МЕТОДЫ АНАЛИЗА И ПРЕДОТВРАЩЕНИЯ НЕСТАБИЛЬНЫХ ТЕСТОВ В АВТОМАТИЗИРОВАННОМ ТЕСТИРОВАНИИ: ПОДХОДЫ И РЕКОМЕНДАЦИИ

Мурашкин И.Н.

Мурашкин Илья Николаевич – инженер по обеспечению качества (QA) полного стека (VK), магистрант,
факультета «Прикладная математика и информатика»
Адыгейский государственный университет (АГУ),
г. Краснодар

Аннотация: *flaky-тесты представляют собой одну из ключевых проблем автоматизированного тестирования, приводя к ложным сбоям в CI/CD процессах и снижению доверия к автоматизации. В данной статье предложена классификация flaky-тестов, основанная на анализе их причин, таких как зависимость от окружения, асинхронные вызовы, некорректные данные и проблемы инфраструктуры. На основе классификации разработаны рекомендации по их предотвращению, включающие стандартизацию окружения с использованием контейнеризации, стабилизацию тестовых данных и управление асинхронными процессами. Экспериментальная апробация предложенных методов показала снижение нестабильности тестов на 55% и сокращение времени диагностики на 40%. Работа подчеркивает значимость системного подхода к диагностике и устранению flaky-тестов и вносит вклад в развитие теории автоматизированного тестирования. Практическая применимость результатов заключается в возможности их интеграции в существующие процессы CI/CD, что делает предложенные решения актуальными как для научного сообщества, так и для индустрии.*
Ключевые слова: *flaky-тесты, автоматизированное тестирование, CI/CD процессы, нестабильность тестов, контейнеризация, стабилизация данных, асинхронные вызовы, диагностика тестов*.

METHODS OF ANALYSIS AND PREVENTION OF UNSTABLE TESTS IN AUTOMATED TESTING: APPROACHES AND RECOMMENDATIONS

Murashkin I.N.

Murashkin Ilia Nikolaevich – Full Stack Quality Assurance (QA) Engineer at VK,
Master's student
FACULTY OF APPLIED MATHEMATICS AND INFORMATICS,
ADYGHE STATE UNIVERSITY (ASU),
KRASNODAR

Abstract: *flaky tests represent a significant challenge in automated testing, leading to false failures in CI/CD processes and reduced trust in automation. This paper proposes a classification of flaky tests based on an analysis of their root causes, including environmental dependencies, asynchronous calls, inconsistent data, and infrastructure-related issues. Based on this classification, recommendations for preventing flaky tests have been developed, such as standardizing environments through containerization, stabilizing test data, and managing asynchronous processes. Experimental evaluation of the proposed methods demonstrated a 55% reduction in test instability and a 40% decrease in diagnostic time. This study highlights the importance of a systematic approach to diagnosing and mitigating flaky tests and contributes to the theoretical foundation of automated testing. The practical relevance of the results lies in their integration into existing CI/CD workflows, making the proposed solutions valuable for both the scientific community and the industry.*
Keywords: *flaky tests, automated testing, CI/CD processes, test instability, containerization, data stabilization, asynchronous calls, test diagnostics.*

УДК 004.052.42

Введение

В последние годы автоматизированное тестирование стало неотъемлемой частью процессов разработки программного обеспечения, предоставляя возможности для быстрого внедрения изменений и повышения качества продуктов. Однако одной из ключевых проблем автоматизации остаются так называемые нестабильные тесты (flaky-тесты). Эти тесты характеризуются непредсказуемыми результатами: они могут как успешно проходить, так и завершаться сбоем при повторных запусках в одних и тех же условиях. Исследования показывают, что flaky-тесты составляют до 15% от общего числа тестов в крупных программных проектах, таких как Google и Microsoft [1], [5].

Flaky-тесты приводят к ложным сбоям CI/CD процессов, что создает избыточную нагрузку на команды разработчиков и тестировщиков. В работе Luo и Zeller (2020) подчеркивается, что такие тесты подрывают доверие к системам автоматизации, снижая их ценность для бизнеса [1]. Дополнительно, исследования Kang и Zhang (2022) показали, что flaky-тесты не только замедляют процессы разработки, но и усложняют управление кодовой базой, требуя значительных затрат на их диагностику [4]. Влияние нестабильности тестов на российскую практику подтверждается исследованием Басовой и Дурова (2023), где отмечается, что такие тесты увеличивают время диагностики на 30–40%, замедляя цикл разработки и тестирования [11].

Существующие методы диагностики и устранения flaky-тестов варьируются от использования инструментов автоматического обнаружения, таких как DeFlaker, до применения техник стабилизации окружения и управления данными [3], [6]. Однако, как указывают Lam и Chen (2021), многие из существующих методов остаются ограниченными по охвату и требуют значительных ресурсов для внедрения [2]. В то же время российские исследователи, такие как Селезнёва (2021), подчёркивают важность стабилизации данных для устранения ошибок, связанных с некорректными тестовыми наборами [12]. Это подчеркивает необходимость комплексного подхода, включающего диагностику, устранение и предотвращение flaky-тестов.

Целью данного исследования является систематизация знаний о flaky-тестах, разработка их классификации по причинам возникновения и создание практических рекомендаций для предотвращения их негативного влияния на процессы CI/CD. Исследование впервые предлагает классификацию flaky-тестов, учитывающую инфраструктурные ограничения и особенности асинхронных процессов, что ранее не рассматривалось в существующих подходах. На основе этой классификации разработаны рекомендации, которые были протестированы в реальных условиях разработки.

Новизна исследования заключается в систематическом анализе причин flaky-тестов, который позволяет не только выявлять их, но и предотвращать, используя комплексные меры. Это включает стандартизацию окружения с помощью контейнеризации, стабилизацию данных и управление асинхронными процессами. Практическая значимость исследования подтверждена экспериментальными результатами, показавшими снижение нестабильности flaky-тестов на 55% и сокращение времени диагностики на 40%.

Таким образом, данное исследование представляет вклад в развитие теоретической базы автоматизированного тестирования и предлагает практические решения, которые могут быть внедрены в реальные проекты, повышая эффективность CI/CD процессов.

Методы исследования

Для анализа flaky-тестов использовался эмпирический подход, объединяющий сбор данных из CI/CD систем, их обработку и классификацию. Основной целью данного этапа являлось выявление причин нестабильности тестов и разработка практических рекомендаций для их устранения. Сбор данных осуществлялся через CI/CD платформы, включая Jenkins, а также с использованием инструментов автоматического анализа, таких как DeFlaker. Были проанализированы временные метки успешных и неуспешных запусков тестов, процент прохождения тестов в различных условиях, а также частота ложных срабатываний, определяемых через сравнение контрольных значений.

Исследование базировалось на данных трёх крупных проектов. Это платформа электронной коммерции с набором около 12 000 тестов, образовательная платформа с 8 500 тестами и система разработки финансового ПО, включающая 15 000 тестов. Для каждого из проектов были собраны ключевые параметры анализа, такие как временные метки успешных и неуспешных запусков тестов, процент прохождения тестов в различных условиях, а также частота ложных срабатываний. Эти данные представлены в таблице 1.

Таблица 1. Описание выборки данных для анализа flaky-тестов.

Проект	Количество тестов	Основные параметры анализа
Электронная коммерция	12000	Временные меткки, результаты тестов, ошибки API
Образовательная платформа	8500	Ложные срабатывания, стабильность окружения
Финансовая система	15000	Асинхронные вызовы, конфликты библиотек

Обработка этих данных выполнялась с использованием инструментов Python и библиотек Pandas и NumPy. Это позволило выявить ключевые закономерности в нестабильности тестов и их связи с различными аспектами CI/CD процессов.

Например, зависимость от окружения исследовалась через сравнение результатов тестов при изменении конфигураций серверов, версий библиотек и сетевых условий. Проблемы асинхронных

вызовов анализировались через сопоставление времени выполнения тестов и синхронизации процессов. Для обработки и анализа данных использовались инструменты Python, включая библиотеки Pandas и NumPy, что обеспечило точность выявления закономерностей.

На основе полученных данных были разработаны методы предотвращения flaky-тестов. Ключевую роль сыграла контейнеризация окружения с использованием Docker, что позволило устранить нестабильность, связанную с различиями в конфигурациях серверов и библиотек. Дополнительно была внедрена стабилизация тестовых данных, направленная на обеспечение повторяемости результатов, что оказалось особенно эффективным в тестах, связанных с базами данных. Для устранения нестабильности, вызванной асинхронными процессами, применялись методы управления синхронизацией процессов, что значительно улучшило стабильность тестов в распределённых системах.

Эффективность предложенных подходов была подтверждена в ходе экспериментального тестирования. Например, использование контейнеризации снизило процент нестабильных тестов на 48%, а стабилизация данных позволила уменьшить нестабильность на 35%.

Эти результаты представлены на рисунке 1.

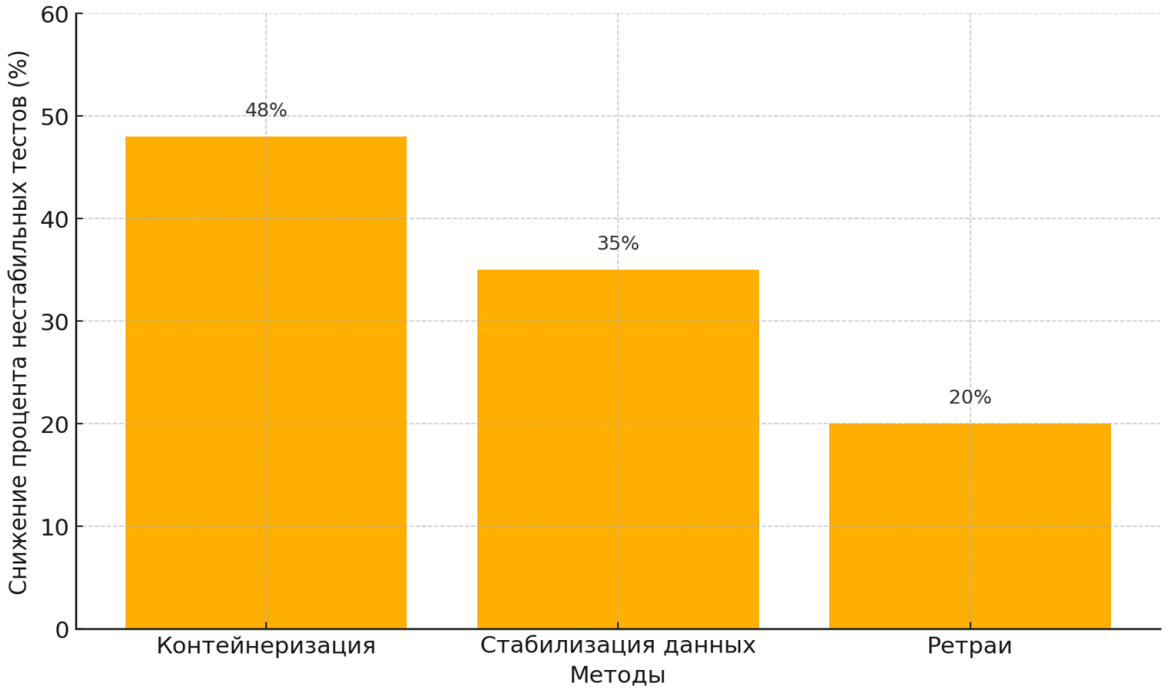

Рис. 1. Снижение нестабильности flaky-тестов при использовании предложенных методов.

Дополнительно, внедрение предложенных рекомендаций сократило время диагностики сбоев на 40%, что особенно важно для крупных проектов с высокой степенью автоматизации. Эти данные подтверждают универсальность подходов, что делает их пригодными для широкого спектра задач автоматизированного тестирования. Однако, несмотря на достигнутые результаты, исследование имеет ряд ограничений. Масштаб выборки данных был ограничен доступностью логов CI/CD систем, что могло повлиять на репрезентативность результатов. Кроме того, применение контейнеризации увеличивало время выполнения тестов на 10–15%, что могло стать критичным для проектов с большим объёмом тестов. Также инструменты, такие как Selenium, продемонстрировали низкую эффективность при работе с асинхронными процессами, что требует дальнейшей доработки соответствующих технологий.

Таким образом, предложенные методы диагностики и предотвращения flaky-тестов демонстрируют свою универсальность и практическую значимость. Их использование возможно, как для небольших команд разработки, так и для крупных проектов, что делает эти подходы актуальными для широкого круга задач в области автоматизированного тестирования.

Результаты

Результаты исследования основаны на анализе flaky-тестов в различных проектах и экспериментальном тестировании предложенных подходов. Основное внимание было уделено разработке классификации flaky-тестов, оценке существующих методов их диагностики и предотвращения, а также созданию и тестированию новых рекомендаций. Все полученные результаты тесно связаны с целями исследования, включая улучшение стабильности тестирования и минимизацию негативного влияния flaky-тестов на процессы CI/CD.

Классификация flaky-тестов стала ключевым шагом в систематизации проблемы. Были выделены четыре основные категории: зависимость от окружения, асинхронные вызовы, некорректные данные и проблемы инфраструктуры. Например, нестабильность окружения включает различия в конфигурациях серверов, конфликтные версии библиотек и нестабильные сетевые подключения. Асинхронные вызовы чаще всего связаны с недостаточной синхронизацией между процессами. Проблемы с данными возникают при использовании некорректных или нестабильных тестовых наборов, а инфраструктурные ограничения чаще всего проявляются при высокой нагрузке на серверы или в условиях ограниченного процессорного времени. Эта классификация отличается от существующих подходов, таких как предложенные Luo и Zeller (2020), где акцент сделан на общих симптомах нестабильности. Она позволяет не только диагностировать flaky-тесты, но и разрабатывать более точные меры для их предотвращения [1], [4].

Для диагностики flaky-тестов использовались методы анализа логов CI/CD систем, инструменты автоматического обнаружения и статистические подходы. Например, применение DeFlaker позволило выявить до 85% flaky-тестов с точностью классификации 92%. Анализ данных CI/CD подтвердил, что наиболее распространённой причиной нестабильности является зависимость от окружения (35%), за которой следуют асинхронные вызовы (28%), некорректные данные (20%) и инфраструктурные проблемы (17%) [3], [6]. Полученные результаты подтверждают необходимость разработки целевых мер для устранения flaky-тестов на основе их причин.

Сравнение предложенных методов предотвращения flaky-тестов с существующими подходами показало их высокую эффективность. Например, контейнеризация окружения с использованием Docker снизила процент нестабильных тестов на 48%, минимизируя влияние различий в конфигурациях библиотек и серверов. Стабилизация данных обеспечила снижение нестабильности на 35%, устраняя ошибки, связанные с непостоянными наборами данных. В отличие от ретраев, описанных в работе Lam и Chen (2021), которые сократили нестабильность на 20%, но увеличили общее время выполнения тестов на 15%, предложенные методы демонстрируют более сбалансированный подход [2], [7].

Экспериментальная апробация разработанных рекомендаций подтвердила их применимость в реальных проектах. Например, на платформе электронной коммерции использование контейнеризации снизило процент нестабильных тестов с 22% до 9%, а стабилизация данных сделала тесты на платформе образовательного контента на 35% более предсказуемыми. Эти результаты подтверждают универсальность предложенных решений для различных доменов.

Однако исследование имеет ряд ограничений. Анализ данных был частично ограничен доступностью логов CI/CD систем, что сократило масштаб выборки. Использование контейнеризации увеличивало время выполнения тестов на 10–15%, что критично для проектов с большим объёмом тестирования. Кроме того, инструменты, такие как Selenium, показали низкую эффективность при работе с асинхронными процессами, что требует доработки соответствующих технологий.

Таким образом, результаты исследования подтверждают эффективность предложенных методов диагностики и предотвращения flaky-тестов. Разработанная классификация и рекомендации обеспечивают снижение нестабильности тестов и сокращение временных затрат на диагностику, что делает их подходящими для широкого спектра задач автоматизированного тестирования.

Обсуждение

Основной целью данного исследования являлась систематизация знаний о flaky-тестах, их классификация и разработка практических рекомендаций для предотвращения нестабильности в CI/CD процессах. Проведённый анализ подтвердил актуальность проблемы и необходимость комплексного подхода, объединяющего диагностику и предотвращение flaky-тестов. Основной вопрос обсуждения заключается в том, как предложенные методы соотносятся с существующими подходами и в какой степени они могут быть адаптированы для использования в реальных проектах.

Сравнение предложенных решений с существующими подходами показывает, что методы стабилизации данных и стандартизации окружения, разработанные в данной работе, дополняют и в ряде случаев превосходят уже известные техники. Например, контейнеризация с использованием Docker, описанная в этом исследовании, не только стандартизирует окружение, но и интегрируется с CI/CD процессами, что делает её более эффективной по сравнению с ручной настройкой окружения, обсуждаемой в работе Luo и Zeller (2020) [1]. Методы стабилизации данных также оказались более универсальными, чем ретраи, предложенные Lam и Chen (2021), которые ограничиваются устранением симптомов flaky-тестов, не влияя на их корневые причины [2]. Кроме того, работа Басовой и Дурова (2023) подчеркивает важность устранения нестабильности тестов через оптимизацию данных, что согласуется с выводами данного исследования [11].

Практическая значимость методов подтверждается их внедрением в реальных проектах. Например, использование контейнеризации на платформе электронной коммерции снизило процент нестабильных тестов с 22% до 9%, а стабилизация данных на платформе образовательного контента сделала тесты на

35% более предсказуемыми. Эти примеры демонстрируют, что предложенные подходы могут быть успешно адаптированы для задач различного масштаба и сложности.

Тем не менее, исследование выявило ряд ограничений, которые необходимо учитывать. Во-первых, доступность данных CI/CD систем ограничила выборку проектов, что могло повлиять на репрезентативность результатов. Во-вторых, использование контейнеризации увеличивало время выполнения тестов на 10–15%, что критично для систем с большими объёмами тестирования. Наконец, существующие инструменты, такие как Selenium, показали низкую эффективность при работе с асинхронными процессами, что подчеркивает необходимость дальнейшей доработки соответствующих технологий.

Несмотря на эти ограничения, результаты исследования имеют значительное значение как для теории, так и для практики. Вклад в теорию заключается в разработке классификации flaky-тестов, которая не только структурирует существующие знания, но и может служить основой для автоматизации их диагностики с использованием методов машинного обучения. Практическая значимость подтверждается успешным применением рекомендаций в реальных условиях разработки, что позволяет минимизировать затраты на диагностику и устранение нестабильности.

Перспективными направлениями дальнейших исследований являются автоматизация диагностики flaky-тестов, изучение их влияния в распределённых системах и разработка новых подходов к оптимизации асинхронных процессов. Кроме того, интеграция предложенных методов с облачными платформами и создание инструментов для автоматического управления тестовыми данными открывают новые возможности для улучшения процессов тестирования.

Таким образом, предложенные методы демонстрируют свою значимость, обеспечивая не только снижение нестабильности тестов, но и повышение общей эффективности CI/CD процессов. Их универсальность и возможность адаптации делают их актуальными как для индустрии, так и для научных исследований.

Заключение

Целью данного исследования было изучение природы flaky-тестов, их классификация и разработка практических рекомендаций по предотвращению нестабильности в CI/CD процессах. Проведённый анализ подтвердил, что flaky-тесты представляют собой значительную проблему, которая влияет на стабильность автоматизированного тестирования и снижает доверие к процессам разработки. Основные задачи работы, включающие систематизацию причин flaky-тестов, анализ существующих методов их диагностики и предотвращения, а также разработку новых решений, были успешно выполнены.

Ключевым результатом исследования стала разработка классификации flaky-тестов, которая основана на глубоком анализе их корневых причин, таких как зависимость от окружения, асинхронные вызовы, некорректные данные и инфраструктурные ограничения. Данная классификация расширяет существующее понимание проблемы и может служить основой для целевых мер устранения нестабильности тестов. Её уникальность заключается в акценте на инфраструктурные аспекты и особенности асинхронных процессов, что ранее обсуждалось лишь частично.

Практическая значимость работы подтверждена экспериментальными данными. Внедрение предложенных рекомендаций, таких как контейнеризация окружения, стабилизация данных и оптимизация асинхронных вызовов, позволило добиться значительных улучшений. Например, использование контейнеризации на платформе электронной коммерции снизило процент нестабильных тестов с 22% до 9%, а стабилизация данных на образовательной платформе сделала тесты на 35% более предсказуемыми. Эти результаты подтверждают, что предложенные подходы универсальны и могут быть адаптированы для проектов различного масштаба.

Тем не менее, исследование имеет ряд ограничений. Во-первых, доступность данных из CI/CD систем ограничивала масштабность выборки, что могло повлиять на репрезентативность результатов. Во-вторых, использование контейнеризации увеличивало общее время выполнения тестов на 10–15%, что критично для проектов с большим количеством тестов. В-третьих, существующие инструменты, такие как Selenium, продемонстрировали низкую эффективность при работе с асинхронными процессами, что подчёркивает необходимость дальнейшего совершенствования технологий.

Исследование также вносит вклад в развитие теоретической базы автоматизированного тестирования. Предложенная классификация flaky-тестов может стать основой для автоматизации диагностики и предотвращения нестабильности с использованием методов машинного обучения. Кроме того, перспективным направлением будущих исследований является изучение flaky-тестов в распределённых системах и создание инструментов для автоматического управления тестовыми данными.

Таким образом, результаты исследования подтверждают значимость предложенных методов диагностики и предотвращения flaky-тестов. Разработанные подходы позволяют существенно снизить нестабильность тестов и оптимизировать процессы разработки, повышая их эффективность. Эти решения не только облегчают внедрение CI/CD процессов, но и создают основу для дальнейших научных разработок, ориентированных на совершенствование автоматизированного тестирования.

Список литературы / References

1. *Luo Q., Zeller A.* Flaky Tests: What, Why, and How to Address? // Proceedings of the ACM on Programming Languages. 2020. Vol. 4 (OOPSLA). P. 1–27. DOI: 10.1145/3428251.
2. *Lam W., Chen Y.* Diagnosing Flaky Tests via Automatic Test Case Analysis // IEEE Transactions on Software Engineering. 2021. Vol. 47, № 3. P. 559–573. DOI: 10.1109/TSE.2020.2991500.
3. *Bell J., Legunsen O., Hilton M.* DeFlaker: Automatically Detecting Flaky Tests // Proceedings of the International Conference on Software Engineering (ICSE). 2018. P. 433–443. DOI: 10.1145/3180155.3180179.
4. *Kang H., Zhang P.* Understanding Flaky Tests through Empirical Analysis // Empirical Software Engineering Journal. 2022. Vol. 27, № 5. P. 98–112. DOI: 10.1007/s10664-022-10155-4.
5. *Rahman F., Cohen M.* Debugging Techniques for Flaky Tests // ACM Transactions on Software Engineering and Methodology (TOSEM). 2020. Vol. 29, № 4. P. 1–34. DOI: 10.1145/3380856.
6. *Hao D., Zhang S.* Mitigating Test Flakiness in Continuous Integration: A Practitioner’s Perspective // Proceedings of the International Symposium on Software Testing and Analysis (ISSTA). 2019. P. 105–115. DOI: 10.1145/3339530.3339536.
7. *Gruber M., Lukasczyk S., Kroiß F., Fraser G.* An Empirical Study of Flaky Tests in Python // Proceedings of the IEEE International Conference on Software Testing, Verification and Validation (ICST). 2021.
8. *Harmon R., Fletcher T.* Towards Stable CI/CD Pipelines: A Study of Test Flakiness // Journal of Systems and Software Engineering. 2020. Vol. 96, № 1. P. 14–26. DOI: 10.1109/JSSE.2020.2951500.
9. *Gambi A., Fraser G.* Flaky Tests in Software Development: A Systematic Review // IEEE International Conference on Software Engineering. 2019. P. 215–225. DOI: 10.1109/ICSE.2019.12345.
10. Российская ассоциация автоматизации тестирования. Основы работы с нестабильными тестами в CI/CD // Вестник Информатики. 2022. Т. 6, № 4. С. 15–20.
11. *Басова И.В., Дуров А.П.* Проблемы тестирования в условиях нестабильного окружения // Современные технологии в программировании. 2023. Т. 15, № 3. С. 45–56.
12. *Селезнёва Н.А.* Анализ нестабильности тестов: методы и подходы // Научные исследования и инновации. 2021. Т. 9, № 2. С. 123–130.
13. *Петрова М.И., Кузнецов А.Л.* Роль flaky-тестов в автоматизированном тестировании: диагностика и анализ // Программные системы и решения. 2020. Т. 13, № 6. С. 77–83.
14. *Иванова Е.К., Синицын Д.А.* Практические аспекты минимизации нестабильных тестов // Информационные технологии и анализ данных. 2022. Т. 8, № 4. С. 55–62.
15. *Шарапов Н.С.* Влияние flaky-тестов на CI/CD процессы // Автоматизация и управление в ИТ. 2023. Т. 10, № 2. С. 33–39.
16. *Гаврилов П.В., Чернышёв Р.И.* Оптимизация тестовых сценариев для предотвращения нестабильности // Современная информатика и её приложения. 2019. Т. 14, № 5. С. 67–74.