УДК 004.4

СОКРАЩЕНИЕ ВРЕМЕНИ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Шакирова А.И., Хасьянов А.Ф., Даутов Э.Ф.

ФГАОУ ВО «Казанский (Приволжский) федеральный университет», Казань, e-mail: public.mail@kpfu.ru

Как известно, тестирование программного обеспечения занимает много времени, требует тщательной проверки и иногда при проведении тестирования происходит пропуск дефектов и ошибок. В рамках данной научной статьи рассмотрено, как проводится тестирование программного обеспечения нескольких видов. Рассмотренные виды тестирования: интеграционное тестирование, функциональное тестирование, тестирование интерфейса, тестирование производительности, тестирование совместимости и тестирование локализации. В рамках работы рассмотрены инструменты, используемые при проведении различных видов тестирования. Произведен опрос и интервьюирование о проведении тестирования в IT-компаниях России, результаты опроса приведены в работе. Проведен анализ опроса и интервью, сделаны выводы. По итогу опроса и интервью разработаны новые методы тестирования программного обеспечения, которые уменьшают время, затрачиваемое на тестирование, при этом количество найденных дефектов и ошибок не уменьшается. Составлена таблица, показывающая ожидаемый результат тестирования и фактический результат тестирования с помощью разработанного метода тестирования. Была выведена формула для подсчета оптимального времени тестирования с помощью нового разработанного метода тестирования программного обеспечения. В результате разработанные методы внедрены в компании, принявшие участие в опросе.

Ключевые слова: тестирование программного обеспечения (ПО), расчет времени тестирования программного обеспечения (ПО), интеграционное тестирование, функциональное тестирование, тестирование интерфейса, тестирование производительности, тестирование совместимости, тестирование локализации

REDUCTION OF TESTING TIME SOFTWARE

Shakirova A.I., Khasyanov A.F., Dautov E.F.

Kazan (Volga region) Federal University, Kazan, e-mail: public.mail@kpfu.ru

As you know, software testing takes a lot of time, requires careful testing and sometimes during testing, there is a omission of defects and errors. In the framework of this scientific article, it is examined how several types of software are tested. Considered types of testing: integration testing, functional testing, interface testing, performance testing, compatibility testing and localization testing. As part of the work considered the tools used in conducting various types of testing. A survey and interviewing about testing in IT-companies in Russia, the results of the survey are given in the work. An analysis of the survey and interviews, conclusions. Based on the survey and interview results, new software testing methods have been developed, which reduce the time spent on testing, and the number of defects and errors found does not decrease. A table was compiled showing the expected test result and the actual test result using the developed test method. A formula was derived for calculating the optimal testing time using a newly developed software testing method. As a result, the developed methods were introduced in the companies that participated in the survey.Keywords: software testing, software testing time calculation, information technology, program quality testing, unit testing, integration testing, functional testing, interface testing, performance testing, compatibility testing, localization testing.

Keywords: software testing, software testing timing, integration testing, functional testing, interface testing, performance testing, compatibility testing, localization testing

Тестирование программного обеспечения – это проверка, соответствует ли реальное поведение программы ожидаемому.

Порядок работ в рамках тестирования ПО можно разделить на несколько уровней: модульное, интеграционное, системное. Каждый уровень включает в себя подуровни (виды тестирования).

В данной статье рассматриваются несколько видов тестирования, которые описаны ниже:

– интеграционное тестирование;

– функциональное тестирование [1, 2];

– тестирование интерфейса (удобства интерфейса);

– тестирование производительности (включает в себя тестирование стабильности; стресс тестирование; объемное тестирование; нагрузочное тестирование) [3];

– тестирование совместимости (кроссбраузерное и мультиплатформенное тестирование);

– тестирование локализации (в том случае, если ПО предназначено для использования с различными языками интерфейса) [4].

Повышение эффективности в виде сокращения времени тестирования ПО без потери качества и количества найденных дефектов является актуальной задачей, имеющей научный и практический интерес [5, 6].

Цель исследования: разработка метода повышения эффективности тестирования ПО, позволяющего сократить время тестирования без потери качества.

В основу метода положены результаты опроса 17 команд, занимающихся тестированием ПО.

Опросу подверглись различные компании России, несколько из них входят в десятку ведущих IT-компаний Республики Татарстан. В целях конфиденциальности названия компаний будут изменены.

В результате опроса выяснено что более половины (52,9 %) опрошенных компаний тестируют ПО полуавтоматизированным способом; 23,5 % опрошенных компаний тестируют ПО исключительно вручную, и такая же доля 23,5 % использует исключительно полностью автоматизированное тестирование.

Виды тестирования, которые фигурировали в опросе, представлены на рис. 1.

В 8 опрошенных командах работают более 10 тестировщиков, в остальных менее 10.

На рис. 2 показаны типы проектов, тестированием которых занимаются опрошенные компании.

Практически все компании используют для организации и документирования тестирования следующие инструменты: тест-планы, «баг-репорты», техническое задание и, в меньшей степени, чек-листы, и ассоциативные карты.

Большинство инженеров тестирования, принявших участие в опросе, отмечают необходимость снижения числа пропущенных ошибок. Многие отмечают, что для этого большую часть тестирования приходится делать ручным способом. Как следствие, тестирование занимает много времени. Однако есть и те участники опроса, которые не отмечают проблем в существующем подходе к обеспечению качества ПО.

Больше всего времени все компании, принявшие участие в опросе, уделяют тестированию функциональности.

Рис. 1. Виды тестирования, которые выполняют опрошенные компании

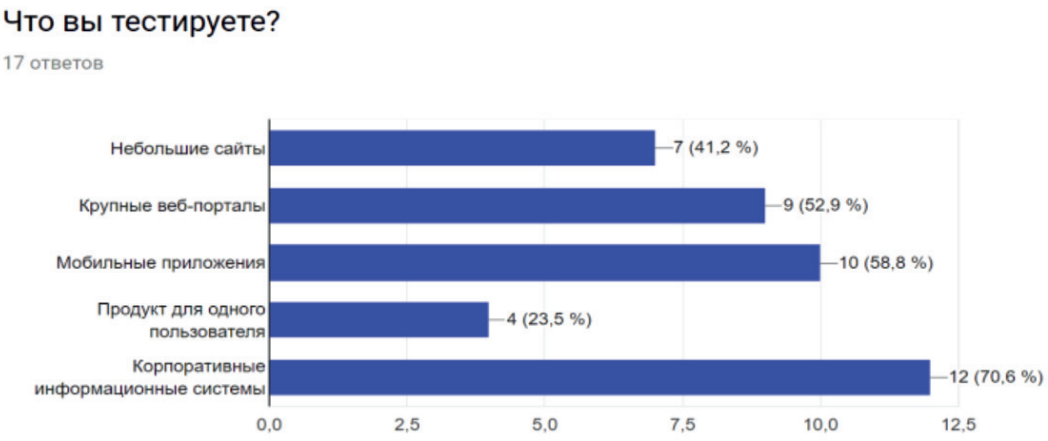

Рис. 2. Основные типы проектов в опрошенных компаниях

Рис. 3. Инструменты, используемые для тестирования ПО

На рис. 3 представлены инструменты, используемые для тестирования ПО.

Результаты опроса

Интеграционное тестирование. Оказалось, что большинство компаний осуществляют данный вид тестирования ПО. В процессе интеграционного тестирования происходит проверка: как взаимодействуют между собой различные компоненты системы после проведенного модульного тестирования, используя следующие подходы:

– восходящий (сначала собираются воедино и тестируются низкоуровневые модули, а затем постепенно добавляются модули более высокого уровня);

– нисходящий (в первую очередь тестируются высокоуровневые модули, а затем добавляются все низкоуровневые);

– комплексный, называемый также «большой взрыв» (модули всех уровней собираются воедино и тестируются).

JUnit, TestNG, NUnit – это те инструменты, которые принявшие в опросе компании используют в описанном выше процессе модульного и интеграционного тестирования приложений чаще всего. В организации интеграционного тестирования в различных компаниях связаны в основном с различиями в платформе разработки, например JUnit для разработки на Java, NUnit для разработки на .Net, TestNG используется с Java, но при этом обладает большей функциональностью чем Junit.

Функциональное тестирование. 52,9% опрошенных компаний используют ручное и автоматизированное тестирование. Одни компании сначала тестируют вручную, а затем повторное тестирование осуществляют автоматизированно. А другие – наоборот: сначала осуществляют автоматизированное тестирование, а затем повторные тесты осуществляют вручную. Для тестирования инженеры по качеству пишут тест-кейсы, далее по ним происходит тестирование. Обнаруженные ошибки записываются в «баг-репорт».

35,3% компаний, среди которых есть маленькие компании, которые тестируют небольшие сайты или мобильные приложения, осуществляют тестирование исключительно вручную. Для этого тестировщики открывают сайт/приложение на устройстве и начинают тестировать функциональность в соответствии с тестовыми сценариями.

11,8% опрошенных компаний занимаются функциональным тестированием только с помощью автоматизации. Для этого инженеры по качеству пишут тест-кейсы, далее по ним пишут автоматические тесты.

При тестировании функциональности используются следующие инструменты: Microsoft Visual Studio, Selenium Web Driver, Selenium IDE.

Selenium IDE – плагин Chrome и Firefox, который используется для быстрого прототипирования тестовых примеров в Selenese/HTML, который также может экспортировать тестовые примеры на разных языках программирования.

WebDriver – это API для программирования тестов на разных языках в Selenium, который может работать с сервером или без него.

Тестирование совместимости. Большинство опрошенных компаний, которые занимаются тестированием совместимости, осуществляют его ручным способом. Для этого они используют различные браузеры, устройства, операционные системы.

Малая часть компаний при тестировании использует частичную автоматизацию, которая выполняется с помощью инструмента TestingBot.

Облако TestingBot позволяет запускать тесты для веб-приложения на более чем 1500 различных комбинациях браузеров и ОС.

Для тестирования совместимости мобильных приложений используется приложение Spoon – фреймворк, который умеет делать скриншоты с экрана устройства или эмулятора во время выполнения тестов и в результате создает из них отчёт.

Тестирование локализации. Данный вид тестирования осуществляют не все компании, потому что не все в нем нуждаются. Компании, которые его осуществляют – тестируют вручную. Для этого сравнивают правильность перевода, расположения и работы кнопок для разных языков. Одна из опрошенных компаний для данного вида тестирования использует аутсорсинг.

Тестирование интерфейса. Лишь одна компания из числа опрошенных производит данный вид тестирования с помощью А/Б тестирования, это метод маркетингового исследования, суть которого заключается в том, что контрольная группа элементов сравнивается с набором тестовых групп, в которых один или несколько показателей были изменены, для того, чтобы выяснить, какие из изменений улучшают целевой показатель.

При тестировании интерфейса инженер тестирования прежде всего обращает внимание на следующие элементы интерфейса ПО: заголовки, тексты, отображаемые ссылки, посадочные страницы или отдельные их элементы, быстрые ссылки, уточнения, изображения и пр.

Лишь один из участников опроса использует автоматизацию при организации тестирования интерфейса. Специфика проектов этой компании заключается в том, что расположение элементов интерфейса всегда известно заранее.

Остальные компании, участники опроса, используют исключительно ручное тестирование интерфейса в соответствии с техническим заданием, логикой, удобством расположения кнопок и грамотностью.

Тестирование производительности. Для этого вида тестирования обычно применяется автоматизация. Этот вид тестов используют не все опрошенные компании. При тестировании производительности опрошенные компании используют Apache JMeter. [7]. Тестируемое приложение при помощи этого инструмента нагружается большим количеством виртуальных пользователей, поддерживается запуск сложных сценариев в большом количестве.

Одна из опрошенных компаний использует виртуальные машины для генерации нагрузки на тестируемое приложение.

Повышение эффективности тестирования

Повышение эффективности тестирования происходит с помощью введения нового метода, включающего в себя 6 рекомендаций, разработанных в рамках исследования.

Для повышения эффективности интеграционного тестирования предлагается:

– разделять тесты на те, где нужна интеграция с внешними сервисами, и те, где важна только внутренняя функциональность продукта. По результатам опроса было выявлено, что более 70% опрошенных компаний не осуществляют такого разделения;

– использовать настраиваемое кэширование для получения данных от внешних сервисов. Обычно при тестировании приходится обращаться к одним и тем же частям приложения неоднократно, при этом не всегда есть потребность в обновленных данных, поэтому можно сэкономить время на вызовах внешних сервисов, «закэшировав» результаты предыдущих;

– применять BDD (behavior-driven development) – подход к разработке и тестированию, при котором особое внимание уделяется поведению системы/модуля в терминах заказчика [8].

Для повышения эффективности функционального тестирования

Предлагается использовать «метод серого ящика», совмещающий подходы к тестированию методом «белого ящика» и «черного ящика». При тестировании методом «серого ящика» инженер по качеству так же, как и при тестировании методом «белого ящика», имеет доступ к коду программы, но в процессе тестирования доступ к коду не использует [9].

Повышение эффективности тестирования интерфейса

Было выявлено, что при тестировании интерфейса можно воспользоваться помощью фокус-групп. Фокус-группы – это метод исследования, при котором группа пользователей приглашается принять участие в тестировании пользовательского интерфейса. Как правило, пользователи, которым предлагается оценить удобство интерфейса, являются потенциальными клиентами. Все действия таких пользователей: реакция на определенные элементы интерфейса, их

реплики и эмоции записываются на аудио- и видеоносители для дальнейшего анализа. Главной задачей проведения фокус-групп является улучшение показателя тестирования интерфейса.

Повышение эффективности тестирования производительности

Для того чтобы сократить время, затрачиваемое на тестирование производительности, которое в свою очередь включает нагрузочное тестирование, тестирование стабильности, стресс-тестирование и объемное тестирование, предлагается [10]:

1) добавление нескольких серверов;

2) использование облачных сервисов/ферм;

3) использование кэширования и сжатия данных;

4) использование выделенного сервера баз данных.

Повышение эффективности тестирования совместимости

Для сокращения времени при тестировании совместимости предлагается построить пирамиду кроссплатформенного тестирования и кроссбраузерного тестирования, где в основании будут лежать те платформы (веб и мобильные) и браузеры, которыми пользуются чаще всего [9]. После чего тестировать нужно начинать с нижнего уровня пирамиды.

Повышение эффективности тестирования локализации

Для сокращения времени тестирования локализации предлагается использовать один из инструментов автоматизированного тестирования локализации (большинство опрошенных компаний делают это вручную, что занимает значительное время):

– EggPlant от компании TestPlant;

– Applitools Eyes;

– Visual Studio .NET 2003.

В табл. 1 приведены данные по представленным инструментам.

В рамках предлагаемой рекомендации повышения эффективности тестирования, которая была доработана с учетом и результатов проведенного опроса и дальнейшего интервьюирования инженеров по качеству из различных компаний, изучения примеров локализации нескольких веб-приложений [9, 11, 12], были разработаны несколько правил для сокращения времени тестирования локализации:

1. Инженеры локализации ПО и инженеры по качеству ПО, осуществляющие тестирование локализации, должны свободно владеть языком локализации ПО.

2. Набор ключевых терминов (это могут быть название, слоган и т.д.) можно оставлять без локализации, за счет этого время тестирования локализации заметно сокращается.

3. При разработке ПО следует учитывать то, что перевод текста на другие языки может занимать значительно больше места, чем оригинал текста [9, 11, 13].

Таким образом, с использованием разработанного в рамках представленного исследования метода, можно существенно сократить время тестирования ПО, при этом не ухудшив качество тестирования программного продукта.

Результаты исследования и их обсуждение

Для подсчета ожидаемой экономии времени тестирования ПО, по каждой из предложенных рекомендаций, применялась следующая формула:

$$E = \sum * P_i * e_i, \quad (1)$$

где $\sum$ – количество предложенных рекомендаций;

e_i – процент, который занимает в проекте i-й этап при ожидаемом тестировании;

P_i – ожидаемый процент экономии i-го этапа тестирования.

Ожидаемое сокращение времени по каждой из предложенных рекомендаций приведено в табл. 1.

Для подсчета реальной экономии времени тестирования ПО, отдельно для каждой предложенной рекомендации используется формула:

$$F = \sum * I_i * S_i * L_i, \quad (2)$$

где $\sum$ – количество предложенных рекомендаций;

I_i – индекс использования рекомендации (1 или 0);

S_i – процент экономии при реальном тестировании i-го этапа;

L_i – процент, который занимает в проекте i-й этап при реальном тестировании.

Предложенные рекомендации тестирования внедрены в несколько компаний, принявших участие в исследовании. С помощью формулы (2) произведен расчет результатов для каждой компании и рекомендации по отдельности. Проведен анализ результатов реального внедрения, усреднен и представлен для каждой рекомендации, в табл. 2.

В итоге, при расчете среднего сокращения времени ожидается, что время тестирования веб-приложения сократится, используя все рекомендации примерно 8,92 %, при этом качество тестирования улучшится.

Таблица 1

Функции рассматриваемых инструментов для тестирования программного обеспечения

	EggPlant(TestPlant)	ApplitoolsEyes	VisualStudio .NET
Количество языков более 100	+	+	–
Бесплатный	–	–	+
Когнитивное зрение	+	+	–
Распознавание текста	+	+	–
Поддерживаемые языки	Java C # Ruby	C # Java PHP Python Ruby	C++ C #
Система отчетов	+	+	+
ОС	Windows Linux Mac	Windows	Windows

Таблица 2

Анализ результатов ожидаемого сокращения времени тестирования и реального внедрения разработанного метода сокращения времени тестирования ПО

Вид тестирования	Ожидаемое сокращение времени тестирования	Реальное усредненное сокращение времени тестирования
Интеграционное	1,2%	6%
Функциональное	3,7%	5,2%
Интерфейса	–2,86% (увеличение времени)	–2,7% (увеличение времени)
Производительности	1,23%	4,1%
Совместимости	3,9%	4%
Локализации	1,75%	1%

При расчете реального сокращения времени получен результат: новый метод тестирования сокращает время тестирования в среднем на 17,6% на проектах, принявших участие в апробации.

Выводы

1. Разработан метод, включающий в себя 6 рекомендаций, который позволяет сократить время тестирования программного продукта, используя описанные выше стратегии.

2. Проведено внедрение, в результате которого получен следующий результат: новый метод сокращает время тестирования в среднем на 17,6%. Сокращение времени тестирования измерено на основе данных, полученных в результате опроса, и оценки эффективности применения нового метода, использования различных техник и инструментов тестирования.

Список литературы

1. Тестирование производительности. [Электронный ресурс]. URL: http://www.nicotech.ru/quality-assurance/performance-testing/. 2019 (дата обращения: 25.05.2019).

2. Copeland L. A Practitioner's Guide to Software Test Design. M.: Artech House, 2004. 314 p.

3. Тестирование с использованием BDD. 2012. [Электронный ресурс]. URL: https://habrahabr.ru/post/139674/ (дата обращения: 21.05.2019).

4. Тестирование при TDD и BDD. 2015. [Электронный ресурс]. URL: http://blog.bausov.pro/tdd-bdd-tests/) (дата обращения: 18.05.2019).

5. Научите посетителей вашего сайта покупать. 2015. [Электронный ресурс]. URL: http://saas.ru/articles/~provieriaiem-naghruzku-na-sait-i-povyshaiem-ieghootkazoustoichivost~3942 (дата обращения: 15.05.2019).

6. Дерягин Д.А. Метод генерации регрессионных модульных тестов // Молодежный научно-технический вестник. 2014. № 4. С. 18–21.

7. Abdulmajeed A., Sonal M., William G. Detecting and Localizing Internationalization Presentation Failures in Web Applications. 2016. vol. 8. no 21. P. 963–970. DOI: 10.1109/ICST.2016.36.

8. Ruiz A. Test-Driven GUI Development with TestNG and Abbot. 2007. vol. 65. no. 9457101 P. 51–57.DOI: 10.1109/MS.2007.92.

9. Шакирова А.И. Сокращение времени автоматизированного тестирования ПО. 2017. [Электронный ресурс]. URL: https://kpfu.ru/student_diplom/10.160.178.20_9227869_SHakirova_A.I._308.pdf (дата обращения: 21.05.2019).

10. Halili E. Apache Jmetr. M.: Packt, 2008. 141 p.

11. Калинов А.Я., Косачёв А.С., Посыпкин М.А., Соколов А.А. Автоматическая генерация тестов для графического пользовательского интерфейса по UML диаграммам действий // Труды Института системного программирования РАН. 2004. Т. 8. № 1. С. 99–116.

12. Harman M., Jia Y. and Zhang Y. Achievements, open problems and challenges for search based software testing. 2015. no. 15111242. P. 18–30. DOI: 10.1109/ICST.2015.7102580.

13. Hak J., Winckler M., Navarre D. PANDA: Prototyping using ANnotation and Decision Analysis. 2016. no. 15111318. P. 171–176. DOI: 10.1145/2933242.2935873.