Федеральное государственное автономное образовательное учреждение

высшего образования

"Российский университет транспорта" (МИИТ)

Кафедра «Цифровые технологии управления транспортными процессами»

**ОТЧЁТ**

**ПО ТЕХНОЛОГИЧЕСКОЙ ПРАКТИКЕ**

«Обзор подходов к тестированию программного обеспечения   
в крупных компаниях»

**Тип практики:** ПРОИЗВОДСТВЕННАЯ ПРАКТИКА

(технологическая - 1)

Выполнил ст. г. УВА-411 Д. Н. Романов

Принял руководитель: Н. М. Нечитайло

Москва  
2022

|  |
| --- |
| УТВЕРЖДАЮ  Руководитель ОП ВО по  направлению 09.03.01  \_\_\_\_\_\_\_\_ Э.К. Лецкий  \_\_.07.2021 г. |

**ЗАДАНИЕ НА ПРАКТИКУ**

**производственную практику**

**(технологическая - 1)**

студента группы УВА-411 Романова Даниила Николаевича

Место прохождения производственной практики: кафедра ЦТУТП ИУЦТ РУТ (МИИТ)

Наименование темы: Обзор подходов к тестированию программного обеспечения   
в крупных компаниях.

Необходимо изучить:

1. Способы тестирования программного обеспечения
2. Ручное тестирование
3. Автоматизированное тестирование

Необходимо самостоятельно выполнить:

1. Анализ жизненного цикла разработки программного обеспечения
2. Обзор процесса автоматизированного тестирования
3. Разбор решения проблем, связанных с автоматизированным тестированием

Основные требования к проделываемой работе:

Для оформления отчета по практике должен быть использован ГОСТ 7.32-2017 «Отчет о научно-исследовательской работе. Структура и правила оформления».

Источники информации:

1. Основы тестирования программного обеспечения. Учебное пособие для СПО, 2-е изд., стер.
2. Software Testing Techniques, 2nd Edition, Борис Бейзер
3. Scrum: Революционный метод управления проектами, Джефф Сазерленд
4. Experiences of Test Automation: Case Studies of Software Test Automation 1st Edition, Dorothy Graham

Материалы к защите:

1. Заполненный, подписанный и утверждённый лист индивидуального задания на производственную практику.
2. Пояснительная записка (отчёт) по технологической практике.

Студент группы УВА-411 Д.Н. Романов

Доцент кафедры ЦТУТП Н.М. Нечитайло

**Оглавление**

[**1** **Вступление** 3](#_Toc94387661)

[**2** **Разбор элемента жизненного цикла разработки ПО “Тестирование”** 4](#_Toc94387662)

[**3** **Разбор процесса автоматизированного тестирования** 8](#_Toc94387663)

[**4** **Разбор проблемы автоматизированного тестирования** 10](#_Toc94387664)

[**Заключение** 12](#_Toc94387665)

[**Источники информации** 13](#_Toc94387666)

1. **Введение.**

В современном мире информационные технологии развиваются с огромной скоростью. Постоянно растёт потребность в новом программном обеспечении, развиваются банки, бюрократические системы, транспортные технологии и т.д. В свою очередь, растут и компании, занимающиеся разработкой программного обеспечения, на спрос необходимо отвечать предложением.

Крупным IT компаниям приходится постоянно решать новые задачи, гибко подстраиваясь под требования рынка. Решения находят быстро, но зачастую они не всегда оказываются наиболее эффективными. Каждая новая программа (система), проходит несколько жизненно важных этапов разработки. Одним из этапов является тестирование программного обеспечения. Существует множество подходов к его осуществлению. Однако рассматривая эти подходы, можно заметить множество важных недочётов, которые в перспективе оборачиваются для компании большими убытками. Решение этой проблем с тестированием остаётся открытым вопросом на сегодняшний день.

Проблема является наиболее актуальной для крупных компаний, которые изначально приняли неверный подход к решению задач по тестированию их программного обеспечения. Чем больше IT компания, тем тяжелее обеспечивать гибкость текущих процессов и решать проблемы с тестированием приходится медленно, зачастую даже с принятием потенциальных убытков.

Цель исследования заключается в том, чтобы проанализировать текущие решения, выявить их сильные и слабые стороны. И уже основываясь на полученных выводах принять наиболее верный подход к решению проблемы тестирования программного обеспечения. Результатом работы будет программа, которая обеспечит наиболее эффективное и гибкое решение для тестирования любых приложений или систем.

**Глава 1.1 Тестирование программного обеспечения. Общий анализ.**

Современный цикл разработки программного совмещает в себе несколько этапов, таких как - анализ, разработка, тестирование и релиз.

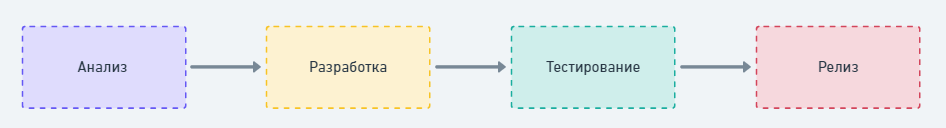


Рис. 1 - Современный цикл разработки ПО

* Анализ: в этап входит обсуждение задач с заказчиком, уточнение требований к конечному продукту, трансляция заявленных требований разработчикам, написание технической документации и постановка задачи. [3]
* Разработка: большой и непрерывный этап жизненного цикла, выполняется реализация технического задания, бизнес требований, планировка архитектуры будущей информационной системы. Происходит корректировка требований, обсуждение новой функциональности, доработка технического задания.
* Тестирование: постоянный процесс тестирования приложения, которое находится в разработке. На этом этапе проверяется не только его соответствие требованиям, но и пропускная способность, отказоустойчивость, готовность к непредвиденным ситуациям, к различному поведению пользователей внутри и снаружи будущей системы.
* Релиз: поставка приложения, которое успешно прошло этап тестирования заказчику для уточнения требований, возможных доработок и непосредственной эксплуатации. [3]

**Разбор подтипов тестирования**

Тестирование программного обеспечения делится на множество подтипов, каждый из которых решает конкретную задачу.



Рис. 2 - Виды тестирования

* Модульное тестирование. Такие тесты пишут разработчики для проработки небольших участков кода. Они позволяют избежать явных логических ошибок. Модульные тесты обеспечивают покрытием около 80% всего кода, они обязательно включают в себя все методы и логические ветвления [1].
* Функциональное тестирование. Обеспечивает проверку заявленного функционала. Проверяется работа приложения согласно написанной технической документации [1].
* Smoke тестирование. Проверка приложения на работоспособность. Проверяется работа приложения независимо от бизнес-требований. Успешное smoke тестирование позволяет приступить к другим типам тестирования [1].
* Интеграционное тестирование или A/B тесты. Пишутся тестовые сценарии, по которым будет осуществляться работа с готовым приложением. Позволяет протестировать возможные действия пользователя.  
  Интеграционный тест считается проваленным, если не выполнился хотя бы один пункт сценария [1].
* Нагрузочное тестирование. Проверяется работа приложения под большой нагрузкой, что позволит увидеть слабые и сильные стороны конечного продукта [1].
* Системное тестирование. Данный подтип тестирования проверяет функционал интеграции с другими приложениями или системами [1].
* Регрессионное тестирование. Подразумевает под собой множество различных тестов, которые прошли успешно в старой версии приложения. Позволяет легко выявлять ошибки, которые появляются в новых версиях приложения. Количество регрессионных тестов увеличивается с каждым новым релизом [1].

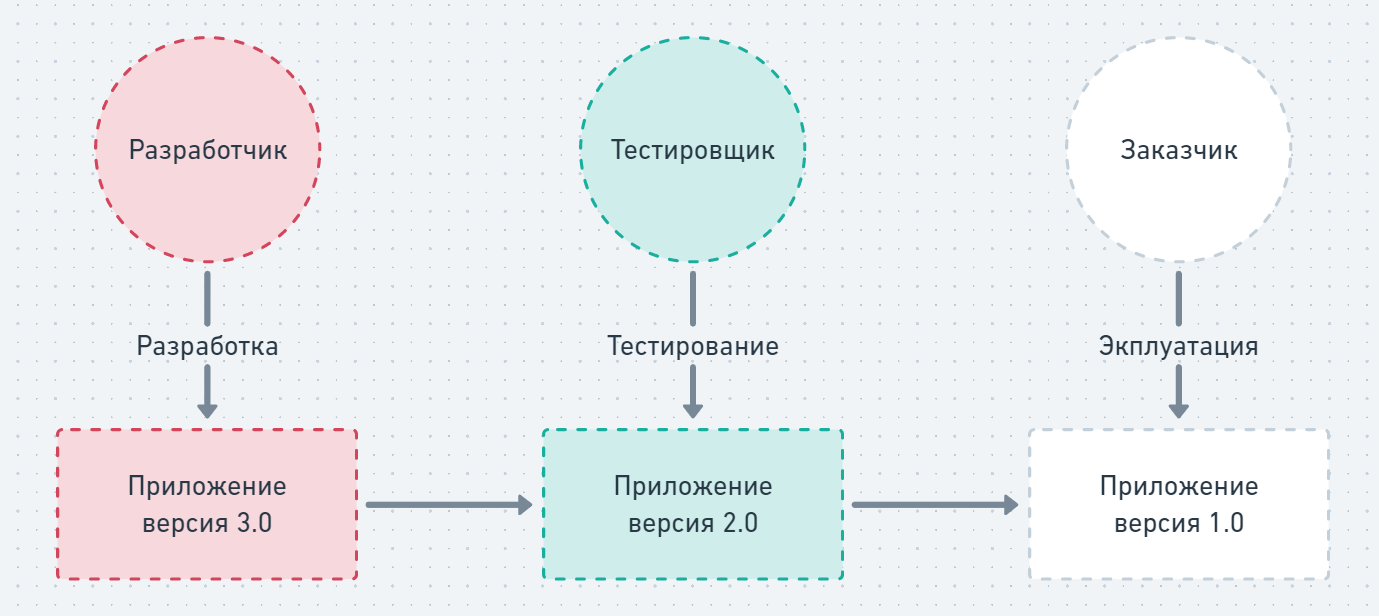


Рис. 3 Схема взаимодействия разработчика, тестировщика и заказчика

Разработчик постоянно реализует новый функционал для приложения и исправляет выявленные ошибки. Тестировщик занимается проверкой работы приложения и передаёт его заказчику. Заказчик в свою очередь получает приложение и предлагает идеи по его улучшению, исправлению не выявленных при тестировании ошибок и т.д.

Такой подход к тестированию приложения является приемлемым только на небольших объёмах производства программного обеспечения. Чем больше приложение, тем сложнее становится его тестировать и тем больше времени уходит на его выпуск заказчику.

Схема на рис. 3 вырождается в схему на рис 4.

**Глава 1.2 Обзор подходов к решению проблем тестирования программного обеспечения.**

В современных системах целевым подходом к разработке больших приложений является разбиение его на модули, которые выполняют небольшую часть конечной функциональности. Такие модули принято называть «микросервисы», а архитектура таких приложений называется «микросервисная архитектура».

Рассмотрим подходы к тестированию такого приложения.

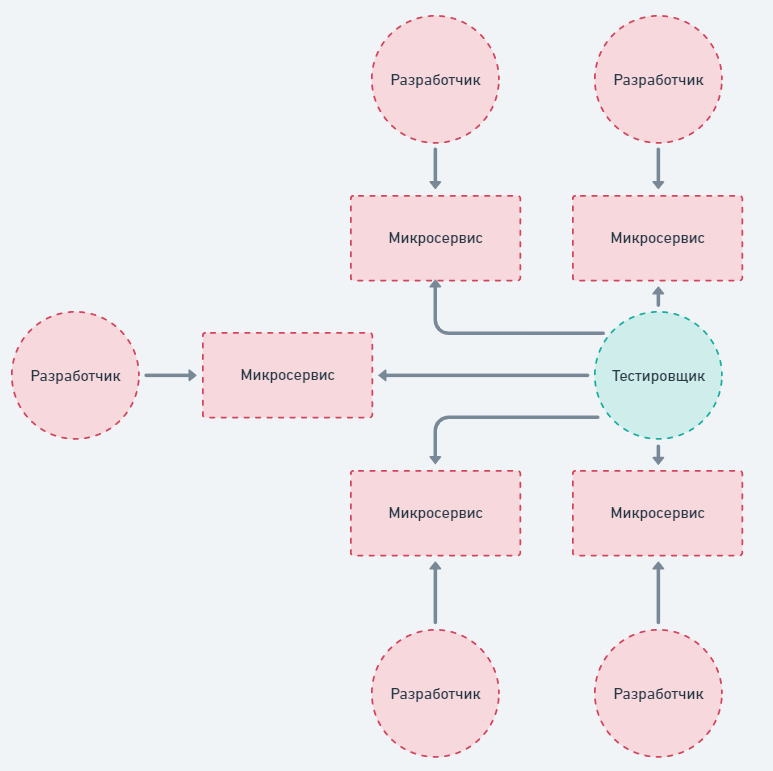


Рис. 4 Схема взаимодействия разработчика и тестировщика №1

**Достоинства:**

* Низкие затраты на тестирование.

**Недостатки:**

* Этап тестирования занимает большое количество времени. С каждым новым релизом приложения это время будет расти линейно.
* Отсутствует возможность роста команды разработки.
* Замедление этапа разработки программного обеспечения, т.к. тестирование не справляется с объёмами текущего приложения.

**Решение:**

Увеличить команду тестирования.

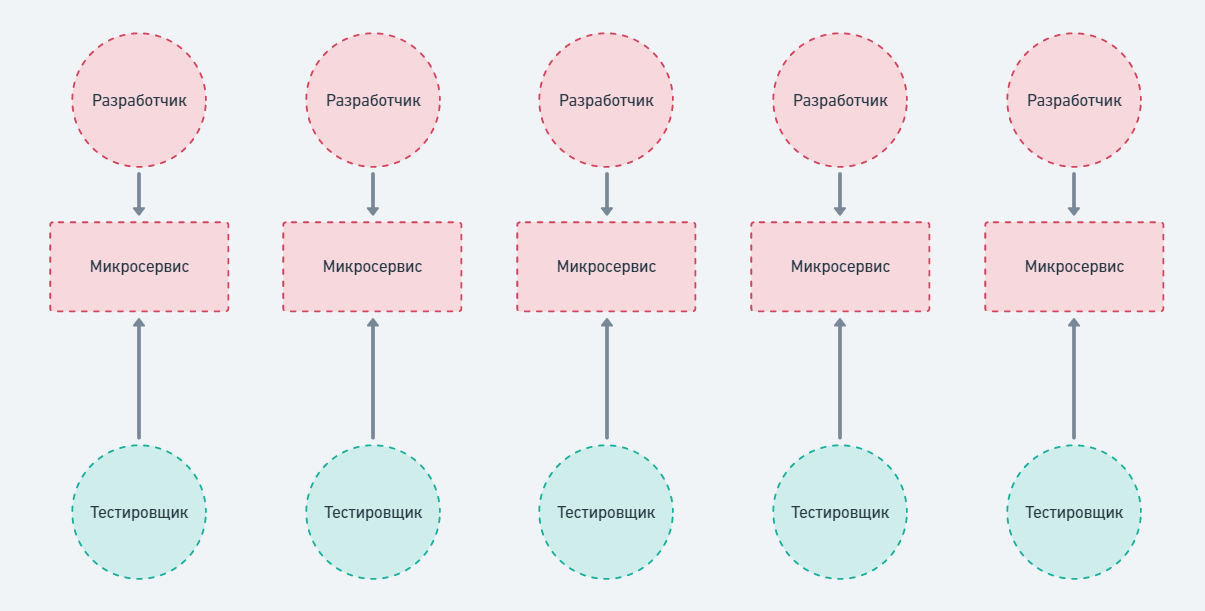


Рис. 5 Схема взаимодействия разработчика и тестировщика №2

**Достоинства:**

* Тестирование проходит быстро.
* Нагрузка на каждого тестировщика распределена равномерно.

**Недостатки:**

* Рост команды разработки влечёт за собой рост команды тестирования
* Высокие расходы на тестирование
* После того, как приложение будет полностью готово останется большая команда тестирования.

**Решение:**

Внедрить автоматизированное тестирование.

**Глава 1.3 Разбор процесса автоматизированного тестирования**

Ручное тестирование – трудоёмкий процесс, который можно автоматизировать. На рынке существует множество решений, которые позволяют работать с браузером, базами данных и API приложений программным путём. Используя автоматизированные тесты можно отказаться от специалистов по ручному тестированию.

Автоматизированные тесты – это программа, которая способна эмулировать поведение конечного пользователя в системе и генерировать отчёты об ошибках и проделанных действиях и полученных результатах. Возможности «автотестов» очень широки.

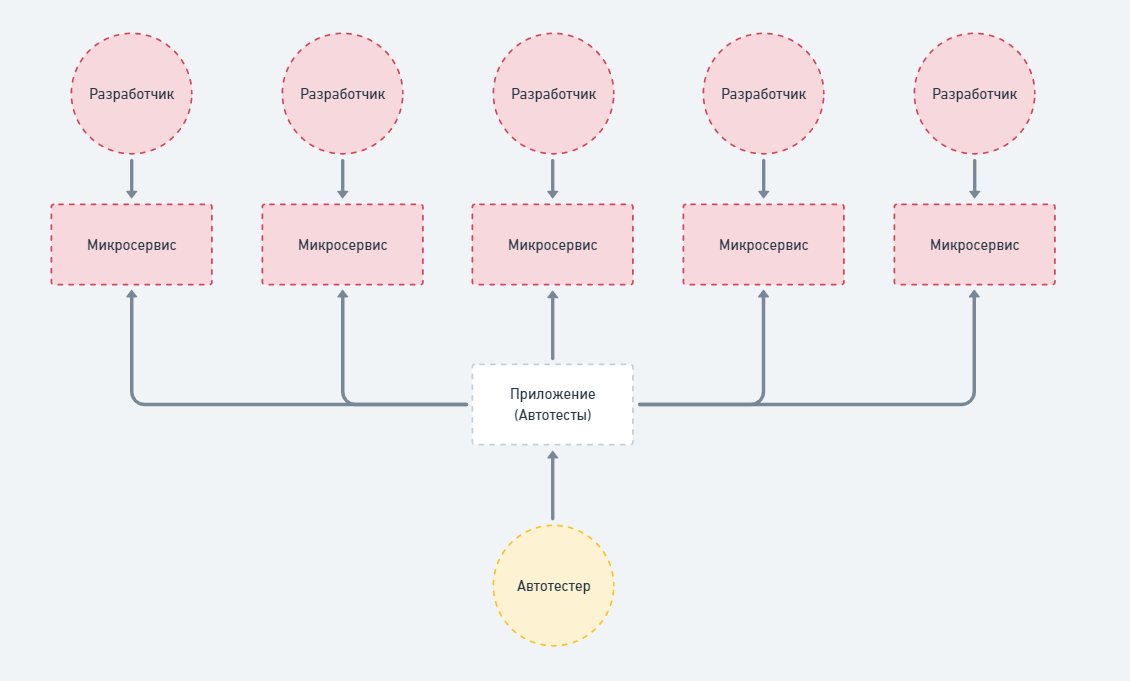


Рис. 6 Схема взаимодействия разработчика и автотестера №1

**Достоинства:**

* Автоматизированные тесты работают быстрее любого человека.
* Есть возможность запускать тесты в обособленной среде.
* Можно быстро выполнять полное тестирование любых версий приложений.
* Тесты работают одновременно, в асинхронном режиме.
* Тесты не имеют человеческого фактора - безошибочны.
* Автоматизировать можно любой процесс, даже отправку отчёта об ошибках разработчикам напрямую.
* Тесты легко поддерживать, есть возможность гибкой настройки.

**Недостатки:**

* На разработку автоматизированных тестов уходит много времени.
* На больших проектах одного специалиста по автоматизированному тестированию будет недостаточно, т.к. необходимо быстро разрабатывать новые тесты.

**Решение:**

Нанять ещё нескольких специалистов по автоматизированному тестированию.

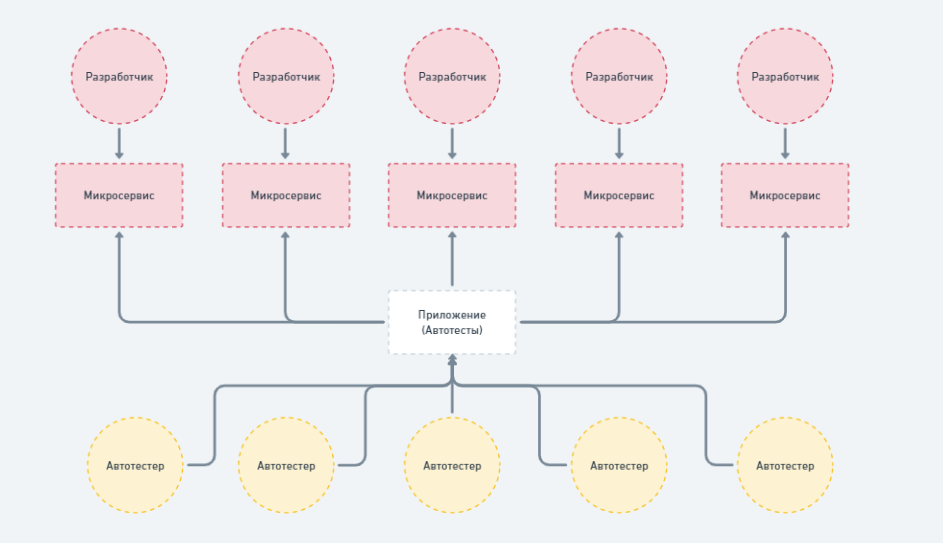


Рис. 7 Схема взаимодействия разработчика и автотестера №2

**Достоинства:**

* Автоматизированные тесты разрабатываются и поддерживаются на высокой скорости.
* Команда по автоматизации тестирования способна надёжно протестировать практически любой крупный проект

**Недостатки:**

* Большие затраты на тестирование
* Приложение практически не проверяется в ручном режиме, из-за этого можно получить ошибки, которые не были предусмотрены автотестом

**Решение:**

Комбинированный подход между ручным и автоматизированным тестированием. Внедрение новой функциональности в автоматизированные тесты, которая позволит специалистам по ручному тестированию писать автоматические тесты.

**Глава 1.4 Разработка структуры приложения для автоматизированного тестирования с гибридным подходом.**

На основе анализа подходов к решению проблемы тестирования программного обеспечения в крупных компаниях, стало понятно, что в большинстве компаний процесс тестирования является не оптимизированным и низкоэффективным. Имеется ряд недостатков, таких как скорость и стоимость данного этапа.

Определенно точно необходим некий подход, который обеспечит непрерывный и быстрый процесс тестирования приложения с минимальными затратами на его реализацию.

На основе обзора и анализа предлагаю следующую реализацию.

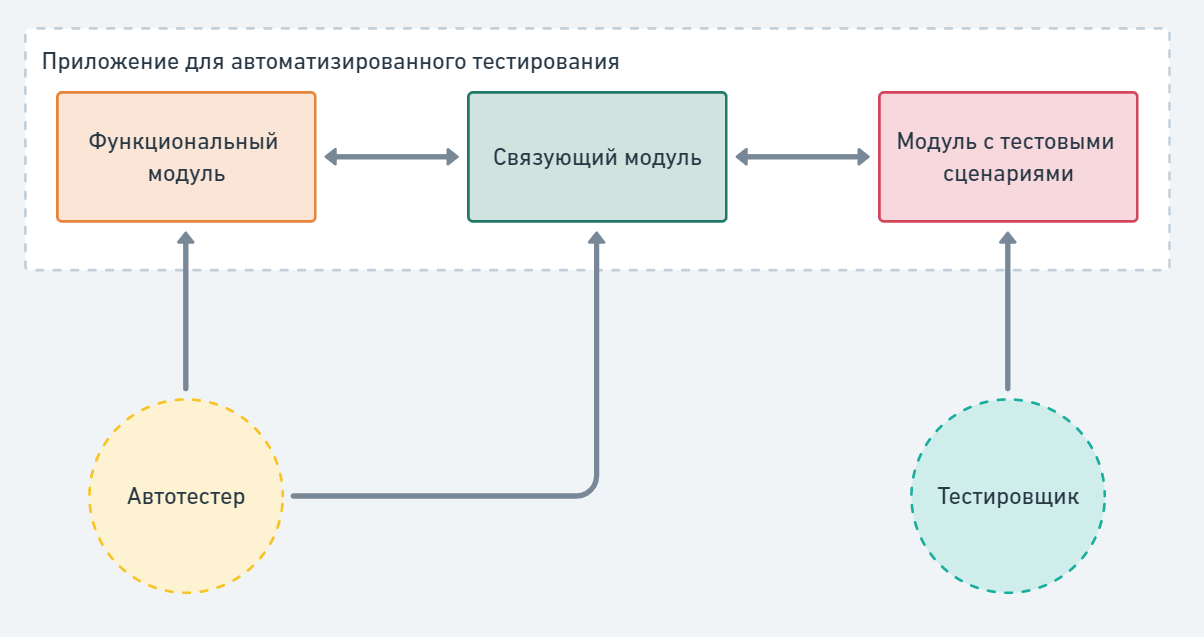


Рис. 8 Схема взаимодействия тестировщика и автотестера

Предлагается разработать приложение из трёх компонентов:

* Функциональный модуль. Содержит основные функциональные возможности автоматизированного теста.
* Модуль с тестовыми сценариями. Содержит тестовые сценарии.
* Связующий модуль. Преобразует тестовые сценарии в полноценный автоматизированный тест.

Предлагаемое приложение позволит специалистам по ручному тестированию функционально конструировать автоматизированные тесты.

Специалист по автоматизированному тестированию реализует функциональный и связующий модули, которые предоставят готовый конструктор по внедрению новых автоматизированных тестов. Обеспечит стабильную работу этих модулей и займётся их поддержкой.

Специалисты по ручному тестированию напишут тестовые сценарии для проверки работоспособности приложения. Сценарии станут основой для вызова функционального модуля, который и обеспечит работу теста.

Целевое решение имеет ряд преимуществ, таких как:

* Тестовые сценарии, написанные на русском языке
* Высокая скорость тестирования.
* Низкая стоимость тестирования.
* Повышение эффективности работы специалистов по ручному тестированию.
* Повышение эффективности работы специалистов по автоматизации тестирования.
* Конструктор для создания автоматизированных тестов.

**Выводы**

В ходе практики были проанализированы бизнес процессы разработки, тестирования и релиза. Выявлены узкие места. Был разобран процесс тестирования современных приложений, выявлены проблемы. Проведен анализ текущих сложностей в проведении тестирования. Предложено решение проблем проведения тестирования в крупных компаниях, была построена схема взаимодействия специалистов ручного и автоматизированного тестирования в рамках нового решения. Спроектированы и описаны ключевые модули целевого решения.

**Источники информации**

1. Основы тестирования программного обеспечения. Учебное пособие для СПО, 2-е изд., стер.
2. Software Testing Techniques, 2nd Edition, Борис Бейзер
3. Scrum: Революционный метод управления проектами, Джефф Сазерленд
4. Experiences of Test Automation: Case Studies of Software Test Automation 1st Edition, Dorothy Graham