Лабораторная работа № 2. Разработка тестов и тестовых сценариев

Цель работы

Изучение понятий чек-лист и тест-кейс и получение навыков их составления.

Краткие теоретические сведения

Чек-лист (check-list) – набор идей тестов.

Тест-кейс (test case) – набор входных данных, условий выполнения и ожидаемых результатов, разработанный с целью проверки того или иного свойства или поведения программного средства.

Тестовый сценарий, тест-сюит (test scenario, test-suite) – набор тест-кейсов, собранных в группу (последовательность) для достижения некоторой цели.

1. Разработка тестов

Существует много подходов к проектированию тестов.

Например, можно создавать:

- Тесты на основе требований (requirements based tests)
- Функциональные тесты (functional test)
- Сравнительные («параллельные») тесты (parallel testing)
- Сценарные тесты (scenario tests)
- Тесты ошибочных ситуаций (fault injection tests)
- Тесты интерфейса (interface tests, GUI tests)
- Тесты удобства использования (usability tests)
- Тесты упаковки и документации (packaging/documentation tests)
- Стрессовые тесты (stress tests)
- Тесты производительности (performance tests)
- Конфигурационные тесты (configuration tests)
- Законодательные тесты (regulation tests)

2. Классы эквивалентности и граничные условия

Класс эквивалентности (equivalence class) – набор тестов, полное выполнение которого является избыточным и не приводит к обнаружению новых дефектов.

Признаки эквивалентности (несколько тестов эквивалентны,

если):

- Они направлены на поиск одной и той же ошибки.
- Если один из тестов обнаруживает ошибку, другие её тоже не обнаружат.
- Если один из тестов НЕ обнаруживает ошибку, другие её тоже, скорее всего, НЕ обнаружат.
- Тесты используют схожие наборы входных данных.
- Для выполнения тестов мы совершаем одни и те же операции.
- Тесты генерируют одинаковые выходные данные или приводят приложение в одно и то же состояние.
- Все тесты приводят к срабатыванию одного и того же блока обработки ошибок («error handling block»).
- Ни один из тестов не приводит к срабатыванию блока обработки ошибок («error handling block»).

Граничные условия (border conditions) – это те места, в которых один класс эквивалентности переходит в другой.

Граничные условия очень важны, и их обязательно следует проверять в тестах, т.к. именно в этом месте чаще всего и обнаруживаются ошибки.

Пример

Проверить реакцию приложения на ввод слишком короткого (менее трёх символов) или слишком длинного (более 20-ти символов) имени пользователя, которое может содержать только английские буквы, цифры и знак подчёркивания.

Классы эквивалентности:

1. Позитивный тест: строка допустимых символов длиной от трёх до 20-ти символов.
2. Негативный тест: строка короче трёх символов.
3. Негативный тест: строка длиннее 20-ти символов.
4. Негативный тест: строка длиной от трёх до 20-ти символов, содержащая недопустимые символы.

Тесты:

1. ABCD, ABCDEFGHIJKLMNOPQRST, abc_12_def
2. AA, {пустая строка}
3. AAAAAAAAAAAAAAAAAAAAAA (21 символ)
4. Abcd#23456% @#&#%^#

Выводы по классам эквивалентности

- Классы эквивалентности не всегда очевидны.
- Как правило, негативных тестов получается больше, чем

позитивных.

- Принадлежность теста к позитивным или негативным зависит от требований.

3. Рекомендации по разработке тестов

- Начинайте с простых очевидных тестов.
- Затем переходите к более сложным тестам.
- Помните о граничных условиях.
- Если остаётся время, занимайтесь исследовательским тестированием.

Последовательность разработки и выполнения тестов

- Простые позитивные.
- Простые негативные.
- Сложные позитивные.
- Сложные негативные.

4. Оформление тест-кейсов

Общие идеи оформления тест-кейсов проиллюстрированы на рис. 2.1.

Приоритет	Связанное с тестом требование		Заглавие (суть) теста		Ожидаемый результат по каждому шагу
UG_U 1.12	A	R97	Галерея	Загрузка файла	Галерея, загрузка файла, имя со спецсимволами
Идентификатор					Приготовление: создать непустой файл с именем #\$\$%^&.j.p
					1. Нажать кнопку «Загрузить картинку»
					2. Нажать кнопку «Загрузить данные, необходимые для выполнения теста»
					3. Выбрать из списка подготовленный файл
					4. Нажать кнопку «Добавить в галерею»
					5. Нажать кнопку «Добавить в галерею»
					Шаги
					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

Рис. 2.1. Общие идеи оформления тест-кейсов

5. Свойства хороших тестов

Специфичность и общность.

Когда все детали прописаны до мелочей:

- при повторных выполнениях теста всегда будут выполняться строго одни и те же действия, что снижает вероятность обнаружить ошибку;

- возрастает время создания и поддержки теста.

Общий тест-кейс сложно выполнять по многим объективным и субъективным причинам, а потому: он сложен для начинающих тестировщиков; он вполне может остаться невыполненным.

Специфичный тест-кейс

1. В поле А ввести 10	4. Значение в поле С равно 25
2. В поле В ввести 15	
3. Нажать кнопку «Сложить»	
4. Проверить значение в поле С	

Общий тест-кейс

1. Проверить, что программа суммирует два числа корректно	4. Суммирует корректно
---	------------------------

Вывод и решение

Сложение А и В 1. В поле А ввести корректное целое число 2. В поле В ввести корректное целое число 3. Нажать кнопку «Сложить» 4. Проверить значение поля С 5. Повторить шаги 1-4 для значений: 0, максимального и минимального допустимого значений.	4. Значение поля С равно сумме А и В
--	--------------------------------------

Здесь мы не привязаны к конкретным значениям. Мы знаем, как проверить результат. Мы сокращаем время написания и поддержки теста ссылкой на шаги 1-4. Мы перечислили значения, представляющие для нас особый интерес.

Простота и сложность.

Простые тесты:

1. Откройте файл «1.txt». Файл открыт.
2. Введите слово «Дом». Появляется слово «Дом».
3. Сохраните файл. Кнопка «Сохранить» становится неактивной.

Сложный тест:

1. В документе размером более 100 Мб создайте таблицу 100x100, в ячейку 50x50 вставьте картинку размером 30 Мб, применив к ней функцию «Авторасположение». Проверьте результат.

Простые тесты оперируют за раз одним объектом. Преимущества простых тестов:

- Их легко выполнять.
- Они понятны новичкам.
- Они упрощают диагностику ошибки.
- Они делают наличие ошибки очевидным.

Преимущества сложных тестов

- Больше шансов что-то сломать.
- Пользователи, как правило, используют сложные сценарии.
- Программисты сами редко проверяют такие варианты.

По названным причинам следует постепенно повышать сложность тестов.

Независимость и связанность.

Независимые тесты не ссылаются ни на какие другие.

Связанные тесты явно или неявно (в рамках сценария) ссылаются на другие (как правило, на предыдущий).

Преимущества независимых тестов

- Легко и просто выполнять.
- Такие тесты могут работать даже после краха приложения

на других тестах.

- Такие тесты можно группировать любым образом и выполнять в любом порядке.

Преимущества связанных тестов и связанных сценариев

- Они имитируют работу реальных пользователей.
- Они удобны для интеграционного тестирования.
- Они удобны для разбиения на части тестов с большим количеством шагов.

- Следующий в наборе тест использует данные и состояние приложения, подготовленные предыдущим.

Промышленным стандартом являются независимые тесты. Использование сценариев не запрещено, но не следует делать их слишком длинными.

Язык написания тестов

Используйте активный залог: («open», «paste», «click»). В русском языке используйте безличную форму: «открыть» (вместо «откройте»).

Описывайте поведение системы: «появляется окно...», «приложение закрывается».

Используйте простой технический стиль.

ОБЯЗАТЕЛЬНО указывайте **ТОЧНЫЕ** названия всех элементов приложения.

Резюмируя вышесказанное можно отметить, что хороший тест:

- обладает высокой вероятностью обнаружения ошибки.
- исследует соответствующую («ту, которую надо») область приложения.
- выполняет какие-то интересные действия.
- не выполняет ненужных действий.
- является не слишком простым, но и не слишком сложным.
- не является избыточным по отношению к другим тестам.
- делает обнаруженную ошибку очевидной.
- позволяет легко диагностировать ошибку.

Процесс разработки тестов

Общие рекомендации по процессу разработки тестов можно определить следующим образом:

- Начинайте как можно раньше.
- Разбивайте приложение на отдельные части/модули.
- Для каждой области/модуля пишите чек-лист.
- Пишите вопросы, уточняйте детали, добавляйте «косметику».
- Получите рецензию коллег, разработчиков, заказчиков.
- Обновляйте тесты, как только обнаружили ошибку или изменилась функциональность приложения.

Тестовые сценарии

Тестовый сценарий (test scenario) – набор тестов (тест-кейсов), собранных в последовательность для достижения некоторой цели. Может быть составлен из связанных или независимых тестов.

Хороший тестовый сценарий всегда следует некоторой логике, например: типичному использованию приложения, удобству тестирования, распределению функций по модулям и т.д.

Общие рекомендации по оформлению тестовых сценариев

- Пишите сценарий для отдельной части приложения.
- Пишите отдельно сценарии для Smoke и Critical Path тестов.
- Постепенно повышайте сложность тестов.
- Организуйте сценарий логично.

Практическая часть

На основе представленного ниже набора требований сформируйте:

- Смоук-тест.
- Чек-лист для теста критического пути.

- Сам тест критического пути (необходимо расписывать идеи из чек-листа в полноценные тесты).

В приведенных требованиях по-прежнему есть ошибки, но сегодня цель не состоит в тестировании требований, необходимо писать тесты – и делать их настолько хорошими, насколько это возможно. Необходимо искать ошибки, противоречащие здравому смыслу.

В работе содержатся 3 блока заданий, каждый из них рассчитан на двухчасовое занятие. По результату каждой части работы

Часть 1.

Требования к разрабатываемому приложению

Общие положения: приложение должно выполнять математические вычисления.

1. Приложение должно работать под всеми версиями ОС Windows.

2. Приложение должно быть максимально похоже на стандартный калькулятор Windows за исключением некоторых особенностей.

3. Несколько приложений должны иметь возможность работать одновременно.

4. При запуске приложения должно отображаться окно со стандартными для калькулятора кнопками и полем ввода и отображения данных.

5. Для начала вычислений пользователь должен нажать кнопку "Начать".

6. Приложение должно позволять легко сохранять вычисления в выбранном пользователем формате.

7. Опционально предусматривается поддержка нескольких языков.

8. Приложение должно позволять выполнять вычисления сразу же после запуска.

9. Скорость вычислений должна быть максимально высокой.

10. Приложение должно позволять выполнять следующие операции: сложение, умножение, вычитание и деление чисел.

11. Приложение должно позволять строить графики простых функций.

12. Приложение должно запрашивать подтверждение ("Результат не сохранён. Выйти?") в случае, если пользователь не сохранил результаты работы.

Часть 2.

Требования к разрабатываемому приложению

Общие положения: Приложение предназначено для сбора информации о семье, т.е. муже, жене, детях. Язык приложения – английский.

1. Приложение должно поддерживать все версии всех операционных систем.

2. Информация о каждой семье ("база данных семьи") должна храниться в 1 файле, который может быть сохранён с пустыми полями.

3. Несколько приложений должны иметь возможность работать одновременно.

4. После запуска приложений окно должно содержать меню с тремя стандартными кнопками со всплывающими подсказками ("New Database", "Open Database", "Save Database"). Закладка "General" должна быть выбрана по умолчанию (см. рисунок 2.2).

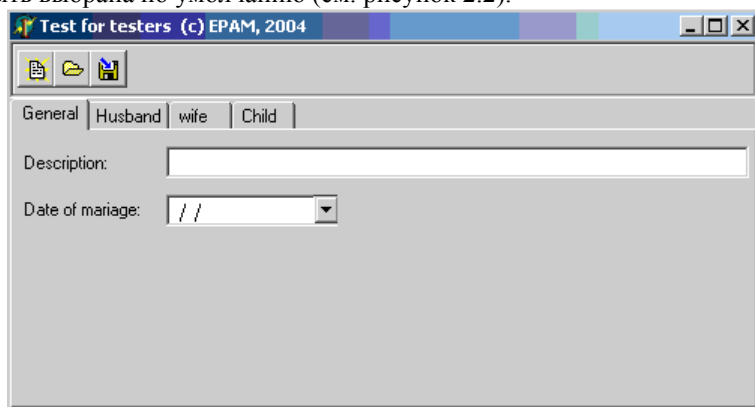


Рис. 2.2. Вкладка «General»

4.1 Эти ("New, Open, Save") кнопки также должны быть доступны для остальных трёх закладок: "Husband", "wife", "Children". Кнопка "Save" должна быть недоступна, пока пользователь не кликнет по кнопке "New" или "Open".

5. Для создания новой базы данных семьи пользователь должен кликнуть по кнопке "New". Пользователь может заполнить описание семьи ("Description", см. приложение 2) и выбрать дату свадьбы ("Date of marriage") с использованием календаря "Calendar" (приложение 1).

6. Чтобы открыть существующую базу данных, пользователь должен кликнуть по кнопке "Open", выбрать файл в диалоговой форме и кликнуть "OK".

6.1. Если пользователь редактирует новую (или открытую) базу данных, а затем кликает по кнопкам "New" или "Open", приложение должно предложить выполнить сохранение текущей базы (изменений).

7. Чтобы сохранить текущую базу данных семьи пользователь должен нажать "Save".

7.1 Имя сохранённой БД должно быть явно видно.

7.2 Если новая БД ещё не редактировалась, кнопка "Save" должна быть недоступна.

7.3 Если пользователь пытается сохранить файл БД под именем уже существующего файла, приложение должно отобразить предупреждение.

8. Опционально должна быть поддержка приложением нескольких языков.

9. Приложение должно позволять только заполнение данных о семье и сохранение таких данных. Приложение не должно позволять работать с ранее созданными базами данных семей.

10. Для ввода информации о муже пользователь должен перейти на закладку "Husband" (см. рисунок 2.3)

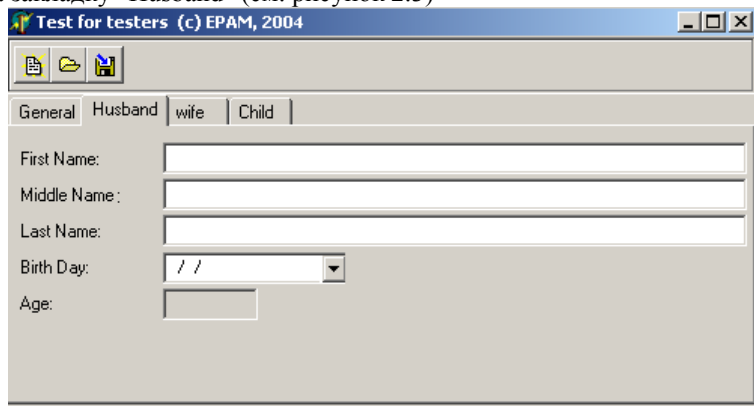
The image shows a screenshot of a software application window titled "Test for testers (c) EPAM, 2004". The window has a standard Windows-style title bar with minimize, maximize, and close buttons. Below the title bar is a toolbar with three icons: a document, a folder, and a save icon. The main area of the window contains a tabbed interface with four tabs: "General", "Husband", "wife", and "Child". The "Husband" tab is currently selected. Below the tabs, there are five input fields: "First Name:" with a text box, "Middle Name:" with a text box, "Last Name:" with a text box, "Birth Day:" with a text box containing "/" and a dropdown arrow, and "Age:" with a text box.

Рис. 2.3. Вкладка «Husband»

10.1 О муже может быть сохранена следующая информация: Имя, Отчество, Фамилия (см. приложение 2), день рождения (приложение 1).

10.2 Поле "Возраст" ("Age") должно иметь значение по умолчанию. Поле возраст должно быть R/O. Его значение должно вычисляться, как только будет заполнено поле "День рождения" ("Birth Day").

11. Приложение должно поддерживать функциональность по вводу и сохранению полной информации о жене.

12. Чтобы ввести данные о детях, пользователь должен перейти на закладку "Ребёнок" ("Children") (см. рисунок 2.4).

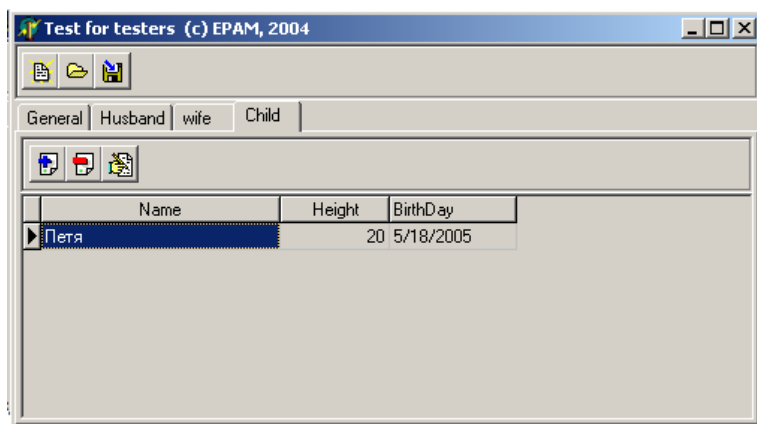


Рис. 2.4. Вкладка «Children»

13. Таблица со списком детей (2 ряда и 3 колонки) должна быть пустой для новой базы данных. Эти три колонки должны иметь имена "Имя" ("Name") и "Рост" ("Height") соответственно.

13.1 Ячейки в таблице нельзя редактировать, но их можно выделять.

14. Пользователь может добавить более одного ребёнка.

15. На этой странице должна быть прокрутка.

16. Закладка "Дети" ("Children") должна иметь три кнопки со всплывающими подсказками: "Add Child", "Del Child", "Edit Child".

16.1 Если в таблице нет записей, кнопка "Del Child" должна быть недоступной.

16.2 Кнопка "Add" должна быть доступна всегда.

17. При клике по "Add Child" должна появляться форма с тремя полями:

- "Имя" ("Name") (см. приложение 2)

- "Рост" ("Height")

- "Дата рождения" ("Birth Day") (см. приложение 1)

17.1 После клика по "OK" новая запись должна появиться в таблице со всеми заполненными полями.

17.2 Если пользователь нажимает "Cancel", ничего не должно происходить.

18. Для удаления ребёнка нужно выбрать запись и нажать "Del Child". Должно появиться соответствующее сообщение. Если пользователь кликает "OK", запись удаляется. Если пользователь кликает "Cancel", запись остаётся.

18.1 После успешного удаления записи, доступным для редактирования остаётся только поле "Дата рождения" ("Birth Day") (но не "Имя" ("Name") или "Рост" ("Height")).

19. Если пользователь кликает "Edit Child", должна появиться форма с тремя атрибутами. Если пользователь кликает "ОК", изменения сохраняются и отображаются в таблице, а если он кликает "Cancel", изменения не сохраняются.

Приложение 1. Поле "Date" должно иметь формат даты. Остальные форматы должны игнорироваться приложением. Дата может быть пустой. Дата может быть введена вручную или выбрана из всплывающего календаря ("Calendar"). Календарь должен иметь нормальный размер и интуитивно-понятный интерфейс.

Приложение 2. Приложение должно позволять вводить в поле буквенные и иные символы

- Максимальная длина поля – 50.
- Поле может быть пустым.

Часть 3.

Требования к приложению «File Searcher»

1. Приложение «File Searcher» (далее FS) предназначено для автоматического поиска файлов по заданному шаблону.

2. Приложение должно быть написано на Delphi 7 и работать под Win XP и Win 7.

3. Для поиска указывается начальный каталог или набор каталогов. FS автоматически сканирует каталоги на неограниченную глубину вложенности и отображает все найденные файлы в правой панели (см. рисунок 2.5).

4. Для поиска доступно три типа файлов (выбор производит вручную или с помощью комбо-бокса «Что искать»):

- a. Аудиофайлы (mp3, ogg, wav, mid).
- b. Видеофайлы (avi, mpg, mpeg).
- c. Офисные файлы (doc, docx, xls,xlsx).

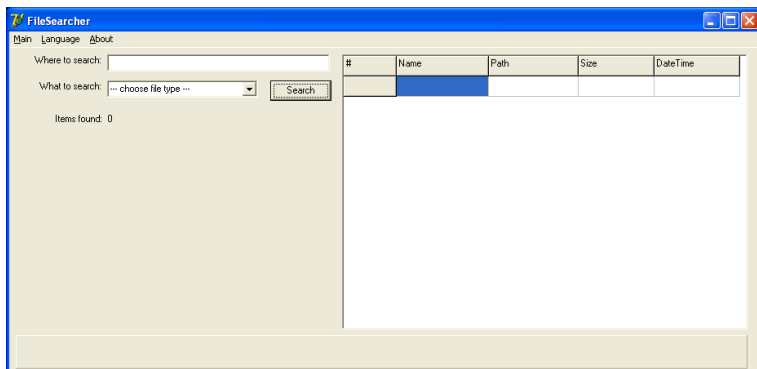


Рис. 2.5. Окно приложения «FileSearcher»

5. По всем найденным файлам отображается:
 - a. Имя.
 - b. Полный путь.
 - c. Размер.
 - d. Дата-время создания файла.
 - e. Скриншот с первым кадром.
6. Производительность.
 - a. FS должно искать не менее 500 файлов в секунду при условии, что скорость чтения/записи на диск превышает 50 Мб в секунду.
 - b. В случае если общее время операции превышает 1 час, приложение не должно начинать работу.
7. Поддержка языков.
 - a. Приложение должно поддерживать русский и английский языки по умолчанию.
 - b. Должна быть возможность добавлять новые языки.
8. Логирование.
 - a. FRS должно вести лог своей работы.
 - b. Если размер лога превышает 1 Мб, ведение лога прекращается.
 - c. Текущий анализируемый каталог должен отображаться в панели «Сейчас проверяется» внизу экрана.
9. Поддержка файловых систем:
 - a. Должны поддерживаться все файловые системы Windows и UNIX.
 - b. В случае обнаружения неподдерживаемой FS, FS должно аварийно завершать работу.
10. Должна быть поддержка сети.

Содержание отчета

В отчете необходимо отразить цель работы, основные полученные результаты с описанием процесса выполнения, комментариями и выводы.

Контрольные вопросы

1. Что такое чек-лист?
2. Что такое тест-кейс?
3. В чем принципиальное отличие чек-листа от тест-кейса?
4. Поясните понятие класса эквивалентности.
5. Сформулируйте общие рекомендации к разработке тестов
6. Перечислите свойства хороших тестов и поясните каждое из

них.