УДК 004.415.53

В.В. Герасимов, Н.Э. Никитин, А.Е. Щербак, Н.В. Карпенко

ТЕСТИРОВАНИЕ API И АКТУАЛЬНЫЕ СРЕДСТВА ЕГО РЕАЛИЗАЦИИ

Аннотация. В свете растущей популярности микросервисной архитектуры приложений востребованным становится тестирование API. В работе изложены особенности тестирования API, описаны преимущества, которые дает его использование при разработке программного обеспечения, в частности при разработке веб-приложений. Приведены отличия тестирования API от других видов тестирования. Дано описание некоторых популярных средств, применяемых при тестировании API, сделаны выводы касательно наиболее эффективных областей их применения.

Ключевые слова: тестирование, API, REST, DDT, Postman, JMeter, Katalon Studio, REST-Assured, RestSharp, PyRestTest.

Постановка проблемы. В наше время редко какое приложение обходится без API (от англ. Application Programming Interface — интерфейс прикладного программирования или интерфейс программирования приложений). Это справедливо как для простого сайта, так и для высоконагруженных распределенных систем. Поэтому тестирование API является одной из главных задач в процессе обеспечения качества программного обеспечения (ПО) и неудивительно, что спрос на тестировщиков, которые умеют тестировать API, повышается изо дня в день.

Интерес к тестированию API неудержимо растёт на протяжении нескольких последних лет, согласно исследованиям Google Trends [1]. Опрос, проведенный компанией Smartbear в 2017 году среди 5000 профессионалов в области разработки ПО, показал, что более 50% опрошенных респондентов используют автоматические средства тестирования API, и ожидается рост их количества на 30% (с 59% до 77%) в течении следующих двух лет, причем 80% участников опроса указали, что отвечают за тестирование API.

Анализ последних исследований и публикаций. Спектр вопросов, затрагиваемых в публикациях, посвященных вопросам тестирования ПО и

обеспечения его качества, достаточно широк. Так, в [2] приведены результаты исследования основных технологий автоматизированного тестирования ПО на предмет их особенностей, достоинств и недостатков, сделаны выводы касательно сфер их применения. В работе [3] приведены результаты подобного исследования основных технологий модульного тестирования. Наконец, в работе [4] рассмотрены разнообразные подходы к процессу создания ПО, вопросы качества, как самого продукта, так и процесса его создания, организация и процедура тестирования проекта, кода, документации и т.д.

Цель работы. Целью данной работы есть знакомство с тестированием API как с востребованным и актуальным направлением тестирования ПО, исследование его особенностей и преимуществ, которое дает это тестирование. Будут рассмотрены и проанализированы наиболее распространенные и популярные средства тестирования API, определена эффективная область использования для каждого средства.

Основная часть. Интерфейс API позволяет осуществлять связь и обмениваться данными между двумя отдельными модулями ПО. API — это набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением (библиотекой, сервисом) для использования во внешних программных продуктах. На сегодняшний день есть два основных подхода к построению программного интерфейса веб-приложений: REST (RESTful) API и SOAP API [5].

REST (от англ. Representational State Transfer — передача состояния представления) обеспечивает общение между клиентом (как правило, это браузер) и сервером с помощью обычных HTTP-запросов (GET, POST, PUT, DELETE и т. д), передавая информацию от клиента в параметрах самих запросов, информацию от сервера — в теле ответа (который может быть, например, JSON-объектом или XML-документом). REST является архитектурным стилем, а не стандартом.

SOAP (от англ. Simple Object Access Protocol — простой протокол доступа к объектам, вплоть до спецификации 1.2) характеризуется использованием HTTP(S)-протокола лишь как транспорта (чаще всего, методом POST). Все детали сообщений (в обе стороны — от клиента к серверу и обратно) передаются в стандартизованном XML-документе. SOAP может работать и с другими протоколами прикладного уровня (SMTP, FTP), но чаще всего он применяется поверх HTTP(S). SOAP является протоколом и имеет спецификацию.

REST API может применяться в сервисах социальных сетей, веб-чатов,

мобильных сервисах и т.д., тогда как SOAP API преимущественно характерен для больших корпоративных систем и используется в финансовых сервисах, сервисах телекоммуникаций и платежных системах. Подход REST API является более распространенным из-за своей относительной простоты и удобства для разработчиков.

Для чего же тестировщику стоит осваивать тестирование API и почему работодатели требуют от современных инженеров ручного и автотестирования умения работать с API? Перечислим возможные выгоды:

1. Точная локализация. Умение работать с API позволяет лучше понимать и точнее описывать возникшие ошибки. Если тестировщик достаточно технически подкован, он может гораздо точнее и информативнее описать найденный баг, тем самым облегчив и ускорив работу разработчика. Предположим, тестировщик открывает некую страницу сайта и видит в пользовательском интерфейсе сообщение об ошибке. Понять, чем конкретно она вызвана, насколько она серьезна, может ли она отразиться на других частях системы будет гораздо проще, если тестировщик предоставит разработчику информацию о том, какую именно функцию не удалось выполнить системе, статус ответа и сообщение об ошибке. Сообщения в графическом интерфейсе не всегда позволяют оценить эту зависимость.

2. Экономия времени при подготовке тестовых данных и ситуаций. Многие тесты требуют подготовки условий для их проведения (Preconditions) [4]. Так, для прохождения тестового случая (Test Case) тестировщику необходимо создавать необходимые для этого условия, используя графический интерфейс. Это может занять большое количество времени и при достаточно сложных условиях могут возникнуть проблемы уже на этапе подготовки. Более того — многочисленные действия в браузере часто являются причиной ложных «падений» автоматизированных тестов, которым на уровне GUI (Graphical User Interface — графический интерфейс пользователя) свойственна «хрупкость». Этот процесс можно оптимизировать, используя API приложения, сформировав запрос программно или воспроизведя его с помощью специальных инструментов, чем можно существенно сократить время подготовки условия проведения тестов и многократно повысить их стабильность.

3. Возможность воспроизводить тесты на больших наборах входных данных. Для некоторых проектов важно проводить тесты с большим количеством разнообразных наборов входных данных, отделенных от кода самого теста и вынесенных в отдельный файл. В этом случае один и тот же сцена-

рий может повторяться многократно для разных значений. Этот подход DDT (Data Driven Testing) [6] сложно реализовать через графический интерфейс с приемлемой скоростью и стабильностью. Напротив, на уровне http-запросов это делается очень быстро и с гораздо более высокой надежностью.

4. Возможность участвовать в проектах, где работа с API является требованием тест-дизайна. Сам тест-дизайн проекта может оказаться таковым, что некоторая часть тестов должна будет проводиться только через программные интерфейсы — например, если на проекте реализована концепция «пирамиды тестирования», при которой лишь немногие тесты используют графический интерфейс. Также стоит отметить, что все большую популярность набирает микросервисная архитектура, когда приложение является не монолитным, а фактически представляет собой набор самостоятельных приложений, обменивающихся данными. И в этом случае тестирование API становится особенно востребованным, так как именно оно позволяет в полной мере убедиться в правильности интеграции отдельных сервисов и в соблюдении логики функционирования системы в целом. Неумение работать с программными интерфейсами существенно ограничивает возможность участия тестировщика в таких проектах.

Тестирование API полностью отличается от тестирования графического интерфейса и в основном концентрируется на слое бизнес-логики архитектуры программного обеспечения. Графический пользовательский интерфейс приложения в API тестировании практически не задействуется, его может даже и не быть, что позволяет внедрять тестирование на самые ранние этапы жизненного цикла программного обеспечения. Тестирование API также очень сильно отличается и от Unit-тестирования (модульного тестирования [3, 4]) (табл. 1).

Таблица 1

Разница между API-тестированием и Unit-тестированием

Unit-тестирование	**API-тестирование**
Выполняется разработчиками	Выполняется тестировщиками
Тестируются отдельные модули	Тестируется end-to-end функционал
Разработчик имеет доступ к исходному коду	Тестировщик не имеет доступа к исходному коду
Возможно тестирование GUI	Тестируется только API
Тестируется только базовый функционал	Тестируется весь функционал
Ограничен в возможностях	Широкие возможности
Запускается до заливки в репозиторий кода	Запускается после очередной версии сборки

 ISSN 1562-9945

Рассмотрим актуальные средства и библиотеки, используемые для тестирования API.

Postman [7], будучи изначально плагином браузера Chrome, теперь расширяет свои технические решения вместе с оригинальными версиями как для Mac, так и для Windows. Основное предназначение приложения — создание категорий (коллекций) с запросами к API приложения. Любой разработчик или тестировщик, открыв категорию, сможет с лёгкостью разобраться в работе сервиса. Тестировщики могут писать тесты и производить автоматизированное тестирование прямо из Postman. Также содержит функционал для автоматического документирования по описаниям.

Из основных особенностей Postman можно выделить:

– Наглядность — все запросы всегда под рукой, мануальное тестирование во время разработки становится легче.

– Быстрый старт — вовлечение нового участника команды, будь то программист или тестировщик, проходит легко и быстро.

– Тесты — возможность писать тесты для запросов, а потом быстро составлять из них, как из пазлов, различные варианты и пути жизни приложения.

– Поддержка непрерывной интеграции — возможность интегрировать тесты в системы CI (Continuous Integration) с помощью newman (легкий консольный клиент для запуска Postman категорий).

Богатый интерфейс этого инструмента делает его простым и удобным в использовании как при автоматическом, так и при исследовательском (ручном) тестировании в приложениях для Mac, Windows, Linux и Chrome. Он не требует изучения нового языка программирования и позволяет пользователям легко делиться опытом и знаниями с другими членами команды, поскольку позволяет упаковывать все запросы и ожидаемые ответы и отправлять их коллегам.

Apache JMeter [8] — это открытое ПО, которое широко используется для функционального тестирования API, однако изначально создавалось для нагрузочного тестирования. JMeter представляет собой полнофункциональную среду тестирования, которая позволяет быстро записывать план тестирования (из браузеров или собственных приложений), создавать и отлаживать тестовые случаи. Может использоваться для тестирования производительности как на статических, так и на динамических ресурсах. Его можно использовать для имитации большой нагрузки на сервер, группу серверов или сеть, чтобы проверить их стабильность или проанализировать общую производительность при различных типах нагрузки.

Из основных особенностей JMeter можно выделить:

– Возможность загрузки и тестирования производительности различных приложений/серверов/типов протоколов: веб — HTTP, HTTPS, SOAP/REST веб-сервисы, FTP, базы данных через JDBC, LDAP, почта — SMTP(S), POP3(S) и IMAP(S), TCP, Java Objects.

– Поддерживает самые популярные форматы файлов ответов, среди которых HTML, JSON, XML и любой другой текстовый формат.

– Благодаря интеграции между JMeter и сторонними библиотеками вроде Jenkins, Maven и Gradle пользователи могут включать тесты API в конвейерные обработки систем непрерывной интеграции.

– Полная многопоточная структура позволяет одновременную выборку многими потоками и одновременную выборку различных функций отдельными группами потоков.

– Содержит плагины для анализа и визуализации данных, что обеспечивают большую расширяемость и персонализацию тестовых проектов.

– Для DDT-тестов позволяет автоматически работать с файлами CSV, что дает возможность быстро создавать уникальные значения параметров для тестирования API.

– Данный инструмент может использоваться как для статического, так и динамического тестирования производительности ресурсов.

Katalon Studio [9] — интегрированная среда для создания и выполнения тестов при работе с API, Web и мобильными приложениями. Представляет собой законченное решение, которое предоставляет автоматизаторам встроенные возможности создания нового проекта, для выполнения тестов и мониторинга результатов выполнения. Каждый рабочий процесс предоставляет множество возможностей и настроек для упрощения обслуживания и увеличения масштаба проекта. Katalon Studio имеет богатый набор инструментов для тестирования и поддерживает множество платформ, включая Windows, Mac OS и Linux. Интегрируя движки от Selenium и Appium со всеми необходимыми компонентами, встроенными ключевыми словами и шаблонами, Katalon Studio предоставляет уникальную среду разработки как для тестировщиков, так и для разработчиков, занимающихся тестированием API и веб-автоматизации. Примечательные особенности инструмента:

– Обработка API, Web и мобильных тестов на разных платформах.

– Возможность для тестировщиков и разработчиков легко разделять их ответственность в процессе разработки и совместно работать над тестами.

– Сотни встроенных ключевых слов для создания тестов.

– Поддержка AssertJ для создания гибких проверок с использованием BDD стиля [4].

– Интеграция с другими инструментами интеграционного тестирования [4].

Katalon Studio предоставляется бесплатно, хотя и не имеет открытого исходного кода.

Указанные выше средства тестирования API имеют графический интерфейс и могут использоваться в работе как мануальными (ручными), так и автотестировщиками. Постигнув принципы работы API, автотестировщик способен также использовать эти навыки как в оптимизации GUI-автотестов, так и для создания независимых API автоматизированных тестов. Остается выбрать инструменты, которые будут воспроизводить нужные запросы и отслеживать содержимое ответов. Если тестировщик умеет создавать автоматизированные тесты для графического интерфейса (например, с использованием Selenium), то хорошим вариантом развития будет интеграция тестов API в существующий тестовый фреймворк. Тесты, содержащие подготовительные/вспомогательные действия и условия, можно будет переписать с использованием API, что увеличит их производительность и стабильность. Для доступа к API посредством языка программирования потребуется библиотека, работающая с HTTP-запросами.

Для платформы Java библиотека REST-Assured [10] — лучший выбор для автоматизации API. REST-Assured — это богатая Java-библиотека, которую можно применять для тестирования REST-служб, основанных на HTTP. Она поддерживает все типы HTTP-запросов и может быть использована для проверки ответов (responses) по конкретно заданным запросам (requests).

Библиотека REST-Assured изначально предназначена для тестировщиков и интегрируется с любым основанным на Java фреймворком автоматизации. Она предоставляет похожий на BDD [4] доменно-специфичный язык, упрощающий создание автотестов для API на Java. Встроенные функции позволяют не программировать с нуля. Она также поддерживает XML и JSON-запросы/ответы. API REST-Assured создан так, что разработчику не требуется быть экспертом по HTTP, достаточно лишь разобраться с несложным синтаксисом.

RestSharp [11] является аналогом REST-Assured под платформу .Net и представляет собой очень функциональный и в то же время простой в использовании HTTP и REST-API клиент. Основные его особенности:

– Простая установка с использованием NuGet для большинства версий .Net.

– Автоматическая десериализация XML и JSON и поддержка пользовательских сериализации и десериализации через реализацию ISerializer и IDeserializer.

– Автоматическое определение типа возвращаемого контента.

– Поддержка основных типов запросов GET, POST, PUT, PATCH, HEAD, OPTIONS, DELETE, COPY, а также других нестандартных методов HTTP.

– Встроены схемы авторизации OAuth 1, OAuth 2, Basic, NTLM и Parameter-based, поддержка пользовательских схем авторизации через реализацию IAuthenticator.

PyRestTest [12] является инструментом для тестирования API на основе Python. Из особенностей можно выделить что библиотека позволяет создавать тесты в базовых YAML или JSON конфигурационных файлах, не требует кода и работает только под системами Mac и Linux.

Выводы. Тестирование API становится особенно востребованным в свете растущей популярности микросервисной архитектуры. Следовательно, даже в том случае, если на проекте пока не используется тестирование на уровне API, имеет смысл присмотреться к возможностям, которые оно предоставляет.

Рассмотренные выше инструменты весьма разнообразны и из них можно подобрать наиболее подходящий для нужд конкретного проекта. Так, если не планируется писать тесты с использованием языка программирования, то для этого, скорее всего, подойдет Postman. Он позволяет воспроизводить и сохранять запросы, а также выстраивать из них тесты, используя современный графический интерфейс.

Если же вам необходимо проводить нагрузочное тестирование с использованием большего количества запросов и имитацией одновременного трафика от нескольких пользователей, то тут подойдет JMeter. К тому же JMeter имеет очень хорошую поддержку сообщества и то, чего в нем нет «из коробки», можно найти в свободном доступе в виде плагинов, которые помогают в создании и анализе скриптов.

Katalon Studio, в свою очередь, является комплексным решением, которое включает в себя достаточный функционал для покрытия большей части приложения тестами. Katalon Studio скрывает все технические сложности и обеспечивает дружественный интерфейс с ручным режимом (пользователь может перетаскивать, выбирать ключевые слова и тест-объекты для формирования этапов тести-

рования), но при этом сохраняет необходимые инструменты для более технически подкованных пользователей, которые могут копнуть глубже в кодирование с помощью режима сценариев, который полностью поддерживает такие функции разработки, как подсветка синтаксиса, предложение кода и отладка.

Что касается внедрения использования API в тестовый проект с GUI автотестами или автоматизации тестирования API с помощью языка программирования, то тут упомянутые выше библиотеки REST-Assured, RestSharp и PyRestTest предоставляют практически одинаковый базовый функционал и являются взаимозаменяемыми, выбор зависит лишь от используемой платформы для тестового проекта.

Можно подвести итог, что внедрение тестирования API может принести пользу практически любому проекту, использующему API, а также сделать из тестировщиков, владеющих этими навыками, более востребованных специалистов в своей области.

ЛИТЕРАТУРА

1. Google Trends “API Testing” [Электронный ресурс]. Режим доступа: https://trends.google.com/trends/explore?date=today%205-y&q=api%20testing

2. Герасимов В.В., Кронфельд М.Ф., Озерова Д.М. Исследование технологий автоматизированного тестирования // Системні технології. Системи і процеси обробки інформації та управління: Збір. наук. праць. — Вип. 1(96) — Дніпропетровськ: "Системні технології", 2015, с. 130-136.

3. Матвеева Н.А., Герасимов В.В., Игнатьева Д.О. Исследование технологий модульного тестирования на платформе Java // Системні технології. Системи і процеси обробки інформації та управління: Збір. наук. праць. — Вип. 1(102) — Дніпропетровськ: "Системні технології", 2016, с. 49-55.

4. Литвинов А.А., Карпенко Н.В. Тестирование информационных систем: модульное, интеграционное, системное : учебное пособие. — Д.: Лира, 2016. — 284 с.

5. Освоение тестирования REST API [Электронный ресурс]. Режим доступа: https://quality-lab.ru/rest-api-testing/

6. Data Driven Testing | JazzTeam Software Development Company [Электронный ресурс].
Режим доступа: https://jazzteam.org/ru/technical-articles/data-driven-testing/

7. Postman | API Development Environment [Электронный ресурс].
Режим доступа: https://www.getpostman.com/

8. Apache JMeter [Электронный ресурс].
Режим доступа: http://jmeter.apache.org/index.html

9. Katalon Studio: Best automated testing tool for web, mobile, API [Электронный ресурс]. Режим доступа: https://www.katalon.com/ https://automated-

testing.info/t/detalnyj-obzor-katalon-studio-nadezhnaya-alternativa-oupen-sors-sistemam-avtomatizaczii/14061

10. REST-Assured [Электронный ресурс].
Режим доступа: http://rest-assured.io/

11. RestSharp - Simple REST and HTTP Client for .NET [Электронный ресурс]. Режим доступа: http://restsharp.org/

12. GitHub - svanoort/pyresttest: Python Rest Testing [Электронный ресурс]. Режим доступа: https://github.com/svanoort/pyresttest

REFERENCES

1. Google Trends “API Testing” [Electronic resource]. Access mode: https://trends.google.com/trends/explore?date=today%205-y&q=api%20testing

2. Gerasimov V.V., Kronfeld M.F., Ozerova D.M. Issledovanie tehnologiy avtomatizirovannogo testirovaniya // Systemni tekhnolohii. Systemy i protsesy obrobky informatsii ta upravlinnia: Zbir. nauk. prats. — Vyp. 1(96) — Dnipropetrovsk: "Systemni tekhnolohii", 2015, s. 130-136.

3. Matveeva N.A., Gerasimov V.V., Ignateva D.O. Issledovanie tehnologiy modulnogo testirovaniya na platforme Java // Systemni tekhnolohii. Systemy i protsesy obrobky informatsii ta upravlinnia: Zbir. nauk. prats. — Vyp. 1(102) — Dnipropetrovsk: "Systemni tekhnolohii", 2016, s. 49-55.

4. Litvinov A.A., Karpenko N.V. Testirovanie informatsionnyih sistem: modulnoe, integratsionnoe, sistemnoe : uchebnoe posobie. — D.: Lira, 2016. — 284 s.

5. Osvoenie testirovaniya REST API [Electronic resource]. Access mode: https://quality-lab.ru/rest-api-testing/

6. Data Driven Testing | JazzTeam Software Development Company [Electronic resource]. Access mode: https://jazzteam.org/ru/technical-articles/data-driven-testing/

7. Postman | API Development Environment [Electronic resource]. Access mode: https://www.getpostman.com/

8. Apache JMeter [Electronic resource].
Access mode: http://jmeter.apache.org/index.html

9. Katalon Studio: Best automated testing tool for web, mobile, API [Electronic resource]. Access mode: https://www.katalon.com/ https://automated-testing.info/t/detalnyj-obzor-katalon-studio-nadezhnaya-alternativa-oupen-sors-sistemam-avtomatizaczii/14061

10. REST-Assured [Electronic resource]. Access mode: http://rest-assured.io/

11. RestSharp - Simple REST and HTTP Client for .NET [Electronic resource].
Access mode: http://restsharp.org/

12. GitHub - svanoort/pyresttest: Python Rest Testing [Electronic resource].
Access mode: https://github.com/svanoort/pyresttest