Министерство образования Республики Беларусь

Учреждение образования

Белорусский государственный университет

информатики и радиоэлектроники

УДК 004.41

Фурсов

Филипп Олегович

Автоматизированное тестирование веб-приложение на основе средств контейнерной виртуализации и непрерывной интеграции

АВТОРЕФЕРАТ

на соискание степени магистра технических наук

по специальности 1-45 80 02 Телекоммуникационные системы и компьютерные сети

_____

Научный руководитель

Бобов Михаил Никитич

Доктор технических наук,

профессор

_____

Минск 2018

КРАТКОЕ ВВЕДЕНИЕ

В настоящее время каждый день появляются новые web-приложения, которые построены на различных платформах и написаны на разнообразных языках программирования. Вместе с этим растут требования, предъявляемые к приложениям и всё большую роль, играет обеспечение качества для каждой из систем.

Тестирование является составляющей частью процесса отладки ПО, после выявления ошибок дефекты в программном коде должны быть устранены разработчиками.

Задачами современного тестирования является не только обнаружение ошибок в программах, но и выявление причин их возникновения. Такой подход позволяет разработчикам функционировать максимально эффективно, быстро устраняя возникающие ошибки.

Понимание важности процесса тестирования приводит к возникновению тенденций, направленных на применение промышленных способов проверки качества программного обеспечения. Наиболее важным направлением здесь является внедрение различных систем автоматизированного тестирования. Основная роль в осуществлении качественного процесса тестирования принадлежит способам организации взаимодействия всех участников разработки и выбору правильной методологии.Автоматизированное тестирование способно поддержать множество видов тестов,поскольку функциональность и возможности средств автоматизированного тестирования расширились за последние годы.

Использование инструментов непрерывной интеграции и контейнерной виртуализации увеличивает возможное количество выполняемых одновременных тестов и делает процесс переноса тестовой инфраструктуры в любое окружение максимально быстрым. Данная работа посвящена разработке фреймворка автоматизированного тестирования, а также методологии развёртывания тестовой среды с использованием непрерывной интеграции и контейнерной виртуализации.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Цель работы

Целью работы является разработка на языке Javaфреймворка автоматизированного тестирования веб-приложений и применения инструментальныхсредств непрерывной интеграции и виртуализации в автоматизированном тестировании.

Задачи работы

Для достижения поставленной цели фреймворк должен отвечать следующим требованиям:

– кроссплатформенность (отсутствие необходимости поддерживать тесты для разных платформ отдельно);

– простота модификации;

– возможность запуска тестов без среды разработки;

– минимальная зависимость от окружения.

В разработку фреймворка входит:

– разработка структурной схемы фреймворка;

– разработка диаграммы классов;

– реализация на языке программирования Java.

Вместе с разработкой фреймворка, осуществляется реализация практики непрерывной интеграции на базе выбранных инструментальных средств, являющаяся ключевым звеном в развёртывании системы автоматизированного тестирования в облачных сервисах, поддерживающих технологию контейнерной виртуализации.

Личный вклад соискателя

Содержание диссертации отражает личный вклад автора. Он заключается в разработке фреймворка автоматизированного тестирования веб-приложений, реализации техник непрерывной интеграции в автоматизированном тестировании и разработке методики использования контейнерной виртуализации в автоматизированном тестировании. Определение целей и задача исследований, интерпретация и обобщение полученных результатов проводились совместно с научным руководителем, доктором технических наук, профессором М.Н. Бобовым

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

В начале работы осуществляется анализ текущего состояния тестирования, приводятся теоретические основы тестирования, проводится анализ значимости автоматизированного тестирования как части процесса разработки ПО, раскрывается важность и роль фреймворка в автоматизированном тестировании. Раскрывается понятийный аппарат непрерывных практик и контейнерной виртуализации. Среди актуальных тенденций автоматизированного тестирования важно отметить следующие:

- Постоянный рост потребности в проведении автотестирования;

– Снижение порога вхождения в автотестирование за счёт создания гибких фреймворков
– Коэволюция процессов создания ПО, его тестирования и автоматизированного тестирования в частности.

В условиях постоянных изменений и добавлений требований и программного обеспечения автоматизированное тестирование служит важным механизмом проверки, обеспечивающим точность и стабильность программного продукта при выходе каждой новой версии.Внедрение автоматизированного тестирования изменяет модель индустрии программного обеспечения. Это изменение не только предполагает применение инструментальных средств и выполнение автоматизированного тестирования – оно пронизывает весь жизненный цикл тестирования и системной разработки.

К преимуществам автоматизированного тестирования можно отнести:

– повторяемость – все написанные тесты всегда будут выполняться однообразно, то есть исключен «человеческий фактор»;

– быстрое выполнение – автоматизированному скрипту не нужно сверяться с инструкциями и документациями, это сильно экономит время выполнения;

– меньшие затраты на поддержку – когда автоматические скрипты уже написаны, на их поддержку и анализ результатов требуется, как правило, меньшее время чем на проведение того же объема тестирования вручную;

– отчеты – автоматически рассылаемые и сохраняемые отчеты о результатах тестирования;

– выполнение без вмешательства – во время выполнения тестов инженер-тестировщик может заниматься другими полезными делами, или тесты могут выполняться в нерабочее время.

В главе 2 рассматриваются вопросы структурного проектирования фреймворка автоматизированного тестирования. В разделе 2.1 «Структура фреймворка автоматизированного тестирования» определяется структура программного средства, его задачи и требования. Определена схема взаимодействия разрабатываемого фреймворка с другими системами, включая тестируемую.

В разделах 2.2 – 2.7 описываются подходы к разработке фреймоворка, а также средства и инструменты, которые используются.В разделе 2.2 описан PageObject – шаблон проектирования, роль которого сводится к упрощению поддержки написанных тестов и уменьшению количества дублируемого кода. В разделе 2.3 раскрываются особенности SeleniumWebDriver – семейство драйверов для браузеров, а также набор клиентских библиотек на разных языках, которые позволяют работать с этими драйверами. Раздел 2.4

описывает фреймворкTestNG, который является своего рода драйвером тестов, управляя последовательностью выполнения тестов и предоставляя интерфейс для проверок. Раздел 2.5 включает в себя сведения о библиотеке SeleniumWebDriverManager, которая решает важную проблему автотестирования – избавляет от необходимости писать код под каждую операционную систему.

Глава 3 полностью посвящена функциональному проектированию фреймворка автоматизированного тестирования. В разделах 3.1 – 3.6 дано подробное описание всех классовосновного модуля фреймворка, которые отвечают за взаимодействие с тестируемым приложением. Эти разделы включают в себя детальное описание каждого метода и являются наглядным примером реализации шаблона PageObject.

Разделы 3.7 – 3.9 описывают сервисный модуль, в задачи которого входит подготовка тестовых данных, настройка окружения и другие задачи, напрямую несвязанные с процессом тестирования. Так, например, раздел 3.9 описывает класс получения тестовых данных, в задачи которого входит сбор, обработка и подготовка тестовых данных для дальнейшего использования в тестах.

Раздел 3.10 описывает конфигурационный файл pom.xml, который хранит в себе все внешние зависимости фреймворка. На основе данных из этого файла осуществляется взаимодействие с центральным репозиториемMaven, а также происходит сборка и выполнение java-проекта.

Глава 4 «Реализация непрерывной интеграции и контейнерной виртуализации» раскрывает сущность непрерывной интеграции и виртуализации в отношении к автоматизированному тестированию. В разделах 4.1 – 4.2 проанализирован доступный на сегодняшний день инструментарии средств непрерывной интеграции. Для анализа были взяты CircleCI, TravisCI, Jenkins, TeamCity. В разделе 4.2 дано подробное описание каждого инструмента, его преимущества и недостатки по сравнению с другими. На основании собранных сведений было принято решение использовать Jenkinsв качестве инструмента для реализации практики непрерывной интеграции.

Раздел 4.3 описывает подготовку фреймворка для использования с Jenkins, а также пошаговое описание способа реализации непрерывной интеграции. Рассмотрены основные функциональные возможности Jenkins. Всё это позволило реализовать процесс тестирования, который способен выполняться на удалённом сервере, отвечающий современным требованиям к скорости выполнения тестов и моментально реагирующий на любые изменения в исходном коде проекта.

Разделы 4.4 – 4.5 рассматривают вопросы контейнерной виртуализации. Приведены сведения о необходимой инфраструктуре. Показано, как платформа Docker решает три основные проблемы развертывания сервисов:

- доставка кода на сервер;
- запуск кода;
- единообразие окружения.

В разделе 4.4 дано исчерпывающее обоснование использования Docker, его преимуществ перед другими способами виртуализации. Раздел 4.5 пошагово описывает использование Dockerв облачной платформе AmazonWebServices. Детально описаны способ взаимодействия Jenkinsи Docker, процесс создания репозитория образов Docker, запуск контейнеров в AmazonElasticContainerService. Даны практические советы по масштабируемости проводимого тестирования, что необходимо при параллелизации тестирования. Рассмотрены основные вопросы менеджмента уровней доступа к ресурсам AWS.

Раздел 4.6 «Запуск тестов с помощью Jenkins в Docker-контейнерах» описывает настройку сервера непрерывной интеграции с AmazonECS. В конечном итоге, это позволяет запускать тесты в отдельных Docker-контейнерах, что обеспечивает изолированность тестов, гибкое использование ресурсов, простоту масштабируемости.

ЗАКЛЮЧЕНИЕ

Результаты выполненной диссертационной работы:

– Выполнен анализ состояния автоматизированного тестирования, обоснована необходимость создания фреймворка, его роли в автотестировании;

– Разработан фреймворкавтоматизрованного тестирования веб-приложений, который отвечает всем предъявленным к нему требованиям и показана его эффективность;

– Проанализированы существующие инструментальные средства непрерывной интеграции

– Осуществлена реализация непрерывной интеграции в рамках автотестирования;

– Реализован процесс тестирования веб-приложений в изолированных Docker-контейнерах под управлением Jenkins

ОПУБЛИКОВАННЫЕ РАБОТЫ

1-А Фурсов, Ф.О. Автоматизированное тестирование web-приложений/Ф.О. Фурсов, И.М. Шигало // Телекоммуникационные системы и сети: материалы 53-й научной конференции аспирантов, магистрантов и студентов – Минск, 2017 – С.37.

2-А Фурсов, Ф.О. Автоматизированное тестирование web-приложений/ Ф.О. Фурсов // Телекоммуникационные системы и сети: материалы 54-й научной конференции аспирантов, магистрантов и студентов – Минск, 2018 – Принято к публикации