УДК 004.415.53

РОЛЬ, НАЗНАЧЕНИЕ И ПРОБЛЕМЫ АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ

Смольский С.С.

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники « филиал «Минский Радиотехнический колледж», г. Минск, Республика Беларусь

Научный руководитель: Сальникова Е.А. – преподаватель первой категории дисциплин общепрофессионального и специального циклов

Аннотация. Исследовано определение автоматизированного тестирования, технологии и методологии, относящиеся к автоматизированному тестированию. Разобраны его виды и особенности. Установлено, какое место в разработке программного обеспечения занимает автоматизация; что необходимо для её оптимального проведения и внедрения в проект.

Ключевые слова: автоматизированное тестирование, тестировщик, автотесты.

Введение. Автоматизированное тестирование – это набор техник, подходов и инструментальных средств для контроля выполнения тестов и сравнения ожидаемого фактического результата работы программы. Этот вид тестирования позволяет автоматизировать часто повторяющиеся, но необходимые для максимизации тестового покрытия задачи [1].

Для составления автоматизированных тестов, Quality Assurance (QA)-специалист должен уметь программировать. Автоматические тесты (автотесты) – это полноценные программы, просто предназначенные для тестирования. Автоматизация тестирования – это понятие довольно обширное. Тестируя приложения мануально, используются различные инструменты. Например, для реверс инжиниринга базы данных используют Sql Change Scanner и MS SQL Server Profiler. Это инструменты мониторинга, которые используются в мануальном тестировании, получается ручное тестирование проведено при помощи специального ПО, что полу-автоматизирует его [2].

Выделяют три основных вида автоматизированного тестирования: автоматизация тестирования кода (Code-driven testing), автоматизация тестирования графического пользовательского интерфейса (Graphical user interface testing / UI testing), автоматизация тестирования API (Application Programming Interface). Ввиду разной интерпретации термина «автоматизированное тестирование» виды могут смешиваться и иметь различные названия, но перечисленные являются общепринятыми.

Основная часть. Code-driven testing или автоматические юнит-тесты – тестирование на уровне программных модулей, классов и библиотек. UI testing – специальная программа, которая позволяет генерировать пользовательские события – клики мышкой или нажатие клавиш, переходы по ссылкам и отслеживать соответствует ли реакция программы на эти действия спецификации. При тестировании API проверяются интерфейсы, предназначенные для взаимодействия с другими программами или с пользователем. В данном случае также используются специальные фреймворки.

Основные задачи автоматизации подразделяются на следующие типы [3]:

- заменить трудоемкое ручное тестирование рабочих процессов и множества сценариев;
- обеспечить прозрачный контроль над процессом тестирования;
- сократить затраты при повторно используемых скриптах;
- получить точные и надежные результаты испытаний;
- сократить время выхода готового продукта на рынок;
- увеличить объём/охват тестируемых областей скриптами.

От поставленных задач зависит выбор фреймворков для автоматизированного тестирования. При необходимости проведения отказоустойчивых сквозных тестов, без использова-

ния кода или его минимума, отдают предпочтение приложению Testim. Он позволяет создавать стабильные тесты без кода, которые используют искусственный интеллект для экспорта тестов в виде кода.

Для проведения регрессионного тестирования используют Selenium. Это достаточно громоздкий, но универсальный инструмент, работающий на языках Java, Ruby, RSpec, Python, C# с использованием таких фреймворков, как JUnit и TestNG.

При создании тестов для приложений iOS и Android используют программу 21 Labs, её преимуществами является быстрая и интеллектуальная разработка – создание с помощью ИИ дает пользователям возможность создавать автоматизированные функциональные тесты и тесты пользовательского интерфейса за несколько минут. Полностью SaaS, не требует установки или устройств для создания или выполнения тестов. Предлагает беспрепятственный доступ к десяткам устройств [4].

Одной из главных проблем разработки является разделение задач между мануальным и автоматизированным тестированием. Если проект большой, в его составе несколько подсистем и «ручные» тест-кейсы уже превышают несколько сотен, автоматизация позволит повысить продуктивность тестировщика, который не будет затрачивать недели на проверку тест-кейсов. Большая команда программистов также нуждается в автоматизированном тестировании, чтобы ускорить выявление багов при взаимодействии разных модулей кода и оперативно их исправить.

Автоматизация процессов тестирования будет особенно актуальной, если в продукте присутствуют регулярно повторяющиеся сценарии, они трудоёмки и не подходят для ручной проверки, а их анализ занимает много времени.

Для определения необходимости использовать автоматизацию выделяют следующие основные факторы: ROI (окупаемость инвестиций), сложность теста, его возможность к автоматизации, насколько приложение и его отдельные компоненты стабильны и какая площадь может быть покрыта автотестами.

Несмотря на то что для написания тест-кейсов требуются ресурсы, окупаемость для больших и долгосрочных проектов может быть огромной. Это объясняется тем, что тесты настраиваются под определённые сценарии, легко запускаются с использованием параметров и могут работать неограниченное время. Внедряя автоматизацию, можно значительно снизить стоимость каждого часа, затрачиваемого на проверки, а также найти наиболее трудно обнаруживаемые баги.

Используя автоматизированное тестирование, разработчики могут без привлечения тестировщиков узнать корректно ли работает реализованный ими функционал, если тесты запускаются сразу же после билда. Следует помнить, что при использовании API тестирования скорость прохождения тестов значительно повышается по сравнению с использованием UI. Автотесты представляют более надежным способом по сравнению со стандартными тестами, ни один из которых нельзя пропустить, поскольку это чревато ошибками.

Одним из преимуществ автоматизации является скорость выполнения проверок, что особенно актуально для DDT, тестов, управляемых данными. Автоматизация позволяет проходить этапы проведения тестов быстрее, чем это делает человек. Это позволяет проводить одни и те же проверки много раз, но с разными наборами данных.

Главная задача автоматизации – найти баги в простых операциях, например, вход в приложение, создание аккаунта или отправка электронного письма, когда пользователь забывает пароль. Этот процесс не устранит специфические проблемы, с которыми могут столкнуться пользователи их все еще должны тестироваться вручную. Еще один недостаток – ограниченное число тестируемых сценариев. Помимо прочего, автоматизация не позволяет провести эффективное эргономичное тестирование дизайна, например, положение кнопки, и, в целом, насколько удобно приложение в использовании.

Автотесты могут вызывать ложную уверенность в качестве, они проверяют только то, на что они были запрограммированы. Тест может пройти успешно, а дефект остаться незамеченным, и всё из-за того, что тест не был запрограммирован на нахождение этой ошибки.

Необходимо соблюдать осторожность при разработке, так как автотесты могут перестать работать из-за многих факторов. Таким фактором может стать незначительное изменение UI-интерфейса, падение сервиса, проблемы с сетью, загруженность тестовой машины. По возможности нужно стараться минимизировать количество таких факторов.

Порой, после первой сборки с новым функционалом, написание автотестов занимает больше времени, чем ручная проверка программы. Тут нужно исходить из ситуации и приоритетов. При возможности рекомендуется начать писать тесты одновременно с написанием разработчиками нового функционала. Можно сделать некий шаблон, а затем туда дописать необходимые проверки или локаторы после сборки.

Заключение. Учитывая изученные особенности автоматизированного тестирования, были сформулированы оптимальные условия на проекте для внедрения автоматизации.

Первое и основное – это хорошая коммуникация между разработчиками и тестировщиками: разработчик должен учитывать пожелания тестировщиков по поводу технических деталей, таких как id элементов и более простая структура программы; должна быть реализована простая и быстрая связь между отделами тестирования и разработки приложения. Желательно, использование одной команды, как в методологии Scrum, чтобы все части команды были осведомлены о технических деталях и планах разработки.

Разработчикам должна быть выгодна автоматизация тестирования. На некоторых проектах чем больше ошибок исправил разработчик – тем лучше. В других ситуациях, разработчики стремятся уменьшить количество багов после сдачи в тестирование. Тут они не только сами тестируют собственный код, но и могут помочь в разработке приёмочных тестов.

Разработчики должны писать, править и понимать результаты прохождения приёмочных автотестов и их лог-файлы. В таком режиме, сами разработчики смогут писать тесты перед реализацией функционала. С учетом этого, они будут заинтересованы в создании более простой верстки страницы и более тестируемого функционала.

Список литературы

1. *Тестирование программного обеспечения. Базовый курс / Святослав Куликов. – Minsk : EPAM Systems, 2015–2022.*

2. *Тестирование Дот Ком, или Пособие по жестокому обращению с багами в интернет-стартапах / Савин Р. // Москва : Издательство «Дело» – 2007.*

3. *Qalight Центр подготовки IT специалистов [Электронный ресурс] / Qalight. – 2012-2022. – Режим доступа: https://qalight.ua/ru/ – Дата доступа : 01.04.2022..*

4. *Logrocon software engineering [Электронный ресурс] / Logrocon. – 2012-2022. – Режим доступа: https://logrocon.ru. – Дата доступа : 01.04.2022..*

UDC 004.415.53

THE ROLE, PURPOSE AND PROBLEMS OF AUTOMATION TESTING

Smolsky S.S.

Educational Institution "Belarusian State University of Informatics and Radioelectronics" branch "Minsk Radio Engineering College"
Minsk, Republic of Belarus

Salnikova E.A. – teacher of the first category of disciplines of general professional and special cycles

Annotation. The definition of automated testing, technologies and methodologies related to automated testing are investigated. Its types and features are analyzed. The place of automation in software development has been established; what is necessary for its optimal implementation and introduction into the project.

Keywords. automated testing, tester, autotests