TOMSK POLYTECHNIC UNIVERSITY

ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа инженерного предпринимательства
Направление подготовки 27.04.05 Инноватика
ООП/ОПОП Прикладной системный инжиниринг

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТРАНТА

Тема работы
Автоматизация этапов жизненного цикла процесса тестирования программного обеспечения

УДК 005.41:004.415.2:004.415.53

Обучающийся

Группа	ФИО	Подпись	Дата
3НМ15	**Смолякова К.В.**		

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ШИП	**Хаперская А.В.**	**к.пед.н.**		

КОНСУЛЬТАНТЫ ПО РАЗДЕЛАМ:

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ШИП	**Черепанова Н.В.**	**к.филос.н.**		

Нормоконтроль

Должность	ФИО	Ученая степень, звание	Подпись	Дата
ст. преподаватель	**Громова Т.В.**	**-**		

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП/ОПОП,должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ШИП	**Жданова А.Б.**	**к.э.н.**		

Томск – 2023 г.

Планируемые результаты освоения ООП

27.04.05 Инноватика
(Прикладной системный инжиниринг)

Код компетенции	Наименование компетенции
	Универсальные компетенции
УК(У)-1	Способен осуществлять критический анализ проблемных ситуаций на основе системного подхода, вырабатывать стратегию действий
УК(У)-2	Способен управлять проектом на всех этапах его жизненного цикла
УК(У)-3	Способен организовать и руководить работой команды, вырабатывая командную стратегию для достижения поставленной цели
УК(У)-4	Способен применять современные коммуникативные технологии, в том числе на иностранном(ых) языке(ах), для академического и профессионального взаимодействия
УК(У)-5	Способен анализировать и учитывать разнообразие культур в процессе межкультурного взаимодействия
УК(У)-6	Способен определить и реализовать приоритеты собственной деятельности и способы ее совершенствования на основе самооценки
	Общепрофессиональные компетенции
ОПК(У)-1	Способен анализировать и выявлять естественно-научную сущность проблем управления в технических системах на основе положений, законов и методов в области математики, естественных и технических наук.
ОПК(У)-2	Способен формулировать задачи управления в технических системах и обосновать методы их решения
ОПК(У)-3	Способен самостоятельно решать задачи управления в технических системах на базе последних достижений науки и техники
ОПК(У)-4	Способен разрабатывать критерии оценки систем управления в области инновационной деятельности на основе современных математических методов, вырабатывать и реализовывать управленческие решения по повышению их эффективности
ОПК(У)-5	Способен проводить патентные исследования, определять формы и методы правовой охраны и защиты прав на результат интеллектуальной деятельности, распоряжаться правами на них для решения задач в области развития науки, техники и технологии.
ОПК(У)-6	Способен осуществлять сбор и анализ научно-технической информации, обобщать отечественный и зарубежный опыт в области управления инновациями и построения экосистем инноваций
ОПК(У)-7	Способен аргументировано выбирать и обосновывать структурные, алгоритмические, технологические и программные решения для управления инновационными процессами и проектами, реализовывать их на практике применительно к инновационным системам предприятия, отраслевым и региональным инновационным систем.
ОПК(У)-8	Способен выполнять эксперименты на действующих объектах по заданным методикам и обрабатывать результаты с применением современных информационных технологий и технических средств
ОПК(У)-9	Способен решать профессиональные задачи на основе истории и философии нововведений, математических методов и моделей для управления инновациями, знаний особенностей формирующихся

Код компетенции	Наименование компетенции
	технологических укладов и четвертой промышленной революции в инновационной сфере
ОПК(У)-10	Способен разрабатывать, комбинировать и адаптировать алгоритмы и программные приложения, пригодные для решения практических задач цифровизации в области профессиональной деятельности
ОПК(У)-11	Способен разрабатывать учебно-методические материалы и участвовать в реализации образовательных программ в области образования
	Профессиональные компетенции
ПК(У)-1	Способность осуществлять разработку и реализацию стратегии продвижения проекта компании в цифровой среде на основе комплексного анализа рынка

TOMSK POLYTECHNIC UNIVERSITY

ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа инженерного предпринимательства
Направление подготовки 27.04.05 Инноватика
ООП/ОПОП Прикладной системный инжиниринг

УТВЕРЖДАЮ:
Руководитель ООП/ОПОП
_____ _____ _Жданова А.Б.
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ

на выполнение выпускной квалификационной работы

Обучающийся:

Группа	ФИО
3НМ15	**Смолякова Кристина Викторовна**

Тема работы:

Автоматизация этапов жизненного цикла процесса тестирования программного обеспечения
Утверждена приказом директора (дата, номер)

Срок сдачи обучающимся выполненной работы:	

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к функционированию (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.)*	**Объект исследования:** бизнес-процессы жизненного цикла тестирования программного обеспечения.
Перечень разделов пояснительной записки подлежащих исследованию, проектированию и разработке *(аналитический обзор литературных источников с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов*	1. Описание и моделирование бизнес-процессов тестирования программного обеспечения; 2. Имитационное моделирование и анализ бизнес-процессов тестирования программного обеспечения; 3. Разработка плана мероприятий по оптимизации бизнес-процессов программного обеспечения; 4. Внедрение мероприятий по оптимизации бизнес-процессов тестирования программного обеспечения;

выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе)	5. Расчет эффективности внедрения мероприятий по оптимизации бизнес-процессов тестирования программного обеспечения.
Перечень графического материала *(с точным указанием обязательных чертежей)*	Рисунок 1 – Структура компании ООО «Neo Stack Technology». Рисунок 2 – Бизнес-процесс подготовительного этапа. Рисунок 3 – Бизнес-процесс этапа анализа требований для коммерческого проекта. Рисунок 4 – Бизнес-процесс этапа анализа требований для продуктового проекта. Рисунок 5 – Бизнес-процесс этапа разработки тестов. Рисунок 6 – Бизнес-процесс этапа тестирования. Рисунок 7 – Бизнес-процесс этапа подготовки документации. Таблица 1 – Вопросы для проведения интервью у специалистов отдела. Таблица 2 – Настройка имитационной модели для подготовительного этапа. Таблица 3 – Настройка имитационной модели для этапа анализа требований коммерческого проекта. Таблица 4 – Настройка имитационной модели для этапа анализа требований продуктового проекта. Таблица 5 – Настройка имитационной модели для этапа разработки тест-кейсов. Таблица 6 – Настройка имитационной модели для этапа тестирования. Таблица 7 – Настройка имитационной модели для этапа подготовки документации.

Консультанты по разделам выпускной квалификационной работы *(с указанием разделов)*	
.Раздел	**Консультант**
Социальная ответственность	Черепанова Н.В.
Названия разделов, которые должны быть написаны на иностранном языке:	
1 Теоретические аспекты тестирования программного обеспечения	
1 Theoretical aspects of software testing	

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ШИП	**Хаперская А.В.**	**к.пед.н.**		

Задание принял к исполнению обучающийся:

Группа	ФИО	Подпись	Дата
3НМ15	**Смолякова Кристина Викторовна**		

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа инженерного предпринимательства

Направление подготовки (ООП/ОПОП) 27.04.05 Инноватика (Прикладной системный инжиниринг)

Уровень образования - магистратура

Период выполнения (весенний семестр 2022/2023 учебного года)

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Обучающийся:

Группа	ФИО
3НМ15	**Смолякова Кристина Викторовна**

Тема работы:

Автоматизация этапов жизненного цикла процесса тестирования программного обеспечения

Срок сдачи обучающимся выполненной работы:	

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
13.03.2023	Теоретическая часть	20
24.05.2023	Практико-ориентированная часть	60
25.05.2023	Социальная часть	10
03.06.2023	Часть на английском языке	10
Итог		100

СОСТАВИЛ:

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ШИП	**Хаперская А.В.**	**к.пед.н.**		

СОГЛАСОВАНО:

Руководитель ООП/ОПОП

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ШИП	**Жданова А.Б.**	**к.э.н.**		

Обучающийся

Группа	ФИО	Подпись	Дата
3НМ15	**Смолякова К.В.**		

Реферат

Выпускная квалификационная работа содержит 123 страницы, 40 рисунков, 16 таблиц, 10 использованных источников, 2 приложения.

Ключевые слова: тестирование программного обеспечения, жизненный цикл тестирования программного обеспечения, бизнес-процесс, имитационное моделирование, оптимизация бизнес-процессов.

Объектом исследования являются бизнес-процессы жизненного цикла программного обеспечения на примере отдела контроля качества (отдела тестирования) компании ООО «Neo Stack Technology».

Цель работы: анализ и оптимизация бизнес-процессов тестирования программного обеспечения на примере отдела контроля качества компании ООО «Neo Stack Techology».

В ходе работы проводились обзор и анализ бизнес-процессов тестирования программного обеспечения отдела тестирования и их имитационное моделирование.

В результате анализа имитационного моделирования был разработан и внедрен план мероприятий по оптимизации бизнес-процессов тестирования программного обеспечение и рассчитана эффективность от внедрения мероприятий по оптимизации бизнес-процессов тестирования ПО.

Область применения: тестирование программного обеспечения.

Экономическая эффективность работы: расчеты эффективности подтверждают целесообразность внедрения мероприятий по оптимизации бизнес-процессов отдела тестирования.

В будущем планируется применять имитационное моделирования с целью оптимизации бизнес-процессов для других циклов разработки ПО.

Содержание

Введение

Большинство компаний, которые имеют в своей структуре IT-отдел задаются вопросом об ускорении процесса разработки, не теряя его качество. Ни один цикл разработки программного обеспечения не обходится без этапа тестирования. Тестирование программного обеспечения – процесс анализа программного средства и сопутствующей документации с целью выявления дефектов и повышения качества продукта. С развитием индустрии разработки программного обеспечения и усложнением технологического процесса, появлением комплексных методов решения задач, роль тестирования возросла. Вместо одной из финальных стадий создания проекта тестирование стало применяться на протяжении всего цикла разработки. В дальнейшем были созданы гибкие методики тестирования, разнообразные оптимизации и была углублена интеграция с процессом разработки. В данный момент в индустрии имеется огромный интерес к тестированию, как таковому, так и к методам улучшения качества программ в целом, ведь конечной целью любого процесса тестирования является обеспечение качества. Тестирование программного обеспечения позволяет определить, выполняет ли программа то, что от неё ожидают. А основной задачей тестирования программного обеспечения является снижение стоимости разработки путем раннего обнаружения дефектов.

Раннее обнаружение дефектов и своевременная обратная связь позволяют создавать высококачественное и надежное программное обеспечение. Кроме того, организация эффективного и непрерывного тестирования ускоряет поставку этого программного обеспечения и снижает затраты компании по причине того, что к разработчикам появляется возможность вносить в код изменения с минимальными рисками, которые способны нарушить работоспособность программы. Именно поэтому основной целью выпускной квалификационной работы являлось анализ и оптимизация бизнес-процессов тестирования программного обеспечения на

примере отдела контроля качества компании ООО «Neo Stack Techology». Для достижения поставленной цели были сформулированы следующие задачи:

1. Обзор бизнес-процессов тестирования программного обеспечения;

2. Анализ бизнес-процессов тестирования программного обеспечения;

3. Мониторинг работы участников отдела тестирования;

4. Моделирование бизнес-процессов тестирования программного обеспечения;

5. Проведение имитационного моделирования бизнес-процессов тестирования программного обеспечения;

6. Анализ результатов имитационного моделирования бизнес-процессов тестирования программного обеспечения;

7. Разработка плана мероприятий по оптимизации бизнес-процессов тестирования программного обеспечения;

8. Внедрение мероприятий по оптимизации бизнес-процессов тестирования программного обеспечения;

9. Расчет эффективности от внедрения мероприятий по оптимизации бизнес-процессов тестирования программного обеспечения.

Так объектом исследования являются бизнес-процессы жизненного цикла программного обеспечения на примере отдела контроля качества (отдела тестирования) компании ООО «Neo Stack Technology».

Практическая значимость исследования состоит в разработке и внедрении мероприятий по оптимизации бизнес-процессов тестирования программного обеспечения, которые позволят сократить время и стоимость процесса тестирования, не теряя качества выпускаемого программного продукта.

Определения, обозначения, сокращения

IT – информационные технологии;

ООО – общество с ограниченной ответственностью;

STLC – жизненный цикл тестирования программного обеспечения;

ПО – программное обеспечение;

ОКК – отдел контроля качества;

ЖЦ – жизненный цикл;

ФС – фича-страница;

ПМИ – программа и методика испытаний.

В данной работе применены следующие термины с соответствующими определениями:

тестирование программного обеспечения: Процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определённым образом.

бизнес-процесс: Совокупность взаимосвязанных мероприятий или работ, направленных на создание определённого продукта или услуги для потребителей.

имитационное моделирование: Метод исследования, при котором изучаемая система заменяется моделью, с достаточной точностью, описывающей реальную систему, с которой проводятся эксперименты с целью получения информации об этой системе.

bpmn: Язык это язык моделирования бизнес-процессов, который является промежуточным звеном между формализацией/визуализацией и воплощением бизнес-процесса.

bpm: Среда, в которой вы непосредственно участвуете в моделировании.

1 Теоретические аспекты тестирования программного обеспечения

1.1 Определение проблемы

Согласно данным аналитического отчета по российскому рынку тестирования программного обеспечения от компании «PerfomanceLab» на 2020-2021 [1] годы 80% респондентов считают, что главной целью отдела тестирования является повышение качества ИТ-продуктов и 69% нацелены на повышение удовлетворенности пользователей. При этом цель по сокращению времени вывода продуктов на рынок занимает третье место и интересует 62% респондентов, что на 26% выше, чем в 2018 году.

Большое количество компаний стремятся сократить выдачу продукта при этом не потерять качество и увеличить удовлетворенность пользователей. По результатам опроса можно сказать, что такая потребность у компаний с каждым годом только увеличивается. Именно оптимизация процессов тестирования может позволить решить проблему быстрой выдачи продукта без потери его качества.

В статье Шакировой А.И. [2], которая посвящена сокращению времени тестирования программного обеспечения рассматривается метод повышения эффективности тестирования ПО, позволяющий сократить время тестирования без потери качества. Для этого метода проводились опросы у 17-ти команд, которые занимаются тестированием для того, чтобы выяснить как в этих командах выстроен процесс тестирования, а именно какой вид тестирования используется и сколько человек участвует в тестировании. В результате исследования автором статьи были разработаны рекомендации, которые позволяют повысить эффективность каждого рассмотренного вида тестирования.

В данной работе будет предложен и описан алгоритм по оптимизации процесса тестирования с учетом опыта, методов оценки эффективности и

расчета экономики на затраты в процессе тестирования, описанные в статье Шакировой А.И.

1.2 Жизненный цикл тестирования программного обеспечения

Тестирование является неотъемлемой частью современной разработки программного обеспечения. Хотя существует очень много подходов к разработке программного обеспечения, тестирование необходимо везде.

Тестирование имеет свой собственный жизненный цикл, который называют жизненным циклом тестирования программного обеспечения или STLC. Жизненный цикл тестирования программного обеспечения помогает в проведении процесса тестирования надлежащим и тщательным образом.

Тестирование программного обеспечения, как и разработка программного обеспечения, включает в себя несколько шагов, следующих в определенной последовательности. Это и называется жизненным циклом тестирования программного обеспечения. Он определяет начало, конец и промежуточные моменты полного процесса тестирования ПО.

Каждый этап STLC имеет определенные результаты, цели и задачи. Хотя разные тестировщики могут по-разному подходить к этапам, таким как объединение одного или нескольких в один, повторение и т.д. основная идея остается той же. Ниже приведены основные фазы жизненного цикла тестирования программного обеспечения, которые используются независимо от выбранного подхода [3]:

- анализ требований;
- планирование тестирования;
- разработка тестов;
- выполнение тестирования;
- подготовка документации.

Каждый из данных этапов так или иначе является обязательным для жизненного цикла тестирования, вне зависимости от того, насколько он

формализован и задокументирован. Далее каждый этап будет рассмотрен более подробно.

Анализ требований.

Анализ требований представляет собой активности по исследованию новых изменений, анализу документации к ним, выполнению статического тестирования требований при необходимости. Чаще всего данный этап начинается после завершения согласования технических требований и спецификаций аналитиком команды. Требования могут быть описаны в явном формализованном виде, а могут быть представлены в виде user story. На данном этапе у тестировщика есть возможность повлиять на реализацию будущих изменений с учетом его виденья системы и процесса в целом с точки зрения тестирования.

На данном этапе тестировщик получает понимание об изменениях, которые планируются к разработке и внедрению. Результатом является фактически сформированный чек-лист тестовых требований, который предстоит проверить тестировщику. Это может быть, как формализованный документ, так и просто знания в голове конкретного специалиста по тестированию.

Планирование тестирования.

На этапе планирования тестировщик определяется с целями тестирования, необходимыми типами тестирования, требованиями к тестовому окружению (тестовой среде), определяет критерии начала и завершения каждого вида тестирования, оценивает предварительные сроки тестирования на основе полученных ранее тестовых требований, идентифицирует потенциальные риски и совместно с другими участниками команды прорабатывает варианты их минимизации, а также любые иные специфические требования к тестированию, которые могут возникнуть в результате выполнения тестов.

Результатом данного этапа может быть формализованный тест-план, который описывает все моменты, перечисленные выше. Данный документ

чаще всего согласовывается дополнительно тестировщиком со всеми заинтересованными участниками проекта. Бывает, что такой документ не формализуется, ы все аспекты проговариваются внутри команды на встречах по планированию и фиксируются в протоколах данных встреч. Это значительно сокращает процесс планирования тестирования, так как есть возможность сразу обсудить открытые вопросы со всеми участникам команды и найти подходящее решение. Результатом в данном случае будет только оценка активностей по тестированию.

После завершения этапа планирования тестировщик приступает в одному из ключевых этапов процесса тестирования – это разработка тестов.

Разработка тестов.

После проработанного тест-плана тестировщик приступает к разработке тест-кейсов и их наполнению. Данный этап может занимать достаточно продолжительное время, особенно если речь идет о большом проекте. Тестировщик первым делом составляет чек-лист проверок, формируют структуру тест-кейсов и после приступает к их наполнению шагами, различными условиями, а также определению тестовых данных для их успешного выполнения. Кроме того, на данном этапе могут определяться тест-кейсы, которые будут подлежать дальнейшей автоматизации. Существует подход, когда тестировщик создает для себя чек-лист проверок с тестовыми данными и условиями, но не переходит к написанию полноценных тест-кейсов.

Результатом данного этапа является сформированное детальное понимание того, что необходимо будет тестировать. Это могут быть полноценный тест-кейсы, чек-листы или виденье того, что нужно будет тестировать.

Помимо этого, необходимо выполнить подготовку к тестированию. Подготовка к тестированию имеет следующие активности:

– Деплой версии приложения с необходимыми для тестирования изменениями;

– Создание тестового окружения с необходимой конфигурацией;

– Подготовка тестовых данных для тестирования;

– Доработка скриптов автоматизации тестирования.

В качестве результата тестировщик получает работоспособную тестовую среду с тестовыми данными и установленным приложением для тестирования.

Выполнение тестирования.

Данный этап предполагает выполнение последовательно нескольких типов тестирования до момента получения положительного заключения и успешного прохождения тестов. Сначала выполняется интеграционное или даже сразу системное тестирование, зависит от специфики тестирования в конкретной компании. После завершения системного тестирования проводится регрессионное тестирование. Чаще всего за счет автоматизации тестирования, тестировщик выполняет только интеграционное и системное тестирования, после чего полностью или частично запускаются автоматизированные регрессионные тесты. Приемочное тестирование выполняется, либо на стороне заказчика, либо на специально отведенной для этого встречи, которую часто называют «Демо». В ходе этой встречи тестировщик показывает реализацию функционала системы, и заказчик принимает данные изменения, либо отправляет на повторную доработку.

Результатами данного этапа можно считать успешно пройденные тест-кейсы или проверки по чек-листу.

Подготовка документации.

Последним заключительным этапом тестирования перед внедрением новых изменений на продуктивные среды – является отчетность о результатах тестирования. Данный этап предполагает создание формальных отчетов по результатам тестирования изменений, который может включать в себя общие результаты тестирования, открытые проблемы и дефекты, а также рекомендации для сопровождения по установки изменений на продуктивную среду.

На этом в большинстве случаев процесс тестирования завершается и изменение передается для установки на продуктивную среду конечным пользователям или заказчикам.

1.3 Виды тестирования программного обеспечения

Виды тестирования принято классифицировать по разным признакам [4]:

– по доступности кода;

– по объекту тестирования;

– по позитивности сценариев;

– по целям тестирования;

– по степени автоматизации;

– по исполнению кода;

– по уровню тестирования;

– связанные с изменениями;

– по исполнителям тестирования;

– и т.д.

В разных источниках классификация может отличаться по признакам, но общая суть видов тестирования остается такой же. В данной работе будут рассмотрены виды по уровню тестирования, по исполнению кода и связанные с изменениями.

Тестирование по исполнению кода разделяют: статическое и динамическое [5].

Статистическое тестирование – тип тестирования, который предполагает, что программный код во время тестирования не будет выполняться. Статическое тестирование начинается на ранних этапах жизненного цикла ПО и является, соответственно, частью процесса анализа требований. Большинство статических техник могут быть использованы для «тестирования» любых форм документации, включая вычитку кода,

инспекцию проектной документации, функциональной спецификации и требований.

Динамическое тестирование – тип тестирования, который предполагает запуск программного кода. Таким образом, анализируется поведение программы во время ее работы. Для выполнения динамического тестирования необходимо, чтобы тестируемый программный код был написан, скомпилирован и запущен. Кроме того, динамическое тестирование может включать разные подвиды, например, уровни тестирования.

Тестирование на разных уровнях производится на протяжении всего жизненного цикла разработки и сопровождения программного обеспечения. Уровень тестирования определяет то, над чем производятся тесты: над отдельным модулем, группой модулей или системой, в целом. Выделяют четыре основных уровня тестирования: компонентное или модульное, интеграционное, системное и приемочное [5].

Компонентное или модульное тестирование проверяет функциональность и ищет дефекты в частях приложения, которые доступны и могут быть протестированы по-отдельности, например, модули программ, объекты, классы, функции и т.д. Обычно компонентное тестирование проводится, вызывая код, который необходимо проверить и с использованием сред разработки. Все найденные дефекты, как правило исправляются в коде без формального их описания.

Интеграционное тестирование предназначено для проверки связи между компонентами, а также взаимодействия с различными частями системы, либо связи между различными системами. Интеграционное тестирования в свою очередь делится на уровни:

– компонентный интеграционный уровень проверяет взаимодействие между компонентами системы после проведения компонентного тестирования;

– системный интеграционный уровень проверяет взаимодействие между разными системами после проведения системного тестирования.

Основной задачей системного тестирования является проверка как функциональных, так и не функциональных требований всей системы в целом. При системном тестировании выявляется неверное использование ресурсов системы, несовместимость с окружением, непредусмотренные сценарии использования, отсутствующая или неверная функциональность, неудобство использования и т.д. Для минимизации рисков, связанных с особенностями поведения в системы в той или иной среде, во время тестирования рекомендуется использовать тестовое окружение, максимально приближенное к тому, на которое будет установлен продукт после выдачи.

Выделяют два подхода к системному тестированию:

– на базе требований, когда для каждого требования пишутся тест-кейсы, проверяющие выполнение данного требования;

– на базе случаев использования, когда на основе представления о способах использования продукта создаются тест-кейсы для проверки каждого сценария.

Приемочное тестирование – это формальный процесс тестирования, который проверяет соответствие системы требованиям и проводится с целью:

– определения удовлетворения системой приемочным критериям;

– вынесения решения заказчиком или другим уполномоченным лицом о принятии приложения.

Приемочное тестирование выполняется на основании набора типичных тестовых сценариев, разработанных на основании требований к данному приложению. Фаза приемочного тестирования длится до тех пор, пока заказчик не выносит решение о выдаче приложения или отправлении его на доработку.

В основном тестировщики участвуют в интеграционном, системном и приемочном тестированиях. Каждый уровень тестирования используется при видах тестирования связанных с изменениями.

После проведения необходимых изменений, таких как исправление дефектов, программное обеспечение должно быть протестировано для

подтверждения того факта, что проблема была действительно решена. Ниже перечислены виды тестирования, которые необходимо проводить после установки программного обеспечения, для подтверждения работоспособности приложения или правильности осуществленного исправления дефекта [5].

Дымовое тестирование пошло из инженерной среды: «При вводе в эксплуатацию нового оборудования считалось, что тестирование прошло удачно, если из установки не пошел дым.»

В области программного обеспечения дымовое тестирование рассматривается как короткий цикл тестов, выполняемый для подтверждения того, что, после сборки кода (нового или исправленного), устанавливаемое приложение стартует и выполняет основные функции.

Вывод о работоспособности основных функций делается на основании результатов поверхностного тестирования наиболее важных модулей приложения на предмет возможности выполнения требуемых задач и наличия критических и блокирующих дефектов. В случае отсутствия таковых дефектов дымовое тестирование объявляется пройденным и приложение передается для проведения полного цикла тестирования, в противном случае, дымовое тестирование объявляется проваленным и приложение уходит на доработку.

Регрессионное тестирование – это вид тестирования, направленный на проверку изменений, сделанных в приложении или окружающей среде, например, исправление дефекта, слияние кода, миграция на другую операционную систему или базу данных, для подтверждения того факта, что существующая ранее функциональность работает, как и прежде.

Как правило, для регрессионного тестирования используются тест-кейсы, написанные на ранних стадиях разработки и тестирования. Это дает гарантию того, что изменения в новой версии приложения не повредили уже существующую функциональность. Сам по себе термин регрессионного тестирования, в зависимости от контекста использования, может иметь разный смысл:

– регрессия багов – попытка доказать, что исправленная ошибка на самом деле не исправлена;

– регрессия старых багов – попытка доказать, что недавнее изменение кода или данных сломало исправление старых ошибок, т.е. старые баги стали снова воспроизводиться;

– регрессия побочного эффекта – попытка доказать, что недавнее изменение кода или данных сломало другие части разрабатываемого приложения.

1.4 Виды тестовой документации

Тестовая документация – это тип документации, которая описывает процесс, цели и результаты тестирования программного обеспечения [6]. Она также может содержать информацию о среде, настройках и конфигурации, необходимых для выполнения тестирования. Тестовая документация используется для доведения деталей плана тестирования или стратегии до сведения заинтересованных сторон, разработчиков и тестировщиков.

Некоторые компании утверждают, что тестовая документация – это необязательный процесс, который только увеличивает затраты на разработку и больше работает на имидж компании-разработчика программного обеспечения, чем на обеспечение реальной ценности. Но правда в том, что тестовая документация является неотъемлемой частью процесса тестирования, которая должна предшествовать, сопровождать и завершать любой бизнес-процесс тестирования.

Основными целями тестовой документации являются:

– управление будущими работами по тестированию и демонстрация заинтересованным сторонам результатов тестирования для того, чтобы они имели возможность принимать обоснованные решения относительно работ по разработке программного обеспечения;

– предотвращение пробелов и проблем в процессах тестирования или дублирования в проведении тестирования, чтобы убедиться, что выполнены все необходимые тесты.

Тестовая документация позволяет сделать тестирование прозрачным, так как документация по тестированию показывает все аспекты тестирования, от постановки целей до методов тестирования и результатов. Это дает заказчику четкое представление о том, какая работа была проделана, и какие результаты были получены. Это также полезно для других тестировщиков, которым передается задача, и для разработчиков, которым приходится решать обнаруженные дефекты или проблемы.

Также документация улучшает коммуникацию в команде, потому что количество ошибок при общении значительно сокращается, если у команды есть единый источник информации о функциональности продукты или даже о процессе работы над проектом. Бывают ситуации, когда все понимают ситуацию по-разному и в таком случае есть на что сослаться и что обсудить. Также нет необходимости совершать лишние звонки, чтобы получить необходимую информацию, потому что она постоянно обновляется, и каждый знает, где искать ее текущую версию. И, конечно, это очень помогает, когда в команду приходят новые люди или, когда команда тестирования меняется в целом, что без качественной документации может быть невозможно.

При надлежащем документировании тестирования риск возникновения неожиданных проблем значительно снижается. Хорошо составленная тестовая документация отображает реальный опыт разработки, на основе которого можно сделать много ценных вещей и использовать лучшие практики на других проектах. Документацию можно использовать для обучения новых членов команды, поиска лучших практик и обмена ими, проведения аргументированных дискуссий и многого другого.

Хорошая тестовая документация является отличной демонстрацией того, как проводится фундаментальное тестирование, какие используются подходы и как тестируются различные части программного продукта.

Существует множество тестовых документов, и они могут соответствовать самым разнообразным стандартам. Наиболее распространенным типом тестовой документации является план тестирования. В плане тестирования описывается подход, который будет применен для тестирования конкретной программной системы. Он включает в себя информацию о том, что будет тестироваться, как это будет тестироваться и кто будет отвечать за проведение тестов.

Другие типы тестовой документации включают тест-кейсы, тестовые сценарии и отчеты о дефектах. Тестовые примеры описывают конкретные шаги, которые необходимо предпринять для тестирования конкретной функции. Тестовые сценарии - это более подробные инструкции, в которых точно описывается, как следует проводить тест. Отчеты о дефектах документируют любые ошибки, обнаруженные во время тестирования.

Типы тестовой документации зависят от конкретной компании, продукта и заказчика. Далее будет подробнее рассмотрены наиболее распространенные типы тестовой документации.

Внутренние тестовые документы необходимы для выполнения рабочих задач и для использования членами команды разработки и тестирования. Он содержит цели, методы и результаты тестирования на техническом уровне.

Стратегия тестирования – это документ, в котором излагаются основные аспекты тестирования, в частности, на каких уровнях проекта оно будет выполняться. Во время тестирования разработчики, проектировщики и менеджеры могут вернуться к этому документу в любое время, чтобы убедиться, что тестирование по-прежнему проводится в рамках первоначальной стратегии и выделенной области.

План тестирования – это документ, в котором подробно описываются все основные аспекты проекта тестирования, такие как объем, подход, ресурсы, расписание, участники тестирования и их деятельность. Поскольку это самый простой документ, он используется чаще всего, распространяется среди всей команды и доводится до сведения клиентов.

Чек-лист – это документ, содержащий краткое описание функций, которые должен проверить тестировщик. Выглядит чек-лист как список функций с указанием результата проверки. Чек-листы могут использоваться вместо тест-кейсов, поскольку их легче подготовить. Но если необходимо более конкретное описание процедуры, без тест-кейсов не обойтись.

Тест-кейс – это документ, в котором содержится подробное описание шагов и действий, которые тестировщик должен выполнить для тестирования какой-то одной части функционала, а также критерии прохождения тестов. Компании могут использовать разные форматы тест-кейсов, но информация в них всегда очень подробная и конкретная [7].

Тестовые данные – это документ, который описывает данные, необходимые для тестирования, чтобы приблизить производительность приложения к реальному использованию. Например, это могут быть наборы сгенерированных пользователей, документы, медиафайлы, метаданные и все остальное, с чем приложение должно работать в реальном мире.

Внешние тестовые документы необходимы для наглядного представления конечных результатов, поскольку использование различных типов визуализаций облегчает их понимание. Они часто предоставляются заказчикам и могут быть предоставлены пользователям в той или иной форме.

Отчеты об ошибках – это документ, который описывает конкретную ошибку в приложении во всех ее возможных аспектах. Это очень важный и часто публикуемый отчет, который непосредственно помогает улучшить продукт в конкретных аспектах.

Отчет по тестированию – это документ, в котором обобщаются результаты цикла тестирования. Он может содержать стоимость обнаружения дефектов, эффективность набора тестов, результативность тестирования, соотношение усилий по доработке и проверке и т.д.

Отчет о приемке пользователем – это документ, содержащий результаты тестирования перед отправкой заказчику на соответствие требованиям. Это гарантирует, что взгляды разработчиков и заказчиков

остаются идентичными во время разработки и тестирования и что техническая реализация полностью соответствует им.

Для того, чтобы документация работала положительно для команды и проекта и приносила пользую необходимо соблюдать следующие правила.

Поддержание документации в актуальном состоянии. Распространенной ошибкой является прекращение обновления документации, даже если с самого начала вы выполняли всю документацию очень хорошо. И это кажется незначительным, но только до тех пор, пока документация не понадобится проекту или команде.

Хранить всю документацию в одном месте. Документация не должна быть разбросана по разным местам в попытке сохранить ее близкой к соответствующей кодовой базе. Это может показаться хорошей идеей, но это может сбить с толку членов команды и создать путаницу в практическом смысле.

Хранить документацию в надежном месте. Не размещать в открытом доступе никакую документацию, которая может негативно повлиять на бизнес-цели. В этом случае может понадобится зашифрованное хранилище и возможность предоставлять доступ определенным лицам.

Использовать стандартные шаблоны для документации. Лучше всего использовать универсальные решения для документации, такие как Word и Excel. Это позволит любому заинтересованному лицу легко открывать и читать их без установки дополнительного программного обеспечения или других сложностей.

Сделать документацию подробной. Документация должна быть хорошо структурированной, содержательной, подробной и ясной. Все участники процесса, использующие документацию должны понимать ее однозначно.

Набор тестовой документации может варьироваться от проекта к проекту. Но существуют неизменяемые тестовые документы, качественное

ведение которых дает компании, ее сотрудникам и клиентам множество преимуществ.

2 Анализ и моделирование бизнес-процессов тестирования программного обеспечения

2.1 Деятельность компании ООО «Neo Stack Technology»

Компания Neo Stack Technology (NST) или НСТ занимается разработкой программного обеспечения на заказ, а также имеет собственную продуктовую линейку. Компания существует на рынке более 12 лет, имеет более 250 реализованных проектов, более 80 высококвалифицированных инженеров и менеджеров и более 8 собственных проектов. Основная деятельность компании:

– разработка ПО "под ключ" (концепт, архитектура, аналитика, разработка, тестирование);

– автоматизация и диджитал-трансформация предприятий и процессов (3D, CAD, GIS);

– разрабатываем и имплементируем свои продукты по автоматизации;

– IT-консалтинг.

Помимо этого, компания NST активно сотрудничает с университетами Томска, участвует в митапах и конференциях на локальном и международном уровнях.

Продуктовая линейка компании – это геоинформационные решения GeoSolution, которые предназначены для сбора, хранения, обработки, преставления и моделирования пространственных данных, и связанной с ними атрибутивной информации. Данная линейка имеет следующие продукты [8]:

– NeoPortal – веб-решение для организации и оптимизации работы с геоданными предприятия в виртуальной среде;

– NeoBim – веб-решение для управления техническим состоянием объекта и получения информации о 3D или BIM модели для принятия решений в течении всего жизненного цикла;

– NeoRent – решение для мониторинга и контроля арендных платежей по земельным отводам;

– NeoQuarry – веб-приложение для решения задач мониторинга жизненного цикла карьеров, хранения и доступа к документации;

– NeoWell – решение для сбора, систематизации, хранения и управления информацией об объектах маркшейдерского обеспечения.

Структура компании представлена на рисунке 1.

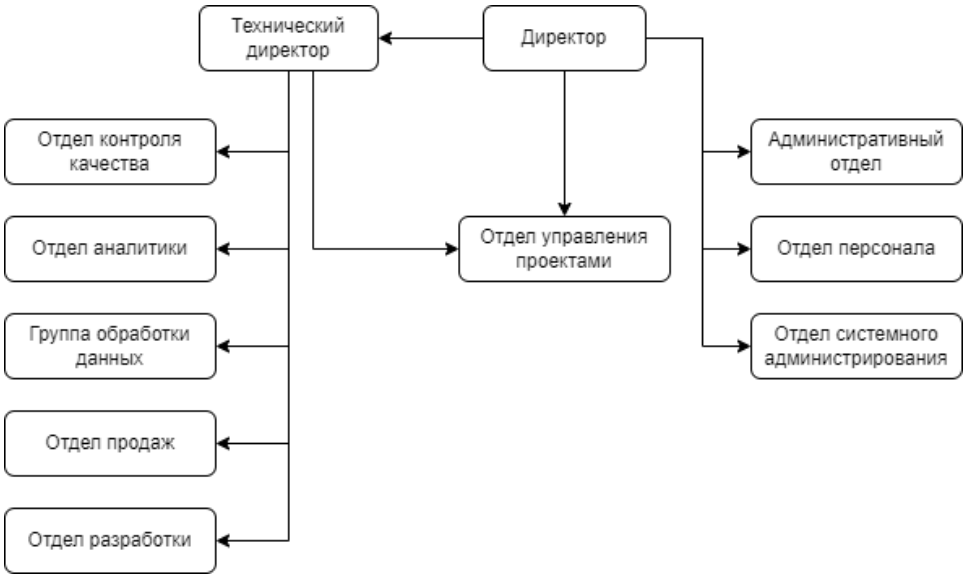


Рисунок 1 – Структура компании ООО «Neo Stack Technology»

В своей структуре компания содержит следующие отделы:

– отдел управления проектами включает руководителей IT-проектов, которые занимаются инициализацией проекта, управлением бюджета и организацией команды;

– отдел контроля качества включает специалистов по контролю качества или тестировщиков, которые отвечают за качество разрабатываемого ПО;

– группа обработки данных включает специалистов, которые отвечают за настройку геоданных в продуктовой линейке;

– отдел персонала включает HR-специалистов, которые занимаются подбором персонала и организаций мероприятий компании;

– отдел системного администрирования включает DevOps специалистов и системных администраторов, которые занимаются настройкой периферии и окружения для функционирования, разрабатываемого ПО;

– отдел разработки включает группу backend и frontend разработчиков;

– отдел аналитики включает специалистов, которые отвечают за документацию к разрабатываемому ПО и выяснение требований у заказчиков;

– отдел продаж включается специалистов, которые занимаются продвижением и продажей решений из продуктовой линейки;

– административный отдел включает бухгалтера, экономиста, помощника директора и директора по общим вопросам.

2.2 Обзор бизнес-процессов тестирования программного обеспечения

Любая IT-компания имеет в своей структуре отдел тестирования, который в основном занимается тестированием выпускаемого ПО и отвечает за качество этого продукта. Организация этого отдела может отличаться в зависимости от специфики компании, например, отдел может разделяться на подразделения, которые отдельно отвечают за безопасность, за автоматизацию, за нагрузочное тестирование, ручное тестирования и так далее.

Компания ООО «Neo Stack Technology» в своей структуре имеет один отдел, выполняющий основные функции тестирования ПО, именуемый как отдел контроля качества. Для того, чтобы погрузится в работы ОКК были изучены имеющиеся регламенты данного отдела. Часть регламентов отдела находится в стадии разработки, но основная часть готова к использованию и позволяет выстроить процесс работы отдела.

Вся документация отдела и компании хранится в так называемой базе знаний, которая расположена в Confluence. Confluence – это рабочее пространство, вики-система для использования организацией с целью

созданий единой базы знаний и обменом этими знаниями с другими участниками. Использование Confluence позволяет создавать и редактировать страницы любым участникам проекта, проводить дискуссии, комментировать рабочие документами и управлять проектами. Все регламенты, хранящиеся в данной системе написаны участниками ОКК и их руководителем и не имеют копий в официальных документах. Они направлены на то, чтобы новые сотрудники быстрее погружались в процессы отдела, а текущие сотрудники обращались к этим регламентам при возникновении спорных или конфликтных ситуаций на проектах.

Основной целью изучения этих регламентов было выделение процессов работы ОКК. Для этого понадобились следующие регламенты:

– «Работа с отделом аналитики в рамках тестирования фиче-страниц» – описывает бизнес-процессы тестирования аналитической документации в зависимости от типа проекта;

– «Единый процесс работы на проекте» – описывает основные этапы работы специалиста ОКК на проекте;

– «Составление тест-плана на проекте» – описание шаблона плана тестирования и его значимости;

– «Определение критериев готовности проекта к релизу» – определение и описание документа релиза и его шаблона;

– «Работа с ЖЦ задач на проекте» – описание основных процессов задач, которыми пользуется специалист ОКК;

– «Работа с руководством пользователя на проекте» – описание значимости документа и этапов его составления;

– «Актуализация тестовой документации при re-teste ошибок» – описание работ, после которых необходима актуализация тестовой документации;

– «Оценка работы специалиста ОКК на проекте» – описание минимальной, вероятной и максимальной оценки на работы ОКК;

– «Проведение аудита проекта как отправной точки для погружения в проект» – описание значимости аудита и порядка его проведения;

– «Предварительная подготовка к ретроспективе» – описание значимости ретроспектив и шаблона для заполнения соответствующей страницы.

В результате изучения регламентов были выделены два типа бизнес-процессов, которые присутствуют в отделе:

– процесс работы специалиста на проекте;

– внутренние процессы отдела.

Процесс работы специалиста на проекте регламентирован и выполняется каждым сотрудником отдела, когда тот заходит на проект. Часть процессов может отличаться в зависимости от типа проекта. В компании в основном выделяются следующие типы проектов:

– продуктовый проект – проект, который входит в продуктовую линейку компании, такие проекты являются проектами компании, которые продаются другим компаниям;

– коммерческий проект – проект заказной разработки, который поступает от определенного заказчика.

Независимо от типа проекта выделяются этапы процесса работы специалистов ОКК на проекте:

– подготовительный этап;

– анализ требований;

– разработка тестов;

– тестирование;

– подготовка документации.

О каждом этапе процесса будет описано подробнее далее.

Внутренние процессы отдела связаны с работами, которые выполняются специалистами внутри отдела и не относятся к проектам компании. Данные работы позволяют оценить работу каждого специалиста ОКК и отдела в целом. Из таких работ выделяют:

– проведение ретроспектив;

– сбор метрик;

– мероприятия, направленные на достижение индивидуального плана развития каждого сотрудника.

2.3 Мониторинг работы участников отдела тестирования

Как упоминалось выше, часть регламентов отдела находится в стадии разработки, поэтому для полноты понимания процессов и работы отдела было решено провести интервьюирование специалистов ОКК. Целью данного интервью является получение информации о процессах отдела у непосредственных его участников. Интервью проводилось в виде личной беседы отдельно с каждым специалистом. Это было сделать с целью того, чтобы участники интервью не ориентировались на ответы друг друга. Процесс интервьюирования позволяет узнать мнения специалистов о процессе, с которым они работаю, его недостатки и достоинства, а также возможные пожелания в изменении и/или улучшении рассматриваемого процесса.

Подготовка к интервью состояла из составления списка вопросов, которые были разделены на блоки в соответствии с основными этапами процессов, используемых в работе специалистов ОКК.

В таблице 1 представлен список вопросов, подготовленный для проведения интервью.

Таблица 1 – Вопросы для проведения интервью у специалистов отдела

Вопрос	**Цель вопроса**
Какую должность вы занимаете?	Сбор общей информации
Сколько времени вы работаете в данном отделе?	

Продолжение таблицы 1

Сколько времени в среднем у вас занимает аудит одного проекта?	Сбор информации о проведении
Как часто вы проводите аудит на своем проекте?	

Какие достоинства и недостатки вы видите в процессе проведения аудита?	подготовительного этапа работ
Считаете ли вы возможным оптимизацию и автоматизацию данного процесса?	
После проведения аудита отслеживаете ли вы устранение замечаний или это будет заметно только после проведения следующего аудита? Считаете ли вы необходимым отслеживать устранение замечаний?	
Выскажите свое мнение о данном процессе. Хотелось бы что-то улучшить/изменить помимо вышесказанного?	
Сколько в среднем времени у вас уходит на составление тест-плана (с нуля)?	
Возникают ли сложности с оценкой работ по тестированию?	
Какова цель хранения и создания карты переходов на проекте?	Сбор информации о проведении этапа анализа требований
Сколько времени в среднем у вас занимает составление карты переходов проекта?	
Расскажите про ваш опыт использования карты переходов? Часто ли приходится обращаться к карте переходов?	
Сколько времени в среднем занимает тестирование одной ФС без учета написания чек-листа?	
Как долго вы ожидаете уточнения требований от аналитика (ответы на ваши вопросы)? Если задержка присутствует, ответить на вопрос: Как вы считаете по каким причинам происходит задержка уточнения требований?	
Расскажите, как происходит ваше взаимодействие с аналитиком?	
Сколько в среднем итераций вопросов-уточнений у вас приходится на одну ФС? Если больше одной итерации, ответить на вопрос: Как вы считаете, почему не получается разрешить все вопросы за одну итерацию?	
Считаете ли вы необходимым этап написания чек-листа при тестировании аналитики? Если да, то почему?	

Продолжение таблицы 1

Составленный чек-лист помогает вам в будущих процессах? Пользуетесь ли вы им?	

Если да, описать в каких случаях и при каких обстоятельствах.	
Видите ли, вы возможность подключения к тестированию ФС раньше? Если да, то каким образом?	
Выскажите свое мнение о существующем процессе. Хотелось бы что-то улучшить/изменить помимо вышесказанного?	
Сколько в среднем времени у вас уходит на написание одного тест-кейса?	Сбор информации о проведении этапа разработки тестов
Сложно ли вам поддерживать тест-кейсы в актуальном состоянии? Если да, то почему и что мешает?	
Сколько в среднем времени у вас уходит на актуализацию тест-кейсов? Как часто это приходится делать на вашем проекте?	
Как вы отслеживаете актуальность написанных тест-кейсов?	
После написание тест-кейсов занимается ли кто-нибудь их приемкой?	
Удобен ли вам текущий формат написание тест-кейсов? Расскажите про его достоинства и недостатки.	
Видите ли вы возможность использования специализированной программы для написания тест-кейсов?	
Как вы подготавливается тестовые данные для тестирования? Где они хранятся?	
Как происходит ваша подготовка к тестированию (выделение тест-кейсов, составление шаблона отчета)?	Сбор информации о проведении этапа тестирования и документирования
Сколько в среднем времени у вас занимает подготовка к тестированию?	
Какие виды тестирования вы обычно проводите?	
Как часто проходит тестирование на вашем проекте (сколько итераций, например, регресс и приемочное)?	
Сколько в среднем времени у вас уходит на заведение одного баг-репорта?	
Баг-репорты у вас заводятся по единому шаблону? Или зависит от специалиста?	
Удобен ли вам текущий формат отчета о тестировании? Расскажите про его достоинства и недостатки.	

Продолжение таблицы 1

Сколько в среднем времени у вас уходит на составление ПМИ?	

В интервью приняло участие 6 человек, из которых: руководитель ОКК, 2 старших специалиста ОКК и 3 младших специалиста ОКК. Подробнее с ответами всех специалистов можно ознакомиться в приложении Б.

В результате проведенного интервью были учтены особенности процессов каждого этапа, которые не были описаны в регламентах или не были уяснены после их прочтения. Выяснены их достоинства и недостатки, получена средняя оценка времени, которое тратится на проведение работ этапов тестирования. Некоторые участники интервью также высказали свое мнение по улучшению и оптимизации существующих процессов, которое будет учтено при разработки соответствующих мероприятий.

2.4 Моделирование бизнес-процессов тестирования программного обеспечения

После изучения регламента и проведения анализа ответов из интервью, проведенного со специалистами отдела была начата работа по моделированию полученных бизнес-процессов.

В качестве инструмента для моделирования бизнес-процессов была выбрана система для моделирования процессов в нотации BPMN – Bizagi. Bizagi – это BPM-система, направленная на моделирование, исполнение, автоматизацию и анализ бизнес-процессов. Система Bizagi включает 3 модуля для полноценной настройки процессов [9]:

– modeler – полнофункциональная среда моделирвания процессов в нотации BPMN;

– studio – среда разработки бизнес-процессов;

– engine – среда исполнения процессов, которая доступна пользователям в любом браузере с любого устройства.

Для моделирования рассматриваемых в этой работе процессов достаточно функций модуля Modeler. Данный модуль предназначен для моделирования последовательности действий и событий. Помимо этого,

Bizagi Modeler поддерживает имитационное моделирование, экспорт модели и результатов моделирования в различные форматы файлов. Модуль необходим для тех, кому необходимо проанализировать и оптимизировать бизнес-процесс.

Как упоминалось выше, работы специалистов отдела на проектах имеют бизнес-процесс, который разделяется на следующие этапы:

– подготовительный этап;

– анализ требований;

– разработка тестов;

– тестирование;

– подготовка документации.

2.4.1 Бизнес-процесс подготовительного этапа

Подготовительный этап описывает процесс вхождения специалиста на проект, то есть необходим для того, чтобы специалист погрузился в проект и оценил его состояние. Стартовым событием данного этапа является готовность проекта к работам и сформированная команда проекта. Процесс подготовительного этапа не зависит от типа проекта, то есть все действия процесса выполняются на любом проекте компании. Модель бизнес-процесса подготовительного этапа представлена на рисунке 2.

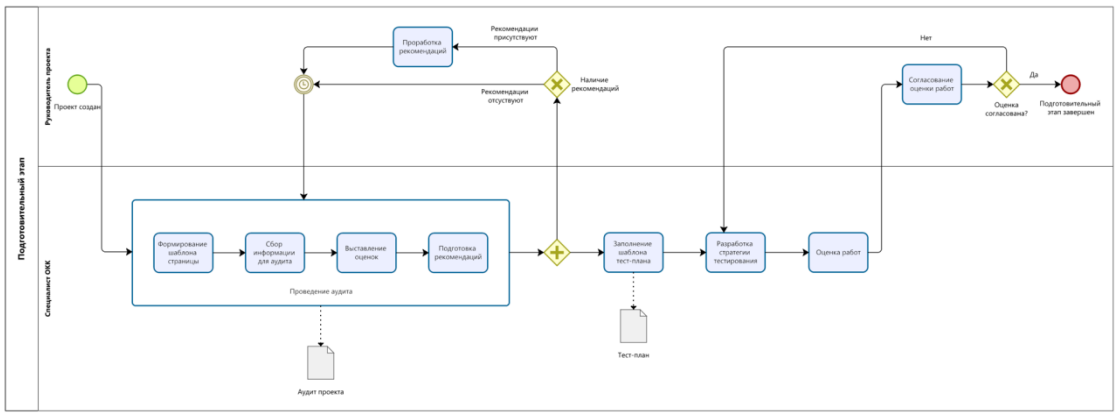


Рисунок 2 – Бизнес-процесс подготовительного этапа

Как только проект готов к работам и к нему подключают специалиста отдела контроля качества, ему необходимо оценить состояние проекта, то есть провести его аудит. Аудит проекта в компании проводится с целью показать общую картину состояния проекта команде и другим заинтересованным лицам. Аудит проекта – это подпроцесс, который включает в себя следующие действия:

– формирование шаблона страницы аудита – аудит ведется в Confluence, поэтому шаблон аудита хранится там же;

– сбор информации для аудита – аудит состоит из нескольких блоков, для каждого блока необходимо получить необходимую информацию и заполнить соответствующий блок;

– выставление оценок – оценки выставляются в виде светофора: красный, желтый и зеленый. Для того, чтобы выставить необходимую оценку необходимо проанализировать собранную для аудита информацию;

– подготовка рекомендаций – по результатам анализа собранной информации и выставленных оценок специалисту ОКК необходимо составить список рекомендаций для руководителя проекта с целью устранения выявленных недостатков проекта.

Как только проведен аудит и составлен список рекомендаций, они направляются руководителю проекта, который занимается проработкой этих рекомендаций. Может также возникнуть ситуация, что рекомендации отсутствуют и тогда руководителю проекта не нужно будет их прорабатывать. Как видно из модели бизнес-процесс проведение аудита цикличный процесс, который срабатывает по таймеру, так как аудит проводится регулярно раз в месяц, чтобы отслеживать состояние проекта не только при входе специалиста в проект, но и на протяжении всего ведения проекта.

После проведения аудита и параллельно проработке рекомендаций руководителем проект, специалист ОКК занимается заполнением шаблона плана тестирования, который также хранится в Confluence. Ценность данного документа в том, что он показывает основные работы специалиста на проекте,

которые будут проведены. В будущем обращаясь к данному документу можно восстановить историю работ специалиста и оценить работу. Для корректного и полного заполнения данного документа специалисту ОКК необходимо разработать стратегию тестирования, то есть план необходимы работ. Список работ индивидуален в зависимости от проекта и может отличаться, но в основном есть список работ, которые совпадают со следующими этапа процесса, которые будут рассмотрены далее.

В компании ведется учет трудозатрат сотрудников, поэтому оценка работ является важной составляющей не только для отчета перед руководством, но и для планирования последующих работ. Фактические и планируемые оценки работ позволяют руководителям оценить стоимость проекта и помогают в планировании будущих работ и проектов. Оценкой работ по тестированию занимается специалист ОКК исходя из своего опыта или памятки подсказки, которая есть в регламенте отдела, где прописаны минимальные, вероятное и максимальные оценки основных работ отдела.

Оценка работ обязательно должна быть согласована с руководителем проекта. По результатам согласования оценка может быть не согласована, тогда специалисту отдела необходимо пересмотреть составленную стратегию тестирования и пересмотреть оценку на работы. Если же оценка согласована со стороны руководителя проекта, то подготовительный этап заканчивается и начинаются работы следующего этапа.

2.4.2 Бизнес-процесс этапа анализа требований

Основная цель этапа анализа требований – это изучить аналитическую документацию, которая может быть представлена в виде требований или фича-страниц, выявить проблемные места в этой документации и составить план проверок. Аналитическую документацию составляет отдел аналитики, как только документация готова начинается процесс ее тестирования.

Стоит отметить, что процесс анализа требований зависит от типа проекта: продуктовый или коммерческий. На рисунке 3 представлен смоделированный бизнес-процесс для коммерческого проекта. В данном процессе участвуют только специалист от отдела аналитики и специалист ОКК.

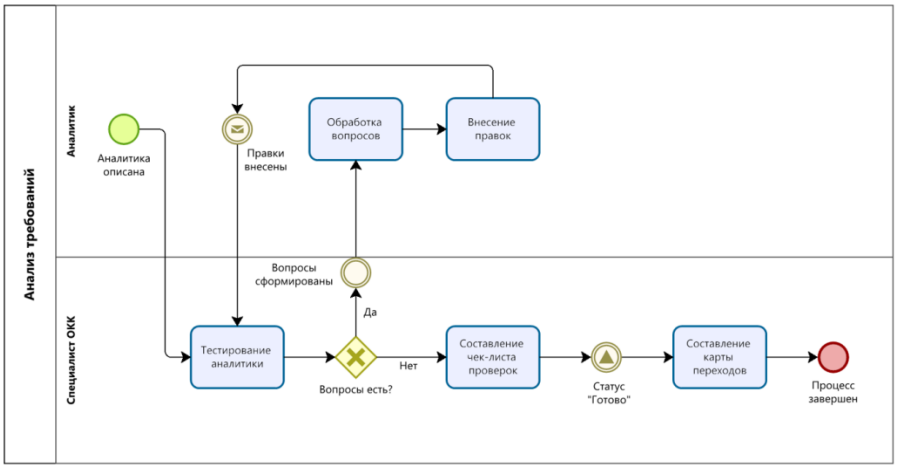


Рисунок 3 – Бизнес-процесс этапа анализа требований для коммерческого проекта

Как только аналитическая документация готова к тестированию она передается специалисту ОКК, который проводит тестирование аналитики. Тестирование аналитики подразумевает вычитку описанных требований и нахождение несоответствий или противоречий в них. Если таковые недостатки в написанной аналитической документации найдены, то в процессе тестирования аналитики составляется пул вопросов или добавляются комментарии к процессу. Так как аналитическая документация также ведется в Confluence, то там поддерживается комментирование части текста аналитики, что позволяет специалисту отдела аналитики отслеживать все изменения на странице.

После того как были сформированы вопросы, страница аналитической документации передается соответствующему аналитику на обработку данных вопросов. Каким образом аналитик будет обрабатывать данные вопросы – это отдельный процесс отдела аналитики, на который специалист ОКК никак не может повлиять. Аналитик может устраивать созвоны или встречи с

заказчиком или уточнять вопросы с разработчиками. После того как все вопросы были обработаны, аналитик вносит правки в документ аналитики, оповещая об этом специалиста ОКК. После чего процесс тестирования аналитики повторяется снова. Таких итераций работы над аналитической документацией может быть несколько.

И только когда все вопросы решены – специалист ОКК приступает к формированию чек-листа проверок. Чек-лист проверок выглядит как нумерованный список, в котором отображаются основные проверки, которые необходимо будет провести для тестирования разрабатываемой системы. Только после написания данного чек-листа аналитическая документация считается готовой и специалистом ОКК проставляется статус тестирования «Готово».

После проверки аналитической документации специалист ОКК составляет карту переходов проекта по фичам, которые были описаны в документации. Такая карта переходов строится с помощью MindMap и позволяет наглядно увидеть весь проект и его основные функции. Иногда карта переходов может заменять составление тест-кейсов для тестирования, о чем будет описано в соответствующем бизнес-процессе.

Далее будет рассмотрен процесс этапа анализа требований для продуктового проекта, модель которого представлена на рисунке 4. Данный процесс отличается тем, что в нем участвует еще одно заинтересованное лицо – Product Owner. Product Owner – это специалист, который владеет продуктом и отвечает за его развитие.

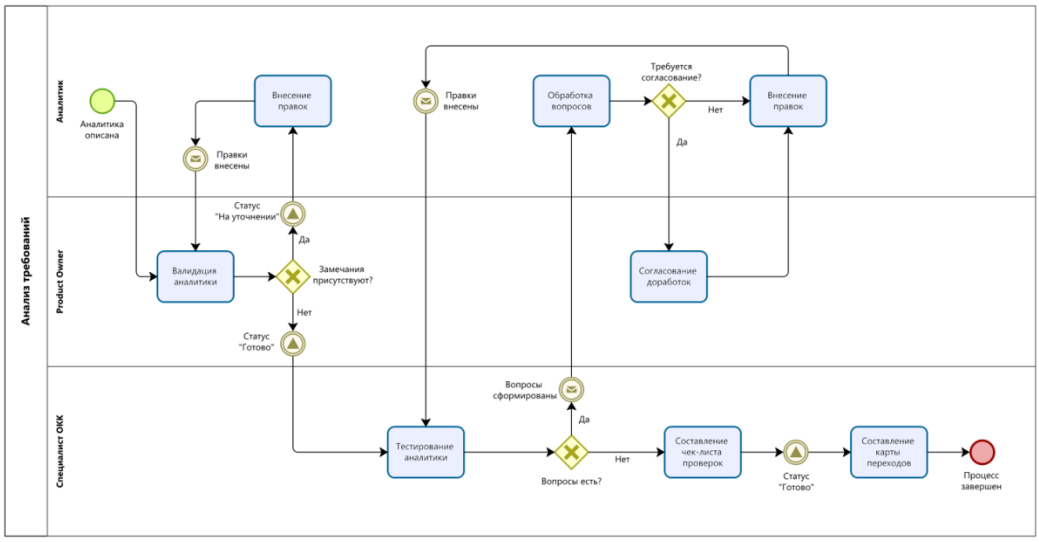


Рисунок 4 – Бизнес-процесс этапа анализа требований для продуктового проекта

После того как аналитическая документация готова к тестированию она передается изначально Product Owner, который проводит валидацию написанной аналитики. Под валидацией подразумевается вычитка и написание замечаний, которые не соответствуют требуемому функционалу системы. Если Product Owner находит замечания, то он оставляет их на странице аналитики в виде комментариев и отправляет аналитику на исправление. Сигнал о том, что аналитическая документация имеет замечания от Product Owner состоит из выставления соответствующего статуса «На уточнении». Аналитик в свою очередь вносит правки по предоставленным замечаниям и оповещая Product Owner отправляет аналитику на повторную валидацию. Итераций такого процесса может быть также несколько. Как только Product Owner принимает аналитику он выставляет статус «Готово».

Статус «Готово» от Product Owner является сигналом для специалиста ОКК о том, что аналитическую документацию можно брать в работу, то есть в тестирование. Процесс тестирования аналитики специалистом ОКК ничем не отличается от процесса на коммерческом проекте, то есть также составляется пул вопросов, которые передаются специалисту отдела аналитики.

Специалист отдела аналитики обрабатывает полученные вопросы, также сам процесс обработки вопросов – это процесс отдела аналитики, на который специалист ОКК не влияет. После обработки вопросов может возникнуть ситуация, что затронуты изменения в функциональной части и тогда аналитику необходимо согласовать эти изменения с Product Owner, так как это его проект. После того как изменения были согласованы или не согласованы аналитик вносит правки в аналитическую документацию и передает ее снова на тестирование специалисту ОКК. Также, как и в коммерческом проекте итераций данного процесса может быть несколько и только после того как будут решены все вопросы специалист ОКК приступает к составлению чек-листа проверок.

Дальнейшие действия специалиста ОКК идентичны процессу на коммерческом проекте, то есть специалист выставляет статус «Готово», который сигнализирует о том, что документация полностью протестирована и составлен минимальный чек-лист будущих проверок, а также составляется карта переходов проекта, которая показывает основную функциональность системы.

Стоит отметить, что все работы, проведенные на данном этапе согласовывают с руководителем проекта на подготовительном этапе и отображаются в плане тестирования, который был также составлен на подготовительном этапе.

2.4.3 Бизнес-процесс этапа разработки тестов

На основе чек-листов, которые были составлены на этапе анализа требований разрабатывается тестовая документация такая как тест-кейсы. Тест-кейсы являются основной документацией специалиста, так как именно они играют одну из ключевых ролей при тестировании продукта. Поэтому данный этап является важной частью работы специалиста на проекте. Модель бизнес-процесса разработки тестов представлена на рисунке 5.

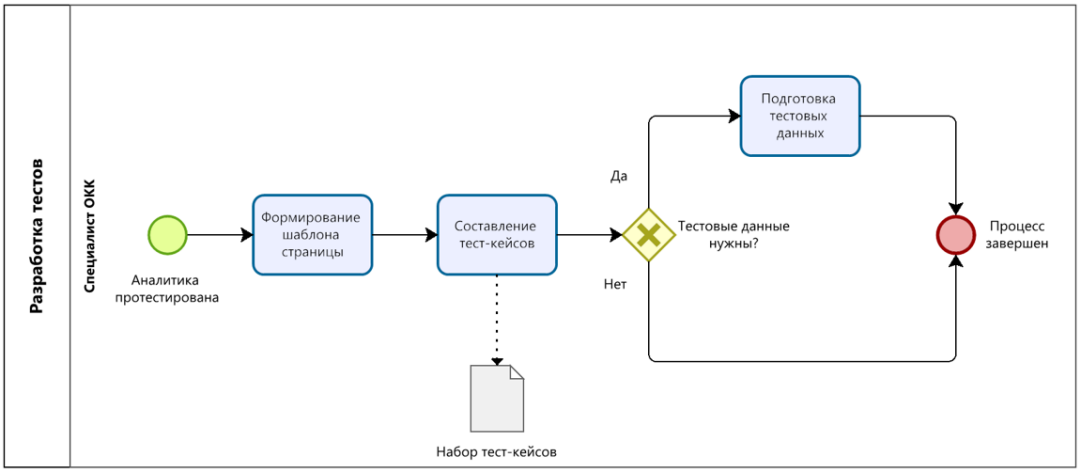


Рисунок 5 – Бизнес-процесс этапа разработки тестов

Тест-кейсы составляются и хранятся в Confluence в виде таблицы, поэтому для начала необходимо сформировать и заполнить шаблон для хранения тест-кейсов. После того, как шаблон сформирован начинается этап составления и разработки тест-кейсов. Хоть данный процесс описан в одно действие, он довольно длительный и требует полного сосредоточения от специалиста ОКК. Данный вид документации очень важен, так как он предназначен не только для специалиста, который непосредственно составляет эти тест-кейсы, но и для новых специалистов в качестве погружения в проект или же в помощи при проведении тестирования. Поэтому тест-кейсы должны быть полными и описывать достаточное количество проверок.

Для проведения тестирования могут также понадобиться различные тестовые данные. Необходимость и структура тестовых данных зависит в большей степени от специфики проекта, например, если это проект разработки геоинформационной системы, то необходимо подготавливать данные, которые будут отображать на карте проекта – это могут быть объекты построек и иных сооружений. Или же в качестве тестовых данных могут служить различные вложения в виде файлов разных форматов, например, изображения документы и т.д. Все эти данные должны соответствовать разрабатываемому тест-кейсу.

Процесс разработки тестов считается завершенным, если написаны все тест-кейсы и подготовлены все тестовые данные для этих тест-кейсов.

2.4.4 Бизнес-процесс этапа тестирования

Этап тестирования подразумевает непосредственное тестирование продукта и выявление ошибок. Тестирование на проекте может проходить несколько раз, что зависит в большей степени от длительности проекта. В каком виде будет проходить этап тестирования зависит от вида применяемого на проекте тестирования. Стоит упомянуть, что каждый из видов может встретиться как на коммерческом проекте, так и на продуктовом. На рисунке 6 представлен процесс этапа тестирования.

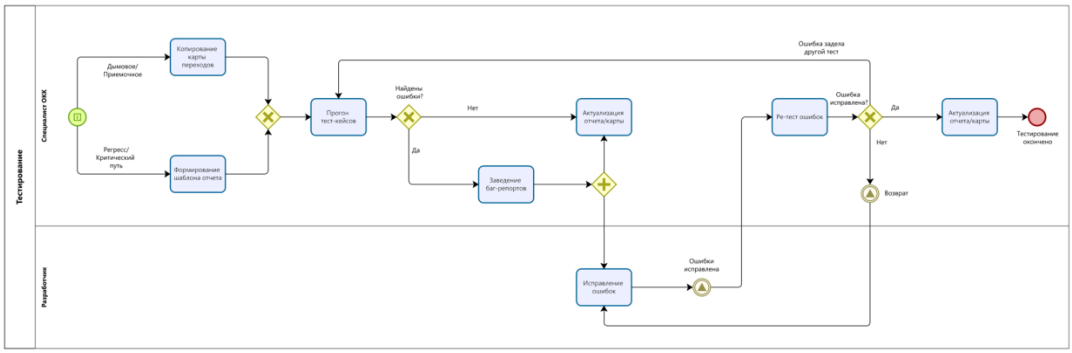


Рисунок 6 – Бизнес-процесс этапа тестирования

Процесс этапа тестирования начинается, как только будет готова тестовая документации и стенд с разрабатываемой системой будет готов к тестированию. Далее в зависимости от бизнес-ценности тестирования выбирается вид проходимого тестирования. Вид тестирования может быть любой, но часто используемые в компании – это регрессионное, тестирование по критическому пути, дымовое тестирование или приемочное. Регрессионное тестирование и тестирование по критическому пути подразумевают большой объем количества проверок и затрагивают всю функциональность системы. Дымовое и приемочное тестирование необходимы для того, чтобы проверить основные функции системы, которыми будет пользоваться конечный пользователь.

Если для системы необходимо регрессионное тестирование или тестирование по критическому пути, то для этого необходимо сформировать шаблон для ведения отчета о тестировании. Такой документ ведется в Confluence и заполняется вручную специалистами ОКК, он состоит из списка тест-кейсов необходимых для прохождения тестирования и результатов их прохождения. На этапе формирования шаблона отчета необходимо указать вид тестирования и выбрать необходимые для тестирования тест-кейсы и перенести названия этих тест-кейсов на страницу отчета.

Если же для системы необходимо дымовое или приемочное тестирование, то для этого не обязательно создавать отчет о тестировании, достаточно просто скопировать MindMap карту переходов, которая была сформирована на этапе анализа требований. Как уже было сказано, данная карта содержит основной функционал системы, а дымовое и приемочное тестирование подразумевают проверку основного функционала, поэтому данная карта способна заменить отчет о прохождении тестирования.

После того как были подготовлены страница и карта переходов для ведения отчетов о тестировании начинается процесс тестирования или прогона тест-кейсов. Во время прохождения тестирования находятся ошибки системы, которые называют баги. Если ошибки найдены, то для них необходимо завести так называемые баг-репорты. Баг-репорт представляет из себя описание возникновения ошибки, который переходит разработчику для устранения найденной ошибки. Баг-репорты заводятся в системе Jira и имеют статусы, которые показывают на каком этапе находится данная ошибка: на работе у разработчиков или в тестировании и т.д. Как только был заведен баг-репорт или тест-кейс пройден без ошибок, необходимо актуализировать отчет о тестировании или карту. Актуализация отчета о тестировании или карты подразумевает проставление статуса прохождения теста, например, пройден или провален и в случае, если тест-кейс был пройден с ошибками, то прикрепление заведенного баг-репорта.

Параллельно этому разработчик приступает к исправлению заведенного баг-репорта. Процесс исправления ошибки – это внутренний процесс отдела разработки и каждого разработчика, который не влияет на рассматриваемый в данной работе процесс. Как только разработчик исправляет ошибку, он изменяется ее статус на «В тестировании» и отправляет специалисту ОКК на проведение ре-теста или проверки функциональности на то, что ошибка была действительно исправлена. Часто возникают ситуации, когда в процессе ре-теста ошибка оказывается не исправленной, тогда тестируемый баг-репорт возвращается обратно разработчику на исправление. Также исправленная ошибка может задеть другую функциональность, тогда необходимо вернуться на процесс прогона тест-кейса и выбрать тот тест-кейс, который данная ошибка заделе, прогнать его и в случае необходимости создать баг-репорт и актуализировать отчет о тестировании или карту. После исправления ошибки необходимо также актуализировать отчет о тестировании или карту, чтобы они находились всегда в актуальном состоянии и показывали текущее состояние продукта, что будет необходимо как руководителю проекта для будущего планирования работ, так и заказчику.

2.4.5 Бизнес-процесс этапа подготовки документации

Этап подготовки документации является последним этапом процесса тестирования и обычно выполняется после того как будут проведены все работы на предыдущих этапах. Обычно вся документация, которая подготавливается на данном этапе необходима заказчику и показывает состояние проекта, изменения, внесенные в проект и испытания проводимые в процессе тестирования. Модель данного этапа представлена на рисунке 7.

Этап подготовки документации может отличаться в зависимости от заказчика и его потребностей, отличаться может с точки зрения набора необходимых для передачи документов. В модель были добавлены основные документы, которые чаще всего требуют заказчики.

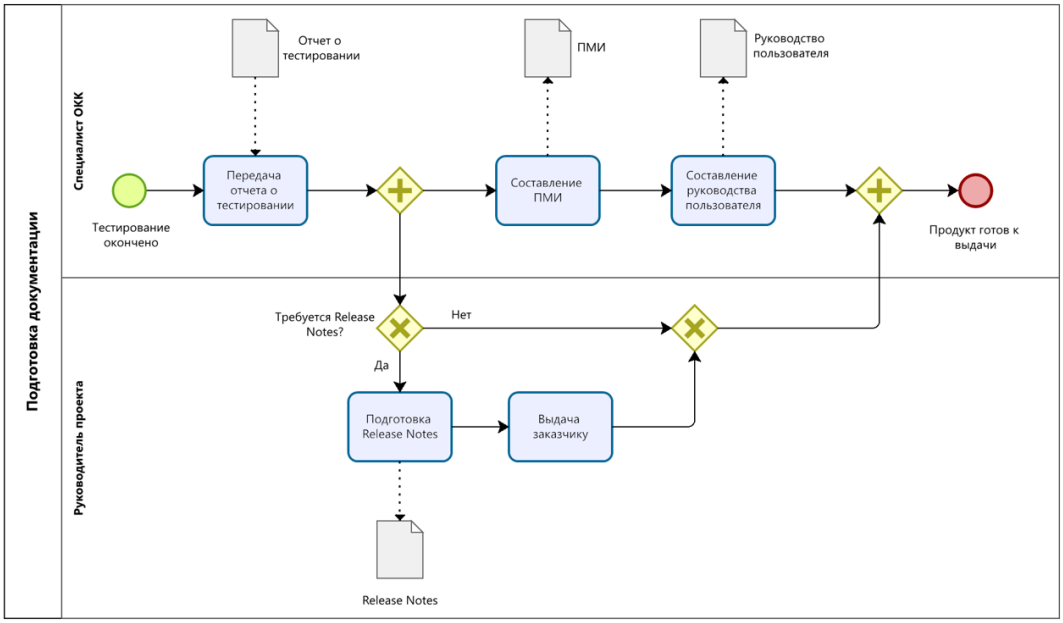


Рисунок 7 – Бизнес-процесс этапа подготовки документации

После прогона тест-кейсов на этапе тестирования составляется отчет о тестировании, как только тестирование завершается необходимо передать данный отчет руководителю проекта для оценки готовности проекта к релизу. После чего происходит работа над документацией, она выполняется разными специалистами, поэтому данный процесс выполняется параллельно.

Руководитель проекта при необходимости составляется Release Notes. Такая необходимость в основном возникает на долгосрочных коммерческих проектах или на продуктовых проектах. Release Notes представляет из себя документ, который содержит информацию о выполненных задачах или исправленных ошибках, что показывает заказчику, какая работы была проведена и какие обновления будут выкатаны на боевые стенды заказчика. После того как такой документ готов, он передается заказчику.

Специалист ОКК занимается составлением ПМИ - программа и методика испытаний. Этот документ показывает какие проверки были проведены при тестировании системы, обычно содержит набор основных тест-кейсов необходимых для проведения тестирования. Помимо ПМИ специалист ОКК также занимается написание Руководства пользователя. Данный документ содержит описание системы и инструкцию по ее использованию.

Как только все документы будут описаны они перелаются заказчику на приемку. Заказчик проверяет данные документы и может вернуть их на доработку, сформировав соответствующие замечания. Процесс считается завершенным, когда все необходимые документы приняты заказчиком.

2.5 Настройка бизнес-процессов тестирования программного обеспечения для имитационного моделирования

Прежде чем приступать к анализу смоделированных бизнес-процессов, необходимо провести имитационное моделирование. Цель имитационного моделирования – это воспроизведение исследуемой системы, имитация ее во времени. Имитационное моделирование позволяет проводить эксперименты с системой для получения информации о ней, которая поможет при дальнейшем анализе и оптимизации.

Для проведения имитационного моделирования в системе Bizagi Modeler необходимо определить ресурсы, которые участвую в данном процессе. В качестве ресурсов были взяты все участники рассматриваемых процессов. Определение ресурсов в Bizagi представлено на рисунке 8.

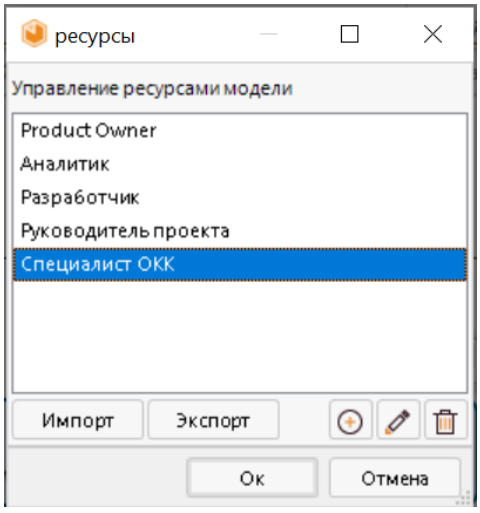


Рисунок 8 – Ресурсы бизнес-процессов

Далее были выставлены настройки для каждого элемента во всех рассмотренных моделях.

Для действий модели выставляются настройки времени обработки и ресурс, который выполняет данную работу. Время обработки действий берется из регламента «Оценка работы специалиста ОКК на проекте» или опыта работы сотрудников на проекте. Каждому ресурсу необходимо выставить затраты за час его работы. Затраты за час работы берутся средние за 2023 год по Томску из сервиса ГородРабот.ру, на рисунке 9 представлена настройка затрат за час работы каждого ресурса.

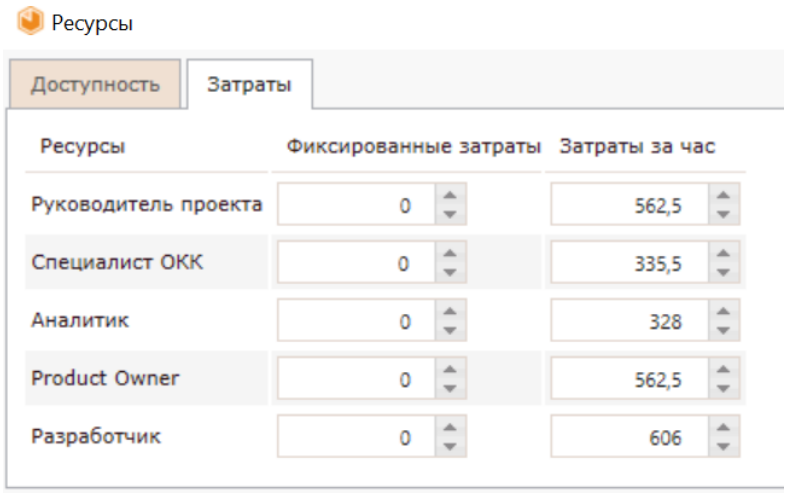


Рисунок 9 – Затраты на работу ресурсов

Шлюзы модели настраиваются как процент вероятности наступления следующего события. Процент вероятности наступления событий берется исходя из опыта сотрудников компании, которые участвую в рассматриваемых процессах.

Настройки имитационной модели для подготовительного этапа представлены в таблице 2.

Таблица 2 – Настройка имитационной модели для подготовительного этапа

Элемент процесса	Параметры настройки
Формирование шаблона страницы аудита	Время обработки – 30 минут. Ресурс – специалист ОКК.
Сбор информации для аудита	Время обработки – 3 часа. Ресурс – специалист ОКК.
Выставление оценок	Время обработки – 45 минут. Ресурс – специалист ОКК.
Подготовка рекомендаций	Время обработки – 1 час.

	Ресурс – специалист ОКК.
Шлюз «Наличие рекомендаций»	Вероятность «Рекомендации отсутствуют» – 10%. Вероятность «Рекомендации присутствуют» – 90%.
Проработка рекомендаций	Время обработки – 8 часов. Ресурс – руководитель проекта.
Событие таймера	Интервал – 30 дней.
Заполнение шаблона тест-плана	Время обработки – 30 минут. Ресурс – специалист ОКК.
Разработка стратегии тестирования	Время обработки – 2 часа. Ресурс – специалист ОКК.
Оценка работ	Время обработки – 1 час. Ресурс – специалист ОКК.
Согласование оценки работ	Время обработки – 2 часа. Ресурс – руководитель проекта.
Шлюз «Оценка согласована?»	Вероятность «Да» – 75%. Вероятность «Нет» – 25%.

Настройки имитационной модели для этапа анализа требований коммерческого проекта представлены в таблице 3.

Таблица 3 – Настройка имитационной модели для этапа анализа требований коммерческого проекта

Элемент процесса	Параметры настройки
Тестирование аналитики	Время обработки – 45 минут. Ресурс – специалист ОКК.
Шлюз «Вопросы есть?»	Вероятность «Да» – 80%. Вероятность «Нет» – 20%.
Обработка вопросов	Время обработки – 30 минут. Ресурс – аналитик.
Внесение правок	Время обработки – 1 час. Ресурс – аналитик.
Составление чек-листа проверок	Время обработки – 30 минут. Ресурс – специалист ОКК.

Составление карты переходов	Время обработки – 2 часа. Ресурс – специалист ОКК.

Настройки имитационной модели для этапа анализа требований продуктового проекта представлены в таблице 4.

Таблица 4 – Настройка имитационной модели для этапа анализа требований продуктового проекта

Элемент процесса	Параметры настройки
Валидация аналитики	Время обработки – 30 минут. Ресурс – product owner.
Шлюз «Замечания присутствуют?»	Вероятность «Да» – 75%. Вероятность «Нет» – 25%.
Внесение правок	Время обработки – 1 час. Ресурс – аналитик.
Тестирование аналитики	Время обработки – 45 минут. Ресурс – специалист ОКК.
Шлюз «Вопросы есть?»	Вероятность «Да» – 80%. Вероятность «Нет» – 20%.
Обработка вопросов	Время обработки – 30 минут. Ресурс – аналитик.
Шлюз «Требуется согласование?»	Вероятность «Да» – 75%. Вероятность «Нет» – 25%.

Продолжение таблицы 4

Согласование доработок	Время обработки – 30 минут. Ресурс – product owner.
Внесение правок	Время обработки – 1 час. Ресурс – аналитик.
Составление чек-листа проверок	Время обработки – 30 минут. Ресурс – специалист ОКК.
Составление карты переходов	Время обработки – 2 часа. Ресурс – специалист ОКК.

Настройки имитационной модели для этапа разработки тест-кейсов представлены в таблице 5.

Таблица 5 – Настройка имитационной модели для этапа разработки тест-кейсов

Элемент процесса	Параметры настройки
Формирование шаблона страницы	Время обработки – 15 минут. Ресурс – специалист ОКК.
Составление тест-кейсов	Время обработки – 3 часа. Ресурс – специалист ОКК.
Шлюз «Тестовые данные нужны?»	Вероятность «Да» – 75%. Вероятность «Нет» – 25%.
Подготовка тестовых данных	Время обработки – 45 минут. Ресурс – специалист ОКК.

Настройки имитационной модели для этапа тестирования представлены в таблице 6.

Таблица 6 – Настройка имитационной модели для этапа тестирования

Элемент процесса	Параметры настройки
Копирование карты переходов	Время обработки – 15 минут. Ресурс – специалист ОКК.
Формирование шаблона отчета	Время обработки – 1 час. Ресурс – специалист ОКК.
Прогон тест-кейсов	Время обработки – 3 часа. Ресурс – специалист ОКК.

Продолжение таблицы 6

Шлюз «Ошибки найдены?»	Вероятность «Да» – 75%. Вероятность «Нет» – 25%.
Заведение баг-репортов	Время обработки – 1 час. Ресурс – специалист ОКК.
Актуализация отчета/карты	Время обработки – 15 минут. Ресурс – специалист ОКК.
Исправление ошибок	Время обработки – 8 часов. Ресурс – разработчик.
Ре-тест ошибок	Время обработки – 1 час. Ресурс – специалист ОКК.

Шлюз «Ошибка исправлена?»	Вероятность «Да» – 40%. Вероятность «Нет» – 40%. Вероятность «Ошибка задела другой тест» – 20%.
Актуализация отчета/карты	Время обработки – 15 минут. Ресурс – специалист ОКК.

Настройки имитационной модели для этапа подготовки документации представлены в таблице 7.

Таблица 7 – Настройка имитационной модели для этапа подготовки документации

Элемент процесса	Параметры настройки
Передача отчета о тестировании	Время обработки – 30 минут. Ресурс – специалист ОКК.
Шлюз «Требуется Release Notes?»	Вероятность «Да» – 80%. Вероятность «Нет» – 20%.
Подготовка Release Notes	Время обработки – 1 час 30 минут. Ресурс – руководитель проекта.
Выдача заказчику	Время обработки – 30 минут. Ресурс – руководитель проекта.
Составление ПМИ	Время обработки – 24 часа. Ресурс – специалист ОКК.
Составление руководства пользователя	Время обработки – 40 часов. Ресурс – специалист ОКК.

2.6 Анализ результатов моделирования бизнес-процессов тестирования программного обеспечения

После проведения имитационного моделирования были получены результаты в виде таблицы затрат на осуществление каждого этапа процесса и времени, которое затрачивается на каждое действие этапа. Далее рассмотрим каждый этап подробнее.

На моделях по результатам имитационного моделирования будет отображена легенда, где значение красного квадрата означает количество

выполнений действия, а значение синего квадрата – это общее время обработки действия.

Результат имитационного моделирования подготовительного этапа представлен на рисунке 10.

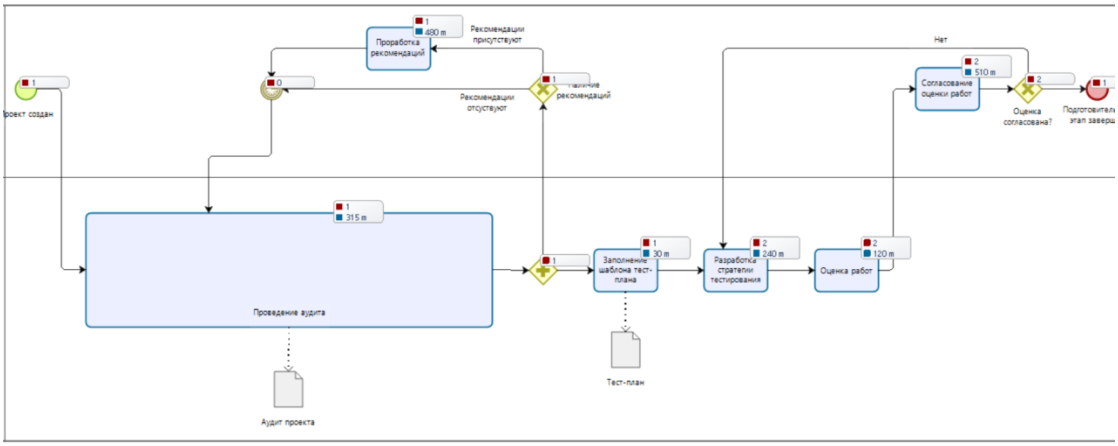


Рисунок 10 – Результаты имитационного моделирования подготовительного этапа

По результатам имитационного моделирования можно отметить, что большое количество времени уходит специалистом ОКК на проведение аудита по проекту. Учитывая, что аудит проводится каждый месяц, а не один раз, есть смысл автоматизировать процесс сбора информации по аудиту. Это позволит сократить время работы специалиста и ускорить процесс прохождения всего подготовительного этапа. Также большое количество времени тратится руководителем проекта на проработку рекомендаций, данный процесс не влияет на другие работы данного этапа и выходит за пределы работы отдела контроля качества.

В данном процессе наблюдается цикл, если оценка не согласуется руководителем, что влечет за собой дополнительное прохождение действий процесса и как следствие увеличение времени выполнения процесса и затрат на него. Зачастую ситуация, когда руководитель проекта не согласует оценку, случается по причине того, что она выходит за рамки бюджета. Чтобы избежать таких ситуаций можно изменить процесс, добавив еще одно действие – это озвучивание рамок бюджета на выполнение работ перед

составлением тест-плана и стратегии тестирования. В таком случае специалист ОКК будет опираться на выданные цифры и стараться не выходить за их переделы, а также строить стратегию тестирования с учетом рамок бюджета.

Мероприятия по оптимизации подготовительно этапа:

– автоматизация процесса сбора информации для аудита;

– добавление действия об информировании специалистов ОКК о бюджете проекта до составления тест-плана и разработки стратегии тестирования.

Результат имитационного моделирования этапа анализа требований для коммерческого проекта представлен на рисунке 11.

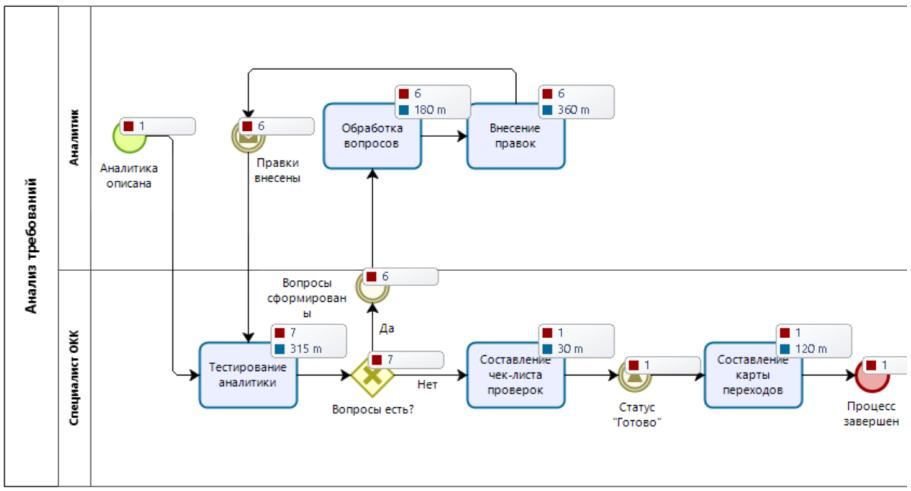


Рисунок 11 – Результаты имитационного моделирования этапа анализа требований коммерческого проекта

По результатам имитационного моделирования видна цикличность возвратов аналитики на доработку – это ситуация, когда во время тестирования появляются вопросы или замечания к написанной документации. На модели видно, что количество возвратов в среднем 6, что является достаточно весомой цифрой и влияет на продолжительность бизнес-процесса рассматриваемого этапа. Данная ситуация не подлежит оптимизации и автоматизации по причине того, что документацию пишут аналитики и при написании могут быть факторы, влияющие на количество вопросов и возвратов, например, аналитик не учел часть функциональности системы или

ошибся при написании сценариев использования. Конечно, в зависимости от вида аналитики количество таких возвратов может быть, как больше, так и меньше или возвратов может и не быть. На данной модели рассмотрен самый часто встречаемый в работе пример. Действия по составлению чек-листа и карты переходов также не подлежат оптимизации.

В заключении анализа бизнес-процесса этапа анализа требований на коммерческом проекте можно сказать, что порядок действий совершаемый при работе на рассматриваемом этапе не подлежит оптимизации и не нуждается в ней.

Результат имитационного моделирования этапа анализа требований для продуктового проекта представлен на рисунке 12.

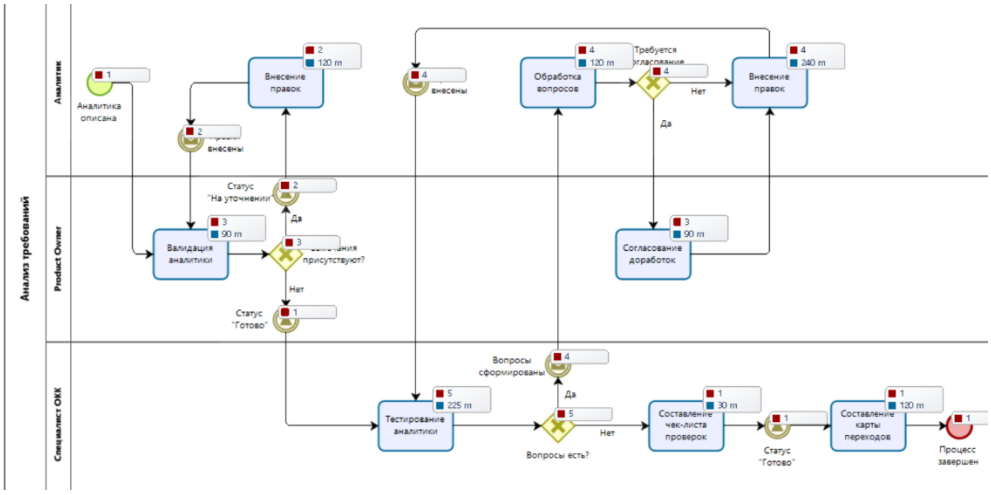


Рисунок 12 – Результаты имитационного моделирования этапа анализа требований продуктового проекта

По результатам имитационного моделирования можно заметить, что валидация аналитики Product Owner происходит на начальных этапах, то есть сразу после ее фактического написания. Это влечет за собой возвраты аналитики на доработки из-за замечаний, оставленных Product Owner. На имитационной модели наблюдается два таких возврата. После валидации аналитики она уходит на тестирование и несмотря на то, что аналитика уже проходила валидацию вопросы со стороны тестирования или ОКК остаются, а значит сохраняются и возвраты, аналогично коммерческим проектам. Вероятнее всего таких возвратов будет меньше, но они не исключаются

полностью и среднее их количество равно четырем. Несмотря на то, что количество возвратов сокращается, они становятся дороже по причине того, что любое внесенное изменение требует согласования с Product Owner. По результатам имитационной модели видно, что таких согласований было 3 из 4 возвратов, что является весомой цифрой.

С целью уменьшения цикличности возвратов аналитики и сохранении смысла существования данного этапа было предложено изменить порядок действий данного процесса, а именно перенести валидацию Product Owner после тестирования аналитики специалистом ОКК. Данное решение позволит Product Owner видеть и валидировать полностью готовую и протестированную аналитику так как зачастую бывает, что аналитики в ходе написания документации упускают часть функционала, которая выявляется на этапе тестирования специалистом ОКК. Помимо этого, данное решение позволит сократить количество возвратов от Product Owner и избавит от согласования доработок. А действия составления чек-листов и карты переходов можно выполнять параллельно валидации Product Owner. В случае изменения функционала после валидации в процессе повторного тестирования можно актуализировать чек-листы и карту, что не займет много времени.

Мероприятия по оптимизации этапа анализа требований продуктового проекта:

– перенос действия валидации аналитики после тестирования и изменение процесса с учетом данного переноса;

– составлять чек-листы и карту переходов параллельно валидации product Owner.

Результат имитационного моделирования этапа разработки тестов представлен на рисунке 13.

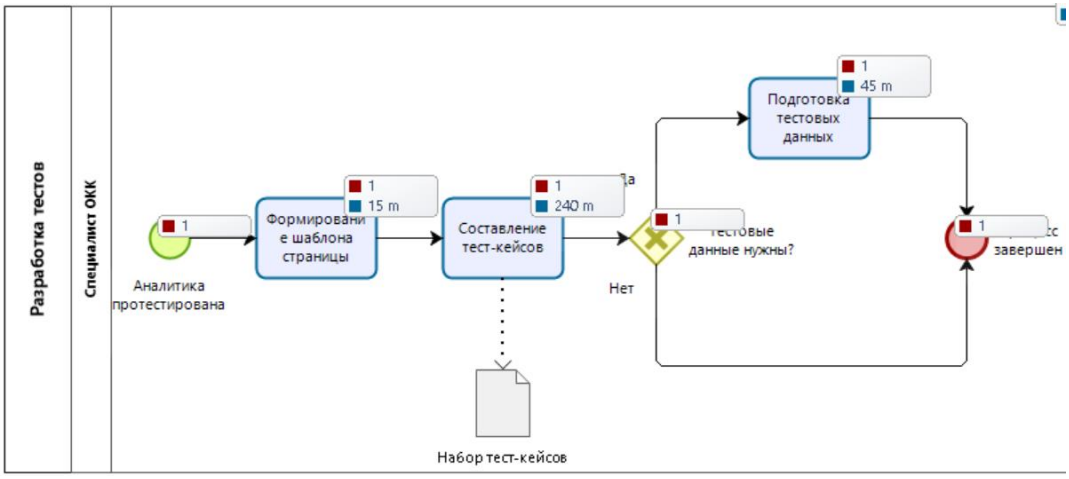


Рисунок 13 – Результаты имитационного моделирования этапа разработки тестов

Хранение и разработка тест-кейсов специалистами ОКК осуществляется в Confluence. Страница в Confluence представляет из себя формат текствого документа, где в виде таблицы хранятся разработанные специалистом тест-кейсы. Прежде чем начать разработку тест-кейсов необходимо сформировать шаблон страницы в Confluence. Стоит отменить. Что для каждой страницы аналитической документации составляется своя страница данного шаблона, поэтому в зависимости от объема проекта количество таких страниц могут варьироваться. Для имитационного моделирования были выставлены настройки для формирования одного такого шаблона.

Существует большое количество специализированных систем для хранения и составления тестовой документации, в том числе и тест-кейсов. Стоит отметить, что Confluence не является такой системой, поэтому необходимо рассмотреть другие варианты для разработки и хранения тест-кейсов. Рассмотрение таких вариантов позволит автоматизировать составление отчетов и структурировать хранение тест-кейсов. Так как основной системой для ведения задач на проектах является Jira, были рассмотрены варианты специализированных плагинов для управления тестированием в Jira.

Инструмент управления тестированием для Jira — это приложение для управления тестированием и обеспечения качества, которое можно

использовать внутри Jira. Эти надстройки Jira обычно предназначены для помощи группам контроля качества в реализации функций автоматизации тестирования, обмена информацией в режиме реального времени и возможности повторного использования тестовых сценариев [10].

Для определения того, какой инструмент лучше использовать были выявлены следующие критерии:

– пользовательский интерфейс. Пользовательский интерфейс является важным фактором при выборе инструментов управления тестовыми сценариями Jira, поскольку он влияет на удобство использования, совместную работу, настройку и возможности отчетности инструмента. Хорошо продуманный пользовательский интерфейс может сократить время обучения и повысить производительность управления тестированием в Jira, упрощая навигацию по функциям инструмента.

– удобство использования. Удобство использования — критический фактор, который следует учитывать при выборе инструментов для управления тестированием в Jira, поскольку он может напрямую влиять на эффективность и результативность процесса тестирования. Инструмент с плохим удобством использования может привести к путанице, разочарованию и снижению производительности среди специалистов.

– интеграция. Интеграцию программного обеспечения важно учитывать при выборе инструментов управления тестовыми сценариями Jira, поскольку они могут расширить возможности инструмента и повысить общую эффективность процесса тестирования. Многие организации используют несколько инструментов для различных задач, таких как отслеживание ошибок, непрерывная интеграция и управление проектами. Интеграция инструмента для управления тестированием в Jira с другими инструментами может упростить процессы, сократить ручной труд и повысить точность.

– цена. Ценообразование – важный фактор, который следует учитывать при выборе инструментов для управления тестированием в Jira, поскольку оно может повлиять на общую рентабельность процесса тестирования.

Инструменты управления тестированием могут сильно различаться по цене: некоторые инструменты предлагают бесплатные или недорогие варианты, в то время как другие могут иметь высокие первоначальные затраты или периодические сборы. Некоторые инструменты управления тест-кейсами Jira могут взимать плату в зависимости от количества пользователей или количества проектов, в то время как другие могут предлагать неограниченный доступ.

Далее будет представлено краткое описание некоторых инструментов.

TestRail — это платформа управления тестированием для групп контроля качества и разработчиков, позволяющая управлять тестовыми сценариями, планировать и выполнять тесты, а также отслеживать результаты как ручного, так и автоматизированного тестирования. Его можно использовать в каскадных или гибких проектах. Тестовые наборы можно упорядочивать по папкам и разделам, а также настраивать тестовые наборы с помощью шаблонов, статусов и полей. TestRail позволяет интегрироваться с Jira, что позволяет управлять тестами и результатами в одном месте. Имеется возможность напрямую искать любые связанные проблемы Jira в TestRail и отправлять отчеты о дефектах из TestRail в Jira. TestRail стоит от 36 долларов за пользователя в месяц и предлагает 14-дневную бесплатную пробную версию.

Zephyr Squad — это самое популярное решение для управления тестированием внутри Jira для небольших Agile-команд, которые хотят получить дизайн тестовых сценариев, выполнение и отчетность прямо в Jira, где Agile-команда координирует работу. Он также включает мощную интеграцию для автоматизации тестирования и BDD, извлекая всю информацию. Zephyr Squad стоит от 10 долларов в месяц до 10 пользователей и 4,55 доллара за пользователя в месяц после этого и становится дешевле по мере масштабирования. Zephyr Squad предлагает 30-дневную бесплатную пробную версию.

Xray для Jira — это полный инструмент управления тестированием, который помогает организовывать, планировать, выполнять и составлять отчеты о ходе тестирования и готовности к развертыванию. Xray использует собственные типы задач Jira. Благодаря Xray, изначально интегрированному в Jira, разработчики и специалисты ОКК работают в одной единой экосистеме, что обеспечивает прозрачность работы, видимость хода тестирования и сотрудничество между разработчиками и специалистами ОКК. Таким образом, каждый тест учитывается, и каждая задача находится в одном рабочем процессе. С помощью Xray менеджеры могут использовать гибкие доски для отслеживания хода выполнения тестов в режиме реального времени. Также имеются надежные возможности отчетности со встроенными отчетами для отслеживания и анализа покрытия, панель инструментов с гаджетами для простой оценки состояния тестирования. Xray стоит от 10 долларов за пользователя в месяц и предлагает 30-дневную бесплатную пробную версию.

В результате проведения сравнительного анализа был выбран инструмент для управления тестированием – Xray. Xray изначально интегрирован в Jira и использует свои типы задач, что позволяет выполнять работы централизованно в одном месте. Также Xray удобен в использовании и позволяет структурировать тест-кейсы. Другие возможности Xray позволяют наблюдать отчеты о тестировании в реальном времени и автоматизировать выгрузку других документов.

Мероприятия по оптимизации этапа разработки тестов:

– внедрение Xray в процесс управления тестированием;

– составление регламентов и инструкций по работе с Xray.

Имитационное моделирование для этапа тестирования проходило 2 раза. В первом случае рассматривался процесс, когда проводится дымовое или приемочное тестирование, то есть в качестве отчета используется карта переходов. А во втором случае рассматривался процесс, когда проводится регрессионное тестирование или тестирование по критическому пути с

полноценным отчетом и прогоном разработанных на предыдущем этапе тест-кейсов.

Результаты имитационного моделирования этапа тестирования для дымового или приемочного тестирования представлены на рисунках 14 и 15.

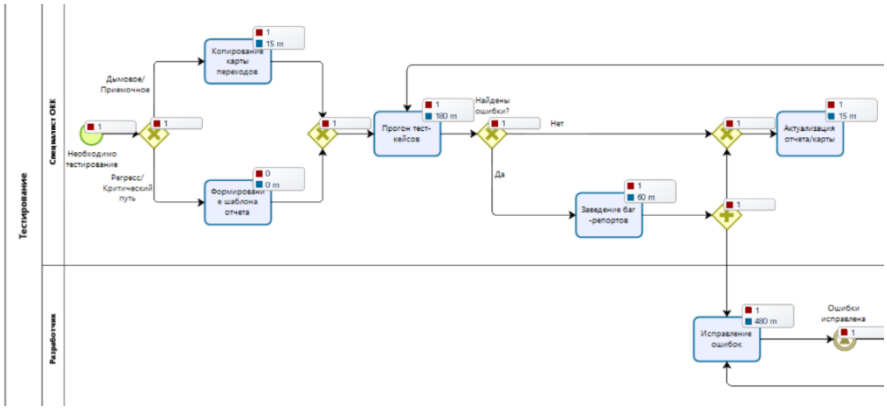


Рисунок 14 – Результаты имитационного моделирования этапа тестирования

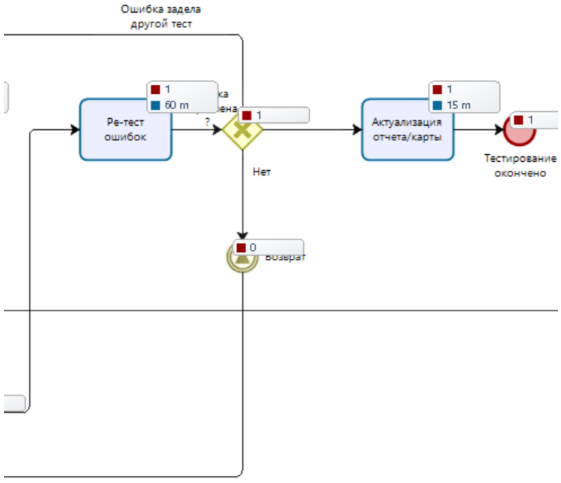


Рисунок 15 – Результаты имитационного моделирования этапа тестирования

По результатам прогона можно отметить, что данный процесс не подлежит оптимизации.

Результаты имитационного моделирования этапа тестирования для регрессионного тестирования или тестирования по критическому пути представлены на рисунках 16 и 17.

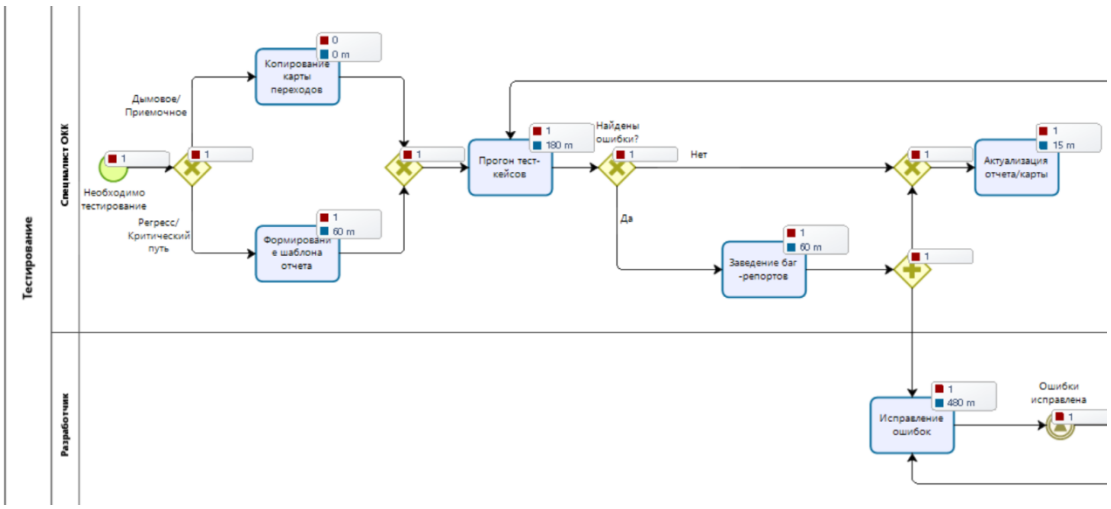


Рисунок 16 – Результаты имитационного моделирования этапа тестирования

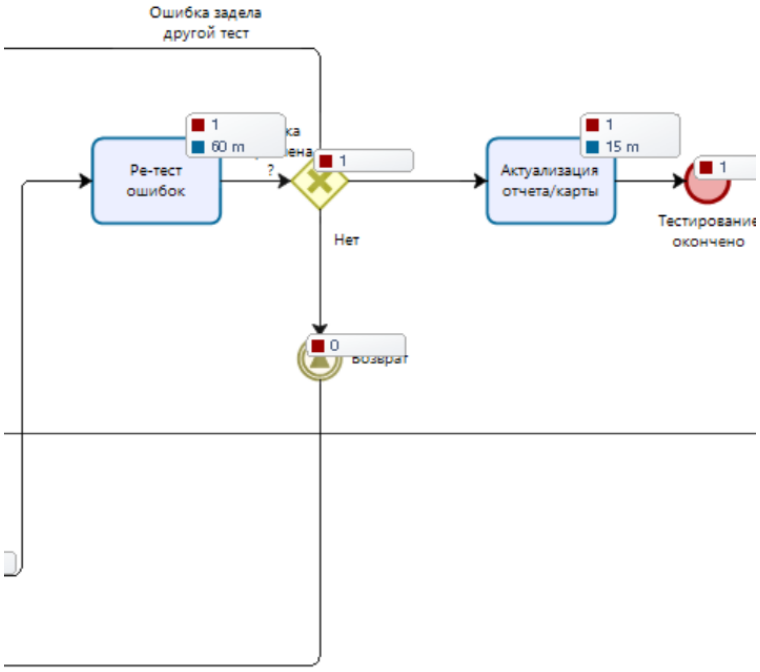


Рисунок 17 – Результаты имитационного моделирования этапа тестирования

Для проведения регрессионного тестирования или тестирования по критическому пути необходимо составлять отчет о тестировании, который хранится в виде страницы в Confluence и заполняется вручную специалистом ОКК. Для того чтобы подготовить шаблон для отчета необходимо его заполнить всеми тест-кейсами, которые были разработаны на этапе разработки тестов, что занимает весомое количество времени, в данном случае 1 час.

Также в последующем такой отчет необходимо всегда актуализировать. Благодаря внедрения специализированной системы для управления тестированием, отчеты о тестировании формируется автоматически и нет необходимости его постоянно актуализировать и следить, чтобы он оставался в актуальном состоянии, так как Xray отображает актуальное состояние отчета в режиме реального времени.

Действие «Заведение баг-репортов» считается частым и важным этапом в тестировании, так как именно на основе баг-репорта разработчик ищет и исправляет полученную ошибку. Важно, чтобы баг-репорты были полными и объясняли всю суть ошибки. Средства Jira позволяют создавать шаблоны при создании задач. Такой шаблон позволить немного сократить время создания баг-репортов и будет гарантировать, что специалист ОКК заполнить все важные части баг-репорта.

Мероприятия по оптимизации этапа тестирования:

– изменение процесса с учетом внедрения Xray;

– формирование шаблона для баг-репортов.

Результат имитационного моделирования этапа подготовки документации представлен на рисунке 18.

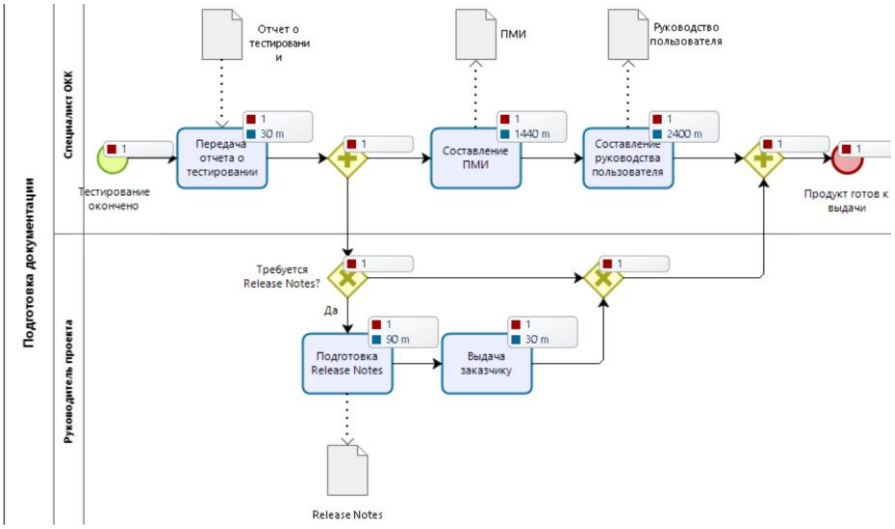


Рисунок 18 – Результат имитационного моделирования этапа подготовки документации

По результатам имитационного моделирования для этапа подготовки документации видно, что большое количество времени затрачивается на составление такой документации, как ПМИ и руководство пользователя. У Xray есть функция Document Generator, которая позволяет настраивать собственные шаблоны и создавать документы, из данных в Jira. ПМИ строится на основе тест-кейсов так как после внедрения Xray тест-кейсы хранятся в Jira, то с помощью Document Generator их можно выгрузить в документ, шаблон которого необходимо настроить. Руководство пользователя – это документ, который предназначит для предоставления пользователям помощи в использовании продукта. Процесс создания такого документа не подлежит оптимизации. Помимо ПМИ с помощью Document Generator можно составить шаблон для выгрузки такого документа как Release Notes. Release Notes строится на основе внесенных в систему изменений, это могут быть ошибки или задачи, которые хранятся в Jira.

Мероприятия по оптимизации этапа подготовки документации:

– подготовка шаблона для Release Notes;

– подготовка шаблона для ПМИ.

3 Результаты проведенного исследования

3.1 Моделирование оптимизированных бизнес-процессов тестирования программного обеспечения

По результатам проведенного анализа имитационного моделирования бизнес-процессов были составлены мероприятия по их оптимизации. Каждый этап работы специалиста ОКК на проекте был изменен в соответствии с этими мероприятиями с целью дальнейшего имитационного моделирования, которое позволит понять целесообразность внедрения рассматриваемых мероприятий.

Оптимизированный процесс подготовительного этапа работы специалиста ОКК на проекте представлен на рисунке 19.

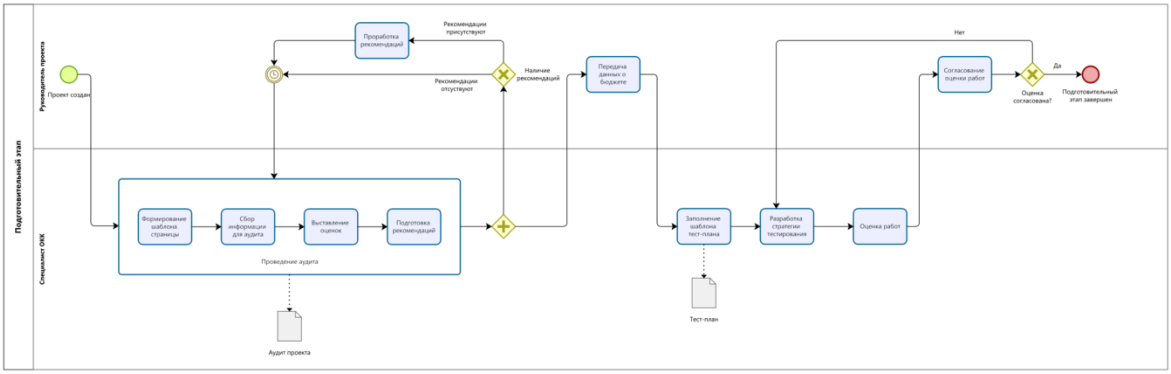


Рисунок 19 – Оптимизированная модель подготовительного этапа

Согласно мероприятиям, выявленным на этапе анализа бизнес-процессов, были осуществлены следующие изменения:

– с учетом автоматизации сбора метрик для аудита, сокращается время обработки действия «Сбор информации для аудита» до 1,5 часов. Так как часть метрик автоматизировать не предоставляется возможности, то сократить время на их сбор еще больше не получится.

– добавлено действие «Передача данных о бюджете» у руководителя проекта, время обработки которого будет занимать в среднем 30 минут.

– время обработки действия «Согласование оценки работ» сократилось до 1 часа за счет того, что было добавлено действие «Передача данных о

бюджете» так как добавленное действие подразумевает, что специалист ОКК будет укладываться в указанные рамки.

– процент вероятности того, что оценка будет не согласована сокращается до 5%. В 5% заложены всевозможные ситуации, которые могут отрицательно повлиять на согласование оценки.

Оптимизированный процесс этапа анализа требований продуктового проекта представлен на рисунке 20.

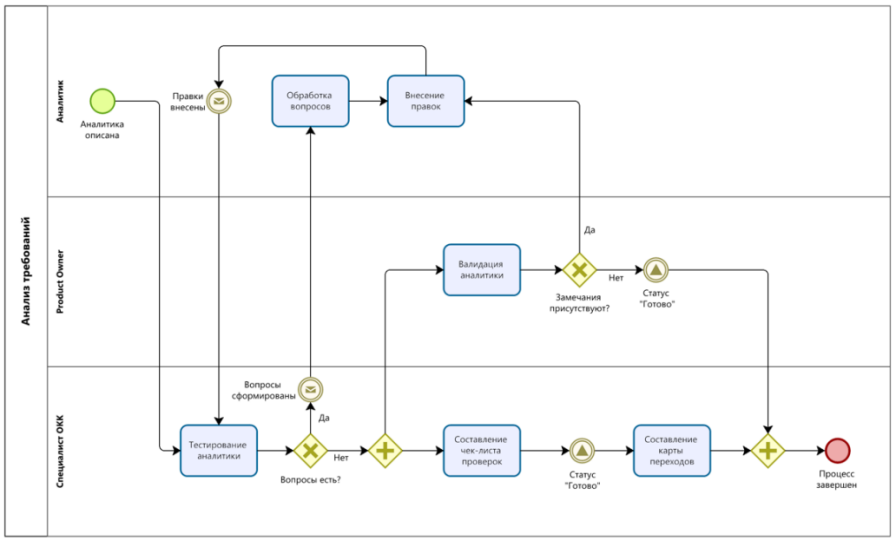


Рисунок 20 – Оптимизированная модель этапа анализа требований продуктового проекта

Согласно мероприятиям, выявленным на этапе анализа бизнес-процессов, был полностью изменен процесс анализа требований продуктового проекта. Основные изменения связаны с тем, что валидации аналитики Product Owner совершается после этапа тестирования, что привело к изменениям в самом процессе.

После написания документации аналитиком специалист ОКК приступает к тестированию. В ходе тестирования могут возникнуть вопросы и замечания по функциональности или структуре составления аналитической документации. Если замечания от специалиста ОКК присутствуют, то аналитик обрабатывает полученные вопросы и вносит соответствующие правки в документацию, после чего аналитика возвращается в тестирование с целью проверки внесенных изменений. Как только вопросы по аналитической

документации от специалиста ОКК заканчиваются, страница переходит на вариацию Product Owner. Если от Product Owner поступили замечании, вероятность наступления чего низка, то аналитика возвращается на доработку к аналитику и далее проходит цикл тестирования. Как только аналитическая документации принята Product Owner, то есть от него отсутствуют замечания выставляется статус «Готово». Параллельно валидации специалист ОКК может приступать к составлению чек-листов и карты переходов. В случае, если аналитика возвращается от Product Owner с замечаниями, специалист ОКК после тестирования и в случае необходимости сможет актуализировать уже написанный чек-лист и составленную карту переходов. Процесс анализа требований продуктового проекта считается завершены, когда аналитика прошла валидацию Product Owner и специалист ОКК выполнил необходимые задачи, то есть составил чек-лист и карту переходов.

Для имитации оптимизированной модели бизнес-процесса были внесены следующие изменения:

– время обработки действия «Валидация аналитики» увеличилось до 1 часа за счет того, что документация, которая приходит в оптимизированном процессе более полная, а, следовательно, и объемная, на проверку такой аналитики в среднем может уходить 1 час.

– вероятность того, что после валидации аналитики будут присутствовать замечания, сократилась до 15%. 15% - это вероятность того, что Product Owner захочет добавить новую или изменить существующую функциональность системы.

Оптимизированный процесс этапа разработки тестов представлен на рисунке 21.

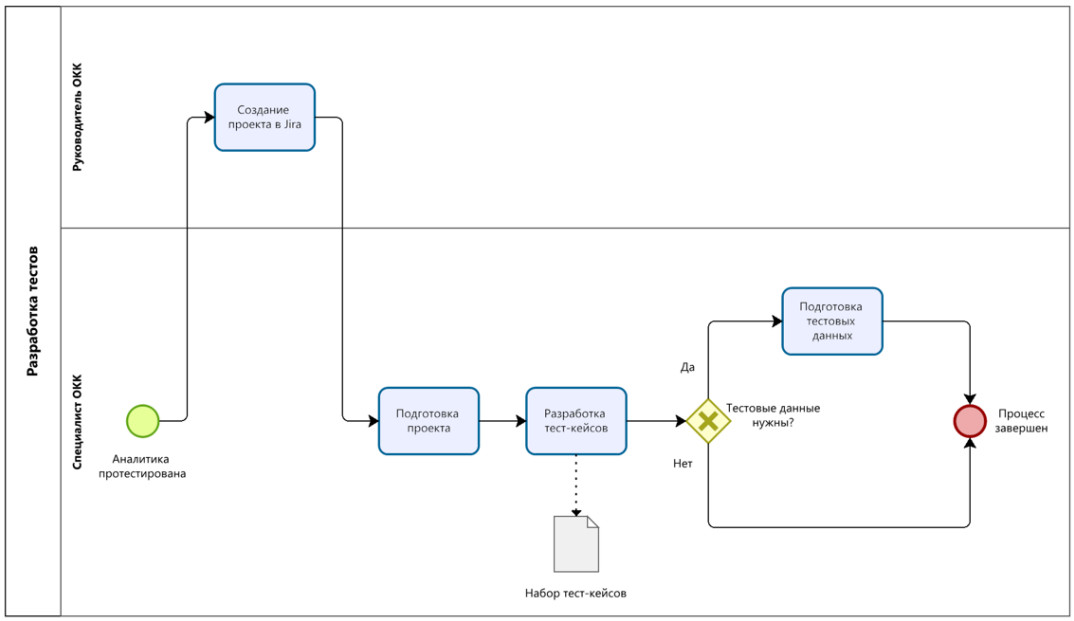


Рисунок 21 – Оптимизированная модель этапа разработки тестов

Для оптимизации бизнес-процесса разработки тестов было решено внедрить специализированный плагин Jira для управления тестированием - Xray. Так как Xray является надстройкой для Jira, то ведение всей документации будет проходить в ней, для чего необходимо создавать новые проекты. Для создания проекта в Xray необходимо подключение руководителя ОКК. Как только проект будет создан, специалисту ОКК необходимо его подготовить в зависимости от принятой структуры и после чего он приступает к разработке тест-кейсов. Также при необходимо специалисту необходимо подготовить тестовые данные.

Для имитации оптимизированной модели бизнес-процесса были внесены следующие изменения:

– добавлен новый ресурс – это руководитель ОКК, затраты за час, которого составляют 812,5 рублей;

– добавлено действие «Создание проекта в Jira» у руководителя ОКК. Время обработки действия – 15 минут;

– действие «Формирование шаблона страницы» заменено на действие «Подготовка проекта»;

– время обработки действия «Подготовка проекта» составляет 15 минут.

Время разработки тест-кейсов и подготовки тестовых данных не изменилось.

Оптимизированный процесс этапа тестирования представлен на рисунке 22.

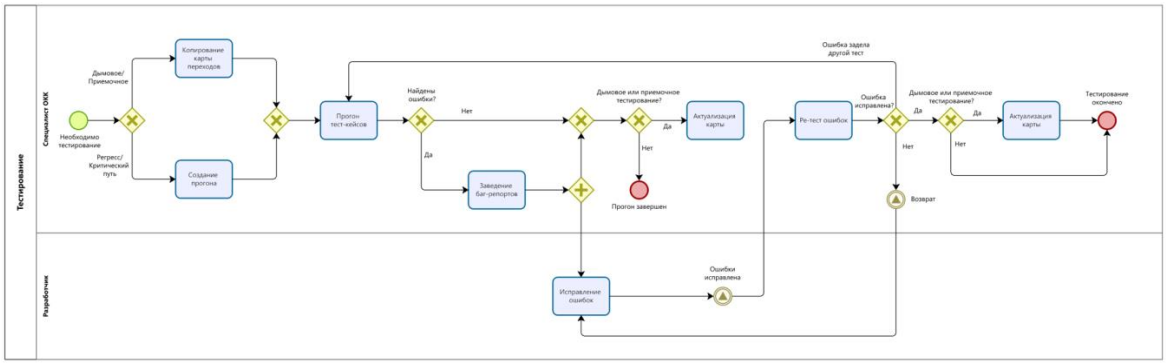


Рисунок 22 – Оптимизированная модель этапа тестирования

Внедрение специализированной программы для управления тестированием позволило изменить процесс для регрессионного тестирования или тестирования по критическому пути. Дымовое или приемочное тестирование не зависят от используемой программы для управления тестированием, поэтому изменений в соответствующей ветке процесса не произошло.

Для имитации оптимизированной модели бизнес-процесса были внесены следующие изменения:

– действие «Формирование шаблона отчета» заменено на действие «Создание прогона». Время обработки действия «Создание прогона» занимаем 15 минут так, как в Xray прогон создается довольно быстро на основе разработанных тест-кейсов.

– время обработки действия «Заведение баг-репортов» сократилось до 45 минут за счет сформированного в Jira шаблона для заведения баг-репортов.

– для регрессионного тестирования или тестирования по критическому пути отсутствует действие актуализации отчета после прогона и после ре-теста, так как Xray отображается отчет о тестировании автоматически в реальном времени.

Оптимизированный процесс этапа подготовки документации представлен на рисунке 23.

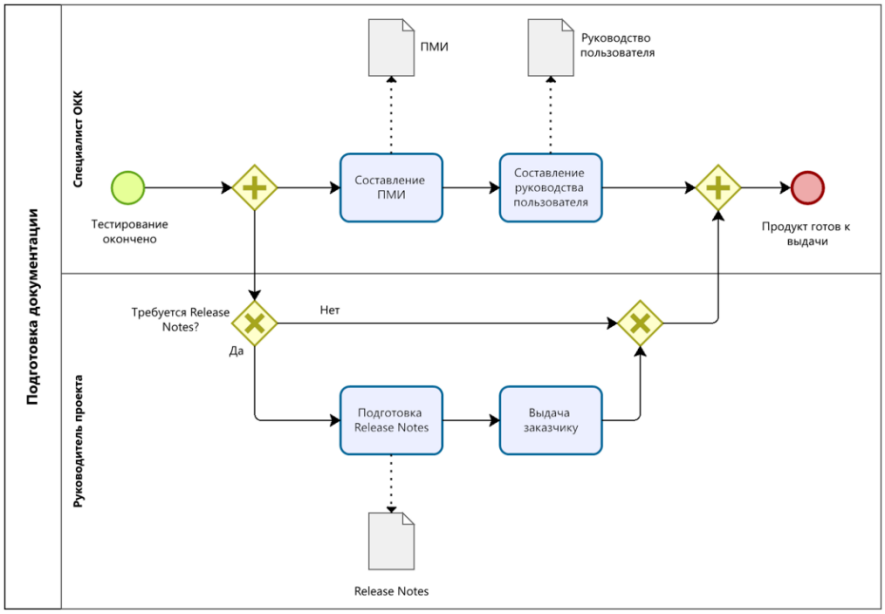


Рисунок 23 – Оптимизированная модель этапа подготовки документации

Внедрение Xray позволило избавиться от действия передачи отчета о тестировании так, как руководитель сам может получать отчет в Jira о прохождении тестирования, которых хранится там в актуальном состоянии в режиме реального времени.

Для имитации оптимизированной модели бизнес-процесса были внесены следующие изменения:

– время обработки действия «Подготовка Release Notes» сократилось до 30 минут, благодаря шаблону, который выгружается в один клик. Руководителю проекта остается только поправить выгруженный документ в случае необходимости;

– время обработки действия «Составление ПМИ» сократилось до 8 часов, благодаря шаблону, который выгружается в один клик. Специалисту ОКК остается только поправить выгруженный документ с учетом требований заказчика. Так как документ ПМИ объемный и может достигать 40 страниц, то время обработки данного действия все равно весомое.

3.2 Анализ результатов имитационного моделирования оптимизированных бизнес-процессов тестирования программного обеспечения

Аналогично имитационному моделированию текущих бизнес-процессов были получены результаты имитационного моделирования оптимизированных бизнес процессов. Также на моделях по результатам имитационного моделирования отображается легенда, где значение красного квадрата означает количество выполнений действия, а значение синего квадрата – общее время обработки действия.

На рисунках 24 и 25 представлен результат имитационного моделирования подготовительного этапа оптимизированной модели.

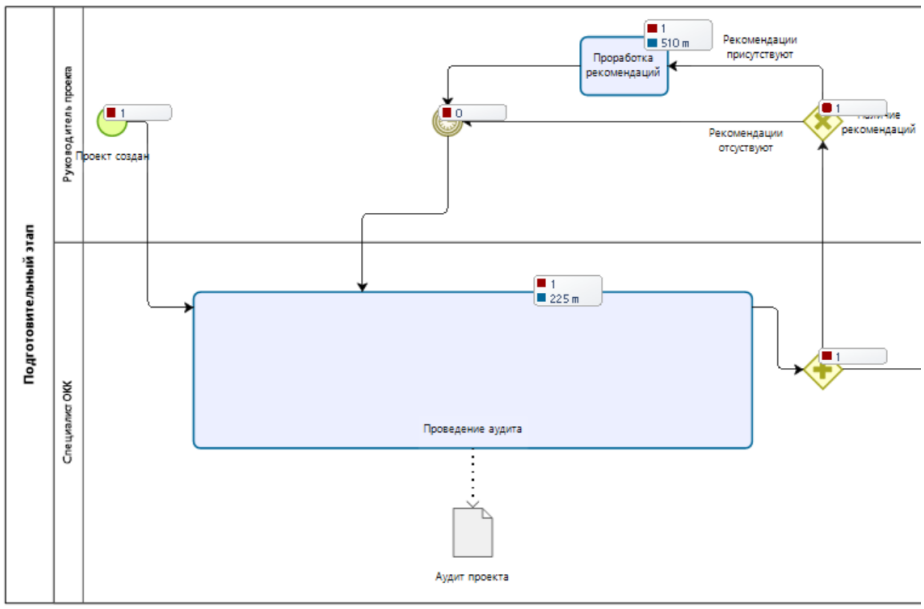


Рисунок 24 – Результаты имитационного моделирования подготовительного этапа оптимизированной модели

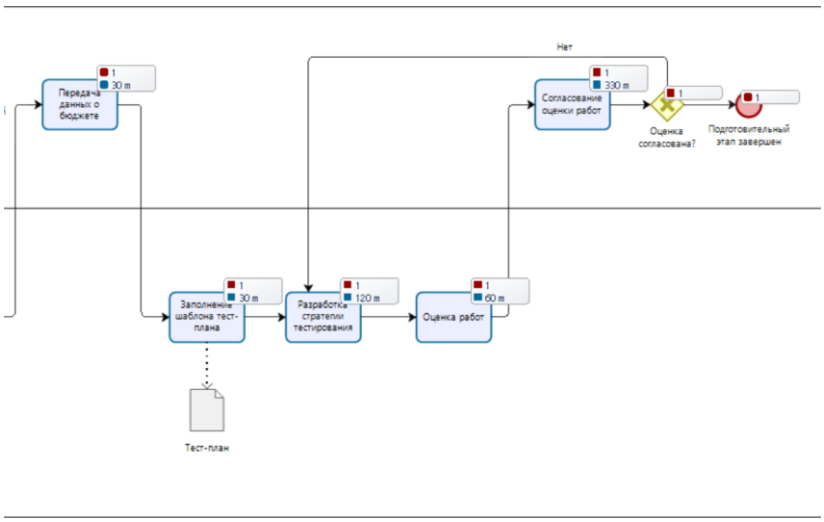


Рисунок 25 – Результаты имитационного моделирования подготовительного этапа оптимизированной модели

В результате имитационного моделирования подготовительного этапа оптимизированной модели сократилось время на проведение аудита, а именно сбора информации для него. Также за счет того, что был добавлен процесс «Передача данных о бюджете» вероятность наступления цикла из-за несогласования оценки работ уменьшилась, а значит уменьшилось время на выполнение таких действий как «Разработка стратегии тестирования», «Оценка работ» и «Согласование оценки работ».

На рисунке 26 представлен результат имитационного моделирования этапа анализа требований продуктового проекта оптимизированной модели.

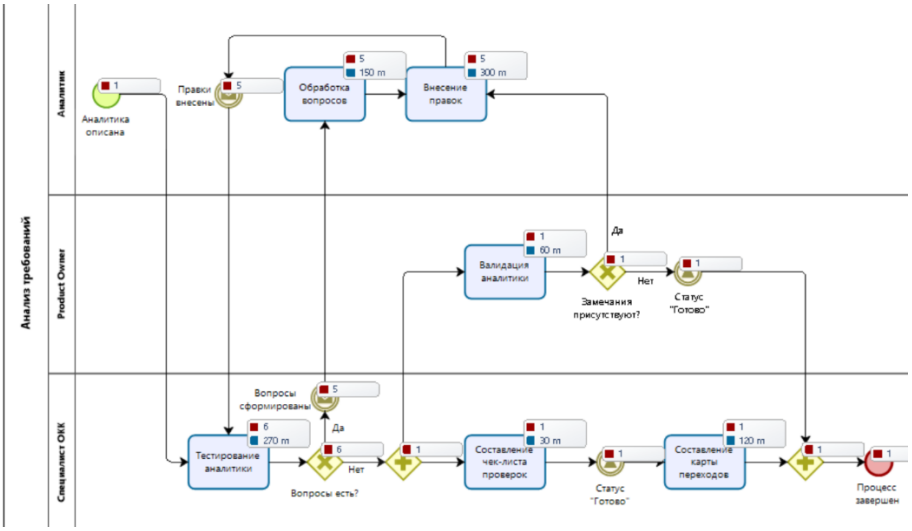


Рисунок 26 – Результаты имитационного моделирования этапа анализа требований продуктового проекта оптимизированной модели

В результате имитационного моделирования этапа анализа требований оптимизированной модели сократилось время работы Product Owner так как он участвует в валидации только на конечном этапе. Также сократилась цикличность возврата замечаний от Product Owner за счет того, что специалист ОКК приступает к тестированию аналитической документации раньше и находит противоречия или отсутствующие сценарии функционирования системы на своем этапе. Это же позволяет Product Owner видеть результаты аналитической документации в полном объеме. В целом процесс сократился по объему, по времени и по затрачиваемым ресурсам.

На рисунке 27 представлен результат имитационного моделирования этапа разработки тестов оптимизированной модели.

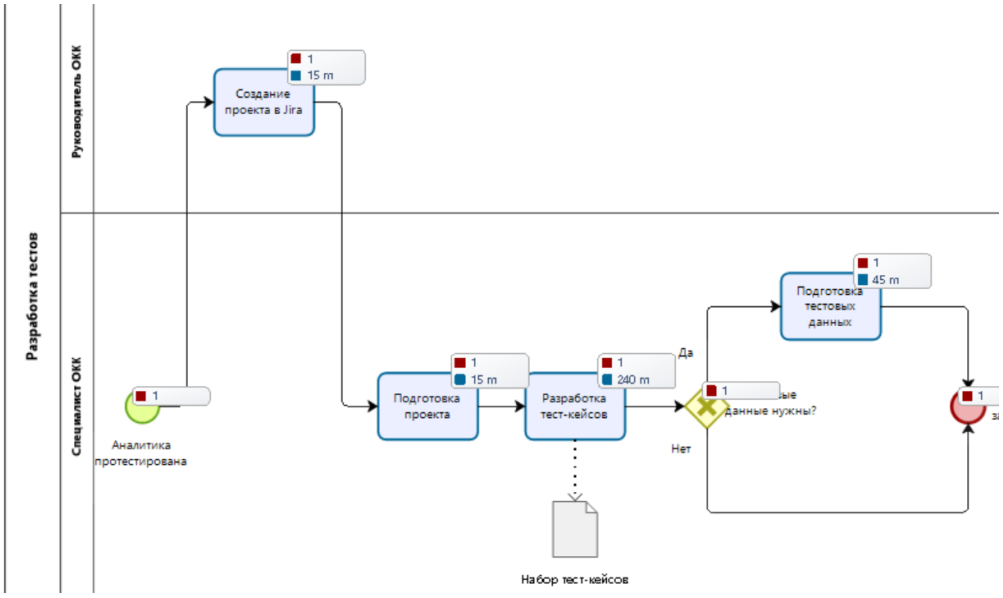


Рисунок 27 – Результаты имитационного моделирования этапа разработки тестов оптимизированной модели

В результате имитационного моделирования этапа разработки тестов оптимизированной модели время работы процесса увеличилось за счет того, что добавляется работа руководителя ОКК. Несмотря на это, такого решение позволит выиграть по времени и затратам в последующих процессах.

На рисунках 28 и 29 представлены результат имитационного моделирования этапа тестирования оптимизированной модели. Имитационное моделирование проводилось только для регрессионного тестирования или

тестирования по критическому пути, потому что только эта часть процесса была изменена.

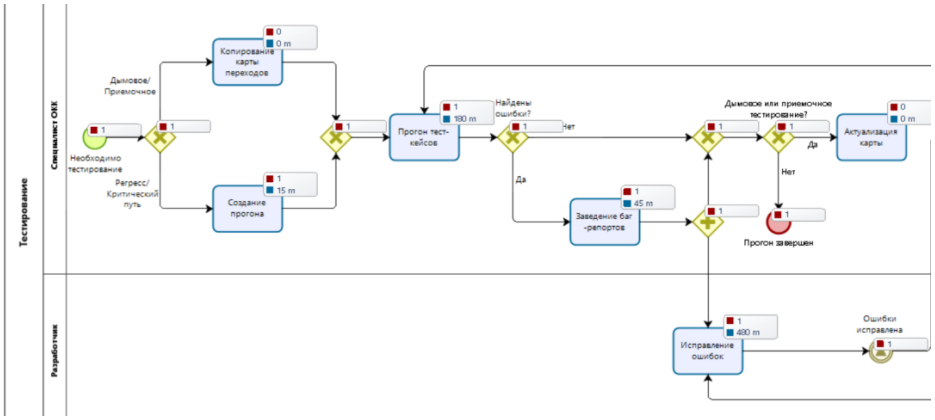


Рисунок 28 – Результаты имитационного моделирования этапа тестирования оптимизированной модели

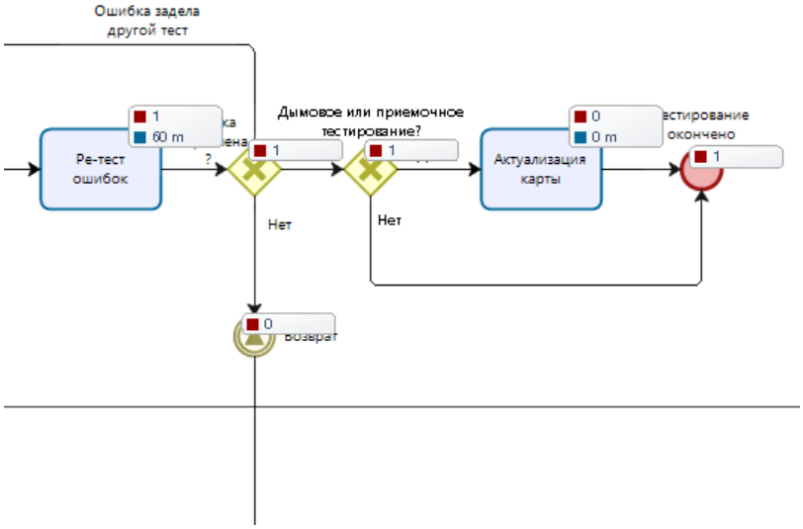


Рисунок 29 – Результаты имитационного моделирования этапа тестирования оптимизированной модели

В результате имитационного моделирования этапа тестирования для регрессионного тестирования или тестирования по критическому пути заметно, что сократилось время на подготовку отчета о тестировании так, как благодаря Xray такой отчет теперь строится автоматически в режиме реального времени. Также нет необходимости заниматься его актуализацией после прогона или ре-теста ошибок, что сокращает время работы специалиста ОКК и самого процесса в целом.

На рисунке 30 представлен результат имитационного моделирования этапа подготовки документации оптимизированной модели.

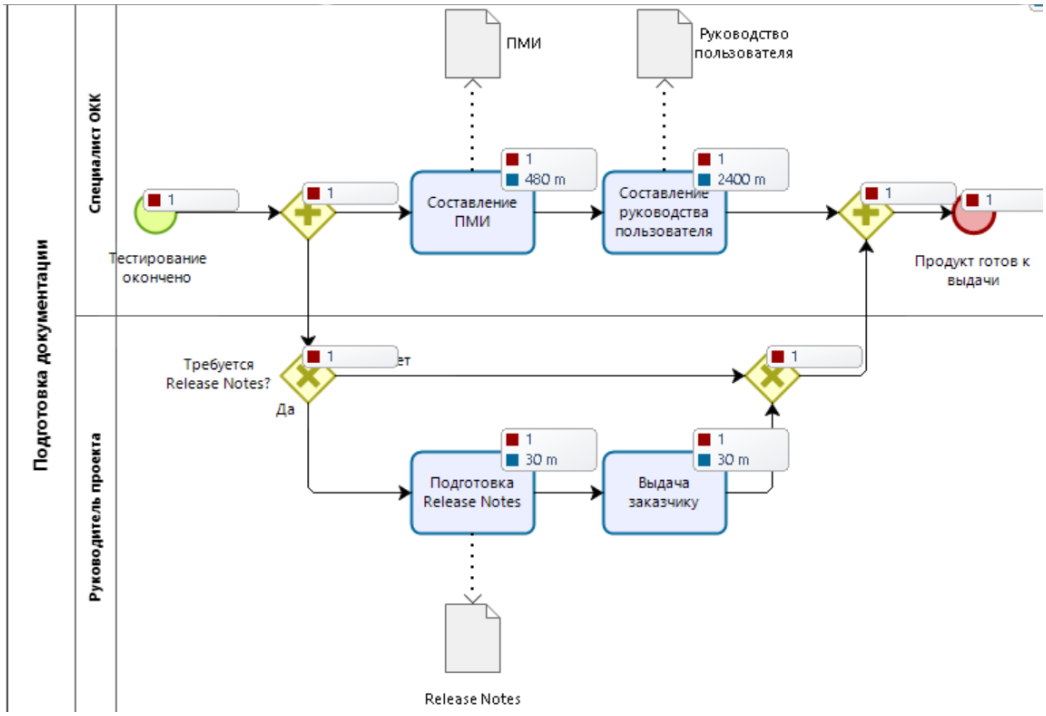


Рисунок 30 – Результаты имитационного моделирования этапа подготовки документации оптимизированной модели

В результате имитационного моделирования этапа подготовки документации оптимизированной модели видно, что сократилось время участия руководителя проекта в составлении Release Notes и сократилось время участия специалиста ОКК в составлении такой документации, как ПМИ. А значит и сократилось время выполнения всего процесса.

3.3 Внедрение мероприятий по оптимизации бизнес-процессов тестирования программного обеспечения

После анализа имитационного моделирования бизнес-процессов работы специалистов ОКК на проектах были выделены следующие мероприятия по их оптимизации:

– автоматизация процесса сбора информации для аудита;

– внедрение Xray в процесс управления тестированием;

– составление регламентов и инструкций по работе с Xray;

– формирование шаблона для баг-репортов;

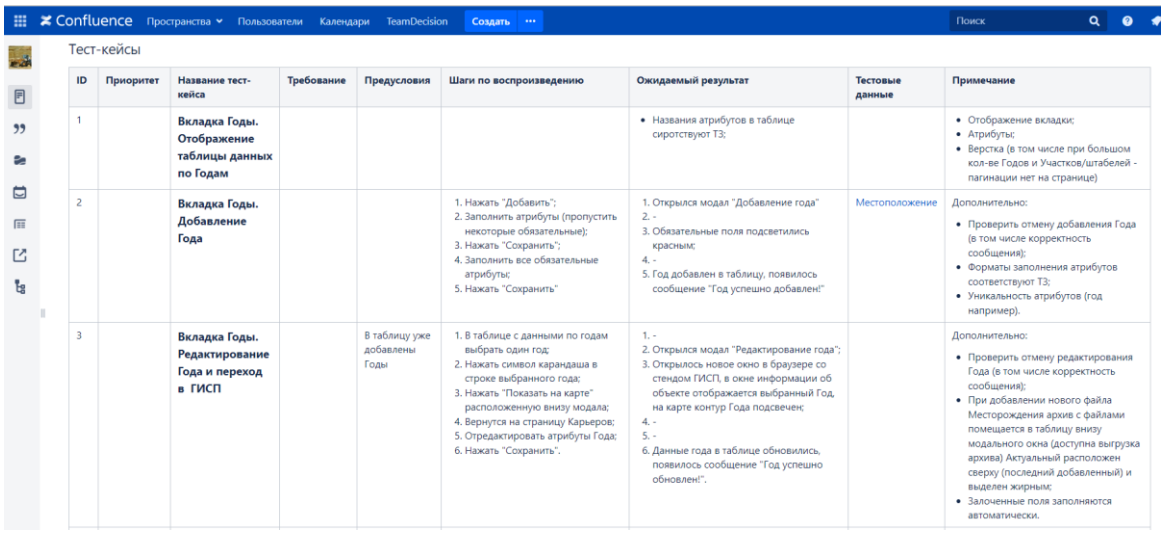
– подготовка шаблона для Release Notes;

– подготовка шаблона для ПМИ.

Стоит отметить, что вышеперечисленные мероприятия – это не все мероприятия, которые были предложены для оптимизации. Часть мероприятий, связанных с изменениями последовательности действий процесса не внесена в список, так как не требует дополнительных ресурсов и затрат на их внедрение. Далее будет рассмотрен процесс внедрения перечисленных мероприятий.

Для внедрения Xray в процесс управления тестированием необходимо выделить специалиста ОКК для изучения системы и составления регламентов и инструкций по работе с Xray. Время, затраченное на данные работы, составило 100 часов. Внедрение Xray изменило следующие процессы этапов тестирования, описанные ниже.

Как уже было сказано выше, тест-кейсы хранились на странице в Confluence. Пример страницы с тест-кейсами представлен на рисунке 31.



ID	Приоритет	Название тест-кейса	Требование	Предусловия	Шаги по воспроизведению	Ожидаемый результат	Тестовые данные	Примечание
1		**Вкладка Годы. Отображение таблицы данных по Годам**				• Названия атрибутов в таблице сиротствуют ТЗ;		• Отображение вкладки; • Атрибуты; • Верстка (в том числе при большом кол-ве Годов и Участков/штабелей - пагинации нет на странице)
2		**Вкладка Годы. Добавление Года**			1. Нажать "Добавить"; 2. Заполнить атрибуты (пропустить некоторые обязательные); 3. Нажать "Сохранить"; 4. Заполнить все обязательные атрибуты; 5. Нажать "Сохранить"	1. Открылся модал "Добавление года" 2. - 3. Обязательные поля подсветились красным; 4. - 5. Год добавлен в таблицу, появилось сообщение "Год успешно добавлен!"	Местоположение	Дополнительно: • Проверить отмену добавления Года (в том числе корректность сообщения); • Форматы заполнения атрибутов соответствуют ТЗ; • Уникальность атрибутов (год например).
3		**Вкладка Годы. Редактирование Года и переход в ГИСП**		В таблицу уже добавлены Годы	1. В таблице с данными по годам выбрать один год; 2. Нажать символ карандаша в строке выбранного года; 3. Нажать "Показать на карте" расположенную внизу модала; 4. Вернутся на страницу Карьеров; 5. Отредактировать атрибуты Года; 6. Нажать "Сохранить".	1. - 2. Открылся модал "Редактирование года"; 3. Открылось новое окно в браузере со стендом ГИСП, в окне информации об объекте отображается выбранный Год, на карте контур Года подсвечен; 4. - 5. - 6. Данные года в таблице обновились, появилось сообщение "Год успешно обновлен!".		Дополнительно: • Проверить отмену редактирования Года (в том числе корректность сообщения); • При добавлении нового файла Месторождения архив с файлами помещается в таблицу внизу модального окна (доступна выгрузка архива) Актуальный расположен сверху (последний добавленный) и выделен жирным; • Залоченные поля заполняются автоматически.

Рисунок 31 – Страница в Confluence с тест-кейсами

С введением Xray тест-кейсы стали хранится в Jira. Тест-кейсы в Jira разбиты по тестовым наборам, которые соответствую аналитической документации. Все хранится централизованной в одном месте и нет

необходимости постоянно создавать шаблоны страниц в Confluence для разработки и хранения тест-кейсов. На рисунке 32 представлено хранение тест-кейсов в Jira с использованием плагина Xray.

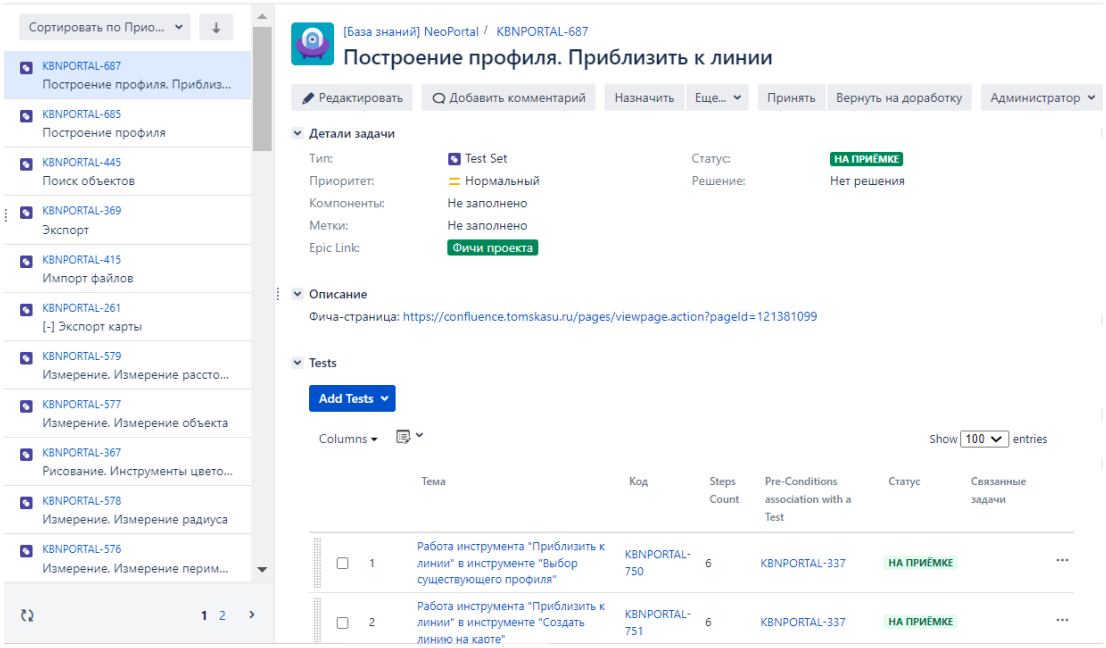


Рисунок 32 – Хранение тест-кейсов в Jira

Помимо изучения Xray были составлены регламент и инструкции по использованию Xray, которые хранятся в Confluence в базе знаний ОКК. Список инструкций представлен на рисунке 33.

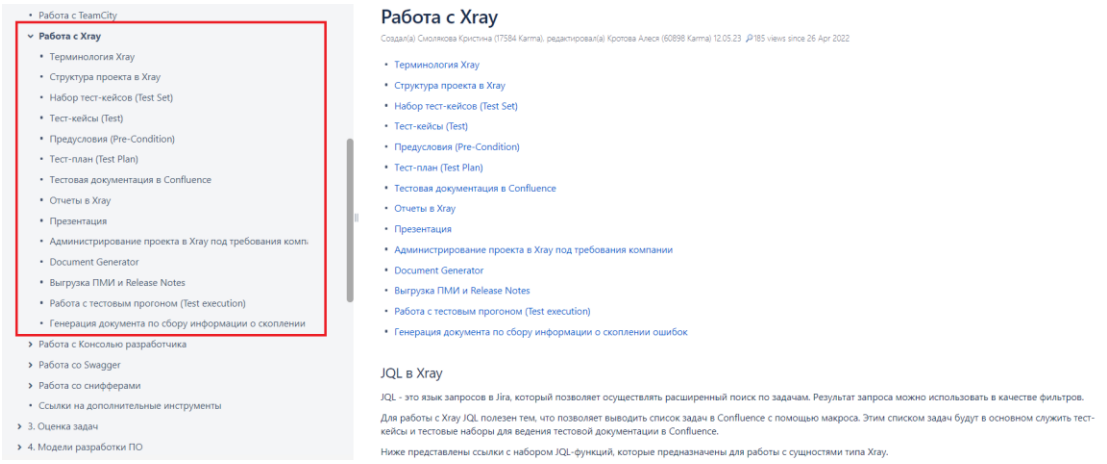


Рисунок 33 – Инструкции для работы с Xray

После прохождения тестирования необходимо составлять отчеты о тестировании. Когда тест-кейсы хранились в Confluence, отчеты о тестировании составлялись там же вручную специалистом ОКК. Вид отчета о тестировании в Confluence представлен на рисунке 34.

№	Название фичи	Название проверки	Результат проверки	Номер ошибки	Серьезность ошибки	Комментарий
1	1. Главное окно	Вход в систему	ПРОВЕРЕНО			
2	1. Главное окно	Главное окно	ПРОВЕРЕНО	61388 61470 61578 61937 61940		• ошибка лого (61388) • *Комментарий: Задача должна быть решена, кодовая база соответствует требованиям, задача возникла из-за недопонимания.* • отсутствуют иконки для кнопок домой, справочник, отчеты (61470) • блок "общие" не отображается при переходе на стенд (61578) • проверить, что везде стоит тюменнефтегаз без мягкого знака (61937) • Изменить "карьеры" на реестр карьеров" (61940)
3	1. Главное окно	Общая информация	ПРОВЕРЕНО	61471		• ошибка отображения в блоке "общие"
4	1. Главное окно	Дизайн стартового окна	ПРОВЕРЕНО	61855		редизайн (61855)
5	1. Главное окно	Работа с деревом карьеров	ПРОВЕРЕНО	61891 62544		• В дереве ЛУ всегда отображаются только "За пределами ЛУ", "Прочие",

Рисунок 34 – Отчет о тестировании в Confluence

Xray автоматически строит отчет о тестировании после прохождения прогона. Во-первых, есть статус-бар, который показывает результаты прогона в ходе тестирования. Пример, такого статус-бара представлен на рисунке 35.

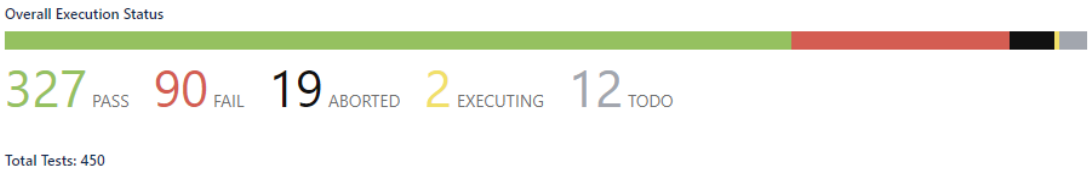


Рисунок 35 – Статус прохождения прогона тестирования

На рисунке 35 видно несколько статус, где зеленый – тест пройден, красный – тест упал с ошибкой, черный – тест пропущен по какой-либо причине, желтый – тест в процессе выполнения, серый – тест еще не был выполнен. Также показано общее количество тест-кейсов, что позволяет наглядно увидеть статус прохождения тестирования и остаток тестирования.

Во-вторых, присутствую отдельные отчеты о тестировании, которые генерируются автоматически. Это отчеты по тест-плану, отчеты по прогону, отчеты по прохождению каждого тест-кейса и отчеты, связанные с требованиями. Пример сгенерированного отчета представлен на рисунке 36.

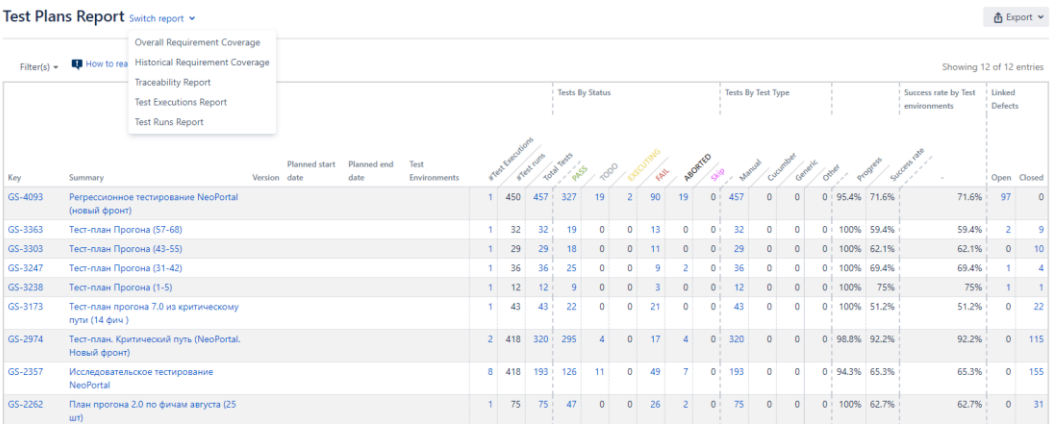


Рисунок 36 – Сгенерированный отчет в Xray

Следующее мероприятие для оптимизации процесса – это составление шаблона для баг-репортов. Шаблон для аг-репорта можно создать с помощью сценария автоматизации в Jira. Сценарий автоматизации содержит: правило для выполнения действий, условие срабатывания действия и само действие. Так для создания шаблона ошибки правило выполнения действия – это когда задача создана, условие выполнения – если задача является ошибкой и само действие – это изменение описания ошибки, то есть добавление обязательных полей для заполнения. Сценарий автоматизации представлен на рисунке 37, а результат на рисунке 38.

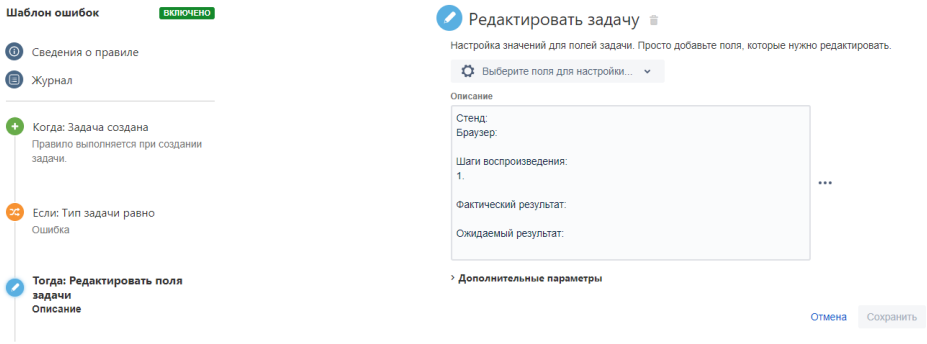


Рисунок 37 – Сценарий автоматизации для шаблона баг-репорта

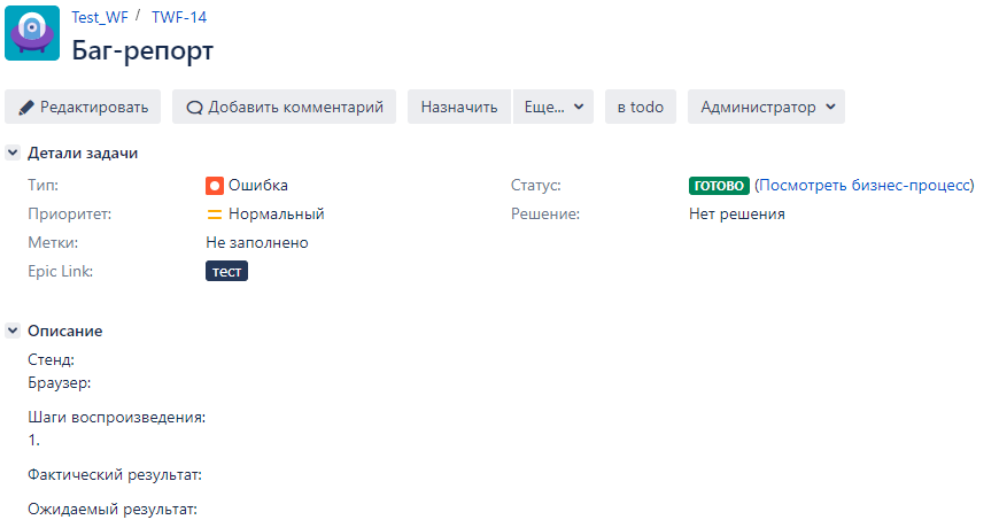


Рисунок 38 – Результат автоматизации создания шаблона баг-репорта

Последние два мероприятия связаны с генерацией шаблона для выгрузки документов из Jira. Выгрузка документов из Jira осуществляется с помощью Document Generator, куда необходимо загрузить шаблон. Прежде, чем загрузить шаблон надо его разработать. Разработка шаблона осуществляется с помощью скрипта. Пример сгенерированного шаблона для Release Notes представлен на рисунке 39, а для ПМИ на рисунке 40.

Наименование Release Notes - ${FixVersions}

Документовед	${Username}
Задача	
Репозиторий	
Версия	
Оповещение Заказчика	
Замечания по релизу	

№ задачи	**Тип задачи**	**Заказчик**	**Наименование задачи**	**Описание**
&{for issues}				
1.	${IssueTypeName}	${Components}	${Summary}	${Description} &{end}

Рисунок 39 – Шаблон документа для генерации Release Notes

ПРИЛОЖЕНИЕ

Таблица 3 - Методика проверки функций

№	НАИМЕНОВАНИЕ МЕТОДИКИ/ИСПЫТАНИЯ	ВЫПОЛНЯЕМЫЕ ДЕЙСТВИЯ	УСЛОВИЯ ОКОНЧАНИЯ ПРОВЕРКИ	РЕЗУЛЬТАТ ИСПЫТАНИЙ (ПРИНЯТО / НЕ ПРИНЯТО)	ЗАМЕЧАНИЯ
#{for j=TestsCount}					
1.	${Tests[j].Summary}	#{if (%{'${Tests[j].PreConditionsCount}'>'0'})} Предусловие: #{for p=Tests[j].PreConditionsCount} ${Tests[j].PreConditions[p].Summary}; #{end} #{end} Шаги: #{for m=Tests[j].TestStepsCount} ${Tests[j].TestSteps[m].StepNumber}. ${Tests[j].TestSteps[m].Action}; #{end}	#{for m=Tests[j].TestStepsCount} ${Tests[j].TestSteps[m].StepNumber}. ${Tests[j].TestSteps[m].ExpectedResult}; #{end}		
#{end}					

Рисунок 40 – Шаблон документа для генерации ПМИ

Стоит отметить, что для ПМИ было разработано три варианта шаблона: без предусловий, с предусловиями для выгрузки по конкретной функциональности и с предусловиями для выгрузки по всему проекту.

3.4 Расчет эффективности внедрения мероприятий для оптимизации бизнес-процессов тестирования программного обеспечения

Для расчета эффективности внедрения мероприятий по оптимизации бизнес-процесса тестирования необходимо рассчитать затраты бизнес-процесса до оптимизации и после нее, а также и затраты на разработку запланированных мероприятий.

В процессе имитационного моделирования были получены общие временные затраты на процесс, а также установлена стоимость ресурса за час работы. Стоит отметить, что в ходе расчетов не будут учитываться постоянные затраты так, как вне зависимости от процесса такие затраты одинаковые и подлежат сокращению. Так как в процессах участвую только сотрудники компании, то необходимо учитывать отчисления в социальные фонды, а именно 30,2%.

В таблице 8 представлены расчеты затрат на процесс до его оптимизации.

Таблица 8 – Расчет затрат на бизнес-процесс

Этап	Ресурс	Общее время работы, час	Стоимость, руб./час	Сумма, руб.
Подготовительный	Руководитель проекта	12	562,5	6 750
	Специалист ОКК	11,75	335,5	3 942,13
Анализ требований продуктового проекта	Специалист ОКК	6,25	335,5	2 096,88
	Аналитик	8	328	2 642
	Product Owner	3	562,5	1 687,5
Разработка тестов	Специалист ОКК	5	335,5	1 677,5
Тестирование	Специалист ОКК	5,75	335,5	1 929,13
	Разработчик	8	606	4 848
Подготовка документации	Руководитель проекта	2	562,5	1 125
	Специалист ОКК	64,5	335,5	21 639,75
Итого (+ взносы в социальные фонды)				62 935,93

В таблице 9 представлены расчеты затрат на оптимизированный процесс.

Таблица 9 – Расчет затрат на оптимизированный бизнес-процесс

Этап	Ресурс	Общее время работы, час	Стоимость, руб./час	Сумма, руб.
Подготовительный	Руководитель проекта	9,5	562,5	5 343,75
	Специалист ОКК	7,25	335,5	2 432,38
Анализ требований продуктового проекта	Специалист ОКК	7	335,5	2 348,5
	Аналитик	7,5	328	2 460
	Product Owner	1	562,5	562,5
Разработка тестов	Специалист ОКК	5	335,5	1 677,5
	Руководитель ОКК	0,25	812,5	203,13
Тестирование	Специалист ОКК	5	335,5	1 677,5
	Разработчик	8	606	4 848
Подготовка документации	Руководитель проекта	1	562,5	562,5

Продолжение таблицы 9

	Специалист ОКК	48	335,5	16 104
Итого (+ взносы в социальные фонды)				49 762,13

В результате проведенных расчетов можно отметить, что затраты на процесс после внедрения оптимизационных мероприятий сократились на 13 173,8 рублей или на 21%.

Отметим, что рассчитанные затраты не показывают стоимость работы на целом проекте, так как на этапах «Анализ требований», «Разработка тестов» и «Тестирование» рассматриваются затраты на работу над одной функцией системы. В среднем система содержит от 10 функций, поэтому сумма затрат на этих этапах будет выше, а значит и сумма все процесса тестирования.

Для расчета эффективности недостаточно посчитать только затраты оптимизированного процесса, необходимо также рассчитать затраты на мероприятия по оптимизации рассматриваемого бизнес-процесса. В таблице 10 представлены расчеты затрат на разработку мероприятий по оптимизации.

Таблица 10 – Расчет затрат на разработку мероприятий

Наименование	Ресурс	Общее время работы, час	Стоимость, руб./час	Сумма, руб.
Автоматизация сбора информации для аудита	Руководитель ОКК	8	812,5	6 500
Внедрение Xray + написание инструкций	Специалист ОКК	100	335,5	33 550
Разработка шаблона для баг-репортов	Специалист ОКК	3	335,5	1 006,5
Разработка шаблона для генерации Release Notes	Специалист ОКК	3	335,5	1 006,5
Разработка шаблона для генерации ПМИ	Специалист ОКК	6	335,5	2 013
Итого (+ взносы в социальные фонды)				57 386,95

Общая сумма затрат на разработку мероприятий по оптимизации бизнес-процесса составили 57 386,95 рублей. Внедрение таких мероприятий будет

эффективно если сумма затрат на них будет меньше разницы сумм затрат неоптимизированного и оптимизированного процессов. Как было посчитано выше эта разница составляет 13 173,8. Рассматривая один проект в целом такая разница некорректна так как не учитывается полнота проекта. Например, возьмем проект, который имеет 10 функций, тогда сумма затрат на этапы «Анализ требований», «Разработка тестов» и «Тестирование» увеличится в 10 раз. В таблице 11 представлена сводная таблица по полученным затратам на проект в целом.

Таблица 11 – Затраты на проект

Наименование	Сумма затрат, руб.
Неоптимизированный процесс	237 311,6
Оптимизированный процесс	211 202,5

Разница составила 26 109 рублей, что меньше затрат на разработку мероприятий по оптимизации. Учитывая, что мероприятия выполняются единожды, а проектов у компании больше одного, то уже на третьем проекте будет видна выгода в стоимостном отношении от внедрения мероприятий по оптимизации.

ЗАДАНИЕ К РАЗДЕЛУ
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Обучающемуся:

Группа	ФИО
3НМ15	Смоляковой Кристине Викторовне

Школа	Школа инженерного предпринимательства		
Уровень образования	магистратура	Направление/ООП/ОПОП	27.04.05 Инноватика/ Прикладной системный инжиниринг

Исходные данные к разделу «Социальная ответственность»:	
1. Описание организационных условий реализации социальной ответственности *−заинтересованные стороны (стейкхолдеры) программ социальной ответственности организации, проекта, инновационной разработки, на которых они оказывают воздействие;* *− стратегические цели организации, проекта, внедрения инновации, которые нуждаются в поддержке социальных программ;* *− цели текущих программ социальной ответственности организации*	Стейкхолдеры: собственник, работники, потребители, конкуренты, инвесторы. Цели организации: – развитие собственного персонала, которое позволяет избежать текучести кадров и привлечь лучших специалистов на рынке; – улучшение имиджа компании и рост репутации; – стабильность и устойчивость развития компании в долгосрочной перспективе.
2. Законодательные и нормативные документы	– ГОСТ Р ИСО 26000-2010 «Руководство по социальной ответственности»; – SA 8000
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Анализ факторов внутренней социальной ответственности:	– стабильность заработной платы; − поддержание социально значимой заработной платы; – развитие человеческих ресурсов через обучающие программы и программы подготовки и повышения квалификации;
2. Анализ факторов внешней социальной ответственности:	– спонсорство и корпоративная благотворительность; – ответственность перед потребителями товаров и услуг (выпуск качественных товаров).
Перечень графического материала:	
Таблица 12 – Определение целей КСО; Таблица 13 – Определение стейкхолдеров КСО; Таблица 14 – Определение элементов программы КСО; Таблица 15 – Затраты на мероприятия КСО;	

Таблица 16 – Эффективность мероприятий КСО.	

Дата выдачи задания к разделу в соответствии с календарным учебным графиком	

Задание выдал консультант по разделу «Социальная ответственность»:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Черепанова Н.В.	к. филос.н.		

Задание принял к исполнению обучающийся:

Группа	ФИО	Подпись	Дата
3НМ15	Смолякова Кристина Викторовна		

4 Социальная ответственность

4.1 Определение целей и задач программы КСО

Корпоративная социальная ответственность – международная бизнес-практика, которая прочно вошла в корпоративное управление в конце XX века. В настоящее время внедрение мероприятий КСО становится неотъемлемой частью успешной компании.

Корпоративная социальная ответственность – это:

– комплекс направлений политики и действий, связанных с ключевыми стейкхолдерами, ценностями и выполняющих требования законности, а также учитывающих интересы людей, сообществ и окружающей среды;

– нацеленность бизнеса на устойчивое развитие;

– добровольное участие бизнеса в улучшении жизни общества.

Иными словами, социальная ответственность бизнеса – концепция, согласно которой бизнес, помимо соблюдения законов и производства качественного продукта/услуги, добровольно берет на себя дополнительные обязательства перед обществом.

Компания ООО «Neo Stack Technology» пока не имеет стратегии КСО. Для того чтобы программы КСО приносили различные социальные и экономические результаты, необходима их интеграция в стратегию компании. Иными словами, деятельность компании и программы КСО должны иметь одинаковый вектор. Тогда программа КСО будет выступать органическим вспомогательным элементом деятельности компании. В таблице 12 представлены цели КСО в компании ООО «Neo Stack Technology».

Таблица 12 – Определение целей КСО

Миссия компании	Создание устойчивой среды для развития сотрудников компании, повышения конкурентоспособности и укрепления позиций на рынке

Продолжение таблицы 12

Стратегия компании	Укрепление позиций в качестве одной из ведущих компании в области разработки геоинформационных систем и создание среды для развития сотрудников и повышения их конкурентоспособности на рынке
Цели КСО	Развитие собственного персонала, которое позволяет избежать текучести кадров и привлечь лучших специалистов на рынке
	Улучшение имиджа компании и рост репутации
	Стабильность и устойчивость развития компании в долгосрочной перспективе

Возможность интеграции целей КСО в стратегию компании:

1. Развитие собственного персонала позволит компании как укрепить свои позиции на рыке в области разработки геоинформационных систем, так и повысить конкурентоспособность не только в качестве разрабатываемых программных продуктов, но и в качестве высококвалифицированных сотрудников с целью работы на аутсорсе. А создание среды для развития сотрудников позволит избежать текучки кадров и привлечь лучших специалистов на рынке, которые будут играть роль в развитии продуктов компании

2. Улучшение имиджа компании и рост репутации позволит укрепить свои позиции на рынке разработки программного обеспечения и привлечь новых высококвалифицированных специалистов. Ведь качество разрабатываемых продуктов повышает спрос у заказчиков и укрепляет их лояльность к компании.

3. Стабильность и устойчивость развития компании в долгосрочной перспективе также интегрируется со стратегией компании, как в укреплении своих позиций на рыке, так и в развитии сотрудников, ведь большинство стремятся к работе в стабильной и устойчивой компании.

4.2 Определение стейкхолдеров программы КСО

Стейкхолдерами или заинтересованными лицами называется любое сообщество внутри организации, или вне ее, предъявляющее определенные требования к результатам деятельности организации и характеризующееся определенной скоростью реакции. Среди множества стейкхолдеров выделяют: собственников, акционеров, органы федеральной и местной власти, поставщиков, топ-менеджеров, работников, профсоюзы, торговые группы, потребителей (внутренних, зарубежных), население, партнеров, инвесторов, кредиторов, конкурентов (внутренних, международных), профессиональные ассоциации, суды и др.

Выбор основных стейкхолдеров проводится исходя из целей программы КСО. К каждой цели программы определяются наиболее влиятельные стейкхолдеров. Стейкхолдеры программы КСО в компании представлены в таблице 13.

Таблица 13 – Определение стейкхолдеров КСО

№	Цели КСО	Стейкхолдеры
1	Развитие собственного персонала, которое позволяет избежать текучести кадров и привлечь лучших специалистов на рынке	Собственник, работники, потребители, конкуренты
2	Улучшение имиджа компании и рост репутации	Собственник, работники, потребители, инвесторы
3	Стабильность и устойчивость развития компании в долгосрочной перспективе	Собственник, работники, потребители, инвесторы

Собственник заинтересован в возможности контроля и управления финансовыми потоками, мощность которых свидетельствует о финансовой состоятельности предприятий, развитии и стабильности. Имидж компании, развитие персонала предприятия и привлечение специалистов непосредственно влияют на рост стоимости самого предприятия и способствуют увеличению продаж и стабильной работе.

Работники ожидают удовлетворения их труда в формах адекватной оплаты, возможностей профессионального роста и построения деловой карьеры, здоровой моральной атмосфере, приемлемых условий и режима труда, хорошего руководства.

Потребителей интересует качество выпускаемого компании программного продукта, от которого зависит их лояльность к компании. Потребители выбирают ту компанию, которая поставляет качественное ПО в долгосрочной для них перспективе.

Конкуренты для компании также играют важную роль, ведь они могут определить путь развития компании и повлиять на развитие навыков сотрудников.

Интересы инвесторов, акционеров и поставщиков связаны с эффективностью управления организацией. Так как инвесторы, готовы вкладывать в развитие компании, если видят ее стабильность и устойчивость в долгосрочной перспективе.

4.3 Определение элементов программы КСО

Следующим этапом разработки программы корпоративной социальной ответственности бизнеса является определение элементов программы КСО. Это будет зависеть от множества факторов, таких как:

1. сфера деятельности компании;
2. финансовые возможности;
3. размер компании;
4. приверженность сотрудников компании;
5. сотрудничество с местными органами самоуправления и местными экологическими организациями;
6. ожидаемые результаты реализации программ т.д.

Для того, чтобы определить необходимый перечень мероприятий, необходимо сопоставить главных стейкхолдеров компании, их интересы,

мероприятия, которые затрагивают стейкхолдеров. Определение элементов программы КСО представлено в таблице 14.

Таблица 14 – Определение элементов программы КСО

№	Стейкхолдеры	Описание элемента	Ожидаемый результат
1	Собственник	Организация оплачиваемых стажировок и практик для студентов	Привлечение специалистов, повышение узнаваемости компании в ВУЗах
2	Сотрудники, собственник	Организация посещения митапов, лекций и семинаров	Повышение квалификации сотрудников
3	Сотрудники	Поддержание стабильной заработной платы	Повышение лояльности к компании сотрудникам, привлечение новых сотрудников
4	Инвесторы, собственник, сотрудники	Участие в конференциях, демонстрации продуктов	Привлечение инвесторов, рост имиджа компании
5	Инвесторы, собственник	Участие в благотворительных пожертвованиях	Привлечение инвесторов, рост имиджа компании

Проведя анализ можно сделать вывод, что основные программы корпоративной социальной ответственности направлены на собственный персонал компании. Соответственно – основным направлением социальной ответственности являются сотрудники.

4.4 Затраты на программы КСО

Затраты на программы КСО в компании ООО «Neo Stack Technology» являются частью ежемесячных или поквартальных отчислений. Затраты на мероприятия КСО представлены на рисунке 15.

Таблица 15 – Затраты на мероприятия КСО

№	Мероприятие	Единица измерения	Цена	Стоимость реализации на планируемый период

Продолжение таблицы 15

1	Организация оплачиваемых стажировок и практик для студентов	Руб.	100 000	400 000
2	Организация посещения митапов, лекций и семинаров	Руб.	30 000	60 000
3	Участие в конференциях, демонстрации продуктов	Руб.	20 000	40 000
4	Участие в благотворительных пожертвованиях	Руб.	200 000	400 000

При организации оплачиваемых стажировок и практик оплата в месяц может составлять 10 000 рублей, если набирать примерно по 10 человек и делать стажировки периодически 4 раза в год, то за год затраты на организацию стажировок или практик составят 400 000 рублей.

Проводятся различные митапы, лекции и крупные семинары для специалистов IT-сферы, вход на такие мероприятия слушателем платный и в среднем составляет 1 000 рублей, если компания будет оплачивать вход на такие мероприятия 30 своим сотрудникам, то общая стоимость за одно мероприятие составит 30 000 рублей. В год на таких мероприятиях можно участвовать два раза, итого 60 000 рублей в год. Если же участвовать как выступающим или организатором таких мероприятий, то участие компании стоит около 20 000 рублей, что в год за два посещения выйдет 40 000 рублей.

Также компания может участвовать в благотворительных акциях и пожертвованиях выделяя в год по 400 000 рублей. Это может быть, как в денежной форме, так и в продуктовой.

4.5 Ожидаемая эффективность программ КСО

Оценка эффективности, разработанной студентом программы КСО, должна строиться на основе принципов эффективности затрат на мероприятия и ожидаемых от мероприятий результатов. В таблице 16 представлена оценка эффективности мероприятий КСО.

Таблица 16 – Затраты на мероприятия КСО

№	Мероприятие	Затраты	Эффект для компании	Эффект для общества
1	Организация оплачиваемых стажировок и практик для студентов	400 000	Повышение интереса студентов ВУЗов к сотрудничеству, повышение узнаваемости компании	Возможность студентов попробовать себя в работе над реальными проектами и получение трудоустройства после прохождения стажировки или практики
2	Организация посещения митапов, лекций и семинаров	60 000	Увеличение профессиональных навыков сотрудников, повышение их конкурентоспособности	Получение опыта и знаний в профессиональной области
3	Участие в конференциях, демонстрации продуктов	40 000	Повышение узнаваемости компании, рост имиджа и привлечение новых инвесторов	Получение новой информации в профессиональной области
4	Участие в благотворительных пожертвованиях	400 000	Привлечение новых инвесторов, повышение имиджа компании	Помощь обществу, решение социальных проблем

Полученные мероприятия позволят компании укрепить свои позиции на рынке в качестве одной из ведущих компании в области разработки геоинформационных систем с помощью участия в различных конференциях и демонстрация своих продуктов. Позволят создать среду для развития сотрудников и повышения их конкурентоспособности на рынке. Организация стажировок и практик привлечет новых специалистов и повысит узнаваемость компании. А также участие в благотворительных акциях и конференциях позволит привлечь новых инвесторов.

Заключение

В ходе выполнение выпускной квалификационной работы была выявлена проблема, связанная с процессом жизненного цикла тестирования. Основная суть проблемы в том, что большинству компаний необходимо сократить время на выпуск реализуемой продукции при это не потерять качество выпускаемого продукта и повысить лояльность клиентов и заказчиков. Исследование данной проблемы было проведено на примере бизнес-процессов жизненного цикла тестирования в отделе тестирования или отделе контроля качества компании ООО «Neo Stack Technology».

Исследование бизнес-процессов отдела тестирования в компании было проведено с помощью изучения регламентов, которые составляются непосредственно сотрудниками отдела и мониторингом работы сотрудников отдела. Мониторинг работы сотрудников отдела проводился с помощью проведения интервьюирования в котором приняли участие 6 сотрудников отдела: руководитель отдела, два старших специалиста и три специалиста. Список составленных для интервьюирования вопросов был разделен на блоки, которые коррелируются с процессом жизненного цикла тестирования программного обеспечения. После исследования процессов работы отдела было проведено моделирование бизнес-процессов. Моделирование позволяет визуализировать бизнес-процессы для дальнейшего их анализа.

Анализ смоделированных бизнес процессов проходил с помощью метода имитационного моделирования, которое заменяет изучаемую систему моделью с целью изучения ее поведения. Для имитационного моделирования были настроены такие параметры модели, как ресурсы и стоимость работы ресурса в час. Проведение имитационного моделирования позволило выявить проблемные места рассматриваемых в работе бизнес-процессов. Благодаря чему был составлен план мероприятий по оптимизации бизнес-процессов работы отдела тестирования с целью сокращения времени и стоимости работы так, чтобы не потерялось качество и все работы, выполняемые отделом, были сохранены в бизнес-процессах.

Основные мероприятия по оптимизации – это внедрение системы управления тестированием Xray, которая позволяет в одном месте хранить и составлять тестовую документацию, а также выполнять тестирование программных продуктов. Помимо этого, внедрение Xray позволило автоматизировать работу составления документации, такой как отчеты о тестировании, ПМИ и Release Notes, что позволило сократить время на ее написание. Также были автоматизированы работы по сбору информации для проведения аудита и составлены шаблоны для заведения баг-репортов с помощью имеющихся у компании программных продуктов таких как Confluence и Jira. С учетом вышеперечисленных мероприятий и анализа были перестроены некоторые бизнес-процессы работы отдела тестирования.

Для того, чтобы понять эффективность от внедрения разработанных мероприятий по оптимизации бизнес-процессов отдела тестирования была рассчитана стоимость работы отдела до внедрения оптимизационных мероприятий. Помимо этого, также рассчитана стоимость внедрения оптимизационных мероприятий. После разработки и внедрения разработанных мероприятий рассчиталась стоимость оптимизированных бизнес-процессов отдела тестирования. Выгода от внедрения оптимизационных мероприятий составила 26 109 рублей, что меньше затрат на разработку мероприятий по оптимизации. Учитывая, что мероприятия выполняются единожды, а проектов у компании больше одного, то уже на третьем проекте будет видна выгода в стоимостном отношении от внедрения мероприятий по оптимизации. Данные результаты можно считать успешными так, как компания имеет более 8 собственных проектов, работа над которыми идет постоянно, а значит рассмотренные бизнес-процессы выполняются постоянно.

Разработанные мероприятия и проведенное исследование в рамках выпускной квалификационной работы можно использовать не только в рамках рассматриваемой компании, но и для сторонних компаний.

Список используемых источников

1. ПерфомансЛаб: официальный сайт. – Москва, 2008-2023. – URL: https://www.performance-lab.ru/rqr (дата обращения: 10.09.2022). Текст: электронный.

2. Шакирова А.И., Хасьянов А.Ф., Даутов Э.Ф. Сокращение времени тестирования программного обеспечения // Современные наукоемкие технологии. – 2019. – № 7. – С. 104-109.

3. Hackr.io: сайт. – 2022. – URL: https://hackr.io/blog/what-is-software-testing-life-cycle (дата обращения: 23.04.2023). Текст: электронный.

4. Intellect.icu Искусственный разум: сайт. – 2021. – URL: https://intellect.icu/tipy-i-vidy-testirovaniya-urovni-testirovaniya-metody-testirovaniya-5187 (дата обращения: 03.05.2023). Текст: электронный.

5. Gitbook: сайт. – 2023. – URL: https://sergeygavaga.gitbooks.io/kurs-lektsii-testirovanie-programnogo-obespecheni/content/lektsiya-2-ch4-vidi-i-napravleniya-testirovaniya.html (дата обращения: 20.05.2023). Текст: электронный.

6. MadDevs: официальный сайт. – 2022. – URL: https://maddevs.io/blog/test-documentation-in-software-testing/ (дата обращения: 20.05.2023). Текст: электронный.

7. Testengineer.ru: сайт. – 2020-2023. – URL: https://testengineer.ru/chto-takoe-testovaya-dokumentaciya-i-zachem-ona-nuzhna (дата обращения: 21.05.2023). Текст: электронный.

8. Neo Stack Technology: сайт. – Томск. – URL: https://www.neostk.com/products (дата обращения: 01.06.2023). Текст: электронный.

9. Хабр: сайт. – 2015. – URL: https://habr.com/ru/companies/trinion/articles/273017/ (дата обращения: 14.02.2023). Текст: электронный.

10. QA Lead: сайт. – 2023. – URL: https://theqalead.com/tools/test-management-tools-for-jira/ (дата обращения: 01.04.2023). Текст: электронный.

Приложение А

(обязательное)

Раздел ВКР выполненный на иностранном языке

Theoretical aspects of software testing

Обучающийся:

Группа	ФИО	Подпись	Дата
3НМ15	Смолякова Кристина Викторовна		

Консультант ШИП (руководитель ВКР)

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ШИП	Хаперская А.В.	к. пед. н.		

Консультант – лингвист ШБИП ОИЯ

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИЯ	Аверкиева Л.Г.	к. пед. н.		

1.1 Problem definition

According to the analytical report on the Russian software testing market from PerfomanceLab for 2020-2021 [1], 80% of respondents believe that the main goal of the testing department is to improve the quality of IT products and 69% are aimed at increasing user satisfaction. At the same time, the goal of reducing the time of bringing products to market ranks third and interests 62% of respondents, which is 26% higher than in 2018.

A large number of companies are trying to reduce the output of the product while not losing quality and increasing user satisfaction. According to the survey results, it can be said that such a need for companies is only increasing every year. It is the optimization of testing processes that can solve the problem of rapid product delivery without losing its quality.

In the article by A.I. Shakirova [2], which is devoted to reducing the time of software testing, a method for improving the effectiveness of software testing is considered, which allows to reduce the testing time without loss of quality. For this method, surveys were conducted among 17 teams that are engaged in testing in order to find out how the testing process is built in these teams, namely what type of testing is used and how many people participate in testing. As a result of the research, the author of the article has developed recommendations that make it possible to increase the effectiveness of each type of testing considered.

In this paper, an algorithm for optimizing the testing process is proposed and described taking into account experience, methods for evaluating efficiency and calculating the economy for costs in the testing process, described in the article by A.I. Shakirova.

1.2 Software Testing Lifecycle

Testing is an integral part of modern software development. Although there are so many approaches to software development, testing is necessary everywhere.

Testing has its own lifecycle which is called the software testing lifecycle or STLC. The software testing lifecycle helps in conducting the testing process in an appropriate and thorough manner.

Software testing, like software development, involves several steps following in a certain sequence. This is called the software testing lifecycle. It defines the beginning, end and intermediate points of the complete software testing process.

Each stage of the STLC has specific results, goals and objectives. Although different testers may approach the steps differently such as combining one or more into one, repeating, etc., the basic idea remains the same. The following are the main phases of the software testing lifecycle that are used regardless of the chosen approach [3]:

- requirements analysis;
- test planning;
- test development;
- performing testing;
- preparation of documentation.

Each of these stages is somehow mandatory for the testing lifecycle, regardless of how formalized and documented it is. Below, each stage is considered in more detail.

Requirements analysis.

Requirements analysis is an activity for researching new changes, analyzing documentation for them, performing static testing of requirements if necessary. Most often, this stage begins after the completion of the coordination of technical requirements and specifications by the team analyst. Requirements can be described in an explicit formalized form, or they can be presented in the form of a user story. At this stage, the tester has the opportunity to influence the implementation of future changes, taking into account his vision of the system and the process as a whole from the point of view of testing.

At this stage, the tester gets an understanding of the changes that are planned for development and implementation. The result is an actually generated checklist

of test requirements which the tester will have to check. It can be either a formalized document or just knowledge in the head of a particular testing specialist.

Test planning.

At the planning stage, the tester determines the testing goals, the necessary types of testing, the requirements for the test environment (test environment), determines the criteria for the start and completion of each type of testing, evaluates the preliminary testing deadlines based on the test requirements received earlier, identifies potential risks and, together with other team members, works out options for minimizing them, as well as any other specific testing requirements that may arise as a result of running tests.

The result of this stage can be a formalized test plan that describes all the points listed above. This document is most often agreed upon additionally by the tester with all interested project participants. It happens that such a document is not formalized, and all aspects are discussed within the team at planning meetings and recorded in the minutes of these meetings. This significantly reduces the testing planning process, as it is possible to immediately discuss open questions with all team members and find a suitable solution. The result in this case will only be an assessment of testing activities.

After completing the planning stage, the tester moves on to one of the key stages of the testing process – the development of tests.

Development of tests.

After a well-developed test plan, the tester starts developing test cases and filling them. This stage can take quite a long time, especially if we are talking about a large project. The tester first of all draws up a checklist, forms the structure of test cases and then proceeds to fill them with steps, various conditions, as well as determining test data for their successful implementation. In addition, at this stage, test cases can be determined, which will be subject to further automation. There is an approach when a tester creates for himself a checklist with test data and conditions but does not proceed to writing full-fledged test cases.

The result of this stage is a detailed understanding of what will need to be tested. These can be full-fledged test cases, checklists or a vision of what needs to be tested.

In addition, it is necessary to prepare for testing. Preparation for testing has the following activities:

- creating a test environment with the necessary configuration;
- the latest version of the application with the necessary changes for testing;
- preparation of test data for testing;
- revision of test automation scripts.

As a result, the tester receives a workable test environment with test data and an installed testing application.

Performing testing.

This stage involves performing several types of testing sequentially until a positive conclusion is received and the tests are successfully passed. Integration or even system testing is performed first, depending on the specifics of testing in a particular company. After the system testing is completed, regression testing is performed. Most often, due to the automation of testing, the tester performs only integration and system testing, after which automated regression tests are fully or partially run. Acceptance testing is performed either on the customer's side or at a specially designated meeting, which is often called a "Demo". During this meeting, the tester shows the implementation of the system functionality, and the customer accepts these changes, or sends them for re-revision.

The results of this stage can be considered successfully passed test cases or points on the checklist.

Preparation of documentation.

The last final stage of testing before introducing new changes to productive environments is reporting on test results. This stage involves the creation of formal reports on the results of testing changes, which may include general test results, open

problems and defects, as well as recommendations for support for installing changes to a productive environment.

At this point, in most cases, the testing process is completed and the change is transferred to end users or customers for installation on a productive environment.

1.3 Types of software testing

Types of testing are usually classified according to different criteria [4]:

– code availability;

– testing object;

– the positivity of the scenarios;

– testing purposes;

– degree of automation;

– code execution;

– level of testing;

– relation to changes;

– test performers; etc.

In different sources, the classification may differ in features, but the general essence of the types of testing remains the same. In this paper, the types of testing level, code execution and related changes are considered.

Code execution testing is divided into static and dynamic [5].

Statistical testing is a type of testing that assumes that the program code will not be executed during testing. Static testing begins at the early stages of the software lifecycle and is, accordingly, part of the requirements analysis process. Most static techniques can be used to "test" any form of documentation, including code proofreading, inspection of project documentation, functional specifications and requirements.

Dynamic testing is a type of testing that involves running program code. Thus, the behavior of the program during its operation is analyzed. To perform dynamic testing, it is necessary for the program code under test to be written,

compiled and run. In addition, dynamic testing can include different testing sublevels.

Testing at different levels is carried out throughout the entire life cycle of software development and maintenance. The level of testing determines what tests are performed: over an individual module, a group of modules, or a system as a whole. There are four main levels of testing: component or modular, integration, system and acceptance [5].

Component or unit testing checks functionality and looks for defects in parts of the application that are available and can be tested separately, for example, program modules, objects, classes, functions, etc. Component testing is usually performed by calling the code that needs to be tested and using development environments. All defects found are usually corrected in the code without a formal description of them.

Integration testing is designed to test the communication between components as well as interaction with different parts of the system, or communication between different systems. Integration testing, in turn, is divided into levels:

– the component integration level checks the interaction between the components of the system after component testing;

– the system integration level checks the interaction between different systems after system testing.

The main task of system testing is to check both functional and non-functional requirements of the entire system as a whole. System testing reveals incorrect use of system resources, incompatibility with the environment, unforeseen usage scenarios, missing or incorrect functionality, inconvenience of use, etc. To minimize the risks associated with the behavior of the system in a particular environment during testing, it is recommended to use a test environment as close as possible to the one in which the product will be installed after checking.

There are two approaches to system testing:

– based on requirements, when test cases are written for each requirement, verifying the fulfillment of this requirement;

– based on use cases, when test cases are created based on the idea of how to use the product to test each scenario.

Acceptance testing is a formal testing process that verifies the compliance of the system with the requirements and is carried out in order to:

– determine whether the system meets the acceptance criteria;

– make a decision by the customer or other authorized person on the acceptance of the application.

Acceptance testing is performed based on a set of typical test scenarios developed according to the requirements for this application. The acceptance testing phase lasts until the customer makes a decision to issue the application or send it for revision.

Basically, testers participate in integration, system and acceptance testing. Each level of testing is used for types of testing related to changes.

After making the necessary changes, such as fixing defects, the software should be tested to confirm that the problem has actually been solved. The following are the types of testing that must be carried out after installing the software to confirm the operability of the application or the right defect correction [5].

Smoke testing came from an engineering environment: "When commissioning new equipment, it was considered that testing was successful if no smoke came out of the installation."

In the software field, smoke testing is considered as a short test cycle performed to confirm that after assembling the code (new or corrected), the installed application starts and performs the main functions.

The conclusion about the operability of the main functions is made based on the results of surface testing of the most important application modules for the possibility of performing the required tasks and the presence of critical and blocking defects. In the absence of such defects, the smoke testing is declared passed and the

application is submitted for a full testing cycle, otherwise, the smoke testing is declared failed and the application goes for revision.

Regression testing is a type of testing aimed at verifying changes made in an application or environment, for example, fixing a defect, merging code, migrating to another operating system or database, to confirm that the previously existing functionality works as before.

As a rule, regression testing uses test cases written at the early stages of development and testing. This ensures that changes in the new version of the application have not damaged the existing functionality. By itself, the term regression testing depending on the context of use may have different meanings:

– bug regression – an attempt to prove that a fixed bug is not actually fixed;

– regression of old bugs – an attempt to prove that a recent change in the code or data broke the correction of old bugs, i.e. old bugs began to be reproduced again;

– side effect regression – an attempt to prove that a recent code or data change has broken other parts of the application being developed.

1.4 Types of test documentation

Test documentation is a type of documentation that describes the process, goals, and results of software testing [6]. It may also contain information about the environment, settings, and configuration required to perform testing. Test documentation is used to bring the details of a test plan or strategy to the attention of stakeholders, developers and testers.

Some companies claim that test documentation is an optional process that only increases development costs and works more for the image of the software developer company than for providing real value. But the truth is that test documentation is an integral part of the testing process, which should precede, accompany and complete any business testing process.

The main objectives of the test documentation are:

– management of future testing work and demonstration of test results to interested parties so that they can make decisions regarding software development work;

– prevent gaps and problems in testing processes or duplication in testing to ensure that all necessary tests are completed.

The test documentation allows you to make testing transparent as the it shows all aspects of testing from setting goals to testing methods and results. This gives the customer a clear idea of what work has been done and what results have been obtained. It is also useful for other testers to whom the task is transferred, and for developers who have to solve detected problems.

Documentation also improves communication in the team, because the number of communication errors is significantly reduced if the team has a single source of information about the functionality of the products or even about the process of working on the project. There are situations when everyone understands the situation differently and, in this case, there is something to refer to and discuss. There is also no need to make unnecessary calls to get the necessary information because it is constantly updated, and everyone knows where to look for its current version. And, of course, it helps a lot when new people join the team or when the testing team changes as a whole, which may not be possible without high-quality documentation.

With proper documentation of testing, the risk of unexpected problems is significantly reduced. Well-written test documentation reflects real development experience on the basis of which you can do a lot of valuable things and use best practices in other projects. The documentation can be used to train new team members, find and share best practices, conduct reasoned discussions, and much more.

A good test documentation is an excellent demonstration of how fundamental testing is carried out, what approaches are used and how various parts of a software product are tested.

There are many text documents, and they can meet a wide variety of standards. The most common type of test documentation is a test plan. The test plan describes the approach that will be used to test a specific software system. It includes information about what will be tested, how it will be tested and who will be responsible for conducting the tests.

Other types of test documentation include test cases, test scenarios, and defect reports. Test cases describe specific steps that need to be taken to test a specific function. Test scenarios are more detailed instructions that describe exactly how the test should be performed. Defect reports document any errors found during testing.

The types of test documentation depend on the specific company, product and customer. Next, the most common types of test documentation are discussed in more detail.

Internal test documents are required to complete work tasks and for use by members of the development and testing team. It contains the goals, methods and results of testing at the technical level.

A testing strategy is a document that outlines the main aspects of testing, in particular, at which levels of the project it is performed. During testing, developers, designers and managers can refer to this document at any time make sure that testing is still carried out within the framework of the initial strategy and the allocated area.

A test plan is a document that describes in detail all the main aspects of a testing project such as scope, approach, resources, schedule, test participants and their activities. Since this is the simplest document, it is used most often, distributed to the entire team and brought to the attention of customers.

A checklist is a document containing a brief description of the functions that the tester should check. The checklist looks like a list of functions indicating the result of the check. Checklists can be used instead of test cases because they are easier to prepare. But if you need a more specific description of the procedure, you cannot do without test cases.

A test case is a document that contains a detailed description of the steps and actions that a tester must perform to test one part of the functionality as well as the criteria for passing the tests. Companies can use different formats of test cases but the information in them is always very detailed and specific [7].

Test data is a document that describes the data needed for testing in order to bring the performance of an application closer to real use. For example, it can be sets of generated users, documents, media files, metadata, and everything else that the application needs to work with in the real world.

External text documents are necessary for visual representation of the final results since the use of various types of visualizations facilitates their understanding. They are often provided by the customer and can be provided to users in one form or another.

Error reports are a document that describes a specific error in an application in all its possible aspects. This is a very important and frequently published report that directly helps to improve the product in specific aspects.

A test report is a document that summarizes the results of a test cycle. It may contain the cost of detecting defects, the effectiveness of the test suite, the effectiveness of testing, the ratio of efforts to finalize and verify, etc.

The user acceptance report is a document containing the results of testing before being sent to the customer for compliance with the requirements. This ensures that the views of developers and customers remain identical during development and testing and that the technical implementation fully corresponds to them.

In order for the documentation to work positively for the team and the project and bring benefits, the following rules must be observed.

Keeping the documentation up to date. A common mistake is to stop updating documentation, even if from the very beginning you made all the documentation very well. And it seems insignificant but only until the time when the documentation is needed by the project or team.

Keep all documentation in one place. Documentation should not be scattered in different places in an attempt to keep it close to the corresponding code base. This

may seem like a good idea but it can confuse team members and create confusion in a practical sense.

Keep the documentation in a safe place. Do not post any documentation in the public domain that may negatively affect business goals. In this case, you may need encrypted storage and the ability to grant access to certain people.

Use standard templates for documentation. It is best to use universal documentation solutions such as Word and Excel. This will allow any interested person to easily open and read them without installing additional software or other difficulties.

Make the documentation detailed. Documentation should be well structured, informative, detailed and clear. All participants in the process using the documentation should understand it equally.

The set of test documentation may vary from project to project. But there are immutable text documents the quality of which gives the company, its employees and customers many advantages.

Приложение Б Ответы специалистов на интервью

Таблица Б.1 – Ответы специалистов на интервью

Вопрос	Ответы
Какую должность вы занимаете?	Руководитель отдела – 1. Старший специалист – 2. Специалист – 3.
Сколько времени вы работаете в данном отделе?	– 1 год. – 2.5 года. – 1.5 года. – 6 месяцев. – 6 месяцев. – 1 месяц.
Блок. Подготовительный этап (аудит проекта)	
Сколько времени в среднем у вас занимает аудит одного проекта?	– 3 часа. – Не делала ни разу еще. – 2 -2,5 часа. – Не проводила еще аудит. – Не проводил аудит. – 3-4 ч.
Как часто вы проводите аудит на своем проекте?	– Пока только при старте \ входе в проект. Цель - 1 раз в месяц. – В теории надо контрольные точки проводить ~ раз в месяц. – Пока только один раз проводила. – Не проводила еще аудит. – Не проводили. – Пока что только один раз.

Продолжение таблицы Б.1

Какие достоинства и недостатки вы видите в процессе проведения аудита?	– Достоинства: Показать команде где и что упущено чтобы считать, что качество разработки проекта поддерживается на уровне принятых регламентов. В идеале - свести все показатели к какой-то одной оценке (например, 6/10 баллов). Недостатки: Неправильное понимание аудита проекта может разнести слухи что проект ужасный и с него надо убегать. – Недостатки - кучу потраченного времени. Достоинства - наглядная динамика развития проекта, подсвечивание критичных моментов. – То, что на это не выделяется отдельная задача. – Достоинства - удобно, можно открыть одну страницу, где будет все прописано о проекте, недостатки - думаю, что составление аудита займет много времени. – Достоинства в том, что прослеживается состояние проекта, можно "подсветить" проблемы на проекте. Минусы в том, что на это нужно выделять время (и думаю не маленькое). – Из достоинств - в целом, позволяет оценить состояние проекта, подсветить проблемные места.
Считаете ли вы возможным оптимизацию и автоматизацию данного процесса?	– Да, как минимум Jira запросы: для отслеживания состояния ошибок и задача в проекте, для отслеживания хода освоения ТРЗ по проекту + автоматизация для health check состояния базы знаний по проекту. – 100%. – Возможно по части экономики, в остальном лучше вручную. – Думаю, оптимизация возможна.
После проведения аудита отслеживаете ли вы устранение замечаний или это будет заметно только после проведения следующего аудита? Считаете ли вы необходимым отслеживать устранение замечаний?	– Сейчас ставка на актуализацию состояния только после проведения следующего аудита. Если получится автоматизировать проверку исправлений - это существенно снизит трудозатраты на повторный аудит. – Обязательно необходимо отслеживать устранение замечаний. – Да, отслеживание необходимо, иначе суть блока с замечаниями теряется.

Продолжение таблицы Б.1

Выскажите свое мнение о данном процессе. Хотелось бы что-то улучшить/изменить помимо вышесказанного?	– У нас отсутствует раздел проверки качества аналитики. Даже пока нет понимания о метриках для отслеживания качества, кроме как субъективной оценки от команды. – Стоит наконец-то его внедрить. Если процесс будет автоматизирован - будет прекрасно. – Аудит как входной процесс в проект специалиста ОКК полезен - позволяет увидеть картину проекта на текущем этапе и помочь указать РП чего на проекте не хватает. – Пока что мне не понятно, почему аудит проекта должен проводить qa, особенно jun.
Сколько в среднем времени у вас уходит на составление тест-плана (с нуля)?	– 1-2 часа. – 3-4 часа. – 30 минут. – 1-2 часа. – Зависит от того , что входит в тест-план. Если брать все запланированные этапы с начала проекта и до конца проекта (как по новому регламенту, где включаются даже ссылки на задачи в рамках, которого делают) , то примерно в районе часа.
Возникают ли сложности с оценкой работ по тестированию?	– Да. Сейчас есть сложности при оценке нового проекта: 1) Нехватка проработанности требований на старте. 2) Нет привлечения спецов ОКК на валидацию оценок (используется только метод 20% доли от разработки). 3) Не учитывается увеличение стоимости работ на старте (тестирование ФС и написание ТК в Xray, которые в последствии дают выигрыш в ходе тестирования). – Еще не было работ по оценкам задач по тестированию. – Да, сложность в том, что оценка = % от разработки. – Когда была джуном возникали, сейчас уже нет.
Блок. Этап анализа требований	

Продолжение таблицы Б.1

Какова цель хранения и создания карты переходов на проекте?	– Я ставлю цель чтобы с помощью построения mind map специалист быстро начинал ориентироваться на проекте - это ускоряет общее погружение спеца в проект. – Помогает изучить тестируемый продукт и его функции, выявить различные взаимосвязи. Поможет новому сотруднику быстрее влиться в проект. – Сверка реализованного проекта с планом реализации. В случаях масштабных проектов можно посмотреть, как добраться до той или иной функциональности. + в редких случаях использовать для тестирования, минуя ТК (в редких, потому что считаю, что это неэффективно и лучше так не делать, имеется провальный опыт). – Считаю, что составлять такую карту нужно в том случае, когда нет четкой аналитики. Я составляла карту такую на стажировке для 2х проектов, и с помощью нее нашла очень много ошибок, т.к. для ее составления нужно протыкать весь проект. – В первом приближении познакомиться с проектом, определить его "слабые зоны", найти ошибки, которые находятся "на поверхности". – На этапе вхождения в проект удобно тем, что ты пристраиваешь связи между всеми элементами системы. Карта содержит в себе не только все элементы, но и наглядно показывает взаимосвязи между ними. Благодаря этому, в случае появления бага в одной части программы, мы не пропускаем все связанные с этой частью места и элементы и проверяем все связанные функциональности. Так же можно указать на карте приоритеты ,чтобы в дальнейшем в первую очередь обращать внимание на них.
Сколько времени в среднем у вас занимает составление карты переходов проекта?	– 2-4 часа в зависимости от сложности и запутанности проекта. – В зависимости от сложности проекта. В среднем 3-5 ч. – Смотря какой проект. Я составляла свои карты не больше 8ч. – в среднем 2-2,5 часа.

Продолжение таблицы Б.1

Расскажите про ваш опыт использования карты переходов? Часто ли приходится обращаться к карте переходов?	– Да, при сжатых сроках сдачи проекта mind map хорошо себя зарекомендовала как индикатор состояния проекта. Так же mind map была тепло встречена руководством при демонстрации результатов этапа проекта в качестве ориентира какой сценарий будет показан. – Составляла при погружении в новый проект, возвращалась к неё для редактирования некоторых пунктов. Если к проекту будет подключаться новый специалист, то ему эта карта может помочь на этапе погружения в проект.
Сколько времени в среднем занимает тестирование одной ФС без учета написания чек-листа?	– От 30 минут до 1 часа (более сложных ФС пока не попадалось). – Зависит от сложности фичи, в среднем ~ мин 40 - час. – 1 час. – 30 мин - 1 час. – 2-3 ч.
Как долго вы ожидаете уточнения требований от аналитика (ответы на ваши вопросы)? Если задержка присутствует, ответить на вопрос: Как вы считаете по каким причинам происходит задержка уточнения требований?	– От 30 минут до 1 часа (более сложных ФС пока не попадалось). – Зависит от сложности фичи, в среднем ~ мин 40 - час. – 1 час. – 30 мин - 1 час. – 2-3 ч.
Расскажите, как происходит ваше взаимодействие с аналитиком?	– Сначала я отмечаю вопрос который надо уточнить в ФС. Затем если рядом (сегодня-завтра) есть митинг - дожидаюсь митинга, чтобы согласовать время на разбор вопросов или если позволяет время - там же и закрыть вопросы. Если митинга нет - согласовываю время для совместного их обсуждения с аналитиком в ЛС. – В чате или в ответах на страницах фич. – Вопросы задаются либо в комментариях в фиче, либо в блоке вопросов в фиче, либо вообще лично подойдя к аналитику. – При тестировании аналитики, я оставляю вопросы в комментариях, аналитик на них отвечает по мере возможности. Если что-то срочное, то пишу в личные сообщения.

Продолжение таблицы Б.1

Сколько в среднем итераций вопросов-уточнений у вас приходится на одну ФС? Если больше одной итерации, ответить на вопрос: Как вы считаете, почему не получается разрешить все вопросы за одну итерацию?	– 2-3, потому что аналитик зачастую не погружается в ответы/отвечает поверхностно, либо отвечает избирательно на вопросы. – В основном обхожусь одной, но иногда больше так как не все влияния на систему например указаны и потом приходится возвращаться к фиче снова и задавать вопросы. – Сложно пока сказать, но точно больше одной. Не всегда у аналитика есть корректное понимание, как должна работать система, приходится обсуждать совместно с разработчиком.
Считаете ли вы необходимым этап написания чек-листа при тестировании аналитики? Если да, то почему?	– Да, считаю важным. Т.к. в ходе обсуждения с директором этой темы, обнаружилась потребность отслеживать не только связку: Требование ТЗ - ФС, но и чек лист - Тест кейс для гарантии что итоговое требование протестировано. – Это удобно, если процесс тестирования аналитики и написания тест-кейсов идет не параллельно + более-менее выделяются требования (наглядно становится). – Зависит от проекта, где-то обходятся только чек-листами без тест-кейсов, что помогает экономить время. Где -то прописывается и чек-лист и тест-кейсы. – Да, потому что когда ты тестируешь аналитику конкретной фичи, ты максимально погружаешься в нее, и составляешь себе много проверок, про которые потом можно забыть. – При работе с ФС писал себе чек-лист, чтобы наметить основные проверки. – Да, помогает выделить основной список проверок.

Продолжение таблицы Б.1

Составленный чек-лист помогает вам в будущих процессах? Пользуетесь ли вы им? Если да, описать в каких случаях и при каких обстоятельствах.	– Да, на моём проекте Healthnet из-за сжатых сроков сдачи этапа проекта пришлось пожертвовать написанием ТК в пользу тестирования стенда на соответствие требованиям из ФС. Чек листы в таком случае были играли роль навигатора при проведении экспресс проверки всей системы. – Только если процесс написания тест-кейсов растянулся (например, аналитик долго отвечал на вопросы или были выходные), то тогда можно пробежаться на чек-листам и тест-кейсам, чтоб удостовериться, не упущено ли что-то. – Да, я по чек-листу потом пишу тест-кейсы. – На основе чек-листа пишу уже ТК.
Видите ли, вы возможность подключения к тестированию ФС раньше? Если да, то каким образом?	– Возможность подключения раньше есть, но при достаточно уникальном стечении обстоятельств: 1) Есть ТЗ по проекту. 2) Есть Аналитик на проекте, который разбивает ТЗ на ФС. 3) У тестировщика этого проекта нет никаких других задач на проектах, в которых уже стадия разработки и где он так же задействован. – Если ФС будут писаться раньше, чем начнется разработка. Если будут свободные ресурсы на это. В нашей компании такое не практикуется. – Здесь необходимо сперва написать какой процесс сейчас есть (так как в текущий момент на разных проектах по разному). Например, сейчас на моём проекте тестировщика подключали как только была написана запущена работа по фичам, то есть аналитик пишет фичу и она переходит на валидацию мидлу отдела аналитики (так как писал джун) и после сразу ко мне . Нет доп валидаций от руководителя аналитиков или продукт оунера и мне не нужно ждать, когда будут написаны 5-10 фич. – В нашей компании - не вижу. Разработка происходит у нас раньше, чем написание ФС.

Продолжение таблицы Б.1

Выскажите свое мнение о существующем процессе. Хотелось бы что-то улучшить/изменить помимо вышесказанного?	– Если процесс написания ФС начался после разработки, то будет здорово, если аналитик сам будет открывать стенд и смотреть, как уже реализовали, а не придумывать своё уже, генерируя ТРЗ таким образом. – Пока меня устраивает данный процесс.
Блок. Этап разработки тестов	
Сколько в среднем времени у вас уходит на написание одного тест-кейса?	– Думаю что 30-45 минут. – 30-45 мин. – Минут 10-15. – 20-30 минут. – 15-20 мин. – 20-30 минут.
Сложно ли вам поддерживать тест-кейсы в актуальном состоянии? Если да, то почему и что мешает?	– Сложно, т.к. у нас матричная система управления ресурсами и пока нет системы управления требованиями, т.е. специалиста могут переключить с проекта на другой проект, а по возвращении быстро не восстановить картину что именно успело поменяться. – Да, т.к. доработки функциональности = новым фичам, которые могут противоречить предыдущей разработке. В итоге +100500 тест-кейсов и часть из них неактуальны уже. – Сложно, чаще всего нет времени, чтобы это поправить. – Нет. – Зависит от проекта иногда выделяют отдельно задачу на актуализацию, иногда нет.
Сколько в среднем времени у вас уходит на актуализацию тест-кейсов? Как часто это приходится делать на вашем проекте?	– Думаю что занимать будет от 20 минут минимум. Частота - 1 раз в 2 месяца точно потребуется. – Я актуализировала, когда получала новую информацию от аналитика и когда сдала ПМИ заказчику, и он нашел там нюансы.
Как вы отслеживаете актуальность написанных тест-кейсов?	– При использовании базы знаний типа Confluence - Только по истории изменения страницы. – По дате обновления фиче-страницы в Confluence, но из-за наложения фич - нужно только знать, иначе сложно сообразить сразу. – С помощью статусов.

Продолжение таблицы Б.1

После написание тест-кейсов занимается ли кто-нибудь их приемкой?	– Сейчас нет. Только в редких случаях. – Должен заниматься Руководитель ОКК. – Нет.
Удобен ли вам текущий формат написание тест-кейсов? Расскажите про его достоинства и недостатки.	– Плюсы метода: скорость написания проверки, гибкость подхода при составлении проверки, простая схема для составления ПМИ через копирование и вставку ТК. Недостатки: не отследить насколько он актуален без вчитывания в аналитику и сам ТК, вставка скриншотов может сделать таблицу нечитабельной, ручное составление и актуализация отчёта о тестировании, нет возможности из нескольких отчётов построить доп статистику ошибок (например, области скопления ошибок). – Недостатки: шаг и результат не удобно отображаются, иногда шаг большой, а результат маленький, а текст идет подряд и становится не очень читабельно.
Видите ли вы возможность использования специализированной программы для написания тест-кейсов?	– Да, безусловно. и в первую очередь для автоматизации отчётности о тестировании перед командой, РП, бизнесом. – Да.
Как вы подготавливается тестовые данные для тестирования? Где они хранятся?	– Тут два варианта: 1) Общие тестовые данные (например, разные файлы для загрузки) - в папке отдела на внутреннем сервере компании. 2) Уникальные тестовые данные для проекта - в пространстве проекта и\или в особом случае в папке отдела на сервере компании (например, если база знаний не поддерживает тяжелые тестовые файлы). – Еще не готовила. Вероятно, буду хранить в КФ, на странице проекта. – В папке на компьютере или в конфлюенс. – На сетевом диске, в папке нашего отдела.
Блок. Этап тестирования и составления документации	

Продолжение таблицы Б.1

Как происходит ваша подготовка к тестированию (выделение тест-кейсов, составление шаблона отчета)?	– Есть этап тестирования аналитики и фиче страницы. Результат тестирования - чек лист проверок для будущих тест кейсов. ТК оформлять в одной странице с группировкой по фичам. Копия этой странице будет служить отчётом о тестировании. – Копируется шаблон отчета о тестировании, вручную вбиваются все фичи, прописываются наименования ТК. – Составляется тест-план, составляется скелет тест-кейсов по выделенным требованиям (могут ещё прописываться чек-листы, но это не всегда), далее прописываются подробно тест-кейсы, делается проверка по тест-кейсам (отмечаем статусом /комментарием какой тест-кейс упал или нет) , заводятся баги, далее проводятся ретесты и собирается отчет по всем проведенным работам.
Сколько в среднем времени у вас занимает подготовка к тестированию?	– От 30 минут до 1-2 часа на одну фичу. – 1-1.5 ч, зависит от объема.
Какие виды тестирования вы обычно проводите?	– Приёмочное, дымовое (обычно под определённый сценарий), регрессионное. – Регрессионное, приемочное. – Дымовое, функциональное, нефункциональное, регрессионное, приемочное.
Как часто проходит тестирование на вашем проекте (сколько итераций, например, регресс и приемочное)?	– Приёмочное после разработки фичи. Дымовое - по согласованию с РП. Регрессионное + ретест ошибок - 1-2 раза за проект. – 1 раз. – В основном две итерации.
Сколько в среднем времени у вас уходит на заведение одного баг-репорта?	– 15 мин. – От 15 минут до 1 часа в зависимости от сложностей с локализацией ошибки. – 30 мин. – 0,25 ч.

Продолжение таблицы Б.1

Баг-репорты у вас заводятся по единому шаблону? Или зависит от специалиста?	– Не сказал бы что прямо по шаблону. Но ключевые моменты обязательно выдерживают (окружение, шаги, ожидаемы и фактический результаты, вложения). – Зависит от специалиста. – У нас в компании все по разному оформляют БР, но шаблон у всех похож. – По единому.
Удобен ли вам текущий формат отчета о тестировании? Расскажите про его достоинства и недостатки.	– Нет. Доп время на актуализацию состояния ошибок. Подсчёт количества ошибок и пройденных/не пройденных кейсов тоже вручную. – Нет, т.к. все вручную требуется вбивать, отслеживать статусы, прикреплять ссылки на баг-репорты. – Долго составлять и это самый большой минус.
Сколько в среднем времени у вас уходит на составление ПМИ?	– От 1 дня. – 40 часов. – 30-50 часов.