Тестирование WEB-приложений

В. В. Гурковская

В данной работе рассматриваются основные способы тестирования Веб-приложений, как ручные так и автоматизированные. В частности рассматриваются некоторые теоретические аспекты ручного тестирования, а так же основные средства автоматизированного тестирования. Проведено изучение средств автоматизированного тестирования на примере конкретного Веб-приложения.

Вступ

Вычислительные и коммуникационные системы используются все чаще и с каждым днем все глубже входят в нашу повседневную жизнь. Компании и отдельные пользователи все больше зависят в своей работе от Веб-приложений. Веб-приложения очень динамичны, а их функциональные возможности непрерывно растут. Непрерывно возрастает потоковый трафик средств информации и запросов, формируемых переносными и встроенными устройствами. Вследствие этого возрастает сложность систем такого рода. Веб-приложения становятся все более распространенными и все более сложными, играя, таким образом, основную роль в большинстве онлайновых проектов. Как и во всех системах, основанных на взаимодействии между клиентом и сервером, уязвимости Веб-приложений обычно возникают из-за некорректной обработки запросов клиента или недостаточной проверки входной информации со стороны разработчика. Для устранения всякого рода неполадок и недочетов и выполняют тестирование во время создания и использования Веб-приложений.

Тестирование - это проверка соответствия между реальным поведением программы и ее ожидаемым поведением в специально заданных, искусственных условиях, осуществляемая на конечном наборе тестов, выбранном определенным образом.

Основные подходы к тестированию Веб-приложений

- Функциональное тестирование(functional testing) - процесс верификации соответствия функционирования продукта его начальным спецификациям. Методы функционального тестирования веб-приложений:
  - Record and Play
  - Functional Decomposition
  - Data-driven
  - Keyword-driven
  - Object-driven
  - Model-based
- Тестирование пользовательского интерфейса
  - Анализ требований к пользовательскому интерфейсу
  - Разработка тест-требований и тест-планов для проверки пользовательского интерфейса
  - Выполнение тестовых примеров и сбор информации о выполнении тестов
  - Определение полноты покрытия пользовательского интерфейса требованиями
  - Составление отчетов о проблемах в случае несовпадения поведения системы и требований, либо в случае отсутствия требований на отдельные интерфейсные элементы
- Тестирование удобства использования;
  Выделяют следующие этапы тестирования удобства использования пользовательского интерфейса:
  - Исследовательское

  - Оценочное
  - Валидационное
  - Сравнительное
- Нагрузочное тестирование имитирует одновременную работу нескольких сотен или тысяч посетителей, проверяя, будет ли устойчивой работа сайта под большой нагрузкой

  Основные цели:
  - оценка производительности и работоспособности приложения на этапе разработки и передачи в эксплуатацию
  - оценка производительности и работоспособности приложения на этапе выпуска новых релизов, патчей
  - оптимизация производительности приложения, включая настройки серверов и оптимизацию кода
  - подбор соответствующей для данного приложения аппаратной (программной платформы) и конфигурации сервера
- Проверка ссылок и HTML-кода
  - Проверка ссылок - актуальна для внутренних ссылок (в случае больших и разветвленных порталов) и для внешних - если это, к примеру, каталог сайтов, или страница "Ссылки"
  - Проверка HTML-кода страниц - проверка корректности HTML-кода, в том числе на соответствие стандартам
- Тестирование безопасности. Тестированию подвергается не только сам конкретный сайт или веб-приложение, а весь сервер полностью не перетвориться на дерево, тобто, ліс, що складається з однієї компоненти зв'язності.

Автоматизированное тестирование

В двух словах, автоматизация тестирования позволяет оптимизировать качество сложных приложений эффективным по стоимости способом за приемлемое время. Это помогает быстрее выпустить программное обеспечение более высокого качества. Процесс автоматизации тестирования делится на три этапа:

- Запись. Сценарий тестирования записывается "на лету"по мере работы пользователя с приложением. .
- Улучшение. Добавление кода, выполняющего разнообразные функции. Типичные изменения сценариев тестирования - условное ветвление, рефакторинг и обработка исключительных ситуаций.
- Воспроизведение. Выполнение сценариев, эмулирующих действия, которые выполнял пользователь приложения при записи теста. Расхождения регистрируются, и тестировщик может сделать вывод о том, хорошо ли функционирует приложение или регрессионное тестирование выявило проблемы.

Средства от IBM Rational:

- IBM Rational Robot - универсальное средство автоматизации тестирования общего назначения для команд разработчиков, выполняющих функциональное тестирование клиент-серверных приложений. Дает возможность обнаруживать неполадки в ПО благодаря расширению сценариев тестирования средствами условной логики, позволяющей целиком охватить тестируемое приложение. Robot позволяет создавать сценарии тестирования с вызовом внешних библиотек DLL или исполняемых модулей.
- IBM Rational Performance Tester - инструмент нагрузочного и стрессового тестирования, с помощью которого можно выявлять проблемы системной производительности и их причины. Позволяет создавать тесты без написания кода и, не требуя навыков программирования. Обеспечивает гибкие возможности моделирования и эмуляции

различных пользовательских нагрузок. Выполняет сбор и интеграцию данных о серверных ресурсах с данными о производительности приложений, получаемыми в режиме реального времени.

- IBM Rational Functional Tester - набор средств автоматизированного тестирования, позволяющих выполнять функциональное и регрессионное тестирование, тестирование пользовательского интерфейса и тестирование, управляемое данными. Инструмент применяет технологию ScriptAssure и функции поиска соответствия по шаблону, позволяющие повысить устойчивость сценариев тестирования в условиях частых изменений пользовательских интерфейсов приложений.
- IBM Rational Quality Manager - решение для реализации процессов управления тестированием и качеством, поддерживает сотрудничество участников групп по разработке программных продуктов, предоставляя им возможность обмениваться информацией, применять средства автоматизации для сокращения графиков выполнения проектов, а также составлять отчеты по проектным показателям для принятия обоснованных решений. Rational Quality Manager может быть дополнен средством управления ресурсами тестирования Rational Test Lab, обеспечивающим учет ресурсов тестирования (серверов), их бронирование, автоматизацию развертывания тестовой среды на сервере и запуск скриптов тестирования, а также отчетность по использованию ресурсов тестирования.
- Rational Quality Manager и Rational Test Lab созданы на базе открытой платформы Jazz, которая предоставляет стандартные интерфейсы и удобные возможности для интеграции с решениями партнеров и других производителей.

Выводы

На сегодняшний день тестирование Веб-приложений является такой же важной составляющей как и создание самого приложения. Тестирование позволяет сделать процесс разработки ПО прозрачным и управляемым для всех участников проекта. Разработчикам тестирование даёт уверенность в верном понимании задач, которые ставит перед ними заказчик. Проектным менеджерам тестирование даёт понимание эволюции проекта, проблемных мест в процессе разработки, а также информацию для принятия оперативных решений о готовности продукта или его версии к продуктивной эксплуатации, продажам и т.д. Если проводить тестирование на каждом шаге создания приложений, то у пользователей не будет претензий к разработчикам. Приложение будет работать корректно и удовлетворять все требования пользователя.

Список литературы

[1] Элфрид Дастин, Джефф Рэшка, Джон Пол Автоматизированное тестирование программного обеспечения
[2] Илья Винниченко Автоматизация процессов тестирования 1989. - 478 с.
[3] Луиза Тамре Введение в тестирование программного обеспечения
[4] Диан Стотлемайер Тестирование Web-приложений
[5] Роберт Калбертсон Быстрое тестирование
[6] В. П. Котляров, Т. В. Коликова Основы тестирования программного обеспечения

Автор

Виктория Валерьевна Гурковская — студентка 4-го курса, факультет кибернетики, Киевский национальный университет имени Тараса Шевченка, Киев, Украина; E-mail: *lonelysmile1@mail.ru*