Министерство Образования Республики Беларусь
Учреждение образования
Белорусский государственный университет
информатики и радиоэлектроники
Кафедра инженерной психологии и эргономики

УДК 331.101.1:004.42

Савосик
Артем Андреевич

ЭРГОНОМИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ФРЕЙМВОРКА АВТОМАТИЧЕСКОГО ТЕСТИРОВАНИЯ ПРОГРАММНЫХ ПРИЛОЖЕНИЙ

АВТОРЕФЕРАТ
на соискание академической степени магистра техники и технологии

1-23 80 08 Психология труда, инженерная психология, эргономика

Магистрант А.А. Савосик,

Научный руководитель
М.М. Меженная, кандидат
технических наук, доцент

Заведующий кафедрой
ИПиЭ
К.Д. Яшин, кандидат
технических наук, доцент

Нормоконтролер
Е.С. Иванова,
Ассистент кафедры ИПиЭ

Минск 2019

ВВЕДЕНИЕ

Первые программные системы разрабатывались в рамках проектов научных исследований или для нужд министерств обороны. Тестирование таких продуктов проводилось формализовано с записью всех тестовых процедур, тестовых данных, полученных результатов.

Позднее много внимания стало уделяться «исчерпывающему» тестированию, которое должно проводиться с использованием всех путей в коде или всех возможных входных данных. Было отмечено, что в этих условиях полное тестирование программного обеспечения невозможно, потому что, во-первых, количество возможных входных данных очень велико, во-вторых, существует множество путей. По этим причинам «исчерпывающее» тестирование было отклонено и признано теоретически невозможным.

В начале 1990-х годов в понятие «тестирование» стали включать планирование, проектирование, создание, поддержку и выполнение тестов и тестовых окружений, и это означало переход от тестирования к обеспечению качества, охватывающего весь цикл разработки программного обеспечения. В середине 1990-х годов с развитием Интернета и разработкой большого количества веб-приложений особую популярность стало получать «гибкое тестирование» (по аналогии с гибкими методологиями программирования).

В 2000-х появилось ещё более широкое определение тестирования, когда в него было добавлено понятие «оптимизация бизнес-технологий». Основной подход заключается в оценке и максимизации значимости всех этапов жизненного цикла разработки программного обеспечения для достижения необходимого уровня качества, производительности, доступности.

Начиная с 2000-х большую популярность в проектах, связанных с информационными технологиями, набирает автоматизированное тестирование.

Автоматизация тестирования позволяет оптимизировать расходы за счет упрощения проведения регрессии и прочих видов тестов, которые в автоматическом режиме проходит гораздо быстрее мануального способа.

Однако удобство работы с современными инструментами, использующимися в этой сфере, по-прежнему остается под вопросом. Уровень входа большинства технологий довольно высок, что мешает молодым специалистам ускоренно вливаться в отрасль. Поэтому повышение эргономичности фреймворков автоматического тестирования программных приложений – актуальная проблема современной автоматизации тестирования.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Целью магистерской диссертации является создание фреймворка, автоматизирующего рабочий процесс тестирования веб-приложений.

Объект исследования – фреймворк автоматического тестирования программных приложений.

Предмет исследования – технологии юзабилити–тестирования фреймворков автоматического тестирования программных приложений.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Выполнить аналитический обзор современных методов и технологий автоматического тестирования программных приложений
2. Разработать фреймворк автоматического тестирования программных приложений
3. Провести юзабилити-тестирование разработанного фреймворка и его модификацию по результатам тестирования.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

В первой главе магистерской работы проводится теоретический анализ технической литературы, изучение современных технологий и методов как мануального, так и автоматического тестирования программных приложений.

Автоматизация тестирования обладает следующими преимуществами:

– повторяемость;
– быстрое выполнение;
– меньшие затраты на поддержку;
– отчеты;
– выполнение без вмешательства человека.

В качестве недостатков автоматизации тестирования можно выделить:

– затраты на поддержку;
– большие затраты на разработку;
– стоимость инструментов для автоматизации тестирования;
– пропуск мелких ошибок.

Однако несмотря на недостатки, автоматизация тестирования довольно выгодна в долгосрочных проектах за счет отсутствия повторяющихся затрат на использование человеческих ресурсов и скорости выполнения.

В качестве основных инструментов и подходов, использующихся в автоматизации тестирования, можно выделить:

– TestNG - скелет фреймворка для модульного тестирования, написанный на Java

– Selenium – наиболее популярный инструмент, применяющийся в автоматизации тестирования веб-приложений. Selenium - это комплект из нескольких инструментов, каждый из которых предполагает свой собственный подход к автоматизации тестирования. Большинство инженеров-тестировщиков, которые работают с Selenium, фокусируются на одном-двух инструментах из этого набора, которые лучше всего отвечают их нуждам.

– Шаблон проектирования Page Object - объектно-ориентированный класс, который выступает интерфейсом веб-страницы тестируемого приложения. Методы данного класса используются в тесте при взаимодействии с элементами пользовательского интерфейса приложения

– Yandex Html Elements Framework - фреймворк, который расширяет функциональность шаблона Page Object и позволяет работать с типизированными и составными элементами на веб-страницах просто, гибко и удобно.

Во второй главе описана разработка фреймворка с помощью вышеперечисленных инструментов и подходов.

Фреймворк базируется на трех модулях: модуль конфигурации и запуска, браузерный модуль и тестовый (клиентский) модуль. Под клиентом в данном случае подразумевается специалист по автоматизации тестирования конкретного продукта. Краткое представление архитектуры, приведенное на рисунке 1, демонстрирует взаимосвязь между модулями.

Рисунок 1 - Диаграмма архитектуры фреймворка

Также для удобства тестирования параметров веб-элементов была разработана Check API, позволяющая в легкочитаемом и легконастраиваемом формате проверить многие из атрибутов отдельного элемента.

Также аждый элемент Page Object может быть помечен аннотацией @Styles, которая позволяет совершить процерки CSS-атрибутов элементов при

инициализации объекта класса, реализующего шаблон проектирования Page Object.

В третьей главе было проведено юзабилити-тестирование разработанного фреймворка. В процессе тестирования отслеживались следующие метрики:

- время выполнения задания;
- ошибки, возникающие при выполнении поставленной задачи;
- частота использования автоматических подсказок, связанных с функциональностью фреймворка, генерируемых интегрированной средой разработки (IDE)
- глубина использования особенностей фреймворка;
- желание автоматизатора использовать особенности отдельных библиотек чаще, чем аналогичные особенности разработанного фреймворка;
- обратная связь автоматизатора от использования фреймворка.

По итогам первичного юзабилити-тестирования были выявлены следующие недостатки:

- недостаточная очевидность части функциональности;
- практически не используемая аннотация @Styles.

В следствие проведенного первичного юзабилити-тестирования фреймворк был доработан, что было доказано улучшившимися результатами в повторном юзабилити-тестировании.

ЗАКЛЮЧЕНИЕ

В ходе магистерской диссертации были изучены процессы тестирования программного обеспечения как мануального, так и автоматизированного. Были изучены методы определения целесообразности введения автоматизации на проекте.

Несмотря на очевидные плюсы были приведены и недостатки перехода на автоматизированное тестирование, такие как:

– необходимость создания фреймворка, что по сложности приравнивается к созданию полноценного ПО;

– большие временные затраты на создание и поддержку автоматизированных тестов;

– пропуск небольших ошибок, на которые тест не запрограммирован.

Для обеспечения автоматизированного тестирования веб-приложений создан специальный фреймворк. В рамках фреймворка был разработан модуль конфигурации и запуска тестов, который предоставляет интерфейс для регистрации конфигурационных параметров (реализуется на стороне клиента). После определения финальных настроек, данный модуль осуществляет запуск указанных клиентом (специалистом по автоматизации тестирования конкретного приложения) тестов.

Также был разработан браузерный модуль фреймворка, который предоставляет API для работы с браузерами различных типов: Google Chrome, Mozilla Firefox и Microsoft Edge. Позволяет осуществлять их конфигурацию под любые нужды и повышает абстрактность кода, оставляя всю низкоуровневую логику взаимодействия с сущностями браузера внутри модуля.

Клиентом же должен быть разработан последний модуль – тестовый, который является уникальным для каждого отдельного проекта.

Основными особенностями и отличительными чертами первой версии фреймворка стали гибкая конфигурация, удобные проверки на странице с помощью аннотации @Styles и гибкие проверки отдельных элементов с помощью Check API.

Был проработан план юзабилити-тестирования фреймворка. Были выделены три группы специалистов, которые должны были выполнить тестовое задание – разработку автоматического теста по заданному тест-кейсу. Задание было разбито на 3 этапа, что позволило проанализировать результаты тестирования более емко.

В итоге, проведенное юзабилити-тестирование фреймворка на трех типах специалистов автоматизации тестирование показало, что отдельные части фреймворка были неудобны и неочевидны в использовании. Так было принято решение отказаться от аннотации @Styles, и объединить ее функциональность с Check API.

Также по результатам тестирования были произведены дополнительные изменения в отдельных частях фреймворка, которые позволили стать ему более простым в освоении, а коду более очевидным в чтении и поддержке.

Преимуществами использования реализованного фреймворка для автоматического тестирования веб-приложения являются:

- высокая абстрактность тестового кода;
- вся низкоуровневая логика выделена в отдельные классы фреймворка;
- обширный API для работы с браузером;
- гибкий интерфейс для работы с настройками;
- фреймворк является расширяемым.

Однако следует отметить имеющиеся ограничения: текущая реализация ориентирована на тестирование веб-приложений, однако в то же время расширяема и для других видов тестирования (веб-сервисов, базы данных и прочих).

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

Список публикаций соискателя

[1-А] Савосик, А.А. Эргономическое обеспечение фреймворка автоматического тестирования программных приложений / А.А. Савосик // Тезисы доклада на 54-й научной конференции аспирантов, магистрантов и студентов. – Минск: БГУИР, 2018. – 1 с.

[2-А] Савосик, А.А. Эргономическое обеспечение фреймворка автоматического тестирования программных приложений: результаты юзабилити-тестирования / А.А. Савосик // Тезисы доклада на 55-й юбилейной научной конференции аспирантов, магистрантов и студентов. – Минск: БГУИР, 2019. – 1 с.