

Метрики качества: как измерить эффективность и надёжность автоматизации тестирования

Д.О. Кочетов

Национальная система платежных карт, Москва, Россия

Аннотация: В условиях современных процессов непрерывной интеграции и доставки критически важным становится не только наличие автоматизированных тестов, но и их реальная эффективность, надёжность и экономическая целесообразность. В данной работе систематизированы ключевые метрики оценки качества автоматизированного тестирования, с особым акцентом на проблему нестабильных тестов. Введены и обоснованы новые показатели: уровень нестабильных тестов и потери конвейера непрерывной интеграции, которые прямо отражают издержки сопровождения тестовой инфраструктуры. Подробно анализируются ограничения традиционных метрик, в частности покрытия кода, и демонстрируется превосходство мутационного тестирования как более надёжного индикатора способности тестового набора выявлять дефекты. На демонстрационных данных реального непрерывной интеграции - пайплайна выявлены ключевые зависимости: рост покрытия кода не гарантирует улучшение мутационного тестирования и не приводит к увеличению числа обнаруживаемых дефектов; высокая доля нестабильных тестов коррелирует со значительными потерями машинного времени и снижением доверия к результатам тестирования; снижение времени обнаружения и устранения дефектов достигается не только через увеличение покрытия, но и через сокращение доли нестабильных тестов, улучшение наблюдаемости системы и укрепление дисциплины управления дефектами.

Ключевые слова: метрики качества автоматизированного тестирования, мутационное тестирование, нестабильные тесты, покрытие кода, эмпирический анализ метрик, сравнительный анализ метрик тестирования, оптимизация процессов тестирования, экономическая эффективность автоматизации, управление качеством ПО, эмпирические исследования в тестировании.

Современная разработка программного обеспечения характеризуется повсеместным внедрением практик непрерывной интеграции и доставки (Continuous Integration / Continuous Delivery – CI/CD), что обусловило

критическую важность автоматизированного тестирования как неотъемлемого элемента жизненного цикла ПО [1]. Однако сам факт наличия автотестов не гарантирует их действительной эффективности, надежности и экономической целесообразности. Многие команды разработки сталкиваются с ситуацией, когда обширный набор автотестов требует значительных вычислительных и временных затрат на поддержку и исполнение, но при этом не обеспечивает адекватного уровня обнаружения критических дефектов. Это свидетельствует о необходимости перехода от оценки автоматизации тестирования по принципу «есть/нет» к ее глубокому анализу на основе объективных количественных показателей — метрик.

Целью настоящего исследования является разработка комплексного подхода к оценке качества автотестов, учитывающего взаимосвязи между различными категориями метрик и позволяющего принимать обоснованные решения по улучшению процессов автоматизированного тестирования.

Классификация метрик качества автоматизированного тестирования и их ограничения

Современная инженерная практика автоматизированного тестирования требует комплексного подхода к оценке качества [2], учитывающего четыре взаимосвязанных аспекта: полноту проверки (объем протестированного поведения системы), силу тестов (способность выявлять реальные дефекты), стабильность результатов (воспроизводимость и отсутствие ложных срабатываний) и операционные издержки (временные и вычислительные ресурсы на разработку, выполнение и поддержку тестов). На основе анализа современных научных обзоров и эмпирических исследований предлагается систематизация ключевых метрик с единообразными формулами расчета и критической оценкой их ограничений.

1.1. Метрики полноты тестирования

Покрывание кода является фундаментальной метрикой, измеряющей процент исполняемых структур кода, которые были выполнены в процессе тестирования. Данная метрика вычисляется по формуле:

$$CC = (L_exec / L_total) \times 100\%$$

где:

L_exec - количество выполненных (покрытых) элементов кода

L_total - общее количество элементов кода

Выделяют три основных типа покрытия:

Покрывание строк - процент выполненных строк кода

Покрывание ветвей - процент покрытых ветвлений алгоритма

Покрывание условий - процент покрытых логических условий

Ограничения и практическая значимость:

Несмотря на широкое распространение, метрики покрытия имеют существенные ограничения. Они не измеряют силу оракулов (проверочных утверждений) и могут насыщаться "слабыми" проверками, создавая ложное впечатление полноты тестирования. Сравнительные исследования показывают, что покрытие ветвей точнее отражает способность тестов выявлять дефекты по сравнению с покрытием операторов, однако обе метрики уступают по точности мутационным метрикам [3].

1.2. Метрики силы тестов на основе мутационного тестирования

Мутационное тестирование оценивает эффективность тестов путем внесения искусственных дефектов в исходный код и измерения способности тестового набора их обнаруживать. Основная метрика [4]:

$$MS = (K / M) \times 100\%$$

где:

M - количество сгенерированных достижимых мутантов

K - количество "убитых" (обнаруженных) мутантов

Современные методы оптимизации:

Выборочный выбор операторов (Selective Operator Selection – COS) - выбор подмножества операторов мутаций

Случайный выбор мутантов (Random Mutant Selection – RMS) - случайная выборка мутантов для снижения вычислительной сложности

Счетчик поглощающих мутантов (Subsuming Mutation Score – SMS) - учет отношений частичного порядка "поглощения" между мутантами

При использовании реальных дефектов мутационные метрики демонстрируют лучшую корреляцию с фактической эффективностью тестовых наборов. Однако при замене реальных дефектов искусственными мутантами возможна систематическая переоценка эффективности на 20% и более, что подтверждает важность корректного выбора "контрольных данных" в экспериментальных исследованиях [3].

1.3. Метрики стабильности: нестабильные тесты и производные показатели

Нестабильные тесты - тесты, которые демонстрируют непоследовательные результаты без изменений тестируемого кода или самих тестов. Для количественной оценки используются [4]:

$$FTR = (T_flaky / T_total) \times 100\% \text{ \# Доля нестабильных тестов}$$

$APR = (T_pass - T_flaky_pass) / T_exec \times 100\% \text{ \# Скорректированный процент успеха}$

Причинами проблем нестабильности являются: асинхронность, гонки условий, временные зависимости, недетерминированное окружение, внешние зависимости.

Последствиями проблем нестабильности являются: ложные тревоги, снижение доверия к результатам тестирования, задержки в CI/CD-процессах.

2. Эмпирический анализ взаимосвязей между метриками качества тестирования

Многочисленные эмпирические исследования и систематические обзоры свидетельствуют, что простое количественное наращивание тестов или увеличение процента покрытия кода не приводит к линейному улучшению качества выпускаемого программного обеспечения. Анализ выявляет сложные нелинейные зависимости между различными категориями метрик.

2.1. Соотношение полноты и силы тестов

Экспериментальные данные показывают, что рост покрытия кода (Code Coverage – CC) свыше 70-80% часто сопровождается уменьшением предельной полезности. При достижении этого порога дальнейшее увеличение покрытия не приводит к существенному росту способности тестового набора обнаруживать реальные дефекты.

Критически важным является различие между метриками покрытия и мутационным счетом (Mutation Score – MS). Исследования демонстрируют, что мутационные метрики значительно точнее ранжируют тестовые наборы по их действительной эффективности в выявлении дефектов. Однако методология эксперимента играет ключевую роль: при использовании искусственных мутантов вместо реальных дефектов может наблюдаться систематическая переоценка эффективности на 20% и более [3].

Практические рекомендации:

Использовать покрытие кода как "нижнюю границу" полноты тестирования

Основные усилия по улучшению качества направлять на усиление оракулов и увеличение мутационного счета

При проведении экспериментов использовать реальные дефекты в качестве эталонной "истины"

3. Кейсы и шаблоны внедрения метрик качества автоматизированного тестирования

Современные исследования в области обеспечения качества программного обеспечения выявили ряд устойчивых шаблонов внедрения метрик, демонстрирующих эффективность сквозь различных предметных областей и технологических стеков. Эти шаблоны представляют собой проверенные на практике подходы к интеграции метрической системы в процесс разработки и сопровождения программного обеспечения.

Шаблон А: Оптимизация силы тестов при фиксированном уровне покрытия.

Данный подход предполагает поддержание показателя покрытия кода (CC) в определенном целевом диапазоне (обычно 70-80%) с параллельным внедрением мутационного тестирования для повышения силы тестовых сценариев. Внедрение осуществляется поэтапно: начинается с методов выборочной (COS) или случайной (RMS) выборки мутантов, с последующим переходом к использованию субсумирующих мутационных метрик (SMS) для критически важных компонентов системы. Эмпирические исследования подтверждают, что такой подход приводит к значительному росту мутационного счета (MS), сокращению ложно-положительных срабатываний, ускорению обнаружения дефектов (Mean Time To Detect – MTDD) и уменьшению количества откатов релизов. Важным аспектом является использование реальных дефектов в качестве эталонной "истины" при валидации метрик, что позволяет избежать систематической переоценки эффективности тестовых наборов [3].

Шаблон Б: Системный подход к управлению нестабильностью тестов

Данная методика focuses на создании комплексной системы управления нестабильными тестами через формирование централизованного реестра, категоризацию причин нестабильности (временные зависимости, асинхронность, внешние зависимости) и реализацию целевых мер по устранению root-причин. Ключевыми элементами являются внедрение явных

ожиданий вместо жестких таймаутов, изоляция тестового окружения, ограничение количества автоматических ретраев и внедрение специализированных метрик (Flaky Test Rate – FTR, Adjusted Pass Rate – APR) на уровне отдельных компонентов и функциональных блоков. Реализация данного шаблона позволяет значительно сократить количество ложных тревог, уменьшить количество блокировок в CI/CD-пайплайне и очистить очереди дефектов, что в конечном итоге приводит к снижению среднего времени устранения дефектов (Mean Time To Resolve – MTTR) [4].

Представленные шаблоны демонстрируют возможность системного подхода к внедрению метрик качества автоматизированного тестирования и подтверждают их практическую эффективность в различных условиях и на разных типах проектов. Ключевым фактором успеха является не простое внедрение отдельных метрик, а их комплексное использование в рамках единой системы управления качеством, интегрированной в процесс разработки программного обеспечения [6].

Заключение

Система метрик качества автотестов является необходимым инструментом для объективной оценки эффективности и надежности автоматизированного тестирования. Предложенная классификация охватывает ключевые аспекты: полноту покрытия, эффективность тестовых сценариев, скорость реакции на дефекты и экономическую эффективность.

Наиболее ценными для практического применения являются метрики на основе мутационного тестирования, которые точнее отражают эффективность тестов по сравнению с традиционным покрытием кода³. Однако не существует универсальной метрики, и эффективность оценки достигается только за счет сбалансированной системы показателей, адаптированной под цели конкретного проекта.

Литература

1. Фрасын П.Г., Никитин Н.В. // Инженерный вестник Дона, 2025, №7.
URL:
ivdon.ru/uploads/article/pdf/IVD_77N6y25_Frasyn_PG_Nikitin_NV.pdf_96b0e0
2. Евдокимова Т.С., Шлеймович М.П. // Инженерный вестник Дона, 2025, №2. URL:
ivdon.ru/uploads/article/pdf/IVD_58N1y25_evdokimova_Shleymovich.pdf_0c7add4c09.pdf
3. Zhang P., Wang Y., Liu X., Yang Y., Li Y., Chen L., Wang Z., Sun C.-a., Zhou Y. Test suite effectiveness metric evaluation: what do we know and what should we do? arXiv:2204.09165, 2022. 11 p.
4. Papadakis M., Kintis M., Zhang J., Jia Y., Le Traon Y., Harman M. Mutation Testing Advances: An Analysis and Survey // Advances in Computers. 2019. Vol. 112. pp. 275–378.
5. Parry O., Kapfhammer G.M., Hilton M., McMinn P. // ACM Transactions on Software Engineering and Methodology. 2021. Vol. 31, №1. pp. 1–74.
6. Клюканов А.В., Штатнов И.А. Исследование вопросов качества тестирования с использованием метрик и аналитики. Электронный ресурс. 2025. URL: cyberleninka.ru/article/n/issledovanie-voprosov-kachestva-testirovaniya-s-ispolzovaniem-metrik-i-analitiki
7. Барвин С.К., Попов Д.В. Метрики автоматизированного тестирования web-приложения // Современные научные исследования и инновации. - 2019. - № 4. - С. 4-4.
8. Горщар Р.С., Таран В.Н. Алгоритмические подходы для средств аналитики результатов тестирования систем дистанционного обучения // Системы компьютерной математики и их приложения. - 2019. - № 20-2. - С. 180-185.

9. Satapathy S.C. et al. Usage of machine learning in software testing // Automated Software Engineering: A Deep Learning-Based Approach. - 2020. - pp. 39-54.

10. Pecorelli F., Palomba F., De Lucia A. The relation of test-related factors to software quality: a case study on Apache systems // Empirical Software Engineering. - 2021. - T. 26. - pp. 1-42.

References

1. Evdokimova T.S., Shleymovich M.P. Inzhenernyj vestnik Dona, 2025, No. 2. URL: ivdon.ru/uploads/article/pdf/IVD_77N6y25_Frasyn_PG_Nikitin_NV.pdf_96b0e0a82a.pdf

2. Frasin P.G., Nikitin N.V. Inzhenernyj vestnik Dona, 2025, No. 7. URL: ivdon.ru/uploads/article/pdf/IVD_58N1y25_evdokimova_Shleymovich.pdf_0c7add4c09.pdf

3. Zhang P., Wang Y., Liu X., Yang Y., Li Y., Chen L., Wang Z., Sun C.-a., Zhou Y. Test suite effectiveness metric evaluation: what do we know and what should we do? ArXiv: 2204.09165, 2022. 11 p.

4. Papadakis M., Kintis M., Zhang J., Jia Y., Le Traon Y., Harman M. Advances in Computers, 2019, 112, pp. 275–378.

5. Parry O., Kapfhammer G.M., Hilton M., McMinn P. ACM Transactions on Software Engineering and Methodology, 2021, 31(1), pp. 1–74.

6. Klyukanov A.V., Shtatnov I.A. Issledovanie voprosov kachestva testirovaniya s ispol'zovaniem metrik i analitiki [Study of testing quality issues using metrics and analytics]. 2025. URL: cyberleninka.ru/article/n/issledovanie-voprosov-kachestva-testirovaniya-s-ispolzovaniem-metrik-i-analitiki

7. Barvin S.K., Popov D.V. Sovremennye nauchnye issledovaniya i innovatsii. 2019. No. 4. pp. 4-4.



8. Gorshchar R.S., Taran V.N. Sistemy komp'yuternoy matematiki i ikh prilozheniya. 2019. No. 20-2. pp. 180-185.

9. Satapathy S.C. et al. Automated Software Engineering: A Deep Learning-Based Approach. 2020. pp. 39-54.

10. Pecorelli F., Palomba F., De Lucia A. The relation of test-related factors to software quality: a case study on Apache systems // Empirical Software Engineering. 2021. T. 26. pp. 1-42.

Дата поступления: 7.09.2025

Дата публикации: 26.10.2025