

Научная статья

УДК 004.415.53

DOI 10.35266/1999-7604-2024-2-7



## РАЗРАБОТКА МЕТОДИКИ ВЫЯВЛЕНИЯ СЛЕПЫХ ЗОН ПРИ ТЕСТИРОВАНИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

**Андрей Олегович Никулин***Пензенский государственный университет, Пенза, Россия**nikulin.science@gmail.com*

**Аннотация.** В статье предложена методика, которая позволит выявить слепые зоны при тестировании программного обеспечения. Описаны общеизвестные проблемы в тестировании ПО и методики их решения. Описан способ, при помощи которого удастся выявить ту или иную проблему. Также продемонстрирован пример реализации этого способа.

**Ключевые слова:** тестирование, тестирование программного обеспечения, регрессионное тестирование, проблемы тестирования

**Для цитирования:** Никулин А. О. Разработка методики выявления слепых зон при тестировании программного обеспечения // Вестник кибернетики. 2024. Т. 23, № 2. С. 57–61. DOI 10.35266/1999-7604-2024-2-7.

Original article

## DEVELOPMENT OF A METHOD FOR IDENTIFYING BLIND SPOTS IN SOFTWARE TESTING

**Andrei O. Nikulin***Penza State University, Penza, Russia**nikulin.science@gmail.com*

**Abstract.** The article proposes a method that will help identify blind spots in software testing. Common problems in software testing and methods for their solution are described. A method to detect a problem is presented. The study also demonstrates an example of implementing this method.

**Keywords:** testing, software testing, regression testing, testing problems

**For citation:** Nikulin A. O. Development of a method for identifying blind spots in software testing. *Proceedings in Cybernetics*. 2024;23(2):57–61. DOI 10.35266/1999-7604-2024-2-7.

### ВВЕДЕНИЕ

Процесс тестирования программного обеспечения основан на одном из фундаментальных принципов – тестирование демонстрирует наличие дефектов, а не их отсутствие [1]. Существование этого принципа формирует одну из важных проблем тестирования программного обеспечения, которая заключается в том, что неясно, каким универсальным способом можно сократить появление ошибок или дефектов.

Сейчас существует большое количество источников, где выделяются проблемы в тестировании программного обеспечения и методики их решения [2, 3]. Главным недостатком является то, что заявленные методики могут решить только конкретную проблему, которая уже известна. Поэтому возникает потребность в методике, которая поможет определить неизвестную проблему.

Чтобы решить эту проблему, была разработана методика, которая позволит выявить

слепые зоны при тестировании программного обеспечения для дальнейшего анализа и устранения, если это будет необходимо и возможно.

## МАТЕРИАЛЫ И МЕТОДЫ

Тестирование программного обеспечения – это испытание, при котором производится проверка соответствия продукта заявленным требованиям [4]. Часто после проведения тестирования обнаруживаются ошибки или дефекты, которые могут возникнуть из-за различных неучтенных случаев.

В рамках данной работы предлагается рассмотреть некоторые проблемы, которые могут способствовать появлению ошибок или дефектов в программном обеспечении.

### Нехватка ресурсов

Данная проблема связана с нехваткой программных или человеческих ресурсов. Чтобы избежать возникновения первого случая, следует обращать внимание на срок действия лицензий у систем управления тестированием и оценивать, какие затраты могут появиться в будущем. Если не производить учет, то через какое-то время может возникнуть нехватка бюджета, и процесс тестирования столкнется с программными ограничениями или полным отсутствием доступа к системам управления тестированием. Также может возникнуть нехватка человеческих ресурсов, что способствует замедлению или полной остановке всего процесса разработки программного обеспечения [2].

## Изменения

С каждой новой итерацией программное обеспечение меняется, поэтому инженеры по тестированию производят постоянное обновление тестовой документации [2], чтобы избежать случаев, когда, например, тестовые сценарии могут оказаться неактуальными. Учитывать эту проблему важно, так как на каждую итерацию отводится определенное количество времени и ресурсов.

### Регрессионное тестирование

Регрессионное тестирование иногда может приводить к такой проблеме, как «регрессионная спираль смерти». Рост функционала с каждой итерацией приводит к увеличению объема регрессионного тестирования, и в этом случае у команды тестирования остается все меньше и меньше времени на другие виды тестирования (представлено на рис. 1), поэтому регрессионное тестирование часто автоматизируют [5, 6].

Также важно отметить, что итеративность обуславливается популярным подходом к разработке программного обеспечения – Agile [7].

В результате были перечислены только некоторые проблемы, которые существуют в процессе тестирования программного обеспечения, а также методики их решения. Понимание самой проблемы дает возможность подобрать методику для ее решения, но проблема не всегда может быть известна, и по-

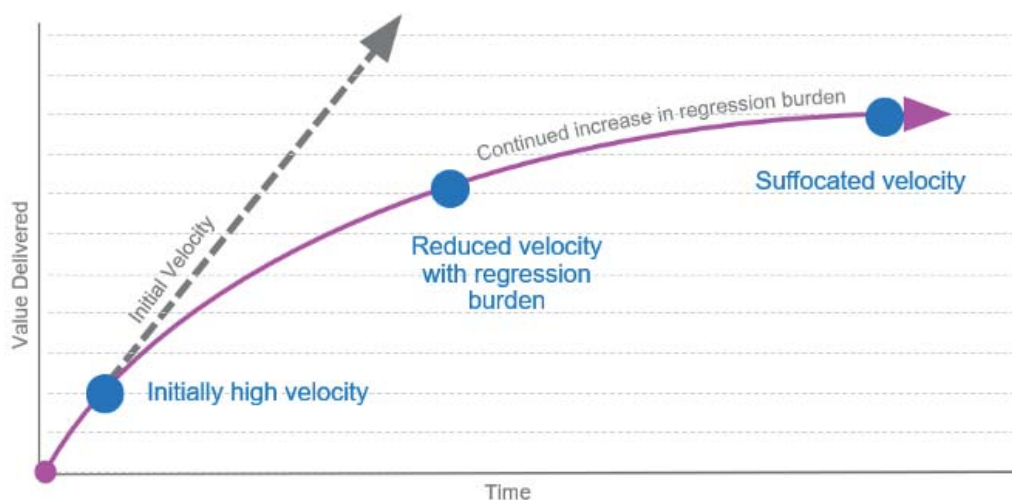


Рис. 1. Регрессионная нагрузка  
Примечание: составлено по [5].

является сложность в ее обнаружении. Для этого была разработана методика, которая позволит выявить слепые зоны при тестировании программного обеспечения.

## РЕЗУЛЬТАТЫ И ИХ ОБСУЖДЕНИЕ

В данном разделе будет представлена разработанная методика, которая позволит уменьшить появление различных ошибок или дефектов в программном обеспечении.

Перед демонстрацией следует выделить требование, без которого методика будет недостаточно эффективной. Суть заключается в том, что процессы на проекте должны быть хорошо выстроены, иначе происхождение большинства найденных ошибок или дефектов можно будет узнать сразу.

Применение методики следует начать с выбора уже исправленных ошибок или дефектов, таким образом, сразу можно избавиться от случаев, когда такого типа тикеты не воспроизводятся, дублируются, специально пропущены или неактуальны. В качестве визуализации будем использовать приложение YouTrack, которое позволяет совершать совместную работу над проектами, управлять задачами, ошибками, дефек-

тами и другими сущностями (представлено на рис. 2) [8].

Также представим небольшую платформу, которая хранит в себе различные документы. Она включает страницу, на которой можно загрузить документы, страницу с общей информацией о платформе и страницу контактов. Тестировщики обнаружили 2 дефекта и 2 ошибки в ней, затем через некоторое время проблемы были устранены, протестированы, и тикеты попали в колонку «Исправлена» (представлено на рис. 2). Таким образом, мы можем перейти к следующему шагу нашей методики.

Начинаем определять критерий, по которому произойдет выбор интересующих ошибок или дефектов. После отбора различных тикетов каждый из них помещается в таблицу для дальнейшего сравнения. Также таблица будет включать разные общеизвестные проблемы, и если потребуется, то можно добавлять собственные. Для примера будут использованы три проблемы, которые были описаны ранее:

1. Нехватка ресурсов.
2. Изменения.
3. Регрессия.

После того как в таблицу внесены ошибки, дефекты и проблемы, то теперь при помощи

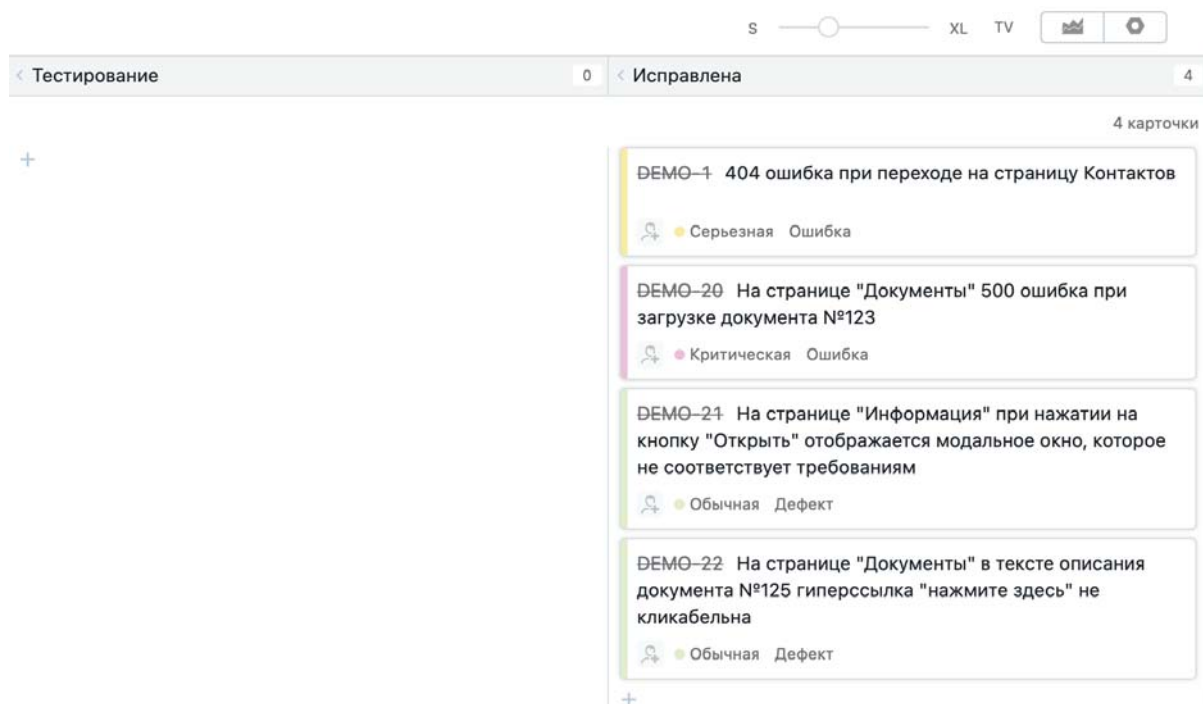


Рис. 2. Доска в YouTrack с ошибками и дефектами  
Примечание: составлено автором.

анализа исправленного тикета следует установить причину его возникновения и на пересечении отметить знаком “+”, если ошибка или дефект не относится к другим проблемам, то на их пересечении будет знак “–”. Также важно упомянуть, что ошибка или дефект может относиться сразу к нескольким проблемам.

Рассмотрим подробнее тикет, где была исправлена ошибка 404 (представлено на рис. 2). В одну из итераций разработки выпускался функционал, связанный со страницей Контактов, на проекте было 2 тестировщика, и планировалось, что тестированием будет заниматься каждый из них. Настал день, когда нужно начинать проверять функционал, и выяснилось, что второй тестировщик не сможет проводить тестирование, и задача полностью ляжет на первого тестировщика. За день до релиза тестировщик поделился результатами, в которых указал, что чек-лист был пройден на 80 %, и по итогу было решено игнорировать оставшиеся 20 % проверок и выпускать функционал для пользователя, так как казалось, что все основные кейсы уже пройдены. Спустя некоторое время после релиза пользователями была обнаружена ошибка 404 при переходе на страницу Контактов. Но если бы тестировщик прошел на 100 % чек-лист, то он бы зафиксировал эту ошибку намного раньше пользователя, так как проверка на это поведение находилась в оставшихся 20 % чек-листа.

Таким образом, мы делаем вывод о том, что проблема возникновения данной ошибки заключается в том, что у команды образовалась нехватка человеческих ресурсов и в таблице на пересечении следует отметить знаком “+”, а другие проблемы – знаком “–”, так как с ними это не связано.

Ошибка 500, возникающая на странице «Документы» при загрузке документа № 123, была связана с тем, что чек-лист на проверку регрессии по функциональности на загрузку документов не обновлялся, и в результате этого при тестировании не был учтен случай, где нужно было загрузить документ № 123,

что привело к ошибке 500. В данном случае в таблице следует отметить знаком “+” проблемы, связанные с изменениями и регрессией, а нехватку ресурсов – знаком “–”.

Дефект, где на странице «Информация» при нажатии на кнопку «Открыть» отображалось модальное окно, не соответствующее требованиям. Это обнаружилось при проведении регрессионного тестирования. В таблице следует отметить проблему, связанную с регрессией знаком “+”, а остальные – знаком “–”.

Дефект, где на странице «Документы» в тексте описания документа № 125 гиперссылка «нажмите здесь» не была кликабельна. На новую функциональность был написан чек-лист, но через некоторое время аналитик частично изменил требования, и тестировщик не обновил чек-лист. Но даже если бы тестировщик внес изменения в чек-лист, то он бы все равно не успел протестировать ту часть чек-листа из-за того, что не хватило времени и было решено выпускать функционал, не продолжая тестирование. В таблице следует отметить проблему, связанную с нехваткой ресурсов и изменениями, знаком “+”, а оставшуюся проблему – знаком “–”.

В результате была сформирована таблица, которая позволит определить, удастся ли в будущем избежать возникновения подобных тикетов или эта зона области не может быть изменена по каким-то причинам. Например, если будет решено исправить проблемы, связанные с регрессионным тестированием, то на нескольких последующих итерациях разработки нужно будет применять методику выявления слепых зон тестирования и проверять, что ошибки или дефекты с такой проблемой больше не появляются. В случае если такие тикеты будут обнаруживаться, то следует менять подход к построению процесса регрессионного тестирования.

Таким образом, для каждой ошибки или дефекта мы можем найти причину возникновения. Если применять данную методику на протяжении нескольких итераций, то удастся уменьшить появление различных ошибок или дефектов.

Таблица

## Результаты применения методики выявления слепых зон при тестировании программного обеспечения

Ошибки и дефекты	Проблемы		
	Нехватка ресурсов	Изменения	Регрессия
На странице «Информация» при нажатии на кнопку «Открыть» отображается модальное окно, которое не соответствует требованиям	–	–	+
На странице «Документы» в тексте описания документа № 125 гиперссылка «нажмите здесь» некликабельна	+	+	–
404 ошибка при переходе на страницу Контактов	+	–	–
На странице «Документы» 500 ошибка при загрузке документа № 123	–	+	+

Примечание: составлено автором.

## ЗАКЛЮЧЕНИЕ

В рамках данной статьи была разработана методика выявления слепых зон при тестировании программного обеспечения. Ее суть заключается в анализе уже исправленных ошибок или дефектов для того, чтобы обнаружить проблему, из-за которой они могли возникнуть, и решить ее. Устранение каких-либо проблем позволит в даль-

нейшем снизить появление ошибок или дефектов.

Методика не сможет гарантировать отсутствие ошибок или дефектов, так как это противоречит принципу тестирования, который описывался ранее. Она способна лишь уменьшить их появление, в том числе за счет этого процессы на проекте будут улучшаться с каждой итерацией применения методики.

## Список источников

1. Принципы тестирования: нас 7. URL: <https://habr.com/ru/post/699990/> (дата обращения: 08.02.2024).
2. 16 software testing challenges: How to handle them. URL: <https://www.softwaretestingmaterial.com/software-testing-challenges/> (дата обращения: 05.02.2024).
3. Do you have blind spots in your software testing? URL: <https://www.testrail.com/blog/blind-spots-software-testing/> (дата обращения: 08.02.2024).
4. Software testing. URL: [https://en.wikipedia.org/wiki/Software\\_testing](https://en.wikipedia.org/wiki/Software_testing) (дата обращения: 03.03.2024).
5. Регрессионная спираль смерти. URL: <https://habr.com/ru/companies/otus/articles/512702/> (дата обращения: 03.03.2024).
6. Никулин А. О. Рекомендации к решению проблем регрессионного тестирования // Надежность и качество : тр. междунар. симпозиума, 29 мая–2 июня 2023 г., г. Пенза. Пенза : Изд-во ПГУ, 2023. Т. 1. С. 465–466.
7. Что такое методология Agile? URL: <https://www.atlassian.com/ru/agile> (дата обращения: 05.03.2024).
8. YouTrack. Powerful project management for all your teams. URL: <https://www.jetbrains.com/youtrack/> (дата обращения: 15.02.2024).

## References

1. Printsipy testirovaniia: nas 7. URL: <https://habr.com/ru/post/699990/> (accessed: 08.02.2024). (In Russ.).
2. 16 software testing challenges: How to handle them. URL: <https://www.softwaretestingmaterial.com/software-testing-challenges/> (accessed: 05.02.2024).
3. Do you have blind spots in your software testing? URL: <https://www.testrail.com/blog/blind-spots-software-testing/> (accessed: 08.02.2024).
4. Software testing. URL: [https://en.wikipedia.org/wiki/Software\\_testing](https://en.wikipedia.org/wiki/Software_testing) (accessed: 03.03.2024).
5. The regression death spiral. URL: <https://habr.com/ru/companies/otus/articles/512702/> (accessed: 03.03.2024). (In Russ.).
6. Nikulin A. O. Recommendations for solving regression testing problems. In: *Proceedings of the International Symposium "Nadezhnost i kachestvo"*, May 29–June 2, 2023, Penza. Penza: Publishing House of the Penza State University; 2023. Vol. 1. p. 465–466. (In Russian).
7. Chto takoe metodologiya Agile? URL: <https://www.atlassian.com/ru/agile> (accessed: 05.03.2024). (In Russ.).
8. YouTrack. Powerful project management for all your teams. URL: <https://www.jetbrains.com/youtrack/> (accessed: 15.02.2024).

## Информация об авторе

А. О. Никулин – аспирант.

## Information about the author

A. O. Nikulin – Postgraduate.