DOI 10.37882/2223-2966.2024.9.16

ВНЕДРЕНИЕ ТЕХНОЛОГИИ АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ НА ПРИМЕРЕ ОТЕЧЕСТВЕННОЙ ИТ-КОМПАНИИ

INTRODUCTION OF TECHNOLOGY OF AUTOMATED TESTING OF INFORMATION SYSTEMS ON THE EXAMPLE OF A DOMESTIC IT COMPANY

D. Kiyan
I. Minkin

Summary. The article considers the importance of the implementation of automated testing of information systems to improve the quality of the products being developed and the optimization of their verification processes. The author stresses that manual testing, despite its effectiveness, is not able to guarantee the detection of all errors in the updated versions of the software due to the human factor, As a result, undetected errors may result in inaccessibility of information systems and the need for operational improvements. The author describes the stages of development and implementation of automated testing, including creation of applications for writing test scripts, browser management, organization of monitoring and parallel running of tests. The main idea of the article is that automated testing of information systems is a key tool for increasing the effectiveness of software verification, reducing the time of system acceptance into operation and eliminating the human factor, which in turn helps to improve product quality, reduce costs and accelerate development processes.

Keywords: automated testing; reception of information systems; software testing; quality of products developed; software verification tool; optimization of the verification process.

Киян Дарья Александровна
Московский государственный технический университет имени Н.Э. Баумана
daria.kiyan@yandex.ru
Минкин Илья Михайлович
Кандидат технических наук, Московский автомобильно-дорожный государственный технический университет
minkin.i@elpts.ru

Аннотация. В статье рассматривается важность реализации автоматизированного тестирования информационных систем для повышения качества разрабатываемых продуктов и оптимизации процессов их проверки. Автор подчеркивает, что ручное тестирование, несмотря на свою эффективность, не способно гарантировать выявление всех ошибок в обновленных версиях программного обеспечения из-за влияния человеческого фактора, в результате чего, не выявленные ошибки могут привести к недоступности информационных систем и необходимости оперативных доработок. Автор описывает этапы разработки и реализации автоматизированного тестирования, включая создание приложений для записи сценариев тестирования, управления браузером, организацию мониторинга и параллельного запуска тестов. Главная идея статьи заключается в том, что автоматизированное тестирование информационных систем является ключевым инструментом для повышения эффективности проверки программного обеспечения, сокращения времени приемки систем в эксплуатацию и исключения человеческого фактора, что в свою очередь способствует увеличению качества продуктов, снижению затрат и ускорению процессов разработки.

Ключевые слова: автоматизированное тестирование; прием информационных систем; проведение испытаний ПО; качество разрабатываемых продуктов; инструмент проверки ПО; оптимизация процесса проверки.

Введение

Стремительное развитие технологий и внедрения процессных инноваций способствует трансформации локальных (отдельных) информационных систем в обширные цифровые платформы и экосистемы, и как следствие, требует изменения подходов к разработке и реализации программных продуктов, приёмка в эксплуатацию которых требует значительных человеческих ресурсов. Данные процессы стимулируют разработку инновационных решений, направленных на значительное улучшение качества внедряемых продуктов и оптимизацию обслуживающих процессов.

В данном контексте радикально встает вопрос эффективной проверки работоспособности информационных систем. Для обеспечения стабильности и качества такой работы, минимизации требуемых ресурсов, высокой скорости, и как следствие, высокого экономического эффекта при реализации этого процесса необходимы прогрессивные методы и технологии проведения испытаний информационных систем в общем, и проверки программного обеспечения (далее — ПО) в частности.

На рынке информационных технологий предлагаются различные инструменты для проведения испытаний информационных систем, охватывающие и ручное (Zephyr, TestRail, Test IT) и автоматизированное (Appium, Jenkins, Cucumber) тестирования, предоставляя широкий спектр возможностей для обеспечения качества разрабатываемых продуктов.

Ручное тестирование в основном использует экстенсивные методы для проверки ПО, и в случае с тестиро-

ванием обновленной (доработанной) версии ПО, такой способ не гарантирует выявления ошибок, приобретенных в отношении той части ПО, которая прежде работала корректно. Данная проблема связана с влиянием человеческого фактора на качество проверок, особенно при реализации большого количества разноплановых сценариев тестирования ПО.

Возникновение критических ошибок, не выявленных на этапе тестирования обновленного ПО, приводит к невозможности эксплуатации доработанной информационной системы и необходимости принятия оперативных решений, связанных с возвратом к предыдущей версии ПО и срочными доработками, а также к недоступности информационной системы в определенный период времени. Таким образом, реальное время на приемку информационной системы в эксплуатацию кратно растет, итерации тестирования сливаются в единый нескончаемый процесс, не всегда приносящий требуемый результат.

Исключение человеческого фактора из процесса тестирования ПО является эффективной мерой устранения описанной проблемы и достигается за счет автоматизации процесса испытаний информационных систем.

Методы

В соответствии с этапами жизненного цикла информационной системы, внесение изменений в ПО является неотъемлемым процессом, требующим проведения функционального и регрессионного тестирования ПО для подтверждения его работоспособности и приемки информационной системы в эксплуатацию.

Эффективность тестирования ПО зависит от:
- качества и полноты описанных сценариев тестирования в соответствии с программой и методикой испытаний;
- качества и полноты реализации сценариев тестирования;
- скорости проведения тестирования.

В данной статье внедрение технологии автоматизированного тестирования рассматривается на примере более чем десятилетнего опыта функционального и регрессионного тестирования информационных систем компанией, занимающейся как разработкой собственного ПО, так и эксплуатацией информационных систем, основанных на партнерском ПО.

Ручное функциональное и регрессионное тестирование версий ПО долгое время являлось единственным методом приемки информационных систем в эксплуатацию. При этом качество и скорость тестирования в полной мере зависели от человеческого фактора.

Описанное положение вещей привело руководство компании к решению о необходимости оптимизации процесса приемки обновленных информационных систем в эксплуатацию путем внедрения технологии автоматизированного тестирования ПО.

При описании подхода к решению проблемы важно ввести уточнение формулировки «автоматизация тестирования». Под автоматизацией тестирования, в данном обзоре понимается автоматизация именно регрессионного тестирования, поскольку, во-первых, оно занимает основную долю трудозатрат при приемке информационных систем в эксплуатацию, и во-вторых, именно регрессионное тестирование наиболее склонно к автоматизации в связи с многократным повторением сценариев в соответствии с программой и методикой испытаний. Автоматизация созданных сценариев тестирования ПО в контексте рассматриваемого вопроса исключения человеческого фактора при проверке работоспособности информационных систем фактически является трансформацией созданных сценариев ручного функционального тестирования в регрессионное, но уже автоматизированное.

Перед началом работы было проведено исследование рынка готовых инструментов для автоматизированного тестирования. Результаты исследования показали, что доступные на рынке решения не подходят для удовлетворения потребностей компании, существующие инструменты не обладают достаточным функционалом, а также не соответствуют специфическим требованиям компании по проведению тестирования ПО. Таким образом, потребовалось разработать собственное решение или модифицировать имеющиеся инструменты для успешной реализации поставленной задачи.

В качестве первого шага было разработано приложение по созданию сценариев автоматизированного тестирования, позволяющее тестировщикам записывать в программируемые сценарии действия (шаги), необходимые для выполнения в соответствии с программой и методикой испытаний. При разработке приложения использовался фреймворк Apache NetBeans [1] — кроссплатформенная интегрированная среда разработки программного обеспечения на различных языках программирования, включая Java, Python, PHP, JavaScript, C, C++.

Одна из основных проблем разработки приложения заключалась в том, что основные пользователи программы — функциональные тестировщики, незнакомые с языками программирования, формулируют главное требование к приложению — возможность создавать сценарии тестирования не путем написания программного кода, а посредством работы в браузере и выбора конкретных действий внутри приложения.

Чтобы реализовать задуманное потребовалось запрограммировать стандартный набор действий, совершаемых при использовании приложения пользователями в браузере («наведение», «щелчок», «двойной щелчок», «послать текст» и «сохранить»), а также множество способов использования данных действий, например, «Текст», «Календарь», «Селектор», «Таблица», «Радиокнопка», «Связи», «Доступность», «Чек-бокс», «Сохранить переменную», «Сгенерировать переменную». Помимо этого, реализованы функции, позволяющие выполнять стандартный набор команд одним шагом сценария тестирования, например, «выбрать сертификат подписи» или «подписать пользовательское соглашение».

Помимо функций, отвечающих за выполнение действий в браузере, в приложении предусмотрено несколько способов работы с направлением необходимого потока данных в информационных системах, таких как загрузка данных непосредственно в шаг сценария тестирования перед его запуском и выбор файлов, которые потребуются приложению для выполнения заложенного функционала загрузки.

Регрессионное тестирование по своей сути стремится к исчерпывающему тестированию и, как следствие, подразумевает осуществление проверки всех возможных способов выполнения одной и той же функции, что объясняет необходимость многократного выполнения сценариев тестирования с небольшими отличиями в одном-двух шагах. Удовлетворение данной потребности реализовано в приложении, как посредством копирования сценариев тестирования целиком, так и дублирования конкретных шагов сценариев тестирования.

Для автоматизации управления работы веб-браузера использован инструмент, позволяющий управлять элементами его страниц. В коде приложения применен модуль управления браузером «Selenium web-driver» [2], который может работать локально или на удаленном компьютере, модули которого включены уже во все ведущие браузеры: Google, Opera, Mozilla Firefox. В результате запуска теста, Selenium web-driver открывает браузер (например, Chrome, Firefox) и взаимодействует с ним, следуя указанным шагам в сценарии тестирования. Таким образом, приложение, используя драйвер, эмулирует действия пользователя в браузере без необходимости фактического вмешательства человека.

В целях обеспечения минимизации потенциальных сбоев в сетевом подключении или задержек во времени при выполнении той или иной функции, разработчики приложения предусмотрели функцию автоматической задержки шагов, которую можно настроить одновременно для всего сценария тестирования или для конкретных выбранных шагов сценария.

Так, если созданный сценарий тестирования подразумевает проверку работоспособности и доступности полей пользовательского реестра (реестр, отображающийся в интерфейсе информационной системы), содержащего множество типов данных, источники которых расположены в разных базах данных системы, то целесообразно применять настройку автоматической задержки для всего сценария, так как отображение информации в интерфейсе реестра выполняется медленнее установленного в технической документации показателя скорости реагирования системы на отображение (прорисовку) интерфейсов. Для сценария, описывающего инициацию создания документа, заполнение его полей, сохранение введенных данных и поиск созданного документа в реестре, предпочтительнее использовать задержку только на двух шагах сценария: ожидание подтверждения сохранения и ожидание отображения результатов поиска документа в реестре.

На данном этапе первая фаза работ завершилась и осуществлен переход от разработки приложения по созданию автоматизированных сценариев тестирования и их запуску к созданию массива сценариев тестирования в соответствии с программой и методикой испытаний и к отладке приложения.

После завершения создания полного массива сценариев регрессионного тестирования приложение было запущено в эксплуатацию

Использование приложения в данной конфигурации продолжалось достаточно длительное время, информационная система регулярно дорабатывалась, в связи с чем создавались новые сценарии регрессионного тестирования. Влияние тестировщиков на качество выполнения сценариев тестирования свелось к минимуму. Однако наличие ручного запуска сценариев, не позволяло достичь необходимой скорости проверки ПО. Требовалось новое решение, которое сократило бы время проведения приемки доработанной информационной системы в эксплуатацию до минимума.

Второй и ключевой фазой создания системы автоматизированного тестирования стала организация мониторинга, обеспечивающего контроль прохождения тестов, выявление ошибок, сбор статистики, сохранение результатов каждого шага сценария тестирования и реализацию функции запуска терминальных клиентов. Для развертывания Мониторинга использовался фреймворк Django [3].

Для выполнения удаленного запуска из Мониторинга множества терминальных клиентов единовременно было разработано специальное приложение. Данное приложение также обеспечило последовательное выполнение сценариев, выстроенных в очередь, органи-

зованную в Мониторинге в соответствии с программой и методикой испытаний.

Необходимо отметить, что данный метод обладает особенностью: запуская тесты в параллельном режиме, каждый тест выполняется в своем собственном браузерном окне, что требует дополнительных ресурсов. Объем оперативной памяти имеет прямое влияние на скорость выполнения сценариев в параллельном режиме. Если памяти недостаточно для запуска нескольких браузеров одновременно, это приведет к замедлению выполнения тестов.

Результаты

Разработка приложения создания сценариев тестирования заняла важную роль в процессе автоматизации тестирования программного обеспечения, использование которого также позволило организовать электронное хранилище сценариев тестирования, благодаря чему стал возможен контроль соответствия созданных автоматизированных тестов и сценариев программы и методики испытаний.

Возможность управления браузером в процессе тестирования обеспечила полное автоматизированное прохождение тестов, что позволило выполнять сценарии тестирования, исключив влияние человеческого фактора, в следствии чего, занятость работников на прохождение тестирования в большинстве случаев свелась только к ручному запуску тестов.

Подключение мониторинга позволило создать логические очереди сценариев и организовать базу данных статистики проведения тестирования, которая хранит информацию о результатах проведенных тестов (время выполнения теста, результат выполнения теста, статистика выполнения теста, скриншоты результата выполнения шагов теста). Это упростило отслеживание процесса тестирования, анализ результатов и выявление потенциально проблемных мест в ПО.

Разработка приложения параллельного запуска терминальных клиентов обеспечила полный переход на автоматизированное тестирование, исключив человека из процесса проверки работоспособности информационных систем.

В результате выполнения комплекса работ скорость проведения тестирования сократилась до 90 %.

По завершению работ по разработке автоматизированного тестирования компания зафиксировала результат в ~95 % покрытия регрессионного тестирования, 5 % сценариев подлежат исключительно ручной проверке, поскольку назначение таких сценариев — защита от использования автоматических скриптов на уровне программного обеспечения, например, ввод в поле сгенерированного кода проверки — капчи (полностью автоматизированный публичный тест Тьюринга для различения компьютеров и людей).

Внедрение автоматизированного тестирования значительно сокращает время проведения испытаний информационных систем и позволяет практически исключить человеческий фактор из процесса регрессионного тестирования. Помимо этого, использование автоматизированного тестирования снижает себестоимость услуг, предоставляемых посредством тестируемой информационной системы, а также ускоряет процесс разработки.

ЛИТЕРАТУРА

1. Официальный ресурс Apache Netbeans [электронный ресурс] // URL: https://netbeans.apache.org/front/main/index.html (дата обращения 15.03.2024)
2. Официальный ресурс Selenium [электронный ресурс] // URL: https://www.selenium.dev/documentation/webdriver/ (дата обращения 15.03.2024)
3. Официальный ресурс Django [электронный ресурс] // URL: https://www.djangoproject.com/ (дата обращения 15.03.2024)

Журнал «Современная наука: актуальные проблемы теории и практики»