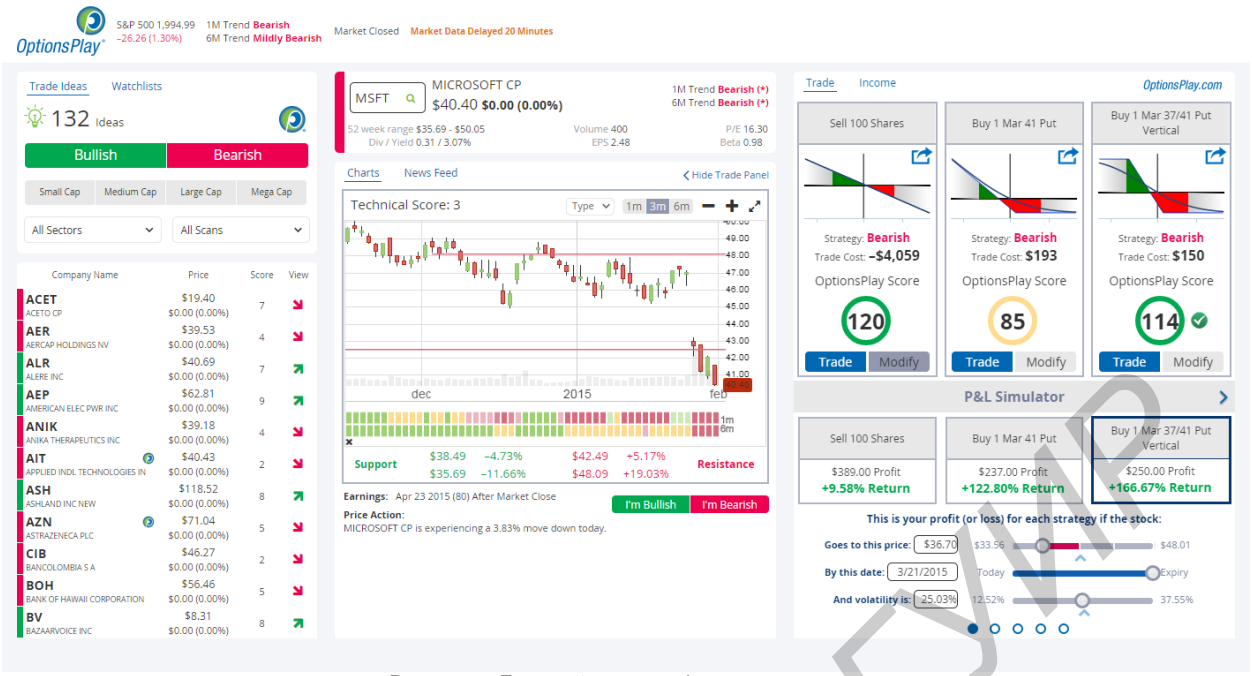

Рисунок 1. Главный экран Web-приложения

Список использованных источников:
1. Макмиллан Л.Г. Опционы как стратегическое инвестирование/Пер. с англ. М.: Евро, 2003 г., 1225 с.
2. TheWebSocketProtocol [Электронный ресурс]. — Режим доступа: http://tools.ietf.org/html/rfc6455. – Дата доступа: 21.05.2014.

ПРОГРАММНАЯ ПОДДЕРЖКА АВТОМАТИЗАЦИИ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Иванова В.Л.
Поттосина С.А. – канд. физ.-мат. наук, доц.

В настоящее время программное обеспечение стремительно совершенствуется и приобретает новые формы. Не редкостью сейчас является то, что одно и то же приложение может быть представлено в нескольких формах: оно может быть настольным, а также иметь web- и мобильную версии (например, Skype, Slack). И, несмотря на то, что для тестирования всех форм приложения может использоваться одинаковый или схожий набор тестов, нет универсального инструмента автоматизации тестирования, который бы позволил протестировать все виды и осуществить одновременную проверку всех компонент программного продукта.

Любая система состоит из модулей, которые взаимодействуют друг с другом и с внешними системами. Для успешной реализации проекта важно, чтобы эти модули работали корректно. Поэтому на протяжении всего жизненного цикла разработки программного обеспечения необходимо производить тестирование на различных уровнях. Уровень тестирования определяет то, над чем производятся тесты: над отдельным модулем, группой модулей или системой в целом [1]. Выделяют следующие уровни тестирования, представленные в таблице 1.

Таблица 1 – Уровни и цели тестирования

Уровень тестирования	Цели тестирования
Компонентное (модульное) тестирование	Проверка работоспособности отдельных модулей системы (функции или класса)
Интеграционное тестирование	Нахождение проблем взаимодействия модулей (компонент) системы
Системное тестирование	Проверка полной (интегрированной) системы на соответствие системным требованиям и показателям качества
Приемочное тестирование	1 Определение уровня соответствия системы приемочным критериям 2 Принятие решения о том принимается система или нет

Автоматизацию тестирования как web-, так и мобильных приложений, можно успешно внедрять на всех перечисленных выше уровнях. Однако, не всегда очевидно, какие тесты автоматизировать не стоит. Если в web-приложении, например, отображается какая-то графика, то наверняка будет лучше и проще взглянуть на нее глазами человека, чем изобретать робота. Автоматизировать стоит лишь то, что хорошо

поддается автоматизации. В противном случае есть риск потратить время впустую, получив нестабильные, сложные, трудно поддерживаемые и не приносящие полезных результатов тесты.

Наиболее успешным является сочетание тестов уровня пользовательского интерфейса (интеграционных) и программного интерфейса (компонентных). API-тесты дают существенное преимущество во времени выполнения и устойчивости к изменениям в продукте. Но одновременно требуют более продуманной разработки, так как они должны имитировать действия пользователей в реальном приложении. Современный уровень развития информационных технологий требует как никогда взвешенного подхода к тестированию вообще и к автоматизации в частности.

Очень важно четко определить, какую функциональность лучше будет протестировать на более низком уровне, а какую на конечном, пользовательском. По этому поводу существует концепция, которая носит название «пирамида автоматизации тестирования».

Пирамида автоматизации тестирования первоначально была описана Майком Коном в книге «Scrum. Гибкая разработка ПО» [2].

Эта пирамида, представленная на рисунке 1, показывает идеальный вариант распределения количества автоматизированных тестов по категориям.

Модульные тесты пишутся, как правило, разработчиками, на том же языке, что и приложение. Это тесты, которые отвечают за тестирование каждого отдельного метода. Таких тестов должно быть больше всего – они позволяют выявить дефекты на самых ранних стадиях разработки.

Компонентные тесты – это тесты уровня API. Они пишутся для тестирования логики приложения не через интерфейс, а через функциональность. Можно использовать различные таблицы для формирования исходных данных. Такие тесты пишут, как правило, на языке предметной области, чтобы они были понятны и заказчику, и разработчикам. Эти тесты взаимодействуют с кодом приложения напрямую, без посредника в виде пользовательского интерфейса.

Интеграционные тесты представляют собой тесты, которые взаимодействуют с приложением через пользовательский интерфейс, имитируя действия пользователя. Например, можно проверить, что кнопки работают, выполняют ожидаемые действия, появляются корректные сообщения об ошибках. Какие-либо тесты, где необходимо тестировать именно пользовательский интерфейс, или, когда без пользовательского интерфейса не обойтись. Таких тестов должно быть немного, так как они, как правило, очень часто перестают работать из-за частого изменения структуры пользовательского интерфейса приложения. Они сложны в поддержке и эксплуатации.

Ручные тесты – это тесты, которые приходится выполнять вручную вследствие того, что проверку этой функциональности невозможно или неоправданно сложно автоматизировать. Нужно стремиться к тому, чтобы таких тестов оставалось как можно меньше, однако, полностью исключать ручное тестирование приложение не стоит, да и, скорее всего, не получится.

Рисунок 1 – Пирамида автоматизации тестирования (согласно Майку Кону)

Нет универсального соотношения всех видов тестов, поскольку это должно определяться для каждого приложения индивидуально, однако, очень полезно руководствоваться этим подходом к автоматизации тестирования при проектировании и написании тестов.

Существует множество готовых инструментов автоматизации тестирования, позволяющих без особого труда разрабатывать тесты определенных уровней. Однако, эти инструменты, как правило, способны работать только с одним видом интерфейса приложения: либо web, либо мобильным. Нет такого готового инструмента автоматизации, который позволил бы разрабатывать, хранить и эксплуатировать автоматические тестовые скрипты всех перечисленных видов и уровней.

Система, разработанная с использованием библиотек языка Java, которые помогают взаимодействовать с web (Selenium), программным (RestAssured) и мобильным (Appium) интерфейсами тестируемого приложения, взаимодействует с системой непрерывной интеграции Jenkins и выполняет следующие основные функции:

- осуществление регрессионного и дымового тестирования web и мобильного интерфейсов на всех уровнях тестирования;
- автоматический запуск тестов сразу после внесения изменений в код тестируемого программного продукта;
- автоматическое формирование отчета о прохождении тестов на сервере непрерывной интеграции.

При выборе инструментов для автоматизации были учтены особенности тестирования web- и мобильных приложений.

Для создания системы автоматизации тестирования были выбраны бесплатные инструменты, что позволило сэкономить на приобретении дорогостоящих лицензий готовых инструментов.

Благодаря разработанной системе было значительно сокращено время регрессионного тестирования приложения, а также ускорено обнаружение дефектов на ранних стадиях разработки.

Список использованных источников:

1.Виды тестирования программного обеспечения [Электронный ресурс]. – Режим доступа : http://www.protesting.ru/testing/testtypes.html.

2. Кон, М. Scrum. Гибкая разработка ПО / Пер. с англ. – М. : ООО «И.Д. Вильямс», 2011. – 288 с.

ОПЕРАЦИОННОЕ УПРАВЛЕНИЕ ОРГАНИЗАЦИЕЙ ПРОЦЕССА ЭЛЕКТРОННЫХ ПРОДАЖ НА ОСНОВЕ ПЛАТФОРМЫ HYBRIS

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Муха И. Н.

Поттосина С.А. – канд. физ.-мат. наук, доц.

В современном мире в условиях всеобщей глобализации экономика получила новые возможности в сети Интернет. Осуществление экономической деятельности, благодаря новейшим информационным технологиям, делают ее эффективней и прибыльней. Актуальность данной темы обусловлена тем, что в экономике зародился новый сектор – электронная коммерция, которая является одной из составляющих «новой экономики», обретающая все большую практическую значимость. Влияние мировых тенденций будет сказываться на увеличении доли электронной коммерции в белорусской экономике в ближайшее время, следовательно, будет возрастать ее положительное воздействие на экономику государства и уровень жизни общества.

Электронную коммерцию часто трактует как «технологию совершения коммерческих операций и управления производственными процессами с применением электронных средств обмена данными». Согласно данной трактовке имеется возможность включить в предметную область такие системы, как MRP, MRP II, ERP благодаря упоминанию управления производственными процессами. Включение понятия «управление производственными процессами» обусловлено эффективностью, которая достигается применением электронной коммерции для организации поставок на предприятие.

Как и любой вид производственной деятельности, организация электронных продаж требует применения правильно подобранной стратегии управления всем процессом продаж. Для достижения желаемых результатов каждое предприятие, ориентированное на долгосрочную работу, разрабатывает определенные планы по завоеванию своей доли рынка. Однако без ежедневного управления продажами реализовать долгосрочные планы невозможно. Именно поэтому в компании должно быть не только стратегическое, но и оперативное управление организацией процесса электронных продаж. Система управления электронными продажами разрабатывается для эффективной реализации стратегии. Она позволяет достичь следующих целей:

1) увеличить объемы продаж;
2) положительно повлиять на продуктивность работы;
3) предотвратить возникновение сложных ситуаций;
4) выбрать соответствующий стиль управления и методы контроля;
5) выделить приоритетную для сбыта продукцию;
6) определить свою целевую аудиторию;
7) адаптировать стратегию под актуальные запросы рынка.

Главная цель деятельности операционных менеджеров заключается в управлении и контроле процесса производства или предоставления услуг. Как показано на рисунке 1, данный процесс представляет собой преобразование входных данных (например, сырье, материалы, человеческие ресурсы, продукция и т.п.) с помощью технологий в продукты или сервисы, предназначенные потребителю.