

# УПРАВЛЕНИЕ СЛОЖНЫМИ СИСТЕМАМИ

DOI: 10.18137/RNU.V9187.25.04.P.24

УДК 004.054

## **Зарубин Илья Борисович**

старший преподаватель кафедры графических информационных систем, Нижегородский государственный технический университет, Нижний Новгород. ORCID: 0000-0002-4870-6171, SPIN-код: 1012-0050, Author ID: 1058887

Электронный адрес: simarglz@yandex.ru

## **Ilya B. Zarubin**

Senior Lecturer at the Department of graphic information systems, Nizhny Novgorod State Technical University, Nizhny Novgorod. ORCID: 0000-0002-4870-6171, SPIN-code: 1012-0050, Author ID: 1058887

E-mail address: simarglz@yandex.ru

## **Филинских Александр Дмитриевич**

кандидат технических наук, доцент, заведующий кафедрой графических информационных систем, Нижегородский государственный технический университет, Нижний Новгород. ORCID: 0000-0003-3826-6771, SPIN-код: 9015-0430, Author ID: 638501

Электронный адрес: alexfil@yandex.ru

## **Alexander D. Filinskikh**

Ph.D. of Engineering Sciences, Docent, Head of the Department of Graphic Information Systems, Nizhny Novgorod State Technical University, Nizhny Novgorod. ORCID: 0000-0003-3826-6771, SPIN-код: 9015-0430, Author ID: 638501

E-mail address: alexfil@yandex.ru

---

## ФОРМИРОВАНИЕ ВЕСОВЫХ КОЭФФИЦИЕНТОВ МЕТОДИК ОТБОРА ДЛЯ РЕГРЕССИОННОГО ТЕСТИРОВАНИЯ

---

**Аннотация.** В статье рассмотрены способы ранжирования и группировки тестов для формирования тестовых наборов при проведении регрессионного тестирования: по приоритету, длительности выполнения, предыдущему результату, общему количеству найденных ошибок, сроку выполнения, приоритету микросервиса, объему внесенных в микросервис изменений, наличию модификаций в смежных сервисах. Предложен подход, позволяющий формировать актуальные значения весовых коэффициентов для каждого из способов отбора и ранжирования тестов с получением результирующего тестового набора, чтобы тесты, которые позволят обнаружить наиболее важные ошибки в информационной системе, находились в верхней половине списка. Описаны способы применения подхода формирования значений весовых коэффициентов для двух систем управления тестированием. Рассмотрен пример использования предложенного подхода при формировании ранжированного списка тестов для регрессионного тестирования реальной микросервисной информационной системы с использованием автоматизированной системы ранжирования тестов, которая произвольное количество способов ранжирования позволяет подключить к современным системам управления тестированием. Приведены графики потери актуальности весовых коэффициентов и появления пропущенных дефектов в зависимости от полноты выполнения тестового набора.

**Ключевые слова:** регрессионное тестирование, отбор тестов для регрессионного тестирования, ранжирование тестов для регрессионного тестирования, методики формирования тестового набора.

**Для цитирования:** Зарубин И.Б., Филинских А.Д. Формирование весовых коэффициентов методик отбора для регрессионного тестирования // Вестник Российского нового университета. Серия: Сложные системы: модели, анализ, управление. 2025. № 4. С. 24–37. DOI: 24 – 37. DOI: 10.18137/RNU.V9187.25.04.P.24

## FORMATION OF WEIGHTING COEFFICIENTS FOR SELECTION METHODS FOR REGRESSION TESTING

**Abstract.** The article discusses methods for ranking and grouping tests to form test suites during regression testing: by priority, execution duration, previous result, total number of detected errors, deadline, microservice priority, volume of changes made to the microservice, and the presence of modifications in related services. An approach is proposed for generating relevant weighting coefficients for each test selection and ranking method, resulting in a resulting test suite, so that tests capable of detecting the most important errors in an information system are located in the top half of the list. Application methods for generating weighting coefficient values for two test management systems are described. An example of using the proposed approach to forming a ranked list of tests for regression testing of a real microservice information system is considered using an automated test ranking system, which allows any number of ranking methods to be connected to modern test management systems. Graphs are provided for depreciation of weighting coefficients and the appearance of missed defects depending on the completeness of the test suite execution.

**Keywords:** regression testing, selection of tests for regression testing, tests ranking for regression testing, methods of forming a test set.

**For citation:** Zarubin I.B., Filinskikh A.D. (2025) Formation of Weighting Coefficients for Selection Methods for Regression Testing. *Vestnik of Russian New University. Series: Complex Systems: Models, analysis, management*. No. 4. Pp. 24– 37. DOI: 10.18137/RNUV9187.25.04.P.24 (In Russian).

В настоящее время потребность человечества в различных информационных системах (далее – ИС) весьма велика – ИС позволяют значительно упростить и ускорить практически все процессы, которые происходят в человеческом обществе. Широкое использование ИС в жизнедеятельности человека – от визуализации сложных географических показателей, например, составления этнографической карты России, до создания интерактивных и доступных образовательных платформ [1–3] – накладывает высокие требования на корректность работы ИС: компании-разработчики тратят значительное количество ресурсов на проверку качества и поиск ошибок на всех этапах жизненного цикла разработки ИС [4]. При этом желание снизить издержки в процессе разработки ИС зачастую приводит к необходимости исключения некоторых важных процедур или уменьшению длительности их проведения.

Важнейшим этапом в процессе разработки ИС, который позволяет определить готовность ИС к передаче заказчику или конечному пользователю, является тестирование [2]. Главными задачами тестирования можно назвать поиск ошибок [3] и проверку соответствия поведения разрабатываемой ИС ожиданиям конечного потребителя.

Для эффективного поиска ошибок применяется значительное количество подходов тестирования, например, модульное тестирование [4], интеграционное [5], нагрузочное [6], тестирование удобства использования [7] и др. В силу применения итерационных подходов к разработке ИС, таких как Scrum [8], Kanban [9] или экстремальное программирование [10], некоторые виды тестирования также должны выполняться регулярно. Применение всего доступного инструментария по тестированию ИС часто осложняется крайне сжатыми сроками разработки, недостаточной квалификацией инженеров по те-

стированию, а также сложными правилами взаимодействия сервисов при использовании микросервисной архитектуры ИС [11]. Сложный механизм влияния сервисов друг на друга значительно усложняет процедуру формирования наборов тестов. Как правило, тестами покрывают непосредственные изменения кода ИС, при этом понять опосредованное влияние модификаций кода того или иного микросервиса на другие микросервисы, которые не подвергались модификациям, бывает крайне непросто.

Вид тестирования, который призван обнаруживать ошибки в микросервисах и блоках кода, которые не подвергались модификациям, но могли утратить корректность работы вследствие модификации смежных структур кода, называется *регрессионным тестированием* (далее – РТ) [12]. Основные задачи, которые решаются при формировании пула тестов РТ, – отбор, минимизация и приоритизация [13].

Процесс разработки ИС со сложной бизнес-логикой и большим количеством интеграций со смежными ИС может занимать значительное время, в течение которого необходимо поддерживать реализованный функционал в актуальном состоянии. Количество тестов для РТ увеличивается с течением времени при добавлении нового функционала, что может привести к необходимости выделения количества ресурсов для РТ, превосходящих наложенные на команду разработки ограничения. В случае чрезмерного количества регрессионных тестов, отобранных для проверки состояния компонентов ИС, возможно, косвенно затронутых внесенными изменениями, необходимо производить ранжирование набора РТ, то есть решить задачу приоритизации. При наличии ранжированного списка тестов для РТ присутствует возможность обнаружения ошибок более высокого приоритета в более короткие сроки, что, в свою очередь, позволит снизить издержки на исправление найденных ошибок и подготовку ИС к промышленной эксплуатации. Для ранжирования тестов необходимо вычислить значимость каждого теста (далее – ЗТ) на основании некоторых признаков и значений метаданных тестов, следовательно, необходимо описать доступные принципы приоритизации тестового набора и сформировать рекомендации по их применению.

В процессе разработки сценариев для РТ в тест добавляется вспомогательная информация, которая в дальнейшем используется для группировки и ранжирования тестового набора. Метаданные тестов, которые впоследствии используются для ранжирования набора РТ, относятся к следующим категориям.

**Приоритет теста.** В процессе разработки и пересмотра теста назначается или анализируется приоритет теста. В подавляющем большинстве систем управления тестированием приоритета теста выбирается один из трех или пяти уровней: высокий, средний и низкий или высший, высокий, средний, низкий и низший соответственно.

*Высший приоритет* имеют тесты, которые непосредственно проверяют важнейший целевой функционал разрабатываемой ИС: включение/выключение устройства, подключение к точке доступа Wi-Fi смартфона, вход в аккаунт интернет-магазина, отображение мультимедийной информации смарт-телевизора и др. Возникновение ошибки в тесте с высшим приоритетом приводит к невозможности использования ИС или устройства. Возвращение функционала возможно только путем перезагрузки устройства или ИС.

*Высокий приоритет* имеют тесты, которые проверяют важный целевой функционал ИС: корректная работа смартфона с различными профилями Bluetooth, стабильная работа устройства в течение целевого времени, отображение специфических данных для опре-

# Формирование весовых коэффициентов методик отбора для регрессионного тестирования

деленного пользователя в интернет-магазине, получение корректного перечня доступных каналов смарт-телевизора и др. Возникновение ошибки в таком тесте приводит к утере ИС или устройством важного функционала с возможностью использования этого функционала каким-либо обходным путем.

*Средний приоритет* назначается тестам, которые покрывают незначительные возможности ИС или устройства: определенное количество сохраненных Bluetooth устройств, определенные характеристики работы устройства, вспомогательная информация в личном кабинете интернет-магазина, особенности структуры меню смарт-телевизора и др. Возникновение ошибки в тесте со средним приоритетом не приводит к утере возможностей ИС или устройства и вызывает лишь неудобство в использовании, а утерянный функционал может быть получен некоторыми обходными путями.

*Низкий приоритет* назначается тестам, которые покрывают визуальную составляющую или также вспомогательные возможности ИС или устройства, которые не относятся к базовым сценариям использования тестируемого объекта. Возникновение ошибки в тесте с низким приоритетом никак не влияет на функциональность и потребительские свойства ИС или устройства.

*Низший приоритет* назначается тестам, которые относятся к особенностям ИС или устройства, которые не могут быть замечены пользователем и не оказывают никакого влияния на функционал тестируемого объекта.

Когда применяется приоритизация тестовых сценариев из трех уровней, принципы назначения высшего и высокого приоритета и низкого и низшего приоритета объединяются.

Отбор сценариев для РТ по приоритету может быть выполнен с помощью стандартных средств систем управления тестированием (TMS), как это показано на Рисунках 1 и 2.

	P	Key	Version	Name	Status	Estimated
<input type="checkbox"/>	■	SQM-T784	1.0	[Политики] Создание кроссплатформенных скрытых политик Wink уров...	DRAFT	
<input type="checkbox"/>	■	SQM-T973	1.0	[Политики] Создание политики по Radius с агрегацией по Количеству а...	DRAFT	30m
<input type="checkbox"/>	■	SQM-T972	1.0	[Политики] Создание политики по Radius с агрегацией по Количеству а...	DRAFT	30m
<input type="checkbox"/>	■	SQM-T974	1.0	[Политики] Создание политики по Radius с агрегацией по Количеству и/...	DRAFT	30m
<input type="checkbox"/>	■	SQM-T968	1.0	[Политики] Создание политики по Vb с агрегацией по Количеству mcast...	DRAFT	2h
<input type="checkbox"/>	■	SQM-T414	1.0	[Политики] Создание политики по Топологии	DRAFT	
<input type="checkbox"/>	■	SQM-T148	1.0	[Политики] ШГД (RADIUS) политики - Создание политики	DRAFT	

**Рисунок 1.** Отбор тестов по приоритету в TMS Zephyr

Источник: здесь и далее рисунки выполнены авторами.

Ранжировать отобранные тесты можно простой сортировкой полученного списка по столбцу «Приоритет».

Фильтр

Название

Введите название

Статус автоматизации

Выбрать

Тип

Выбрать

ID

Введите глобальный ID

Секция

Выбрать

Приоритет

Самый высокий

Высокий

Рисунок 2. Отбор тестов по приоритету в TMS TestIT

**Длительность выполнения теста.** Очевидно, что при прочих равных параметрах и метаданных теста предпочтительно выполнять более короткие тесты. Таким образом, за единицу рабочего времени будет выполнено большее количество проверок, что позволит быстрее сделать вывод о корректности работы ИС или быстрее обнаружить ошибки. Кроме того, в случае недостаточных ресурсов для выполнения всего перечня сценариев РТ большее количество выполненных тестов позволит точнее понять степень готовности разрабатываемой ИС к промышленной эксплуатации. Примерная длительность теста назначается в процессе разработки теста, а впоследствии уточняется при выполнении и пересмотре. Длительность теста не учитывает время исследования найденной проблемы и время создания описания ошибки.

В зависимости от специфики проекта ориентировочное время выполнения теста может меняться в широком диапазоне, например, от пяти минут до одного часа и более. В таких случаях необходимо группировать длительности теста (например, не более пяти минут, не более десяти минут, не более двадцати минут) и в зависимости от длительности теста производить ранжирование пула тестов.

Отбор тестов в зависимости от длительности выполнения также может быть выполнен стандартными средствами некоторых систем TMS (см. Рисунки 3 и 4).

Add Criteria: (1)

Choose saved filter

Estimated Time: All

Reset

Min

hh:mm

Max

hh:mm

DB] Журналирование в базе данных по типу события: Удаление

DRAFT

15m

DB] Журналирование в базе данных по типу объекта: Весовые коэффи...

DRAFT

15m

SQM-T1432 1.0 [DB] Журналирование в базе данных по типу объекта: Глобальные филь...

DRAFT

15m

SQM-T1430 1.0 [DB] Журналирование в базе данных по типу объекта: Очереди

DRAFT

15m

Рисунок 3. Отбор тестов по длительности выполнения в Zephyr

Формирование весовых коэффициентов методик отбора для регрессионного тестирования

Автор	<input type="text" value="Выбрать"/>
Автор изменений	<input type="text" value="Выбрать"/>
Продолжительность	<input type="text" value="от"/> → <input type="text" value="до"/> Ожидаемый формат: 1d 8h 30m 10s
Дата создания	<input type="text" value="Начальная дата"/> → <input type="text" value="Конечная дата"/> <input type="button" value="📅"/>
Дата изменения	<input type="text" value="Начальная дата"/> → <input type="text" value="Конечная дата"/> <input type="button" value="📅"/>

**Рисунок 4.** Отбор тестов по длительности выполнения в TestIT

**Негативный результат предыдущего выполнения.** Простой и при этом надежный подход, позволяющий обнаруживать ошибки при выполнении РТ. Это тест, который, как правило, позволяет определить корректность исправления ошибки или внесение другой ошибки в попытке исправить исходную ошибку. Информация о результатах предыдущего выполнения данного теста легкодоступна как при использовании систем управления тестированием, так и при работе с тестовыми наборами в виде электронных таблиц. Однако наиболее популярные в настоящее время системы TMS, такие как TestIt, Zephyr, TestLink, DoQA, не имеют соответствующих фильтров и, соответственно, данный подход при формировании пула РТ используется только в ручном режиме – поэлементным добавлением подходящих тестов.

**Общее количество найденных ошибок.** Очевидно, что сценарий РТ, с помощью которого было обнаружено много ошибок, представляет большую значимость с точки зрения подтверждения корректного функционирования ИС. К сожалению, в современных системах TMS возможность фильтрации тестов по количеству найденных ранее ошибок отсутствует, и указанный признак может быть использован для ранжирования только добавлением и поддержкой соответствующей информации в метаданных теста.

**Срок предыдущего выполнения.** В силу недостатка ресурсов для выполнения всего доступного пула РТ некоторые сценарии могут не выполняться длительное время. При этом необходимо понимать, что в зависимости от специфики разрабатываемой ИС и процессов, принятых в компании-разработчике, периоды между циклами РТ могут быть весьма значительными. Поэтому более корректно выполнять отбор и ранжирование сценариев РТ не в абсолютных временных отрезках, а в циклах РТ, в рамках которых данный сценарий не выполнялся. Данный подход также не может быть применен при использовании стандартных средств систем TMS.

**Приоритет микросервиса.** При использовании микросервисной архитектуры разрабатываемой ИС важно учитывать, какой микросервис покрывает данный тест. Эту информацию можно указывать как в специальных полях сущности теста, так и в полях, которые могут содержать в себе любую вспомогательную информацию, например, в поле «Тэги» или Label. Используя фильтрацию по данным полям, можно также отбирать соответствующие сценарии стандартными средствами систем TMS.

Однако в подавляющем большинстве случаев каждый сценарий покрывает не один, а несколько микросервисов, при этом степень их участия может сильно различаться. Для этого, например, можно использовать специфические тэги с указанием степени покрытия сцена-

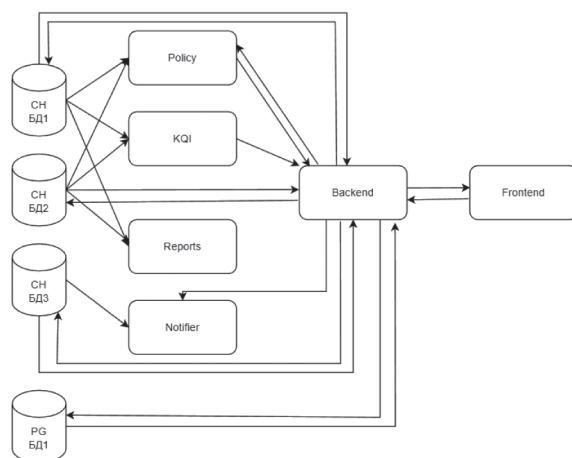


рием тех или иных микросервисов или создать специальные поля сущности «тест для хранения соответствующей информации». Степень покрытия сценарием нескольких микросервисов не может быть отфильтрована и ранжирована стандартными средствами TMS.

**Объем внесенных изменений в микросервис.** Объем внесенных изменений в существующий микросервис является маркером, показывающим степень риска, которую влекут модификации кода. Абсолютные значения модифицированных строк малополезны, при этом, оценивая процент модифицированных, удаленных и добавленных строк, можно сделать вывод о степени модификации того или иного микросервиса. При ранжировании сценариев по степени модификации микросервиса необходимо учитывать специфику разрабатываемой ИС, а также средний процент модифицированных строк. Ранжирование по объему внесенных изменений использует ранжирование по приоритету сервиса или фильтрацию тестов по названию микросервиса.

Стандартные системы TMS не позволяют выполнять отбор и ранжирование сценариев РТ по объему внесенных изменений в силу отсутствия достаточной степени интеграции с системами разработки и контроля версий.

**Модификация смежных микросервисов первого, второго уровня.** В микросервисных ИС взаимодействие микросервисов между собой позволяет решить вопросы упрощения разработки каждого отдельного микросервиса. При этом большое количество микросервисов может значительно усложнить выявление степени влияния внесенного изменения в один из микросервисов на взаимодействующие с ним микросервисы (Рисунок 5). Количество микросервисов и степень их интеграции зависит от специфики разрабатываемой ИС.



**Рисунок 5.** Схема взаимосвязей микросервисной ИС с учетом направления взаимодействий

Из архитектуры микросервисной ИС можно определить, на какие смежные микросервисы может оказать влияние модификация того или иного микросервиса. С точки зрения формирования метаданных для сценариев РТ для учета архитектуры ИС добавляется соответствующая информация в дополнительные поля сущности теста в системы TMS, что позволяет отбирать тесты, проверяющие микросервисы первого, второго и последующих уровней смежности.

Формирование весовых коэффициентов методик отбора для регрессионного  
тестирования

Применение методик отбора по отдельности для всего пула регрессионных тестов не представляет сложной задачи. При этом очевидно, что одновременное применение нескольких методик отбора даже при использовании современных систем ТМС крайне затруднительно – не все подходы отбора и ранжирования могут быть реализованы через фильтрацию. Кроме того, ранжирование пула тестов предполагает возможность определения (установки) ЗТ с точки зрения использующихся при отборе и ранжировании тестов [14].

Значимость отдельного теста может быть вычислена по формуле

$$I_t = I_1 + I_2 + \dots + I_n,$$

где  $I_t$  – итоговая значимость теста;  $I_1$  – значимость теста, найденная способом 1;  $I_2$  – значимость теста, найденная способом 2;  $I_n$  – значимость теста, найденная способом  $n$ .

### Результаты экспериментов

Когда пул РТ состоит из значительного количества тестов, вычисление ЗТ можно выполнить с помощью разработанной автором автоматизированной «Системы отбора регрессионных тестов» [15], которая позволяет получать полный набор параметров и мета-данных каждого сценария РТ и гибко настраивать используемые подходы к ранжированию пула РТ, а также значения весов для того или иного подхода (см. Рисунок 6).

**Рисунок 6.** Конфигурирование параметров ранжирования пула РТ

Результатом работы системы отбора регрессионных тестов является ранжированный в порядке убывания ЗТ список РТ. Сценарии РТ выполняются в порядке убывания ЗТ. В зависимости от специфики процессов разработки ИС может быть установлен порог ЗТ, при котором тесты, имеющие ЗТ ниже этого порога, выполнять не нужно (см. Рисунок 7).

Значимость теста в зависимости от подхода ранжирования не может быть универсальной величиной, подходящей для ИС любой специфики и сферы применения.

Руководствуясь общими предположениями, определить весовое значение для каждой ЗТ не представляется возможным в силу специфики каждой ИС, процессов и подходов, которые применяются в процессе разработки данной ИС. Для получения значений применительно к конкретной ИС предлагается выполнить максимально возможный перечень сценариев для РТ, получив, таким образом, пул РТ с информацией об ошибках, найденных каждым сценарием. Далее необходимо подобрать значения весовых коэффициентов для каждого подхода ранжирования тестов таким образом, чтобы тесты, с помощью



которых были найдены ошибки, были в начале списка. В зависимости от специфики процессов разработки и наличия ресурсов для выполнения пула РТ можно выбрать значения весовых коэффициентов таким образом, чтобы тесты, позволившие обнаружить ошибки, находились в перечне первых 50 или 25 % тестов, или же использовать абсолютное количество сценариев.

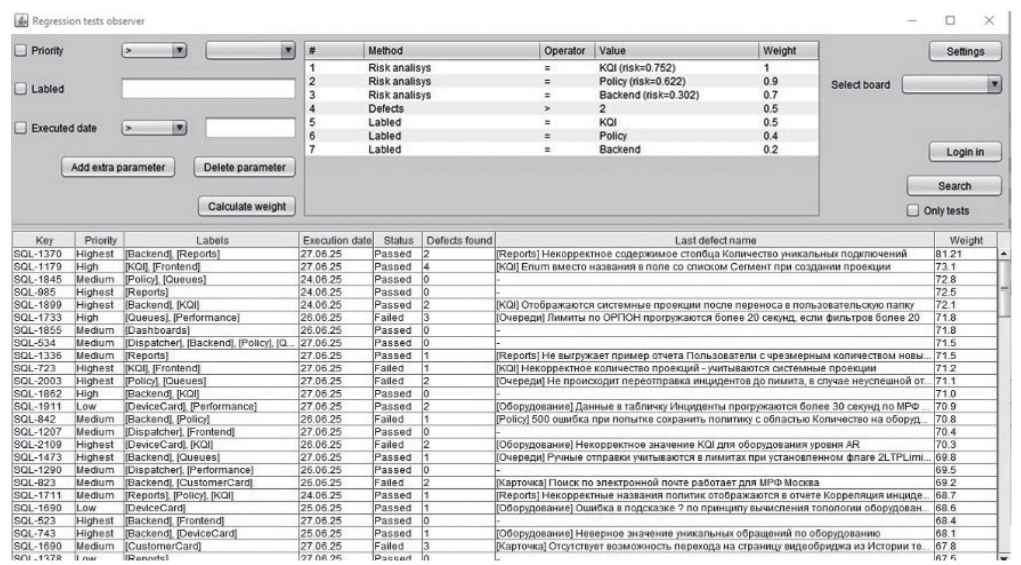


Рисунок 7. Общий вид ИС по ранжированию тестов для РТ

Указанный подход был применен для формирования наборов тестов для РТ в процессе разработки ИС по контролю качества передаваемого мультимедийного сигнала цифрового телевидения и широкополосного доступа в сеть Интернет крупного российского оператора. Архитектура ИС содержит более 180 микросервисов. Общее количество строк кода составляет более 10 млн. Выпуск новых версий происходит в среднем один раз в два месяца. Общее количество сценариев РТ – около 2 тыс. В результате получены следующие значения весовых коэффициентов:

- приоритет теста (см. Таблицу 1);
- длительность выполнения теста (см. Таблицу 2);
- негативный результат предыдущего выполнения, значение весового коэффициента 0,8;
- общее количество найденных ошибок (см. Таблицу 3);
- период предыдущего выполнения (см. Таблицу 4);
- приоритет микросервиса (см. Таблицу 5);
- объем внесенных изменений в микросервис (см. Таблицу 6);
- модификация смежных микросервисов первого, второго уровня в случае наличия модификаций смежных микросервисов первого или второго уровня – значение весовых коэффициентов, которые покрывают данный микросервис, 0,2.

Формирование весовых коэффициентов методик отбора для регрессионного  
тестирования

Таблица 1

**Значения весовых коэффициентов в зависимости от приоритета теста**

Приоритет теста	Высокий	Средний	Низкий
Весовой коэффициент	0,9	0,5	0,2

Источник: здесь и далее таблицы выполнены авторами.

Таблица 2

**Значения весовых коэффициентов в зависимости от приоритета теста**

Длительность теста, мин	≤5	>5≤10	>10≤20	>20≤30	>30
Весовой коэффициент	0,8	0,7	0,6	0,4	0,2

Таблица 3

**Значения весовых коэффициентов в зависимости от количества найденных ошибок**

Количество найденных ошибок	0	1	2	3-4	>4
Весовой коэффициент	0	0,3	0,5	0,7	1

Таблица 4

**Значения весовых коэффициентов в зависимости от периода предыдущего выполнения**

Период от предыдущего выполнения теста (в количестве циклов РТ)	1	2	3-4	5-7	>7
Весовой коэффициент	0	0,1	0,2	0,4	0,6

Таблица 5

**Значения весовых коэффициентов в зависимости от приоритета микросервиса**

Название микросервиса	Backend	PolicyEngine	KQI	Audit	WinkComposer	Прочие
Весовой коэффициент	0,8	0,9	0,3	0,2	0,6	0

Таблица 6

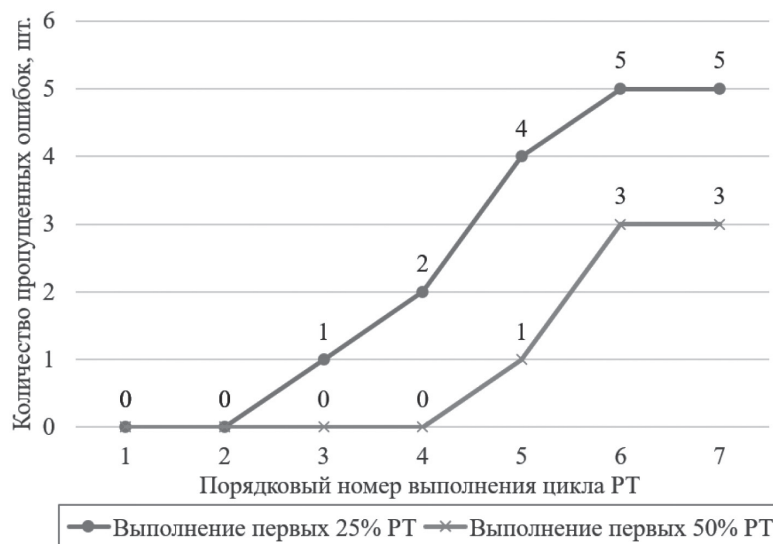
**Значения весовых коэффициентов в зависимости от объема внесенных изменений**

Процент внесенных изменений, %	<0,1	<0,5	<1	<3	<5	≥5
Весовой коэффициент	0	0,2	0,4	0,6	0,8	1

На основании приведенных весовых коэффициентов формировался пул тестов объемом 25 % от полного набора сценариев для РТ. Исследования показали, что для данной ИС актуальность выбранных весовых коэффициентов снижается с каждым новым запуском выполнения РТ в силу внесенных модификаций, новых микросервисов, изменения архитектуры взаимодействия микросервисов и др. Для 25 % тестов с высшей ЗТ актуальность выбранных весовых коэффициентов составляет три запуска выполнения РТ, для 50 % – пять запусков (см. Рисунок 8).

Потеря актуальности весовых коэффициентов приводит к появлению пропущенных во время проведения РТ ошибок, то есть ошибки покрыты тестами, не входящими в вы-

борку. В зависимости от принятых пределов допустимого количества известных ошибок в передаваемой заказчику версии ИС количество циклов выполнения РТ без уточнения весовых коэффициентов может быть увеличено. Уточнение значений весовых коэффициентов производится путем выполнения всего пула РТ в совокупности с анализом пропущенных дефектов (Escaped Defect Analysis – EDA).



**Рисунок 8.** Динамика возникновения пропущенных дефектов при потере актуальности весовых коэффициентов

Представленная методика позволяет формировать весовые коэффициенты для выбранных подходов к отбору и минимизации пула тестов для РТ таким образом, чтобы при формировании ранжированного списка сценариев тесты, которые помогут обнаружить ошибки или подтвердить корректность работы значимого функционала ИС, находились вверху списка. Это позволит получать ошибки в более сжатые сроки и выполнить меньшее количество тестов для определения корректности работы разрабатываемой ИС, что, в свою очередь, приводит к экономии ресурсов в процессах разработки ИС. Количество сценариев, достаточных для вынесения вердикта о готовности ИС к промышленной эксплуатации, зависит от специфики разрабатываемой ИС и процедур, принятых в компании – разработчике ИС.

Необходимо отметить, что предложенная методика формирования весовых коэффициентов требовательна к корректному формированию метаданных сценариев РТ, а также может быть применена только в совокупном использовании с автоматизированными системами ранжирования тестов для РТ.

### Литература

1. Куркин А.А., Епифанов С.А. К вопросу визуализации географического распределения коренных малочисленных народов России // Экономическая безопасность России: проблемы и перспекти-

Формирование весовых коэффициентов методик отбора для регрессионного  
тестирования

- вы : Материалы XII Международной научно-практической конференции. Нижний Новгород, 29 мая 2024 г. Н. Новгород : Изд-во НГТУ, 2024. С. 247–251. EDN WSJGWE.
2. Шутов А.А., Герасимова А.Е., Филинских А.Д., Глумова Е.С. Создание интерактивной образовательной среды с помощью технологии виртуальной реальности // КОГРАФ-2024 : Сборник материалов 34-й Всероссийской молодежной научно-практической конференции по графическим информационным технологиям и системам, Нижний Новгород, 15–17 апреля 2024 г. Н. Новгород : Нижегородский государственный технический университет имени Р.Е. Алексеева, 2024. С. 157–164. EDN BPGCZQ.
3. Куркин А.А., Епифанов С.А. Опыт применения современных инструментов геоинформационного картографирования для составления этнографической карты коренных малочисленных народов России // Будущее технической науки : Сборник материалов XXIII Всероссийской молодежной научно-технической конференции. Нижний Новгород, 31 мая 2024 г. Н. Новгород : Нижегородский государственный технический университет имени Р.Е. Алексеева, 2024. С. 778–779. EDN VGAPLE.
4. Хориков В. Принципы юнит-тестирования / Пер. с англ. Е. Матвеева. СПб. : Питер, 2022. 316 с. ISBN 9785446116836.
5. Савин Р. Тестирование Дот Ком, или Пособие по жестокому обращению с багами в интернет-стартапах. М. : Издательские решения, 2017. 312 с. ISBN 978-5-4485-4551-1.
6. Мясников С.О., Намиот Д.Е. Инструменты нагрузочного тестирования // Прикладная информатика. 2018. Т. 13. № 1 (73). С. 92–102. EDN YSCDZU.
7. Khaleeva U. Evaluating the Interface Using Expert-Heuristic Method // CEUR Workshop Proceedings. Series 31: “GraphiCon 2021” – Proceedings of the 31<sup>st</sup> International Conference on Computer Graphics and Vision. Nizhny Novgorod, September 27–30, 2021. Vol. 3027. Nizhny Novgorod, 2021. P. 1114–1124. EDN SMVQXH.
8. Сазерленд Дж. Scrum. Революционный метод управления проектами / Пер. с англ. М. Гескиной. М. : Манн, Иванов и Фербер, 2022. 288 с. ISBN 5000577221.
9. Барроуз М. Канбан Метод. Улучшение системы управления / Пер. с англ. М. Витебского. М. : Альпина Паблишер, 2021. 304 с. ISBN 978-5-9614-3454-5.
10. Бек К. Экстремальное программирование. Разработка через тестирование : Пер. с англ. СПб. : Питер, 2020. 224 с. ISBN 978-5-4461-1439-9.
11. Ding Yong. Design and Practice of Software Architecture in Agile Development // Journal of Physics: Conference Series. 2021. Vol. 2074. Article no. 012008. DOI: 10.1088/1742-6596/2074/1/012008
12. Verma S.A., Choudhary A., Tiwari S. Regression Test Case Selection: A Comparative Analysis of Metaheuristic Algorithms. Conference paper // Gupta D., Khanna A., Kansal V., Fortino G., Hassaniien A.E. (Eds) Proceedings of Second Doctoral Symposium on Computational Intelligence. Series: Advances in Intelligent Systems and Computing. 2021. Vol. 1374. Singapore : Springer. Pp. 605–615. DOI: [https://doi.org/10.1007/978-981-16-3346-1\\_49](https://doi.org/10.1007/978-981-16-3346-1_49)
13. Shin Yoo, Harman M. Regression testing minimization, selection and prioritization: A survey // Software Testing, Verification & Reliability. 2012. Vol. 22. No. 2. P. 67–120. DOI: <https://doi.org/10.1002/STV.430>
14. Зарубин И.Б., Филинских А.Д. Автоматизированная генерация весовых коэффициентов способов отбора при формировании пула регрессионных тестов // Информационные системы и технологии ИСТ-2021 : Сборник материалов XXVII Международной научно-технической конфе-

ренции. Нижний Новгород, 23–24 апреля 2021 г. Н. Новгород : Нижегородский государственный технический университет им. Р.Е. Алексеева, 2021. С. 436–442. EDN GPWRJW.

15. Долбунов Д.А., Зарубин И.Б. Разработка сетевого restful интерфейса для интеграции автоматизированной системы отбора регрессионных тестов // КОГРАФ-2021 : Сборник материалов 31-й Всероссийской научно-практической конференции по графическим информационным технологиям и системам. Нижний Новгород, 19–22 апреля 2021 г. Н. Новгород: Нижегородский государственный технический университет им. Р.Е. Алексеева, 2021. С. 156–159. DOI: 10.46960/43791586\_2021\_156. EDN BNBLZJ.

## References

1. Kurkin A.A., Epifanov S.A. (2024) On the issue of visualizing the geographical distribution of Russia's indigenous peoples. In: *Ekonomicheskaya bezopasnost' Rossii: problemy i perspektivy* [Economic security of Russia: Problems and prospects] : Proceedings of the XII International Scientific and Practical Conference. Nizhny Novgorod, May 29, 2024. Nizhny Novgorod : Nizhny Novgorod State Technical University named after R.E. Alekseev Publ. Pp. 247–251. (In Russian).
2. Shutov A.A., Gerasimova A.E., Filinskikh A.D., Glumova E.S. (2024) Creation of an interactive educational environment using virtual reality technology. In: Glumova E.S. (Ed) *KOGRAF-2024* [KOGRAF-2024] : Proceedings of the 34<sup>th</sup> All-Russian Youth Scientific and Practical Conference on Graphic Information Technologies and Systems, Nizhny Novgorod, April 15–17, 2024. Nizhny Novgorod : Nizhny Novgorod State Technical University named after R.E. Alekseev. Pp. 157–164. (In Russian).
3. Kurkin A.A., Epifanov S.A. (2024) Experience in Applying Modern Geoinformation Mapping Tools for Creating an Ethnographic Map of Russia's Indigenous Peoples. In: Belyakov V.V. (Ed) *Budushchee tekhnicheskoi nauki* [The Future of Technical Science] : Proceedings of the XXIII All-Russian Youth Science and Technology Conference. Nizhny Novgorod, May 31, 2024. Nizhny Novgorod State Technical University named after R.E. Alekseev. Pp. 778–779. (In Russian).
4. Khorikov V. (2020) *Unit Testing: Principles, Practices, and Patterns: Effective testing styles, patterns, and reliable automation for unit testing, mocking, and integration testing with examples in C#*. Manning. 304 p. ISBN 9781617296277. (Russian edition: transl. by E. Matveev. St. Petersburg : Piter Publ., 2022. 316 p.).
5. Savin R. (2017) *Testirovanie Dot Kom, ili Posobie po zhestokomu obrashcheniyu s bagami v internet-startapakh* [Dot Com Testing: A Guide to Cruelly Handling Bugs in Internet Startups]. Moscow : Izdatel'skie resheniya Publ. 312 c. ISBN 978-5-4485-4551-1. (In Russian).
6. Myasnikov S.O., Namiot D.E. (2018) On load testing tools. *Prikladnaya informatika* [Journal of Applied Informatics]. Vol. 13. No. 1 (73). Pp. 92–102. (In Russian).
7. Khaleeva U. (2021.) Evaluating the Interface Using Expert-heuristic Method. In: CEUR Workshop Proceedings. Series 31: "GraphiCon 2021" – Proceedings of the 31<sup>st</sup> International Conference on Computer Graphics and Vision. Nizhny Novgorod, September 27–30, 2021. Vol. 3027. Nizhny Novgorod, 2021. Pp. 1114–1124.
8. Sutherland J. (2014) *Scrum: The Art of Doing Twice the Work in Half the Time*. New York : Crown Business. 384 c. ISBN 038534645X. (Russian edition: transl. by M. Geskina, Moscow : Mann, Ivanov, and Ferber Publ. 2022. 288 p.).
9. Burrows M. (2014) *Kanban from the Inside: Understand the Kanban Method, connect it to what you already know, introduce it with impact*. Blue Hole Press. 270 p. ISBN 0985305193. (Russian edition: transl. by M. Vitebskiy. Moscow : Alpina Publisher, 2021. 304 p.).

10. Beck K. (2003) *Test-driven development by example*. Addison-Wesley. 220 p. ISBN 0321146530. (Russian edition: St. Petersburg : Piter Publ., 2020. 224 p.).
11. Ding Yong. (2021) Design and Practice of Software Architecture in Agile Development. *Journal of Physics: Conference Series*. Vol. 2074. Article no. 012008. DOI: 10.1088/1742-6596/2074/1/012008
12. Verma S.A., Choudhary A., Tiwari S. (2021) Regression Test Case Selection: A Comparative Analysis of Metaheuristic Algorithms. In: Gupta D., Khanna A., Kansal V., Fortino G., Hassanien A.E. (Eds) *Proceedings of Second Doctoral Symposium on Computational Intelligence. Series: Advances in Intelligent Systems and Computing*. Vol. 1374. Singapore : Springer. Pp. 605–615. DOI: [https://doi.org/10.1007/978-981-16-3346-1\\_49](https://doi.org/10.1007/978-981-16-3346-1_49)
13. Shin Yoo, Harman M. (2012) Regression testing minimization, selection, and prioritization: A survey. *Software Testing, Verification & Reliability*. Vol. 22. No. 2. P. 67–120. DOI: <https://doi.org/10.1002/STV.430>
14. Zarubin I.B., Filinskikh A.D. (2021) Automated generation of selection method weights when forming a regression test pool. In: Glumova E.S. (Ed) *Informatsionnye sistemy i tekhnologii IST-2021* [Information Systems and Technologies IST-2021] : Proceedings of the XXVII International Scientific and Technical Conference, Nizhny Novgorod, April 23–24, 2021. Nizhny Novgorod : Nizhny Novgorod State Technical University named after R.E. Alekseev Publ. Pp. 436–442. (In Russian).
15. Dolbunov D.A., Zarubin I.B. (2021) Development of a network restful interface for integrating an automated regression test selection system. In: *KOGRAF-2021* [KOGGRAPH-2021] : Proceedings of the 31<sup>th</sup> All-Russian Youth Scientific and Practical Conference on Graphic Information Technologies and Systems, Nizhny Novgorod, April 19–22, 2021. Nizhny Novgorod : Nizhny Novgorod State Technical University named after R.E. Alekseev. Pp. 156–159. DOI: 10.46960/43791586\_2021\_156 (In Russian).

Поступила в редакцию: 28.10.2025

Received: 28.10.2025

Поступила после рецензирования: 21.11.2025

Revised: 21.11.2025

Принята к публикации: 12.12.2025

Accepted: 12.12.2025