Санкт-Петербургский Государственный университет

Математико-механический факультет

Разработка приложения для автоматизации тестирования пользовательского интерфейса

Курсовая работа студента 243 группы

Елфимовой Марии Алексеевны

Научный руководитель Зуева Т.

Санкт-Петербург

2014

Оглавление

Введение

В настоящее время большое количество пользовательских приложений имеют графический интерфейс. При тестировании этих проектов разработчики часто используют средства автоматизации тестирования, позволяющие ускорить и облегчить процесс написания тестов.

Тестированием пользовательских интерфейсов, в том числе и созданием автоматических тестов, могут заниматься работники, имеющие достаточно малый навык программирования. Ввиду этого становится целесообразным создание приложения для автоматической генерации тестов на основе составленного человеком тестового сценария, имеющего несложный, но включающий в себя всю нужную функциональность, интерфейс.

Обзор существующих решений

Для облегчения постановки задачи были рассмотрены уже существующие системы автоматизации тестирования пользовательского интерфейса и проведен их краткий анализ.

Test Complete – система для автоматизации тестирования пользовательских интерфейсов, позволяющая писать тесты вручную, а также генерировать их по действиям пользователя с тестируемым интерфейсом. Эта система позволяет сохранять последуемость тестируемых действий в виде программного кода тестов, а так же редактировать результирующий тестовый код.
Selenium IDE - дополнение к браузеру Firefox для создания и запуска тестовых сценариев в собственной среде выполнения. Создано для тестирования функциональности веб-страниц.

QuickTest Professional - один из ведущих инструментов автоматизации тесторования функциональности пользовательского интерфейса. При записи тестов позволяет контролировать генерируемый код, что значительно уменьшает время создания тестов. Этот продукт обладает широкой функциональностью, но вместе с тем и высокой стоимостью.

Постановка задачи

В отличие от существующих средств тестирования, планировалось создать продукт с возможностью построения, сохранения и загрузки тестовых сценариев, а не только генерирующих сами тесты. Такая функциональность нужна для того, чтобы иметь возможность не строить заново тестовые сценарии при небольших изменениях приложения, а изменять уже существующие, и по измененным сценарием генерировать уже исправленные тесты.

Таким образом, была поставлена задача создания приложения для генерации кода тестовых сценариев, включающего в себя графический интерфейс, генерация промежуточного кода по графическому представлению и генерацию кода тестов на языке c# по промежуточному коду.

В частности, задача включала в себя:

- Разработку схемы в формате xsd для промежуточного представления тестовых сценариев
- Реализацию генератора тестов с использованием схемы на основе данных, полученых из промежуточного представления сценариев
- Реализацию класса, подключающего сгенерированные файлы с тестовыми классами к проекту в VS

Описание решения

Промежуточное представление тестовых сценариев было решено представлять в формате xml. Файлы в этом формате имеют древовидную структуру, что облегчает графическое представление сценариев. Также есть возможность создать на основе такого файла объект описываемого разработчиком класса с помощью построенной для этой цели xsd-схемы - этот процесс называется десериаллизацией.

На основе полученного объекта тестового сценария необходимо было создать код тестового файла с помощью библиотеки CodeDom. После этого файл нужно было добавить к уже готовому тестовому проекту.

Тестовый проект, к которому было решено присоединять сгенерированные файлы сценария, должен иметь описание базового тестового класса, от которого наследуются все сгенерированные классы. Базовый класс должен содержать в себе информацию о пути доступа к тестируемому приложению, а так же запрашивать стартовое окно приложения.

В генерируемых тестов стартовое окно приложения передается конструкторам специальных классов, которые облегчают доступ к функциональности окон.

Используемые средства реализации

В ходе решения существующих задач были использованы следующие программные средства:

- LinqToXsd – средство для генерации описания класса на основе xsd-схемы и серриализации объектов этого класса в формат xml, а так же десерриализации из этого формата в объект созданного класса.
- CodeDom – средство для построения дерева кода и его записи в файл.

Реализация

Для хранения промежуточного предсавления была разработана схема в формате xsd, которая описывает xml-файлы, содержащие всю необходимую для генерации тестовых сценариев информацию, то есть данные о последовательности действий с различными контролами окна, имя и тип используемых контролов, их методов и свойств, а также передаваемых параметров и методов проверки. В схеме xsd это было отражено в древовидной структуре со всеми необходимыми элементами.

Для серриализации по разработанной схеме генерируется средствами LinqToXsd файл формата *.cs, описывающий класс, объекты которого создаются на основе промежуточного представления. С помощью этого класса и схемы создается объект, содержащий всю информацию, включенную в xml-файл тестового сценария, и представляющий из себя вложенные списки, соответствующие дереву, описанному схемой

По объекту последовательным просмотром списков средствами Codedom строится дерево кода, которое потом сохраняется в генерируемом файле формата *.cs

Для уменьшения описания необходимых действий был также создан класс «помощник», конструктор которого принимает на вход путь до промежуточного представления тестового сценария, выполняет генерацию кода теста, запись в файл и подключение файла к тестовому проекту.

Результаты

- Реализована серриализация xml файлов в объекты, для этого разработана схема
- Реализован генератор, создающий код тестого класса, использующего в своих методах некоторые проверки прохождения тестов
- реалиизован линковщик генерируемого файла к тестовому проекту, в котором уже добавлен базовый тестовый класс

Список литературы

список литературы