АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ

новые ноты для гармонии оркестра

epam

ВВЕДЕНИЕ

Информационные системы в современной компании сегодня превращаются в оркестр. Каждое приложение – это отдельный инструмент, который может играть как соло (решать определенную задачу), так и в составе камерного ансамбля (вместе с другими системами поддерживать сквозные бизнес-процессы). Разные партии – то исполняемые параллельно, то со сдвигом на несколько тактов - напоминают синхронный и асинхронный режим работы всего комплекса систем.

Для удачного выступления каждый элемент оркестра должен играть совершенно, при этом образовывая единое гармоничное звучание с другими инструментами. Точно также: для удачной работы информационных систем необходимо протестировать как каждый отдельный компонент решений, так и их работу вместе. На помощь дирижеру-тестировщику приходят средства автоматизации, использование которых способно придать свежее звучание даже классическим мелодиям.

Эксперты EPAM примерили на себя роль музыкальных критиков и проанализировали современные инструменты автоматизации.

1. Ускорение темпа: когда автоматизация необходима
2. Мотивы разработки для лучшего звучания
3. Новые инструменты для исполнения
4. Практическое исполнение

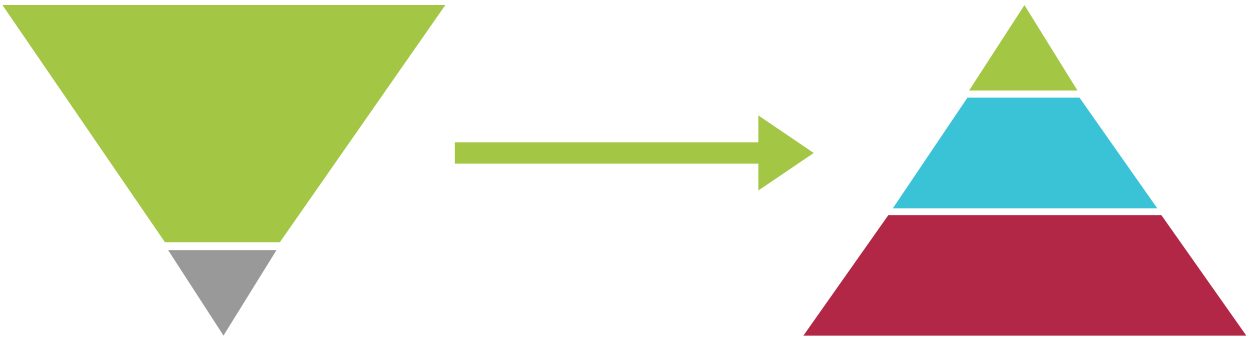
1 УСКОРЕНИЕ ТЕМПА: КОГДА АВТОМАТИЗАЦИЯ НЕОБХОДИМА

Accelerando, stringendo, stretto – темп и характер современного бизнеса с каждым днем изменяются так быстро, что за ними не угнаться без использования информационных технологий. Выпуск раз в год новых версий систем, с которыми работают конечные клиенты, уже не годится для бизнеса. Новый функционал должен появляться постоянно, например, ежемесячно. При этом от его качества зависит очень много: по мнению аналитиков Forrester, именно ПО стало ключевой точкой взаимодействия между клиентом и компанией, превращается в бренд и ее «лицо». Гибкие методологии разработки и практики CI/CD позволяют выпускать новый продукт в еще более короткие сроки. К примеру, Facebook предлагает обновления еженедельно. В Amazon пошли еще дальше и сделали электронный магазин - свой «продукт» - полностью динамичным: новая версия выходит каждые 11 секунд.

Учащение релизов и, как следствие, уменьшение длительности фазы тестирования влекут изменения в расчете возврата инвестиций в автоматизацию. При выпуске нового релиза 1 раз в 6 месяцев вкладываться в автоматизацию неэффективно. Так, для каждого обновления выполняется 1 цикл регрессии. В типичном случае автоматизация тестирования на уровне пользовательского интерфейса занимает в 10 раз больше времени, чем ручное выполнение тех же тестов. Поскольку тесты будут выполняться 2 раза в год, нам понадобится 5 лет только на то, чтобы окупить затраты на автоматизацию тестирования. Но за это время наш продукт изменится кардинально, что приведет к огромным затратам на поддержку тестов в актуальном состоянии.
В случае с 10-12 релизами в год мы, напротив, не можем себе позволить не инвестировать в автоматизацию и тратить месяцы на регрессионное тестирование. Автоматизация становится быстроокупаемой (нужен примерно 1 год) и жизненно необходимой. В результате она постепенно начинает использоваться на все большем количестве проектов.

В сами приложения стремительно проникают новые технологии. Mobile, Cloud, Big Data, Social, распределенная и модульная архитектура - технологии, которые быстро набирают популярность и по отдельности или в комплексе находят свое отражение в современных решениях. Усложнение программных продуктов и переход к модульной и распределенной архитектуре позволяют организовать тестирование каждого модуля по отдельности. Такой подход дает возможность выявлять дефекты раньше и исправлять их быстрее. Возможность перенести тестирование на уровень отдельных компонентов приводит к росту количества автотестов и самого проекта автоматизации.

ПОКРЫТИЕ ПРИЛОЖЕНИЯ ТЕСТАМИ, ДО И ПОСЛЕ ВНЕДРЕНИЯ ТЕСТИРОВАНИЯ НА УРОВНЕ МОДУЛЕЙ



- Системные тесты (end-to-end tests)
- Интеграционные тесты (integration/API tests)
- Модульные тесты (unit tests)
- Непокрытая тестами часть

Переход от централизованных приложений к модульным и децентрализованным влияет на тестовые сценарии. Большие end-to-end сценарии трудно поддаются автоматизации, а если и поддаются - являются нестабильными. В таких случаях выгодно сместить фокус автоматизации на покрытие атомарных бизнес-функций, нежели полного пути пользователя. Из-за уменьшения количества действий в тест-сценарии время его выполнения уменьшается. Наконец, в этом случае идентифицировать ошибки и затем устранить их становится проще. Тем самым эффект от применения средств автоматизации возрастает.

2 МОТИВЫ РАЗРАБОТКИ ДЛЯ ЛУЧШЕГО ЗВУЧАНИЯ

Новым трендом в автоматизации тестирования становится заимствование тестировщиками «лучших мелодий» - практик и инструментов - из арсенала разработчиков. Если в музыке подобное сочли бы плагиатом и подвергли бы остракизму, то в сфере ИТ этот подход даже приветствуется. Отметим, что применение при реализации проектов джазовых импровизаций - гибких методологий разработки (Agile) -меняет схему взаимодействия между разработчиками и тестировщиками. Если ранее разработка и тестирование существовали раздельно, то теперь это единая, тесно взаимодействующая команда.

Современный проект автоматизации содержит десятки и сотни тысяч строк кода и имеет многоуровневую модульную архитектуру. Поддерживаемость такого проекта жизненно необходима. Обеспечить ее можно с помощью таких практик разработчиков, как:

- применение объектно-ориентированной архитектуры проекта (отсутствует необходимость использовать функции для повторяющегося кода, функции уже «встроены» в объекты»), возможность использовать стандартные инструменты для сборки и отладки и подключать дополнительные библиотеки и фреймворки, которые входят в используемый язык программирования (Java, C#, Python и др.).
- использование системы контроля версий, когда проект автоматизации хранится в одном репозитории с основным продуктом. При этом если раньше в качестве инструмента для управления версиями использовался централизованный SVN, то сейчас акцент все больше смещается в сторону децентрализованной и распределенной системы Git.
- применение паттернов проектирования разработки и создание новых паттернов, например, Page Object паттерн в инструменте Selenium Web Driver
- частые просмотр и инсп екции кода, применение coding guidelines.

Одновременно с практиками, специалисты по автоматизации тестирования начинают использовать и инструменты разработчиков. Раньше достаточно было беглого взгляда на монитор сотрудника, чтобы понять, кто он- автоматизатор-тестировщик или разработчик. Сейчас разобраться в этом стало проблематичней. Тестировщики все чаще отказываются от привычных продуктов QTP, TestComplete, Silk Test в пользу широко распространенных, мощных и знакомых каждому разработчику сред, как Eclipse или Visual Studio. При автоматизации тестирования веб-приложений популярностью пользуется стек технологий – Java + Selenium +JUnit. Для расширения возможностей проектирования, оптимизации работы с базами данных и веб-сервисами часто применяются библиотеки Spring и Hibernate, которые перестали быть исключительной прерогативой разработчиков. Тем самым использование и в разработке, и в тестировании одинаковых технологий, языков, инструментов сборки и формирования отчетов позволяет убрать границу между двумя процессами и обеспечить их бесшовную интеграцию.

3 НОВЫЕ ИНСТРУМЕНТЫ ДЛЯ ИСПОЛНЕНИЯ

Свежие ноты в классическую мелодию автоматизации тестирования добавляет развитие программного обеспечения, которое идет в геометрической прогрессии. За пару лет технология может родиться, развиться и завоевать рынок, как это было в случае, например, с браузером Google Chrome или операционной системой Android. Как только появляется новая технология, возникают и инструменты для ее тестирования, в том числе новые средства автоматизации.
Многие инструменты автоматизации, которые мы сейчас знаем как свободное программное обеспечение, начинали свою жизнь в инновационных компаниях, которые использовали эти продукты для тестирования собственных решений. Так, Selenium Core был придуман в Thought Works, а WebDriver - в Google. Решение поделиться собственным продуктом с сообществом и открыть его код имеет ряд преимуществ. Во-первых, это представляет компанию в выгодном света как инноватора и мецената. Во-вторых, позволяет использовать ресурсы мирового ИТ-сообщества для исправления ошибок и разработки нового функционала.

В результате бурное развитие инструментов с открытым исходным кодом и децентрализованных систем контроля версий открывает возможности для появления ответвлений от продуктов автоматизации и их дополнения другим функционалом. В сочетании с бесплатным хостингом на таких платформах, как github, bitbucket, sourceforge, это привело к возникновению множества небольших «оберток» и плагинов для популярных инструментов разработки и автоматизации (например, проект Selenium имеет более 300 ответвлений). Возможность публиковать готовые решения и расширения для средств автоматизации порождает большое количество готовых к работе фреймворков. Часть таких ответвлений в последствии перерастает в самостоятельные продукты и обгоняет по популярности своих «родителей». Наиболее характерные примеры для автоматизации веб-приложений - Thucydides от Wakaleo Consulting (интеграция приемочных тестов и отчетности с WebDriver) и HtmlElements от Яндекса (расширение паттерна Page Object в WebDriver).

Одним из трендов последнего времени становится тот факт, что все чаще маленькие млекопитающие – решения с открытым исходным кодом – выигрывают в эволюционной борьбе у больших динозавров – коммерческих инструментов, даже несмотря на то, что открытое ПО, к примеру, предъявляет более высокие требования к техническому уровню специалистов. Как правило, основной фактор успеха свободных ИТ-решений – это концентрация на более узких задачах и способность их быстро решить с лучшим результатом.

Такой естественный отбор особенно хорошо заметен среди инструментов автоматизации мобильного тестирования. Если применимо к тестированию веб-приложений признанным лидером является Selenium WebDriver, то в мобильной сфере явного победителя пока нет. Наиболее явным претендентом на лидерство с середины 2013 года можно назвать решение Appium. В его пользу говорят факторы схожести с лидером близкой сферы Selenium и возможности использования в качестве инструмента для тестирования и нативных, и гибридных, и веб-приложений, а также кросс-платформенность и стабильность работы инструмента. Но может ли в ближайшие полгода произойти смена лидера автоматизации мобильного тестирования? Вполне: рынок предельно динамичен.

Наконец, нельзя не упомянуть о еще одном обстоятельстве. Сейчас бизнес-специалисты начинают принимать все большее участие в процессе разработки и тестирования ПО, поскольку при высокой частоте релизов необходимо не менее часто определять набор изменений, который будет реализован в новой версии продукта. Тесное взаимодействие бизнеса и разработчиков промышленного ПО привело к возникновению новой методологии разработки и тестирования программного обеспечения - Behavior-driven development. Она предполагает, что бизнес-специалисты предоставляют требования к ИТ-решению в специальном формате Gherkin language, который одинаково хорошо читается как человеком, так и фреймворком. В результате для создания нового автотеста нет необходимости вносить изменения в код – достаточно лишь передать фреймворку новую спецификацию в формате Gherkin. При применении этой методологии новые автотесты создаются бизнес-аналитиками, что позволяет снизить риск потери информации при коммуникациях между заказчиком и исполнителем и минимизировать затраты на разработку и тестирование.

4 ПРАКТИЧЕСКОЕ ИСПОЛНЕНИЕ

Тенденции, о которых мы говорили выше, хорошо иллюстрируются примером из практического опыта одного из заказчиков EPAM Systems – системы заказа путешествий через Интернет. Это решение разрабатывается десятками команд по всему миру. Его простой в течение нескольких часов означает для владельцев бизнеса потери миллионов долларов упущенной прибыли. Новые версии системы выходят каждый месяц.

Важной частью поставки каждого обновленного компонента решения являются автоматизированные тесты. При автоматизации используются те же технологии, что и при разработке системы – Java, Spring, Hibernate, JUnit, что облегчает процесс. Испытания проводятся на нескольких уровнях (юнит-, компонентные и end-to-end тесты), причем все они автоматизированы. Сборка и тестирование новой версии производится в автоматическом режиме по коммиту (запись изменений в репозиторий) в систему версионного контроля. Для управлением интеграцией используется система Jenkins.

В целом процесс тестирования выглядит следующим образом:

1. Происходит коммит изменений в систему контроля версий
2. Производится статический анализ и инспекция кода автотестов
3. Сервер непрерывной интеграции выделяет машину для тестов и подготавливает окружение
4. При необходимости создаются заглушки нижестоящих и вышестоящих уровней, чтобы изолировать тестируемый компонент от других модулей
5. Запускается группа автотестов. Запуск можно сделать как изолированным (с заглушками остальных компонентов), так и среди реальных модулей. Выполнение автотестов идет параллельно в несколько потоков.
6. В случае успешного выполнения тестов компонент переходит на сервер тестирования интеграции, где выполняются регрессионные тесты, end-to-end сценарии и негативные ручные тесты.
7. При успешной интеграции компонентов выпускается новая версия продукта.

Тестирование сборки занимает не больше часа, благодаря чему в день в среднем выходит около 4-х сборок.

ЦИКЛ ТЕСТИРОВАНИЯ

Мелодия автоматизации тестирования активно набирает силу и обогащается все новыми мотивами. Усложнение программных продуктов и переход к модульной и распределенной архитектуре позволяют организовать тестирование каждого модуля системы или решения по отдельности, что дает возможность сократить сроки проведения тестирования. Применяемые разработчиками инструменты и практики все больше «перетекают» в сферу тестирования, что помогает специалистам с обеих сторон работать в единой команде и повышает эффективность как тестирования, так и разработки. Появление новых технологий порождает новые инструменты автоматизации, самый яркий пример – Appium из сферы автоматизации тестирования мобильных приложений. Коммерческие продукты автоматизации все больше уступают место под солнцем свободному ПО. В выигрыше от этого остаются компании-заказчики: оркестр их информационных систем в итоге звучит слаженно и гармонично.

Основанная в 1993 году компания EPAM Systems (NYSE: EPAM) специализируется на разработке программных решений. Штаб-квартира компании расположена в США. EPAM работает с клиентами по всему миру, используя свои отмеченные наградами центры разработки и офисы в 19 странах Северной Америки, Европы, Азии, Австралии. EPAM помогает клиентам повышать эффективность бизнеса, предлагая услуги по цифровому преображению и бизнес-трансформации, созданию умного предприятия и внедрению передовых технологий. По мнению ведущего независимого аналитического агентства, EPAM является лидером в сфере разработке программных решений, входит в Топ 10 поставщиков услуг для торговли и в число ключевых игроков на рынке Agile в мире. По данным издательского дома «Коммерсант» EPAM Systems возглавляет список крупнейших разработчиков программного обеспечения в России.

Более подробная информация доступна на www.epam-group.ru

РОССИЯ И СНГ

ул. 9-ая Радиальная, д.2
Москва, Россия, 115404

Тел.: +7-495-730-63-62
Факс: +7-495-730-63-61

ЕВРОПА

Corvin Offices I. Futó street
47-53
Budapest, H-1082 Hungary

Тел: +36-1-327-7400
Факс: +36-1-577-2384

США

41 University Drive Suite 202
Newtown, PA, USA 18940

Тел: +1-267-759-9000
Факс: +1-267-759-8989