МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

09.04.03 Прикладная информатика
(код и наименование направления подготовки)

Управление корпоративными информационными процессами
(направленность (профиль))

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ)

на тему Исследование методов и подходов процесса управления по переходу программного обеспечения телеком-оператора на микросервисную архитектуру

Обучающийся Е.Н. Моисеева
(Инициалы Фамилия) (личная подпись)

Научный руководитель канд. пед. наук, доцент Е.А. Ерофеева
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2023

Содержание

Введение

Сектор телекоммуникаций одна из наиболее эффективных и динамично развивающихся отраслей мировой экономики. Телекоммуникационные компании обязаны задуматься над тем, как управляют своими процессами и провести реинжиниринг по необходимости.

В условиях пандемии в 2020 году проведенный опрос британской компанией YouGov показал, что интернет стал критически важным элементом для мировой экономики. Он позволил почти половине (48 %) опрошенных продолжать работать и учиться, а 40 % опрошенных во время пандемии обращались к интернету за услугами, которыми не пользовались ранее [28].

Также старший директор консалтинговой фирмы Simon-Kucher & Partners Middle East пишет о том, что телекоммуникационные компании по всему миру должны не только предоставлять на высоком уровне работу своих сетей и услуг, но также развивать другие направления бизнеса. Например, развивать услуги, которые не относятся к основным – такие как платное телевидение, медиа, реклама, умный дом, облачные технологии, безопасность, финансовые услуги, товары для жизни и другие решения [75]. По данным GSM Association (GSMA) из отчета Global Mobile Trends 2021 такие услуги составляют 20 % от общего дохода ведущих операторов связи [74].

Для реализации таких идей может понадобиться создание экосистемы с использованием информационных технологий. Как правило, для достижения таких целей инициируются IT проекты [2].

Проектным менеджерам в телекоммуникационной сфере требуются знания и опыт управления проектами, что является применение процессов, методов, навыков и опыта в соответствии с критериями приемки проекта для достижения целей проекта в рамках согласованных переменных. Управление проектом гарантирует, что проект будет реализован в соответствии с поставленными целями и сможет уложится в заложенные сроки его выполнения и объемы бюджетных средств.

Компания «ИТ Гильдия» отмечает, что даже при тщательном планировании могут возникать внештатные проблемы, к решению которых должны быть готовы менеджеры ИТ-проектов [50].

Анализ актуальности обусловили выбор темы исследования: «Исследование методов и подходов процесса управления по миграции программного обеспечения телеком-оператора на микросервисную архитектуру». В данной работе научной проблемой – это отсутствие эффективной методологии разработки программного обеспечения для проектов по миграции на микросервисную архитектуру.

Гипотеза исследования состоит в том, что миграция программного обеспечения на микросервисную архитектуру будет успешной, если использовать модель улучшенной гибридной методологии разработки.

Цель работы: исследовать методы и подходы процесса управления по переходу программного обеспечения телеком-оператора на микросервисную архитектуру и разработать улучшенную модель методологии разработки программного обеспечения. Для достижения поставленной цели необходимо решить следующие задачи:

- провести анализ предметной области;
- провести анализ существующих стандартов и методологий в сфере разработки ПО;
- разработать модель улучшенной гибридной методологии разработки ПО;
- провести апробацию данной модели.

Объект исследования: процесс по миграции программного обеспечения телекоммуникационных компаний на микросервисную архитектуру.

Предмет исследования: гибридная методология разработки программного обеспечения.

Методы исследования перечислены ниже:

– теоретические при анализе источников по методам управления

процессом разработки ПО;

- сравнительного анализа;
- опрос.

Теоретической основой исследования являются источники, перечисленные ниже:

- отечественные и зарубежные исследования по управлению разработкой, особенностям микросервисной архитектуры программного обеспечения;
- публикации на сайтах, посвященные управлению разработкой программного обеспечения и инженерным практикам.

Практическая значимость работы заключается во внедрении новой модели гибридной методологии по разработке программного обеспечения в компанию ООО «НетКрэкер».

Научная новизна работы состоит в создании новой модели улучшенной гибридной методологии разработки программного обеспечения, которая поможет командам проектов миграции ПО на микросервисную архитектуру увеличить свою эффективность и предоставить своевременную выдачу решения заказчику.

На защиту выносятся:

- разработанная модель улучшенной гибридной методологии разработки программного обеспечения и ее описание;
- результаты апробации разработанной модели.

Магистерская диссертация состоит из введения, четырех глав и заключения на 89 страницах, а также состоит из списка 87 использованных источников и трех приложений.

1 Описание предметной области

Специалисты редакции «РБК» рекомендует телекоммуникационным компаниям подумать о взаимодействии с клиентом удаленно, это можно реализовать с помощью программного обеспечения, где хранится информация о всех клиентах [48].

В этой связи телекоммуникационным компаниям по всему миру необходимо подумать об эффективности используемого программного обеспечения. В настоящее время производители программ выбирают микросервисную архитектуру для своих программных продуктов.

Системы поддержки бизнеса и операционной деятельности OSS/BSS (Operations Support System / Business Support System) позволяют управлять бизнес-процессами и сетевым оборудованием. Основой для их создания стал процессный подход, позволяющий проследить и оценить работу всех подразделений компании на всех уровнях».

Некоммерческая организация Tele Management Forum форум разработала модель операций поставщика услуг связи eTOM. Некоторые исследователи отмечают, что: «В ней сформулирована общая для всех операторов терминология и подходы к описанию внутренних процессов. Что касается построения открытой модели бизнес-процессов, то карта eTOM может быть использована в качестве инструмента анализа как существующих в компании бизнес-процессов, так и для разработки новых. Она позволяет выявить процессы, выполняющие одни и те же функции и устранить существующее дублирование, обнаружить недостающие или избыточные шаги бизнес-процессов, ускорить разработку новых процессов. Прозрачная и универсальная структура карты eTOM делает ее удобным и незаменимым средством моделирования внутренних процессов компании, вне зависимости от особенностей ведения бизнеса, оказываемых услуг и применяемых технологий. Процессы, описанные в карте eTOM, служат основой для построения бизнес процессов, существующих или планируемых к внедрению

в компании» [28].

Укрупненная схема структуры eTOM представлена на рисунке 1.

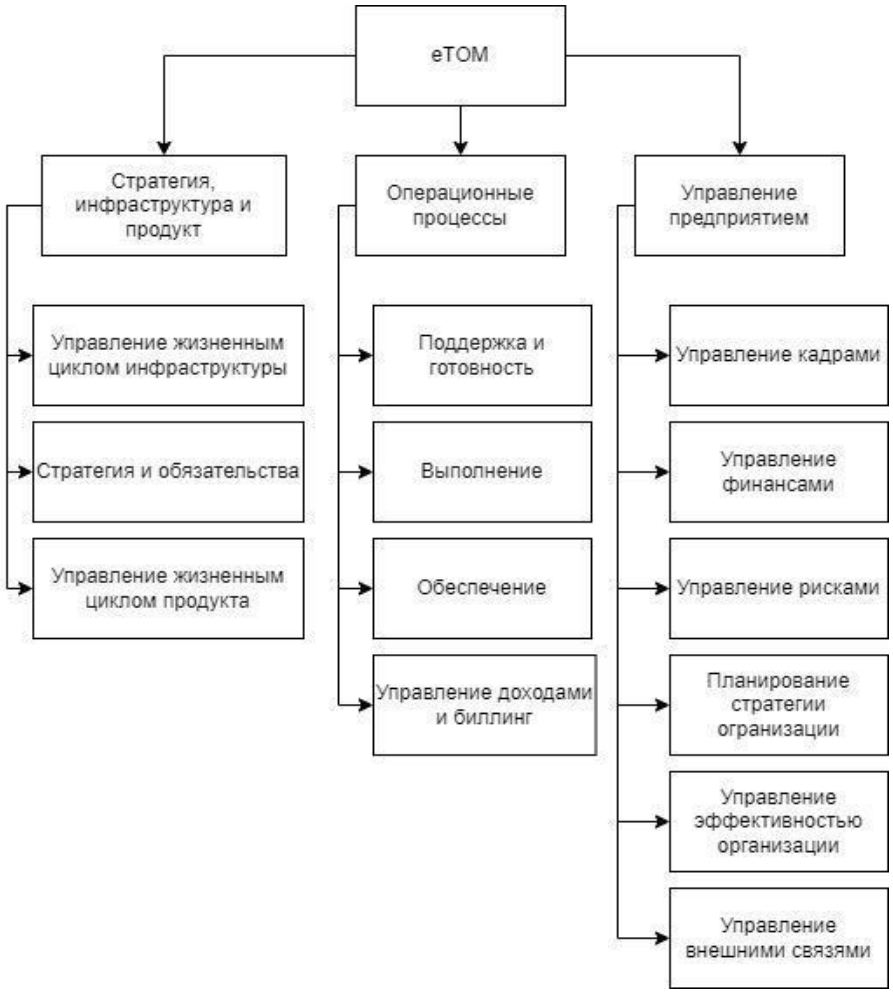

Рисунок 1 – Укрупненная схема структуры eTOM

Зная образец бизнес-процессов телекоммуникационных компаний, можно разработать программное обеспечение для автоматизации этих операций. Можно представить каждую операцию в виде одного модуля. Таким образом, необходимо объединить данные модули, чтобы получилась информационная система, которая поможет автоматизировать их бизнес [29].

Основной целью автоматизации такого бизнеса является:

– минимизация ручных операций на всех этапах деятельности оператора связи (от получения заказа на услугу до ее полного подключения);

– упростить задачу руководства по контролю над сложным большим процессом;
– обеспечить непрерывность процессов.

Существует ряд факторов, определяющих наиболее эффективный способ для конкретного телекоммуникационного оператора: основные цели компании, задачи, которые накладываются на автоматизацию, размер компании (количество ее клиентов), наличие или отсутствие определенных модулей системы и так далее.

Исследователи из университета ИТМО провели анализ бизнеса телекоммуникационных компаний и выявили, что у разных организаций могут быть свои отделы, но в целом они выполняют одинаковые задачи.

Например, взаимодействие с абонентами осуществляет отдел по работе с клиентами, который предоставляет всю информацию об услугах и тарифах, организует подключение новых услуг клиенту и внесение изменений в существующие услуги.

Отдел взаимодействия с оборудованием проводит профилактику сети, прием и обработку заявок на устранение неисправностей, подключением и отключением оборудования.

Ядром информационный системы является модуль – Order management (управление заказами). Этот модуль выполняет следующие основные функции:

– определение основной управляющей бизнес-логики для всех услуг;
– последовательное выполнение задач, определенных бизнес-логикой. Например, когда заказ поступает в модуль, то система определяет его тип и начинает выполнять характерные для него задачи;
– получение и обработка ответов от других модулей/систем.

В результате своей работы модуль получает отчеты от других модулей и систем, с которыми он взаимодействует для выполнения задач по заказу. После обработки этих отчетов модуль обновляет статус заказа и следит за обновлением информации в системе технического контроля.

Компонент системы «Определение технических возможностей» может определить возможность подключения услуги к клиенту благодаря наличию сетевого расположения.

Модуль автоматической активации сетевых элементов и модуль управления действиями технической поддержки осуществляют автоматическую и ручную настройку и активацию оборудования в данном месте.

Модуль ввода заказа осуществляет прямую продажу услуг путем формирования заказа – элемента, с которым система будет работать дальше. Модуль представляет собой визуальное представление, содержащее всю необходимую информацией о продуктах и услугах оператора связи.

В целом, системный процесс состоит из следующих этапов. Клиент инициирует создание заказа. Как правило, клиент может сделать это несколькими способами:

– прийти и обратиться к продавцу в торговой точке;

– позвонить операторам контактного центра;

– воспользоваться сайтом оператора.

В первых двух случаях сотрудники оператора внесут все данные в систему и создадут заказ самостоятельно. В третьем случае помощь персонала не требуется, так как клиент может создать заказ автоматически. Далее модуль ввода заказов (order entering) относит заказ к одному из типов и отправляет его в модуль управления заказами [49].

В зависимости от масштаба компании модулей системы может быть больше. В целом, их последовательное обновление и интеграция – достаточно длительный и дорогостоящий процесс. Для решения аналогичной задачи производителю программного обеспечения необходимо:

– изучить существующие модули систем;

– собрать требования для их доработки;

– решить проблему интеграции новых и старых модулей системы.

В работе исследователей из ИТМО рекомендуется последовательное

обновление существующей системы [49].

Телекоммуникационные компании активно внедряют управляемые услуги (Managed services). Клиенты покупают управляемые услуги, чтобы повысить производительность персонала, преодолев барьер сложности. Согласно контрольным показателям консалтинговой фирмы Arthur D. Little, клиенты (телекоммуникационные компании) готовы платить в пять раз больше за управляемые услуги, чем их неуправляемые услуги [28].

Большая часть программных решений, направленных на решение проблем бизнеса (enterprise решения), построена на многоуровневой архитектуре. Такой подход разделяет программу на несколько уровней «клиент-сервер». В отличие от многослойной архитектуры, такой подход предлагает масштабируемость [1].

По мнению компании OTUS: «концепция монолитной архитектуры обеспечения заключается в том, что различные компоненты приложения объединяются в одну программу на одной платформе. Обычно монолитное приложение состоит из базы данных, клиентского пользовательского интерфейса и серверного приложения. Все части программного обеспечения унифицированы, и все его функции управляются в одном месте» [20].

Компания Microsoft пишет, что основная часть функций монолитного приложения сосредоточены в одном процессе или контейнере [78].

Отмечаются следующие недостатки такой архитектуры:

- компоненты тесно связаны между собой, при внесении изменений в один из компонентов требуется остановка и обновление всего решения целиком;
- по мере наращивания функциональности возникает большое количество взаимосвязей, что увеличивает трудозатраты бизнес аналитиков, разработчиков и специалистов по контролю качества и так далее;
- недостаток в решении может оказывать влияние на функциональность всего продукта и даже приводить к его полному отказу;

- перечень технологий ограничен средствами, которые доступны в среде разработки [18].

В трехзвенной «клиент-сервер» архитектуре присутствует клиентская часть, то есть отображение результатов действий серверной части в рабочей станции, например, в браузере пользователя. Серверная часть представляет собой совокупность программ, которые отвечают за генерирование результатов запросов пользователя. Система управления базами данных (СУБД) – это комплекс языковых и программных средств, нужных для создания, ведения и общего пользования баз данных многими пользователями [11]. Схема такой архитектуры представлена на рисунке 2.

Рисунок 2 – Схема трехуровневой архитектуры «клиент-сервер»

Микросервисная архитектура программного обеспечения представляет собой распределенную систему простых и свободно заменяемых модулей, исполняющих маленькую функцию. При этом сервисы связываются между собой и с клиентами с применением протоколов или текстовых сообщений [4].

Микросервисное приложение складывается из множества небольших самостоятельных и слабо связанных между собой сервисов, в то время как в другом подходе все его компоненты непосредственно взаимосвязаны и работают как единый сервис. Сервисы – это компоненты, выполняемые в

отдельном процессе, общение между ними происходит через веб-запросы [26].

При разработке приложений с монолитной архитектурой обычно используется сервер баз данных, единый для всего приложения. В случае с микросервисами каждый из них может иметь собственный набор технологий и сервер баз данных (необязательно), оптимизированные для конкретного процесса. Схема представлена на рисунке 3.

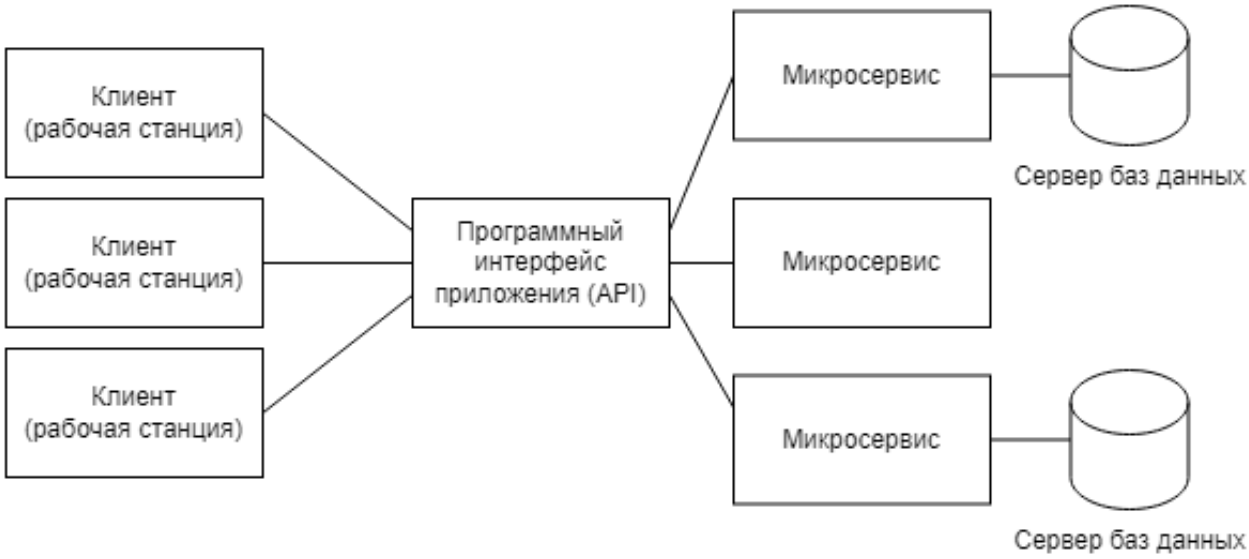

Рисунок 3 – Схема микросервисной архитектуры

Однако есть и некоторые недостатки, такие как: повышение сложности разработки и развертывания, внедрение механизма межсервисного взаимодействия, сложность проведения сквозного тестирования [72].

Также важную роль отводят DevOps, это сочетание разработки и эксплуатации, объединение людей, процессов и технологий, позволяющее постоянно предоставлять решения клиентам. Распределенный доступ к сервисам означает увеличение сетевых задержек и потенциальных сбоев, но можно исправить это путем асинхронности и сокращения числа вызовов [55].

Если говорить о технической стороне вопроса, чаще всего в enterprise решениях используются продукты компании Oracle. Например, это язык программирования Java, система управления базами данных Oracle, сервер

приложений WebLogic. Для больших компаний лицензия на продукты Oracle становится дорогой и более требовательны к ресурсам [35].

В микросервисной архитектуре наоборот разработчики стараются уходить от таких технологий и заменять их на Open Source технологии [33]. «Open Source – это подход к разработке и лицензированию программного обеспечения, при котором каждый желающий имеет право на свободное распространение приложения, свободную модификацию, свободную публикацию собственных улучшений и свободный доступ к его исходным кодам» [16].

Сравнение архитектур по нескольким показателям представлено в таблице 1.

Таблица 1 – Сравнение трехуровневой и микросервисной архитектур

Критерий	Архитектура «клиент-сервер»	Микросервисная архитектура
Громоздкость	Общая большая база данных	Для каждого сервиса может быть своя база данных
Набор технологий	Один набор технологий	Для каждого сервиса может быть свой набор
Наличие Open Source технологий	Как правило, их мало в больших решениях	Да
Обновление компонент	Как правило, приводит к недоступности всего программного обеспечения	Не создает осложнение в работе других частей программного обеспечения
Зависимость компонент	Присутствует	Отсутствует

Таким образом, один микросервис может иметь собственный набор технологий, модель и базу данных, оптимизированные для конкретного процесса.

Так архитектуру программного обеспечения телекоммуникационных компаний можно построить в виде микросервисов, которые выполняют и

декомпозируют процессы, обозначенные в eTOM.

Американская издательская компания O'Reilly в 2020 году провела опрос среди сотрудников ИТ компаний (программисты, системные архитекторы, проектными менеджерами и так далее). В ходе опроса была опубликована следующая статистика:

- около 70 % компаний используют микросервисы в своей работе;
- более половины (55 %) называют свои усилия по переходу на микросервисы в основном успешными;
- около 30 % компаний начали изучать микросервисы [77].

Крупные мировые компании такие, как Netflix, Amazon, Google и другие заявили об успешном внедрении микросервисов. В России также крупные компании поделили опытом, в том числе компания «Мегафон».

Например, в 2019 году проходила дискуссия на тему «От монолита к микросервисам» с участием ведущих российских компаний, где они делились своим опытом и критерии успешного перехода на микросервисы. Руководитель центра исследования и разработки бизнес-систем «Мегафона» отмечает, что они вели планомерную работу по переводу монолитных программ в микросервисы. Также он считает, что: «разработка занимает примерно 40 %, а большую часть времени аналитика, правильные интеграции и правильная архитектура. Особое внимание стоит уделить принципам построения безопасных приложений и тестированию новых решений» [33].

Исследователь из «Белорусского государственного университета информатики и радиоэлектроники» в своей работе пишет о том, что: «Создание высокоэффективных приложений требует эффективных методов управления проектами. Чтобы продукты продавались быстро, группы разработчиков полагаются на эффективные методы планирования и контроля управления проектами для оптимизации своих рабочих процессов» [21].

Таким образом, часть успеха по управлению таким проектом – это выбор правильных инструментов и методов планирования проекта. Для начала нужно определить бизнес-процессы, которые разрабатывая система должна

автоматизировать [29].

Учитывая вышеперечисленные факты, командам проектов по переходу на новую архитектуру необходимо выбрать подходы, которые доведут до успешного завершения проектов.

Исследователи со всего мира описывают технический переход, но почти нигде не говориться об управлении такого рода проектов, потому что они начали появляться не так давно, как классические ИТ проекты.

При планировании ИТ проектов следует учитывать базовые виды работ. Кандидат технических наук Ермаков А. А. и Павлов И.А. провели сравнение методов планирования и выделили следующие виды работ:

- начало реализации проекта;
- разработка технического задания;
- исследование предметной области;
- проектирование классов и основных алгоритмов;
- разработка основных модулей ИС;
- тестирование основных модулей ИС;
- разработка и дизайн интерфейса;
- тестирование и исправление ошибок;
- демонстрация продукта;
- составление программной документации;
- завершение проекта [12].

А также в проекте стоит учесть такие аспекты, как: время, ресурсы и качество. Исследователи Уппсальского университета Абхинав Чандрабабу и Ануша Муддангула провели анализ литературы по видам методологий ИТ проектов. Они выделили следующие методологии: традиционная; гибкая (agile): Scrum; Kanban; гибридная методология [71].

В любом проекте присутствуют риски, которые определены сводом знаний по управлению проектами (PMBOK), как неопределенное явление или условие, которое, если оно происходит, оказывает положительное или отрицательное воздействие на цель проекта [81].

Также PMBOK выпускает рекомендации по управлению рискам, например:

- по разработке критериев оценки рисков, специфичных для продукта;
- проведению сессии по идентификации рисков с использованием методов, доказавших свою успешность в многочисленных коммерческих и военных программах разработки продукции;
- по оценке рисков по критериям оценки рисков, специфичным для продукта/программы, без перегруженности анализом;
- по применению стратегии реагирования на риски и разработка планов реагирования на риски для устранения данного элемента риска;
- мониторингу и контролю процесса выполнения плана реагирования на риски с помощью триггерных событий для минимизации затрат при максимальном увеличении эффекта [70].

Гибкие методологии ориентированы на бизнес-процессы клиента. Присутствует высокая степень вовлечения команды и заказчиков проекта. Команды должны самоорганизовываться, чтобы обеспечивать лучшие результаты для проектов. Клиенты могут увидеть результаты, что снижает некоторые риски в процессе разработки [59].

Например, отмечается, что популярные методологии Scrum и Kanban призваны сделать управление проектами более гибким и адаптивным к изменяющимся условиям. С помощью непрерывной коммуникации между людьми и командами.

Так в гибких методологиях в приоритете бизнес-ценности заказчика, а не только выполнение задач. Проект может успешно выполниться даже при отсутствии полной и точной спецификации требований. С помощью гибких методологий заказчик может участвовать в процессе разработки продукта.

В целом, гибкие методологии управления проектами помогают сделать создание продукта более результативным, инновационным и адаптивным к изменяющимся требованиям клиента.

Однако при таком подходе стимулируются постоянные изменения

проекта и существует сложность итоговой оценки стоимости проекта.

Строгие методологии позволяют увидеть простую структуру процесса разработки, осуществляют строгую этапность процесса разработки и детальную документации проекта. Также задачи ставятся команде разработки с самого начала, а оценка стоимости и сроков могут быть определены до момента запуска разработки

Соответственно такой подход лишен гибкости, можно увидеть приоритет формального подхода к последовательности процесса работы. В случае нехватки времени может сократиться время выполнения какого-либо этапа, что приведет к потере качества [61].

Начиная с 2009 года Институтом управления проектами (PMI) предлагается гибридный вариант методологии управления проектами, как стандарт. Для компаний важно вовлечение в бизнес со стороны разработчиков, поэтому выбор может быть сделан в сторону гибридной методологии. Такой подход позволяет взять преимущества нескольких методологий. Также для компаний необходимо высокое качество продукта, поскольку только российские операторы обслуживают более 100 млн. клиентов [29].

2 Анализ существующих стандартов проектного управления и методологий разработки программного обеспечения

По мнению доктора экономических наук и директора бизнес-школы Уральского Федерального Университета, успешным проект можно считать, когда были достигнуты цели проекта, заданные критерии качества при ограниченных затратах временных и финансовых ресурсов.

Эффективность оценивает соответствие проекта целям и интересам его участников, прибыльность, срок окупаемости, рентабельность проекта [22].

Согласно The Standish Group, которая исследует успешность проектов, связанных с информационными технологиями, только 29 % ИТ-проектов закончились успешно [83].

ИТ-проекты терпят неудачу, так как намного сложнее обычных проектов. Они включают в себя трудности управления, свойственные обычным проектам: дедлайны, ограничения бюджета и недостаточное количество людей, которые могут быть задействованы в проекте. Более того, этот тип проектов зачастую сталкивается с уникальными технологическими вызовами, связанными с техническими устройствами, операционными системами или проблемами с базами данных [53].

Компания «ИТ Гильдия» выделяет основные проблемы в управлении проектами:

- корректировки в процессе выполнения проекта;
- плохая коммуникация между участниками проекта;
- непрозрачность процесса реализации проекта;
- взаимодействие с удаленными участниками и заинтересованными лицами;
- отсутствие готовых методов управления проектами [50].

Директор бизнес-школы Уральского Федерального Университета выделяет некоторые проблемы:

- предлагаются идеи, конфликтующие с требованиями заказчика;

– неоцененность технологий управления проектами;
– планируются задачи без согласования с заказчиком, выбираются только доступные решения и ресурсы;
– команда не вовлекается в проект и не взаимодействует с заинтересованными сторонами;
– составляется план без учета мнения команды;
– недостаток использования средств коммуникации для полного информирования участников проекта обо всех актуальных состояниях;
– отсутствует ретроспектива успехов и неудач команды в ходе реализации проекта [22].

Редакция ECM-Journal пишет о том, что обязательным условием выбора методологии управления ИТ-проектами являются первичные цели, ограничения (границы) и возможные сроки реализации. При этом мы также оцениваем такие аспекты, как:

– возможность или невозможность итерационного развития конечного решения, для реализации которого создается проект;
– начальные условия (наличие персонала, степени проработанности задачи);
– и многое другое, что может повлиять как на выбор самой методологии, так и на конечный результат проекта в целом [51].

Bilginc IT Academy считает, что управление проектами в телекоммуникационной отрасли не только отличается от других отраслей, но и является очень критичным, если вспомнить о сложностях управления командой и необходимости обеспечения эффективной коммуникации между распределенными заинтересованными сторонами [87].

Такая организация предусматривает следующие особенности проектов в телекоммуникационной сфере:

– длительные циклы планирования и исполнения, чтобы допускать как можно меньше ошибок и пробелов;

– серьезные изменения обычно не вносятся в середине проекта из-за стоимости исправлений, поэтому подход к планированию должен быть серьезным;
– должна быть частая коммуникация основных заинтересованных сторон проекта;
– у пользователей большая потребность в телекоммуникационных услугах, здесь большая цена ошибки.

Для многих программных продуктов актуален следующий жизненный цикл программного продукта: сбор и анализ требований, проектирование, разработка, тестирование и отладка, выпуск, сопровождение.

«В управлении проектами разработки ПО существует набор стандартных (типовых) задач, которые могут решаться одинаково, вне зависимости от технических и организационных особенностей проекта и выбранной для него методологии» – считает один из исследователей. Методология включает в себя набор методов по управлению разработкой и модели жизненного цикла ПО [29].

Таким образом, важно, чтобы цели, задачи, затраты и сроки каждого проекта были четко определены, чтобы избежать проблем в дальнейшем. Также надежный финансовый план необходим для любого проекта. Важно с самого начала правильно распределить всю ответственность и обеспечить здоровую коммуникацию между командами, чтобы не сорвать телекоммуникационные проекты. Необходимо предвидеть изменения и проявлять осторожность, чтобы не тратить время и деньги на адаптацию к новым условиям [87].

2.1 Стандарты проектного управления

Таким образом, важно подобрать стандарт, который покрывает вышеописанные требования. Для сравнения были выбраны 2 наиболее известных международных стандарта: PRINCE2 (Projects In Controlled

Environments) и PMI PMBOK (Project Management Body Of Knowledge) и российский ГОСТ Р ИСО 21500-2014.

PMI PMBOK содержит множество процессов и общепринятых методов управления проектами, с помощью которых можно оценить или завершить способ выполнения проектов или методологию, которую вы используете. Поэтому это более теоретический справочник.

PRINCE2 содержит подробную моделью процесса и шаблоны. Это дает пошаговое руководство о том, как организовать и запустить проект. Это руководство, чем справочное руководство. Он фокусируется только на ограниченном наборе методов.

ГОСТ Р ИСО 21500-2014 – национальный стандарт Российской Федерации. Руководство по проектному менеджменту содержит основополагающее руководство по проектному менеджменту и может применяться организациями любого типа, включая государственные, частные или общественные организации, в отношении проектов любых видов, независимо от их сложности, масштаба и продолжительности.

Стандарт содержит общее описание принципов и процессов, которые рассматриваются в качестве составляющих рациональной деятельности по проектному менеджменту. В стандарте проекты рассматриваются в контексте программ и портфелей проектов. Стандарт не содержит детальных указаний относительно управления программами и портфелями проектов. Вопросы, относящиеся к области общего менеджмента, рассматриваются только с точки зрения их связи с проектным менеджментом [7].

Сравнительный анализ приведен в таблице А.1 в приложении А [7, 15, 25].

Также консалтинговая компания Vinsys собрала статистику по предпочтению сертификации по этим направлениям в разных регионах мира (таблица 2). PMP – сертификация, которая подтверждает знания и навыки работы с PMI PMBOK.

Таблица 2 – Предпочтение PMP и PRINCE2 в регионах мира

Регион/Страна	1-е место по предпочтению	2-е место по предпочтению
Азия	PRINCE2/ PMP	PRINCE2/ PMP
Африка	PRINCE2/ PMP	PRINCE2/ PMP
Америка	PMP	PRINCE2
Австралия	PRINCE2	PMP
Европа	PRINCE2	-
Ближний Восток	PRINCE2/ PMP	PRINCE2/ PMP
Россия	PRINCE2/ PMP/ ПМ СТАНДАРТ	PRINCE2/ PMP

Можно увидеть, что во многих странах требуется сертификат PRINCE2 [43, 79].

Однако в России есть собственная сертификация ПМ СТАНДАРТ, который основан на ГОСТ Р ИСО 21500 – 2014. Она разработана автономной некоммерческой организацией «Центр оценки и развития проектного управления», которая активно содействует развитию проектного управления в Российской Федерации и СНГ, повышению эффективности применения принципов, методов и инструментов проектного управления в коммерческих и государственных организациях [44].

ПМ СТАНДАРТ опирается на российские и признанные в России международные стандарты, учитывает лучшие наработки и тренды развития сертификаций в мире [42].

Также британский телекоммуникационный оператор Vodafone использует PRINCE2 в своих проектах [79, 80]. Сотрудник международной компании-разработчик IT-решений для телекоммуникационного бизнеса «НетКрэкер» поделился положительным опыт использования PRINCE2, как логичным, структурированным и масштабируемым инструментом [62].

Компания по разработке программного обеспечения «Бастион Интегратор» поделилась опытом внедрения IT проекта для российской компании ООО «Т2 Мобайл». Их проект выполнялся по стандарту PMI PMBOK, а методология для разработки решения был выбран Microsoft Solution Framework [3].

В докладах организации «Вымпелком» отмечается, что для управления проектами ИТ департамента был выбран PMBOK PMI [52]. ПАО «Ростелеком» решили транслировать рекомендации PMBoK на их организационное окружение [3,14].

Исследователи из Уппсальского университета (Швеция) выделяют следующие фазы жизненного цикла проекта на рисунке 4.

- идея – это первая фаза, которая помогает руководству оценить идеи проекта и определить их приоритетность;
- фаза предварительного исследования включает в себя проведение выявления заинтересованных сторон, бизнес-ценностей и анализ структуры для выполнения проекта;
- фаза планирования связана с выбором методов достижения запланированных целей;
- в фазе выполнения происходят запланированные мероприятия. Они анализируются на предмет отклонений или изменений и обрабатываются. На этой фазе создаются результаты, которые передаются в конце фазы выполнения;
- фаза завершения – это оценка проекта, утверждение итогового отчета, завершение проекта и постепенный уход проектной команды;
- фаза влияния или воздействия помогает обеспечить получение выгод от проекта и результатов обучения, полученных на различных этапах проектного цикла, которые могут быть продолжены.

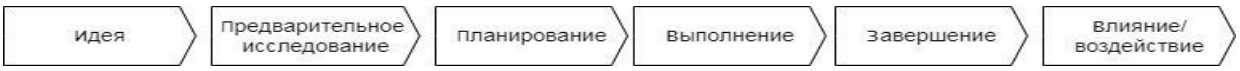

Рисунок 4 – Жизненный цикл проекта

После каждой фазы определяются результаты этапа, чтобы принять решение о возврате, продолжении или прекращении проекта соответственно. Сравнение плана проекта с фактическим планом может помочь в отслеживании хода проекта. Это может обеспечить достижение ожидаемого результата в установленные сроки.

Менеджер проекта должен следовать за проектной группой и фокусировать ее на достижении цели проекта. Управление проектом предполагает открытое общение между всеми сторонами, участвующими в проекте, поскольку оно может помочь достичь ожидаемых результатов для конечных пользователей или заинтересованных сторон [71].

Ведущий преподаватель университета Нортистерн (Северо-Восточный университет в Бостоне, штат Массачусетс) пишет, что что правильная методология должна выбираться исключительно на основе проекта и потребностей организации.

Ключевые факторы для рассмотрения:

– отрасль;

– конкретные цели или задачи проекта;

– сложность проекта.

Руководители проектов пытаются применить единый подход к различным ситуациям, не уделяя должного внимания тому, что принесет наибольшую пользу в данной конкретной работе – такая практика может нанести вред конечному результату проекта, если они не будут осторожны. По этой причине предпочтителен более разносторонний подход к использованию методологий управления проектами [66].

2.2 Традиционные методы управления

Традиционные методы управления проектами предполагают высокую степень детализации плана проекта, чтобы обеспечить достижение результатов проекта с определенным объемом, стоимостью и временем. Этот метод последовательно повторяет жизненный цикл проекта. Состоит из следующих этапов:

- сбор требований к проекту, составление технического задания и плана работ;
- проектирование принципов, архитектуры, логики продукта, а также подбирают инструменты;
- разработка;
- тестирование;
- эксплуатация и поддержка, где продукт выпускается в использование, а затем устраняют ошибки, добавляют новые функции.

Портал «ProКачество» отмечает следующие особенности данного подхода:

- все этапы идут друг за другом, то есть на следующий этап проекта переходят только после завершения на предыдущего;
- после завершения этапа вернуться к нему нельзя;
- перед переходом на следующий этапа результат должен пройти проверку и приемку;
- все процессы проекта описываются в документах, и участники проекта не могут их менять во время работы.

Также выделяются разные методы выполнения работ в классической методологии:

- в параллельном подходе работы разных этапов делают одновременно, значительно ускоряет выполнение проекта, но и увеличивает бюджет.

– в поточном методе соблюдается баланс между последовательным и параллельным методом. В нем команда передает часть работы на следующий этап и сразу начинает делать следующую часть. Так проект идет быстрее, а затраты на команду не увеличиваются [47].

В целом традиционное управление проектами характеризуется большим объемом документации, минимальным взаимодействием с клиентами, формальным общением и уникальным результатом в качестве конечного итога.

Водопадная модель (Waterfall) известна, как модель, управляемая планом, поскольку она может быть реализована в небольших и сложных проектах. Каждый запланированный вид деятельности должен быть завершен и утвержден, прежде чем переходить к следующему виду деятельности, не допуская наложения различных фаз.

В данной методологии водопада необходимо детальное планирование фаз, чтобы избежать ошибок, иначе могут возникнуть незапланированные изменения во время выполнения каждой фазы. Этот метод требует много времени, что делает его непригодным для небольших проектов, а документация должна быть эффективно объяснена в самом начале, поскольку это может привести к провалу проекта [71].

2.3 Гибкие методы управления

2.3.1 Scrum

Существуют общие практики по гибкому управлению. Однако все практики разработаны в соответствии с манифестом Agile.

Управление проектами Agile имеет итеративный характер, и изменения вносятся членами команды и менеджером проекта в план проекта и в связи с требованиями. Самоуправляемые команды создают большой фокус с вовлеченностью, преданностью плану проекта. Эти самоуправляемые команды требуют вовлеченности, самоотдачи при мониторинге и обновлении

хода проекта.

Четкого определения Agile-управления проектами не существует, и развитие теории Agile-управления проектами все еще находится на переходном этапе. Решающими составляющими успеха Agile-управления проектами являются планирование, контроль и коммуникация. Ниже представлены и описаны основные управленческие характеристики Agile-управления проектами:

- отношения с клиентами;
- планирование и контроль;
- требования, разработка и тестирование;
- содержание устанавливается путем переговоров на предстоящей итерации.

В течение последних двух десятилетий наблюдается все больший переход к Agile-методам управления проектами. Agile-практики используются в проектах программного обеспечения благодаря надежному планированию, гибким итерациям, прогрессивным улучшениям по мере развития и быстрой доставки. Каждая задача должна быть протестирована должным образом, чтобы снизить риски при выполнении следующих задач [71].

Наиболее часто используемой Agile-методологией для разработки программного обеспечения в рамках проекта является Scrum. Scrum допускает гибкость в сроках выполнения проекта, частые итерации в плане проекта, выполнение работ в соответствии с требованиями заказчика.

В этой методологии основное внимание уделяется временным отрезкам, которые называются спринтами с фиксированной продолжительностью.

Существует бэклог продукта – это список приоритетных требований к продукту, в котором есть функции и итерации, которые должны быть выполнены проектной командой, клиентами, продажами и маркетингом.

Продолжительность спринтов составляет от одной до четырех недель, и этот принцип соблюдается на протяжении всего срока реализации проекта. Команда проекта выбирает из бэклога продукта те элементы, которые, по их

мнению, будут завершены в текущем спринте, и создает бэклог спринта.

Scrum-мастер, владелец продукта и команда – это три основные роли в Scrum. Scrum-мастер выступает в роли посредника в команде, а также ведет переговоры и налаживает коммуникацию внутри команды, устраняя трудности.

Владелец продукта – это голос заказчика, который работает с командой проекта и принимает необходимые решения относительно продукта. Владелец продукта владеет тем, что он выбирает для создания функций продукта.

Scrum-команда является главным ответственным за доставку продукта и состоит из пяти-семи человек. Команда ежедневно отслеживает ход выполнения задач, которые они выполняют, и отчитывается на ежедневных Scrum-собраниях.

Пять основных действий команды проекта:

- начальное совещание по приоритетам из бэклога продукта в начале проекта;
- совещание по планированию спринта;
- спринт;
- ежедневный Scrum;
- совещание по обзору спринта.

В каждом спринте Scrum-мастер обеспечивает потенциально пригодный для использования продукт как результат спринта. Scrum-мастер стремится к инкрементальному развитию в течение каждого спринта, что является основной характеристикой всех Agile-методов управления проектами. Ежедневный Scrum включает в себя членов Scrum-команды вместе со Scrum-мастером для проведения ежедневных Scrum-собраний. Эти встречи проводятся для отслеживания хода ежедневной работы и обсуждения прерываний и ответственности. Команда обсуждает работу, которую они сделали вчера, работу, которую нужно сделать сегодня, прерывания и результат, который они получили.

Каждый спринт завершается собранием по обзору спринта, на котором

Scrum-команда проводит самооценку и представляет ее владельцу продукта. Во время самооценки происходит понимание сильных и слабых сторон, что закладывает основу для дальнейших улучшений в следующих спринтах в рамках плана проекта [71].

Компания-разработчик программного обеспечения для управления разработкой программного обеспечения Atlassian выделяется следующие артефакты у Scrum:

- бэклог продукта – это основной список задач проекта. Это непрерывно изменяющийся перечень функциональности, требований, улучшений и корректировок, из которых возникают задачи на спринт. Владелец продукта обычно обращается в бэклог;
- бэклог спринта – это список выполняемых задач, историй пользователей или корректировки дефектов, которые отобрали команда разработчиков, чтобы реализовать в этом цикле. Перед каждым спринтом проводится совещание о планировании, в котором команда определяет, какие из задач из бэклога продукта должны быть выполнены;
- цель спринта, инкремент – готовый к применению конечной продукции по результатам спринта. Обычно демонстрируется команде продукт за спринт.

«Сложные задачи могут быть упорядочены в простые истории пользователей, роли четко разделены, весь цикл разработки сохраняет прозрачность, коллективную ответственность в целом. Частые выпуски продуктов мотивируют коллектив и гарантируют удовлетворенность заказчика» – отмечает Atlassian [57].

2.3.2 Kanban

Kanban четко обозначает, какая работа должна быть выполнена и когда она должна быть выполнена с помощью квалифицированного набора разработчиков. Разработчики проекта начинают с применения компонентов проекта, которые добавляют ценность и позволяют избежать ненужных

потребностей, сокращая время и усилия.

Работа построена вокруг доски Kanban, которая применяется для наглядного и оптимизированного процесса работы. Часто используют виртуальные, онлайн доски: с помощью них проще отслеживать процессы, организовывать совместные действия и получить доступ из различных мест.

Такие доски необходимы для:

- визуализации работы команды,
- стандартизации процессов,
- поиска и устранения критических задач и затруднений.

На стандартной доске процесс состоит их трех этапов: «В плане», «В работе» и «Сделано». Однако доска может быть настроена в зависимости от процесса, выбранного командой.

Методология Kanban основана на полной прозрачности и дискуссиях о производительности в реальном времени. Таким образом, доска должна быть единственной достоверной информацией о деятельности команды.

Команда при таком подходе ориентирована исключительно на текущую работу. Завершая рабочую задачу, команда забирает следующий пункт с самого верха списка задач. Владелец продукта может изменить приоритет задач на бэклоге без ущерба для работы команды, так как изменения вносятся за пределы текущей рабочей задачи. Необходимость в фиксированных спринтах, как в Scrum, не наблюдается.

Длина цикла показатель, который является ключевым для команды, это время проведения рабочей работы в цикле продукта от начала рабочей работы до ее поставки. Оптимизация длительности цикла позволит команде в дальнейшем уверенно предсказать срок доставки продукта.

Если задач в списке много, то приходится переключаться между ними, что сказывается на сроках их завершения. Таким образом, одним из главных принципов Kanban является ограничение объемов незавершенных работ. Такой элемент позволяет быстро найти в команде узкие, проблемные участки, которые вызваны недостатком внимания, персонала или навыков.

Непрерывная поставка – это подход, при котором команды в автоматизированном режиме часто и предсказуемо выпускают качественные продукты в рабочую среду из репозитория исходного кода.

«Kanban и непрерывная поставка идеально дополняют друг друга, поскольку обе методики основаны на своевременной (и последовательной) поставке ценности. Чем быстрее команда сможет выпустить инновационное решение на рынок, тем более конкурентоспособным будет ее продукт. Kanban-команды сконцентрированы именно на оптимизации процесса поставки продуктов клиентам» [56, 58].

Метод Kanban работает на основе приоритезации задач, определения рабочего процесса и завершения результатов в запланированные сроки. Kanban создает гибкость между задачами и снижает риски невыполнения задач, способствуя непрерывному потоку в проекте. В организации рабочий процесс меняется в зависимости от состояния проекта.

«Этот метод является одной из самых сложных концепций, используемых для управления потоком, который может быть представлен с помощью диаграмм сгорания, карт потока создания ценности, доски Kanban и кумулятивных диаграмм потока» [71].

2.3.3 eXtreme Programming

В экстремальном программировании (eXtreme Programming, XP) основой проектной документации считается код с комментариями. XP является гибкой методологий.

Выделяются следующие практики экстремального программирования:

- все участники проекта, в том числе представитель заказчика – одна команда;
- сначала происходит планирование выпуска и планирование итераций. Обычно разработчики конкретизируют требования заказчика и дробят на части, реализация которых занимает не более одного дня, а клиент расставляет приоритеты на итерацию;
- версии ПО выпускаются часто, но с небольшими функциональными

возможностями;

- заказчик сам определяет приемочные тесты, а команда их реализует;
- коллективное владение кодом;
- непрерывная интеграция и стандарты кодирования;
- сравнение разрабатываемого продукта с чем-то знакомым, чтобы сформировать у команды общее видение;
- команды работают на максимуме продуктивности, сохраняется устойчивый темп;
- использование разработки, основанной на тестировании;
- парное программирование, рефакторинг – улучшение кодовой базы проекта;
- простой дизайн и код

«Большое внимание в методологии уделяется разработке, основанная на тестировании. Для каждого нового метода сначала пишется тест, а сам метод остается пустым. Затем разрабатывается код метода до тех пор, пока тест не выполниться успешно. Эти тесты сохраняются в коллекциях, которые автоматически выполняются после любого изменения кода. Данная методология требует вовлечение только опытных специалистов и обязательно заказчика продукта» [29].

2.4 Rational Unified Process (RUP)

«Строгая методология Rational Unified Process (RUP) основана на итеративной модели. Считается архитектурно-центрической методологией. Как правило, реализация и тестирование архитектуры системы должны начинаться на самых ранних стадиях проекта. В ходе жизненного цикла проекта распределение усилий проектной команды между дисциплинами постоянно меняется. Однако в целом дисциплины выполняются параллельно» [29].

Все участники проекта используют единую базу знаний, в целом единый

процесс, взгляд на разработку. Все модели методологии в можно найти в нотации UML. Основные особенности перечислены ниже:

- это итеративный процесс;
- предполагает сквозное применение сценариев использования;
- внимание уделяется разработке архитектуры;
- включает в себя управление требованиями и изменениями;
- базируется на визуальном моделировании.

Данная методология применима как в небольших и быстрых проектах, где за счет отсутствия формализации требуется сократить время выполнения проекта и расходы, так и в больших и сложных проектах, где требуется высокий уровень формализма, например, с целью дальнейшей сертификации продукта. Это преимущество дает возможность использовать одну и ту же команду разработчиков для реализации различных по объему и требованиям [24].

2.5 Сравнительный анализ методологий разработки

В таблице Б.1 в приложении Б приведен сравнительный анализ методологий разработки программного обеспечения.

Видно, что все методологии имеют свои достоинства и недостатки. Часть из них подходит под объект исследования, другая часть нет. В этой связи необходима разработка гибридной методологии.

Из исследования RQR (Russia Quality Report) Перфоманс Лаб в 2020-2021, в котором было опрошено более 350 организаций из телекома, ИТ, финансовой сферы, ритейла, ТЭК, промышленности о ситуации на рынке обеспечения качества ИТ-систем и тестирования рынка ПО в России следует, что «гибкие методологии разработки (Agile) популярны не во всех областях экономики, кто-то не знает, что с ними делать, обладаем небольшим опытом или просто пока не готов к использованию. Несмотря на это, гибкие методологии разработки (Agile) продолжают завоевывать симпатии

российских компаний и организаций. По данным прошлых отчетов компании, еще четыре года назад такой подход применяли всего 43 % опрошенных игроков рынка. В 2020 году их количество увеличилось до 80 %» [68].

Поставщик профессиональных ИТ-услуг Datix Inc. дает следующие рекомендации для проекта по обновлению и внедрению новой версии программного обеспечения необходимо:

- выделить отдельную команду с возможностью приглашения экспертов;
- вовлечь заказчика и активно взаимодействовать с конечными пользователями;
- поддержка со стороны исполнительного руководства;
- четкое формулирование требований;
- планирование, разделение проекта на этапы, чтобы сделать его более управляемым;
- проведение миграции данных, объединив данные-дубликаты и избавьтесь от бесполезной информации;
- тщательное тестирование;
- описывать варианты использования;
- обучение пользователей и наличие справочной информации.

Обновление имеет решающее значение для продления срока службы программного обеспечения и получения максимальной отдачи от инвестиций. Вдумчивый, совместный подход к проектам корпоративного программного обеспечения может сгладить любые потенциальные препятствия. Такой проект может быть завершен безболезненно и привести к значительным улучшениям в бизнесе [65].

3 Разработка модели улучшенной гибридной методологии и проектирование информационной системы

3.1 Описание стадий и процессов проекта

В ГОСТ Р 54869 – 2011 рассматриваются следующие процессы ведения проекта:

а) инициация проекта – формальное открытие проекта;

б) планирование проекта:

1) планирование содержания проекта – определение требований проекта и состава работ проекта;
2) разработка расписания – определение дат начала и окончания работ проекта, ключевых событий, этапов и проекта в целом;
3) планирование бюджета проекта – определение порядка и объема обеспечения проекта финансовыми ресурсами;
4) планирование персонала проекта – определение порядка обеспечения проекта человеческими ресурсами;
5) планирование закупок в проекте – определение порядка и объема обеспечения проекта продукцией и услугами, приобретаемыми у сторонних организаций;
6) планирование реагирования на риски – определение основных рисков проекта и порядка работы с ними;
7) планирование обмена информацией в проекте – определение порядка обмена информацией между лицами, участвующими в реализации проекта и заинтересованными в результатах проекта;
8) планирование управления изменениями в проекте;

в) организации исполнения проекта – организация выполнения проекта согласно разработанным планам;

г) контроль исполнения проекта – проверка соответствия процессов и продукта проекта установленным требованиям;

д) завершения проекта – формальное закрытие проекта [5].

В Scrum вместо итераций происходят спринты, в которых происходит работа над списком дел. Во время спринта происходит анализ, разработка и тестирование функциональности. По итогам команда проекта показывает демонстрацию промежуточного продукта и собирает обратную связь [84].

3.2 Описание гибридного подхода

Гибридный подход к управлению проектами часто рассматривается для проектов, чтобы увеличить количество обратной связи заинтересованных сторон, снизить риск и неопределенность. Хотя у каждой методологии есть многочисленные преимущества, руководители проектов сталкиваются со многими проблемами при реализации крупномасштабных проектов. Отсутствие эффективного использования методологии управления проектами приводит к плохой эффективности проектов. Внедрение различных проектных методологий не дает общих преимуществ, которых ожидают руководители проектов.

Существует несколько вариантов гибридных методологий. Абхинава Чандрабабу и Анушы Муддангула из Уппсальского университета предложили гибридную модель для разработки ИТ-проектов, в которой смешиваются Scrum и классическая методология, может быть использована на этапе тестирования. Классика используется для определения требований к проекту, а Scrum применяется во время проектирования, реализации и закрытия проекта [71].

Каждая организация имеет различные структуры управления и жизненные циклы проектов. Таким образом, одна методология управления проектами не подходит для всех. Организациям необходимо адаптировать методологию, которая соответствует их бизнес-процессам, чтобы достичь успеха проекта [73]. Применяя гибридный подход, может быть получена рабочая эффективность [82].

Наиболее часто используемый процесс в гибридной модели состоит из пяти фаз: планирование, фазы первоначальных требований и проектирования, итеративные спринты, обеспечение качества и развертывание.

Проект начинается с традиционной водопадной модели планирования. Во время первоначального анализа проводятся совместные сессии по разработке. На этом этапе планирования члены команды определяют и уточняют дизайн, требования проекта, работая с заказчиком.

На этой фазе 80 % дизайна и требований должны быть покрыты, а остальное остается в ходе спринтов. В соответствии с требованиями остальные и будущие задачи будут определены командой.

Гибридный подход рекомендует проводить итерации с ограничением объема, то есть запланированный объем первой итерации завершается до начала второй итерации.

Схема улучшенной гибридной методологии разработки программного обеспечения на основе модели Абхинава Чандрабабу и Анушы Муддангула из Уппсальского университета представлена в виде рисунка В.1 в приложении В.

Команда проекта должна состоять из менеджеров направлений, проектного менеджера и исполнителей по направлениям (бизнес-аналитики, инженере программисты, по тестированию и так далее), а также сотрудники заказчика такого программного обеспечения. В данном случае владельцем продукта становятся сотрудники заказчика, а скрам-мастером – проектный менеджер.

3.3 Планирование и первоначальный список задач

3.3.1 Управление рисками проекта

Данный этап начинается с модели Waterfall, потому что необходимо провести большую часть анализа по требованиям заказчика к ПО и его архитектуре и технологиям после миграции на новую архитектуру.

Специалисты McKinsey & Company также выявили актуальную

проблему, связанную с безрисковым управлением в ИТ-проектах, а именно необходимость такого методического обеспечения для управления ИТ-проектами, при котором фактические результаты максимально сходились бы с запланированными. Подобное безрисковое управление позволит значительно увеличить долю успешных ИТ-проектов [30].

Для работы с рисками можно взять методику карточек Кроуфорда. Здесь собираются небольшое количество экспертов проекта. Ведущий встречи задает вопросы экспертам, на которые отвечает каждый участник.

Существуют несколько способов реагирования на риски в ИТ проектах:

- уклонение от риска в полной мере исключение рисков воздействия на проект, изменяя характер проекта или план управления проектом;
- передача риска – исключает риск передачей негативного последствия с ответственностью реагирования третьей стороне;
- принятие риска – при пассивном принятии команда ничего не предпринимает в отношении риска и в случае его возникновения разрабатывает способ его обхода или исправления последствий. При активном принятии план действий разрабатывается до того, как риск может произойти, и называется планом действий в непредвиденных обстоятельствах;
- принятие риска – команда не делает ничего с риском, но в случае возникновения разработает способ его устранения или исправления последствия;
- снижение риска предполагает усилия, направленные на уменьшение вероятности риска и его последствий до допустимых границ.

Таким образом, под каждый риск необходимо выбрать свою стратегию.

3.3.2 Методы приоритизации и оценки проектных задач

Необходимо оценить приоритет задачи, предлагается выбрать подход Value (ценность для бизнеса) и Effort (усилия, трудоемкость) название подхода: Lean Prioritization.

На рисунке 5 представлена матрица задач [13].

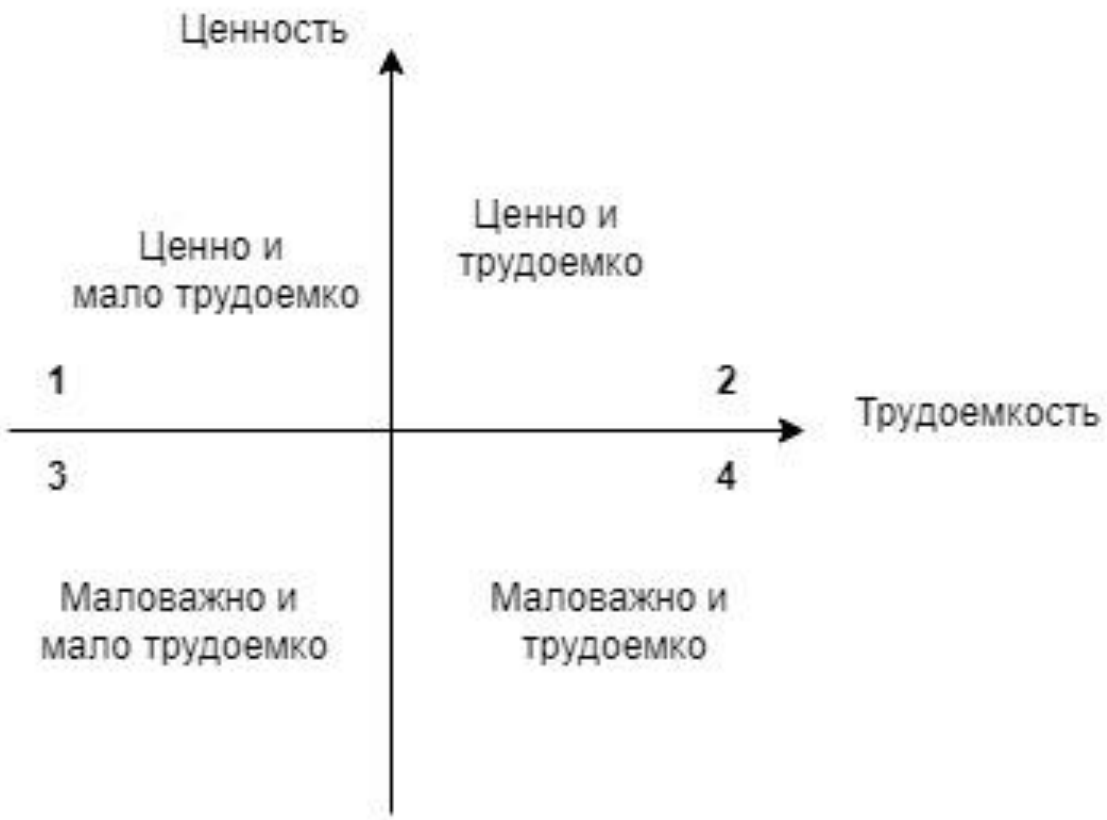

Рисунок 5 – Матрица задач

В данном подходе рекомендуется давать приоритет задачам в следующем порядке:

– задача несет ценность и мало трудоемка;

– задача несет ценность и трудоемка;

– задача маловажная и мало трудоемка;

– задача маловажная и трудоемка.

Для оценки задач можно использовать метод PERT (Техника Оценки и Анализа Программ и проектов). Он является инструментом, который вычисляет ожидаемое значение продолжительности проекта или отдельного процесса. Такой подход позволяет учесть все необходимые работы и подобрать оптимальный состав команды специалистов.

Метод PERT – расчет выполняется по трем точкам (рисунок 6) в часах.

$$t_e = \frac{1}{6}(t_o + 4t_m + t_p)$$

Рисунок 6 – Расчет PERT

Где переменные, это:

– to – оптимистическое (минимальное) время;

– tp – пессимистическое (максимальное) время;

– tm – наиболее вероятное время;

Итого ожидаемое время – это оценка длительности выполнения задачи на основе оценок оптимистического, пессимистического и наиболее вероятного времени [23].

3.3.3 Бизнес-анализ и определение требований

В проекте по миграции программного обеспечения как правило бизнес-процессы остаются такими же, поэтому необходимо согласование документации по новой архитектуре, какие бизнес кейсы могут добавиться в объем работ.

80 % дизайна решения нужно проработать до начала спринтов, а именно: анализ и проектирование существующих процессов, перенос текущей функциональности монолитного решения на новую архитектуру, а также и новых функций. Также необходимо привлекать команду архитекторов, разработчиков для обсуждения технический деталей.

Также для данного этапа предлагается использовать CASE средства. Это совокупность методов и способов проектирования и разработки программ с помощью автоматизированных инструментов, которые ориентированы на реализацию отдельных процессов ЖЦ для разных предметных областей.

Нотация – это описание системы, элементов данных и функций с помощью графов, диаграмм, таблиц, блок-схем и другое [19].

BPMN (Business Process Model and Notation) – это стандарт ISO с 2013 года и нотация для проектирования бизнес-процессов, которые можно

автоматизировать в информационных системах класса BPMS (Business Process Management System).

Нотация BPMN может быть использована для проектирования, анализа, регламентации и автоматизации бизнес-процессов компании [39].

Для большинства пользователей BPMN наиболее важным аспектом является графическое представление моделей. BPMN предоставляет три типа диаграмм:

- схема процесса или совместной работы;
- моделирование обмена данными между различными партнерами;
- диаграмма беседы – обзор нескольких партнеров и их взаимоотношений. диаграмма процесса или совместной работы является наиболее часто используемым типом диаграммы.

Спецификация BPMN объясняет различные элементы обозначения не только устно, но и определяет их в метамодели. Метамодель документируется с помощью диаграмм классов UML, которые графически показывают особенности различных конструкций BPMN и их взаимосвязи. Такая метамодель более точна и определенна, чем строго вербальная описания [69].

После анализа бизнес-процессов необходимо спроектировать информационную систему. В данном проекте происходит переход на другое архитектурное решение, поэтому здесь важно обратить особое внимание на проектирование новой системы. Для проектирования архитектуры необходимо использовать UML (Унифицированный язык моделирования) [37]. Как только бизнес-анализ и дизайн готовы на 80 %, следует переходить к процессу имплементации.

3.4 Итерация (спринт)

3.4.1 Планирование задач на итерацию

Имплементацию рекомендуется проводить по Scrum из-за его следующий преимуществ: гибкость и взаимодействие различных команд.

Планирование рекомендуется проводить с помощью подхода Lean Prioritization, который помогает сосредоточиться на наиболее ценных функциях для клиентов по сравнению с усилиями для их реализации.

Компания Atlassian пишет о том, что для успешного собрания по планированию спринта требуется подготовка. Например, руководителям необходимо освежить в памяти выводы с прошлых собраний, ознакомиться с мнениями заинтересованных сторон проекта.

Также рекомендуется привести бэклог продукта в соответствие с современными реалиями и скорректировать. Как правило, большинству команд стоит собраться перед планированием спринта, чтобы просмотреть и скорректировать бэклог.

На планирование спринта должно отводиться не более двух часов за каждую неделю спринта. Необходимо, чтобы каждый участник собрания понимал ограничения по времени. Таким образом, минимальная продолжительность собрания не задается, а максимальная ограничена.

В основе Scrum лежит эмпирический подход, который используется в данной модели, рекомендуется учиться на практике и затем использовать полученную информацию.

Цель спринта показывает, что ожидается от этого спринта в общих чертах, однако нужный результат можно сформулировать и в содержании бэклога. Оценка сложности по своей сути является прогнозом на основе имеющихся знаний. В целом рекомендуется сосредоточится на цели спринта и внести в бэклог ровно то количество задач, сколько понадобится для начала работы [34].

Основные сложности при разработке: большой набор технологий, инфраструктурные проблемы при разработке и тестировании [27]. В этой связи необходимо предусмотреть риски. Компания ООО «Облачные технологии» пишет, что машинное обучение подразумевает анализ данных для выявления в них закономерностей и имитации процесса получения опыта с пошаговым повышением точности. Также оно может применяться для

классификации данных [76].

В данном случае использовался подход обучения с подготовленными данным, где собрана статистика по реализации задач в разных ИТ-проектах. Выборка для обучения содержит 200 записей.

Задача – по входным параметрам задачи определить к какому классу задач ее можно отнести. Классы: задержка до 4 часов, ок, задержка более 4 часов.

Предварительно данные должны быть нормализованы и отсечены атрибуты: номер задачи, ожидаемое время, реальное время. Поскольку эти данные подсчитываемые и не нужны для анализа.

В данном случае следует использовать алгоритм: Дерево принятия решений (ДПР). Это метод представления решающих правил в иерархической структуре. В узлах находятся решающие правила и производится проверка соответствия примеров этому правилу по какому-либо атрибуту обучающей выборке [10]. Дерево должно строится с помощью рекурсивного алгоритма построения бинарного дерева решений ID3.

Пользователь вводит информацию о задаче или задачах в формате .csv по вышеперечисленным параметрам, алгоритм нормирует данные. Пример входной нормированной строки на рисунке 7.

#	Код задачи	Код направления	Оценка исполнителя задачи (1 (низкий) - 5 (высокий))	Приоритет (1 (высокий) - 4 (низкий))	Код компонента	Пессимистичное время, ч
1	1	1	4	1	3	3
2	2	2	2	4	1	12
3	3	1	4	3	2	4

Рисунок 7 – Пример входной строки

Далее алгоритм определяет по вопросам из дерева к какому классу риска относится эта задача и выводит информацию пользователю. Пример результата входной строки на рисунке 8.

Рисунок 8 – Пример результат входной строки

Пользователь может отредактировать значения: поменять оценку по времени или исполнителя, чтобы получить результат «ок». Однако пользователь может принять результат и заложить риск по выполнению данной задачи.

3.4.2 Составление дизайна и разработка ПО

Как правило, в микросервисной архитектуре используется несколько различных технологий, могут быть разные языки программирования. Для того, чтобы разработка велась с минимальным количеством ошибок необходимо использовать технологии по контролю версий и изменений. Система управления версиями – это система, сохраняющая изменения в одном или нескольких файлах так, чтобы потом можно было восстановить определенные старые версии [60].

В спринтах идет работа по написанию дизайна (оставшиеся 20 %) по процессу, описанному выше. В предпоследнем спринте необходимо закончить весь анализ и дизайн. Важно, чтобы к предпоследнему спринту весь дизайн был уже готов, в данном спринте допускаются небольшие работы, которые могут быть имплементированы командой. В последнем спринте оставляется только разработка и тестирование.

Рекомендуется использовать технику разработки через тестирование программного обеспечения test-driven development. Сначала пишется тестовые сценарии, часто автоматизированные, затем код, который направлен для успешного прохода тестов. А рефакторинг нового кода приводит к текущим стандартам [86].

Отчеты исследовательских центров показывают, что плотность дефектов снижается на 40–60 % в обмен на рост усилий, при котором время на выполнение возрастает на 15–35 % [41]. Такие тесты позволяют проверить

части компонентов по отдельности, являются дополнением для других видов тестирования. Подход помогает выявить дефекты на более ранних стадиях, что увеличить скорость и уменьшить стоимость исправления.

Также был рекомендован подход «разработка, ориентированная на бизнес» (Business-Driven Development, BDD). Это парадигма разработки программного обеспечения, которая фокусируется на потребностях бизнеса, а не на технологиях. Она предназначена для корпоративного программного обеспечения. BDD основывается на моделировании потребностей бизнеса и использует эти модели для проектирования.

BDD включает в себя цикл действий и состоит из четырех этапов: моделирование, создание, развертывание, управление. При таком подходе бизнес-команды и технические специалисты тесно сотрудничают, чтобы результатом успешного применения стал бизнес, в котором программное обеспечение открывает возможности, повышают эффективность и снижает затраты [85].

3.4.3 Тестирование

Для осуществления деятельности по тестированию рекомендуется использовать методы «белого ящика» и «черного ящика». Тестирование «белого ящика» основано на анализе структуры программы. Таким образом, программа считается полностью протестированной, если проведено исчерпывающая проверка маршрутов ее графа управления.

В этом случае формируются тестовые варианты, где:

– гарантируется проверка всех независимых маршрутов программы;

– выполняются все циклы;

– анализируется правильность внутренних структур данных.

При тестировании «черного ящика» рассматриваются системные характеристики программ, игнорируется их внутренняя структура. Такие тесты демонстрируют как выполняются функции ПО, принимаются исходные данные, вырабатываются результаты и сохраняется целостность внешней информации. Исчерпывающее тестирование, как правило, невозможно [32].

Также для проверки бизнес сценариев можно воспользоваться сравнительным тестированием. Такой вид проверки представляет собой сравнение действий из системы со старой архитектуры на действия из новой системы с новой архитектурой. Цель: удостовериться, что действия не отличаются друг от друга. При этом пользовательский интерфейс может отличаться по согласованию с пользователями данной системы.

В случае непрерывной поставки (CD) и непрерывной интеграции (CI) компания Atlassian рекомендует использовать также автоматическое тестирование, то есть применение программных инструментов для автоматизации ручных процессов проверки и утверждения результатов такого тестирования с участием человека.

В целом основная цель непрерывной поставки – максимально ускорить поставку новых релизов заказчикам. Непрерывная поставка следует за фазой непрерывной интеграции, которая требует автоматизированного тестирования новых изменений в коде. Таким образом, можно быть уверенным, что изменения не нарушат работу существующих функций и не создадут новых дефектов. Если все запланированные автоматические тесты успешно пройдены в процессе непрерывной интеграции, начинается фаза доставки.

При выходе внесении изменений в программный код необходимо запускать автоматические тесты, а затем команда разработчиков знакомится с результатами, чтобы продумать план работы по исправлению проблем.

Предлагается следующие приоритеты по автоматизации тестирования при непрерывной поставке:

а) сквозное (E2E) тестирование, которое имитирует работу пользователя во всем программном продукте. С их помощью обычно разрабатываются пользовательские истории. Замечено, что наибольшую выгоду можно получить, внедрив сквозное тестирование среди наиболее важных бизнес-процессов;

б) модульные тесты охватывают работу отдельных частей кода. Модульный тест охватывает функцию. Они проверяют, что при

определенных входных данных функция дает ожидаемый результат. Внедрение юнит-тестов рентабельное и быстрое, что позволяет добиться высокой окупаемости инвестиций;

в) интеграционные тесты проверяют различные взаимодействия с внешними элементами, они могут послужить экономичной альтернативой комплексному тестированию. Однако необходимо просчитать окупаемость, если используется сочетание модульного и комплексного тестирования;

г) тесты производительности измеряют скорость работы и отклика программного проекта. Например, по следующим критериям относительно времени:

1) загрузки страницы;
2) предварительной обработки;
3) вывода результатов поиска.

Также Atlassian обращает внимание на виды тестов программного обеспечения, которые следует проводить вручную:

- глубокое тестирование характеризуется нестандартной последовательностью, позволяющей выявлять ошибки и незапланированные результаты или реакции программного обеспечения. Эффективнее выделить для этой работы специалиста из группы контроля и оценки качества;
- визуальное регрессионное тестирование – это, например, перемещение элементов интерфейса, некорректный подбор шрифта или цветов и далее. Разработка автоматических тестов для вышеперечисленных проверок обходится достаточно трудоемкая;
- частота релизов для проектов с фиксированным графиком выпуска. Если они происходят часто, автоматизированное тестирование становится основным преимуществом, поскольку от него зависят непрерывная интеграция и непрерывная доставка.

«Таким образом, автоматическое тестирование – стандартная практика

в современной разработке ПО. Автоматические тесты применяют все передовые команды и компании. Они необходимы для организации процессов непрерывной интеграции и непрерывной поставки (CI/CD), которые помогают самым эффективным командам поставлять надежное многофункциональное ПО для клиентов» – говорится авторами Atlassian [38].

3.4.4 Демонстрация решения команде и обзор прошедшего спринта

В индустрии существует проблема в высокой сложности социотехнических систем, в результате составляются гипотезы, подлежащие проверке. Назначение демонстрации (демо) – это показ результатов команде или заинтересованным лицам проекта. С помощью него можно понять, соответствует ли программный продукт ожиданиям заказчика, решает ли поставленные задачи и приносит ли ценность.

Как правило, демо проводится в виде встречи с командой проекта и со всеми заинтересованными лицами согласно рекомендациям Scrum Guide. Важно, чтобы вся обратная связь по ценности результата должна быть не только получена, но и донесена до всей команды.

Ретроспектива (обзор прошедшего спринта) – это внутренняя встреча, где каждая команда направления проекта и ее руководитель самостоятельно оценивают прошедший спринт, предлагает улучшением процесса. Здесь необходимо предложить и получить: позитивное мышление, вовлеченность всех членов команды, план улучшений [54].

В манифесте agile говорится: для лучшего соответствия ценностям гибкой методологии разработки команды должны регулярно встречаться для проведения проверок и внесения поправок.

Компания Atlassian рекомендует: «ретроспектива должна давать команде проекта положительный опыт и заряжать ее энергией. Она помогает участникам команды делиться важными отзывами, смиряться с разочарованиями и сообща находить решения. Организаторы тоже могут многое узнать для себя, в том числе лучше понять, как работает команда и какие трудности (и успехи) она пережила в последнем спринте. Результатом

успешной ретроспективы становится список улучшений, за которые участники команды берут ответственность и к которым стремятся в следующем спринте».

По времени можно отвести на эту встречу от тридцати минут до часа в зависимости от того, сколько длился спринт и какой объем работы предстоит обсудить в ходе встречи. Каждый член команды должен присутствовать на ретроспективе. Человек, который строит успешную коммуникацию группы людей (фасилитатор собрания) ведет обсуждение и приглашает сотрудников, которые участвовали в текущей итерации.

Общий порядок встречи может выглядеть следующим образом:

- короткий список того, что было хорошо и что можно улучшить;
- обсудить способы и тактики для улучшения приоритетных элементов, сосредоточившихся на конечных результатах;
- составить план действий, где команда должна подготовить несколько практических идей по улучшению выбранных направлений. Для таких элементов должны четко определены владельцы и завершающие сроки;
- необходимо убедиться, что все участники имеют четкое представление о последующих действиях в работе и проекта.

После запуска крупного проекта можно пригласить представителя руководящей команды и сфокусироваться на обсуждении совместной работы команды [40].

Если обзор итогов спринта в команде проходит недостаточно конструктивно, это может быть сигналом о следующих проблемах:

- команда берет на себя слишком много работы и не успевает выполнить ее за итерацию;
- у команды накопился технический долг – это проблемы в коде, которые связанны с пренебрежением к качеству при разработке ПО, и они могут вызвать дополнительные затраты труда в будущем. При этом конечный пользователь может этого не заметить;

– при разработке не учитывается долгосрочная перспектива;

– процессы команды к разработке не отлажены до конца;

– руководство может менять приоритеты в течение итерации.

Следует учитывать, что неудачные результаты итерацию могут случаться. В таких случаях необходимо определить причины изменения итерации в ходе ретроспективы команды и составьте план действий, которые предупредили бы возникновение данных проблем в следующей итерации [31].

3.5 Приемочное тестирование и внедрение ПО

Завершение итерационного подхода происходит с завершением имплементации. После активной разработки гибридная модель предполагает переход на Waterfall, чтобы сэкономить ресурсы команды на поддержку и исключить имплементацию нового функционала.

Таким образом, внешнее тестирование происходит только в конце фазы разработки. Допускается устранять дефекты. На этом этапе максимальная концентрация на выдаче решения заказчику. Дополнительные требования от заказчика запускают новый процесс: анализ – разработка – тестирование – внедрение. Затем происходит внедрение с поддержкой небольшой части команды разработки.

Качество системы – это степень удовлетворения системой заявленных и подразумеваемых потребностей различных заинтересованных сторон, которая позволяет, таким образом, оценить достоинства [8].

Цель процесса управления качеством – соответствие продуктов, услуги и реализации процесса управления качеством организационным и проектным целям управления качеством и достижение удовлетворенности заказчика [6].

Существуют несколько моделей качества. ГОСТ Р ИСО/МЭК 25010-2015 предлагает следующий подход. В данный момент в серии SQuaRE три модели качества:

а) модель качества при использовании определяет пять характеристик:

1) результативность;
2) производительность;
3) удовлетворенность;
4) свободу от риска;
5) покрытие контекста;

б) модель качества продукта имеет восемь характеристик:

1) функциональная пригодность;
2) уровень производительности;
3) совместимость;
4) удобство пользования;
5) надежность;
6) защищенность;
7) сопровождаемость;
8) переносимость (мобильность).

в) модель качества данных.

Совместное использование моделей качества дает основание считать, что учтены все характеристики качества [8].

Затем после приемно-сдаточных работ программное обеспечение устанавливается на ресурсы заказчика, по согласованию установкой может занимается компания подрядчик или сотрудники клиента.

3.6 Моделирование АИС

Для решения задач управления проектом с помощью улучшенной гибридной методологии необходимо программное обеспечение. На рисунке 9 представлена BPMN-диаграмма процесса «Управления задачами».

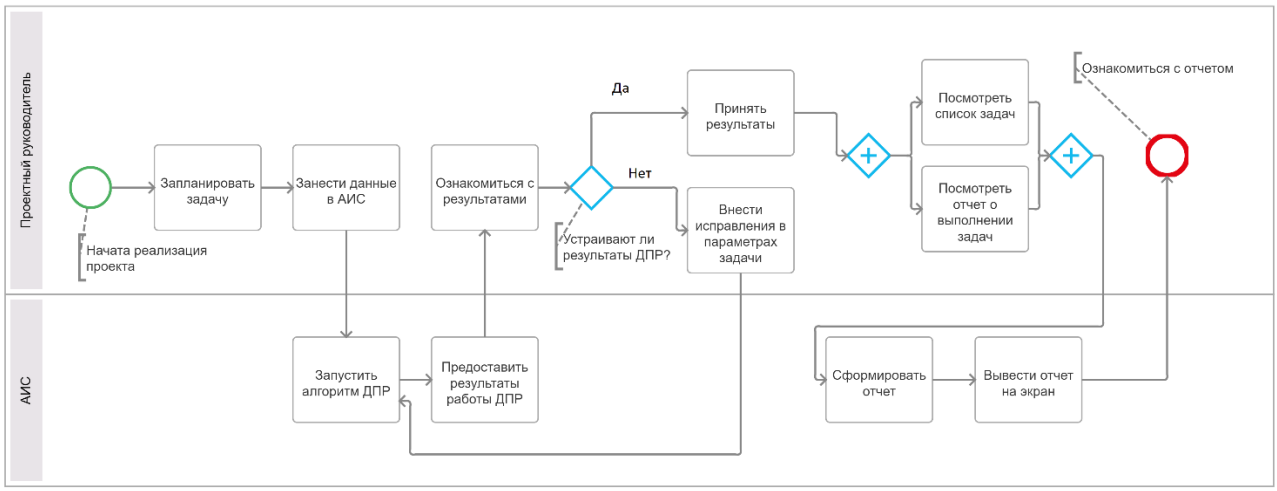

Рисунок 9 – BPMN-диаграмма процесса «Управления задачами».

В данном случае проектный руководитель создает объект «задача» в системе, далее работает алгоритм дерева принятия решений. Руководитель может принять результаты, а может изменить параметры задачи и АИС автоматически обработает указанные параметры и выведет новый результат.

Как только руководитель соглашается с результатами, он может посмотреть список задач, отчет о ходе выполнения задач. В этих случаях АИС формирует отчет и выводит его на экран. Пользователь может с ним ознакомиться.

На рисунке 10 представлена диаграмма вариантов использования (use case).

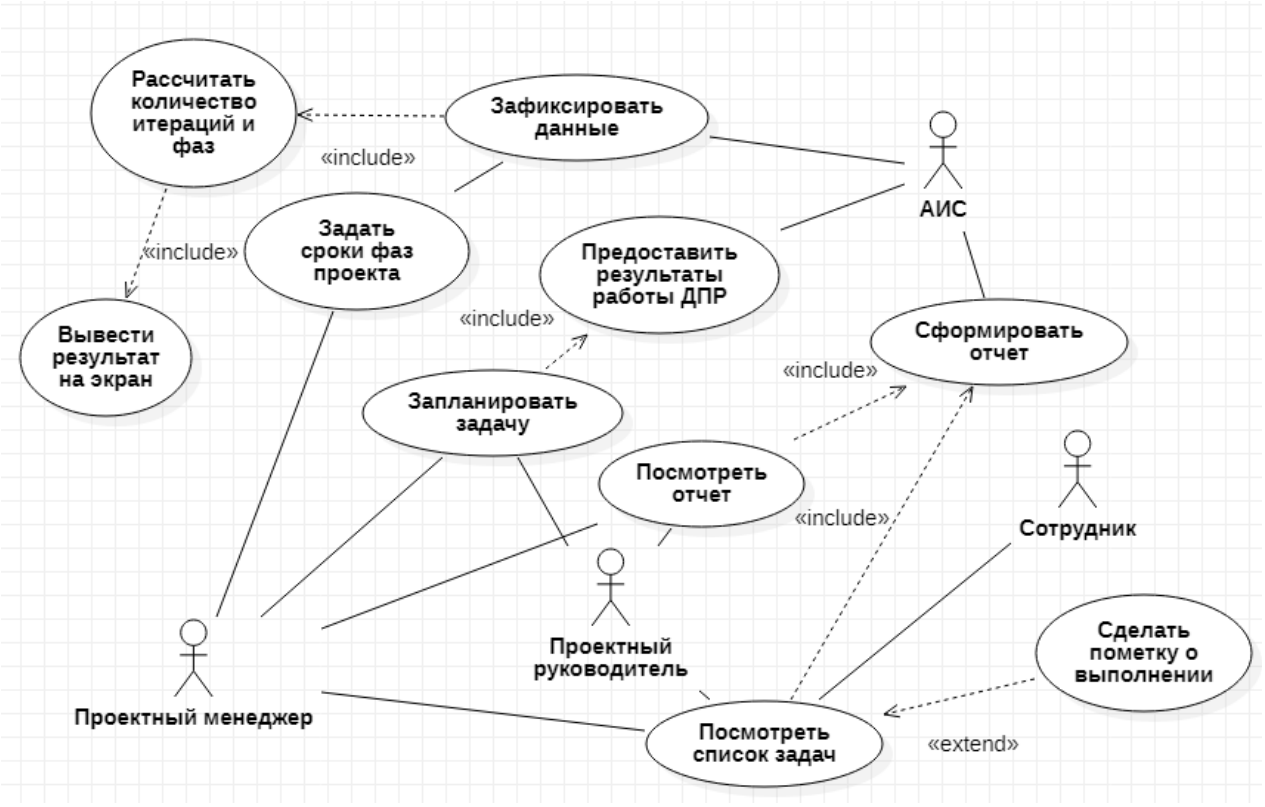

Рисунок 10 – Диаграмма вариантов использования АИС

По диаграмме видно, что проектный менеджер может:

– задать сроки фаз проекта;

– запланировать задачу;

– посмотреть отчет;

– посмотреть список задач, дополнительно можно сделать пометку о выполнении.

Проектный руководитель может всё, что проектный менеджер, кроме функции «задать сроки фаз проекта». Сотруднику доступен просмотр списка задач.

В случае следующих действий АИС:

– задать сроки фаз проекта – фиксирует данные, рассчитывает количество итераций и фаз, выводит результат на экран;

– запланировать задачу – предоставляет результаты работы ДПР;

– посмотреть отчет и список задач – формирует отчет.

В таблице 3 представлено описание сценария выполнения варианта

использования «Запланировать задачу».

Таблица 3 – Описание сценария выполнения варианта использования «Запланировать задачу».

Вариант использования	Запланировать задачу
Актеры	Проектный менеджер, проектный руководитель
Цель	Создание задачи в системе и предоставление результатов работы дерева принятия решений
Краткое описание	Проектный менеджер или проектный руководитель создает задачу в системе. АИС должна предоставить результаты работы дерева принятия решений
Тип	Базовый
Ссылки на другие варианты использования	Включает в себя следующий вариант использования: предоставить результаты работы дерева принятия решений

В итоге по диаграммам можно увидеть основные сценарии использования АИС для управления проектами по улучшенной гибридной методологии разработки программного обеспечения, а также описан важный и основной сценарий выполнения варианта использования «Запланировать задачу».

4 Апробация модели улучшенной гибридной методологии разработки программного обеспечения

4.1 Алгоритмы машинного обучения в планировании задач

Основные сложности при разработке: большой набор технологий, инфраструктурные проблемы при разработке и тестировании. В данном случае использовался подход обучения с подготовленными данным, где собрана статистика по реализации задач в разных ИТ-проектах. Выборка для обучения содержит 200 записей.

Атрибуты данных содержат:

- номер задачи – целочисленное значение;
- код задачи – целочисленное значение;
- проектный код – целочисленное значение;
- код заказчика – целочисленное значение;
- длительность проекта – целочисленное значение;
- код направления – целочисленное значение;
- оценка исполнителя задачи (1 (низкий) – 5 (высокий)) – целочисленное значение;
- приоритет (1 (высокий) – 4 (низкий)) – целочисленное значение;
- код компонента – целочисленное значение;
- оптимистичное время, ч – целочисленное значение;
- пессимистичное время, ч – целочисленное значение;
- наиболее вероятное время, ч – целочисленное значение;
- ожидаемое время, ч – вещественное значение;
- реальное время, ч – вещественное значение;
- результат – категориальное значение.

Задача – по входным параметрам задачи определить к какому классу задач ее можно отнести. Классы: задержка до 4 часов, ок, задержка более 4

часов.

Предварительно данные должны быть нормализованы и отсечены атрибуты: номер задачи, ожидаемое время, реальное время. Поскольку эти данные подсчитываемые и не нужны для анализа.

Были выбраны следующие алгоритмы для классификации данных:

- дерево принятия решений (ДПР) – это метод представления решающих правил в иерархической структуре. В узлах находятся решающие правила и производится проверка соответствия примеров этому правилу по какому-либо атрибуту обучающей выборке [10];
- случайный лес – это ансамбль Деревьев решений, где каждое дерево дает предсказание класса, и набравший наибольшее количество голосов класс, становится предсказанием [45];
- метод k-ближайших соседей (KNN**)** – цель вычислить расстояние до каждого из объектов обучающей выборки и отобрать минимальное расстояние. Результат в данном случае – это класс, наиболее часто встречающийся среди k-ближайших соседей [17].

Деревья строятся с помощью рекурсивного алгоритма построения бинарного дерева решений ID3.

Пользователь вводит информацию о задаче или задачах в текстовом формате Comma-Separated Values по вышеперечисленным параметрам, алгоритм нормирует данные. Далее алгоритм определяет к какому классу относится эта задача и выводит информацию пользователю.

Пользователь может отредактировать значения: поменять оценку по времени или исполнителя, чтобы получить результат «ок». Однако пользователь может принять результат и заложить риск по выполнению данной задачи.

Реализация алгоритмов была выполнена с помощью библиотеки машинного обучения для языка программирования Python Scikit-learn.

4.2 Дерево принятия решений

Дерево принятия решений было построено с помощью рекурсивного алгоритма построения бинарного дерева решений ID3.

Тестирование проводилось на выборке в 50 записей показало следующие результаты. Ограничение глубины дерева – это максимальное число разбиений в ветвях, по достижении которого обучение останавливается [10].

Таблица 4 – Оценка эффективности глубины дерева принятия решений.

Глубина дерева	Точность
2	Примерно 90 %
3	Примерно 100 %
4	Примерно 90 %

В таблице 4 указаны результаты испытаний с разной глубиной дерева. Таким образом, выбрана глубина дерева равная трем.

Приоритеты по атрибутам выборки представлены на рисунке 11.

Рисунок 11 – Диаграмма приоритетов критериев дерева принятия решений

В данном случае важным показателем в задаче является код направления и код компонента. Затем заданное пессимистичное время и приоритет задачи. Другие критерии модели неважны.

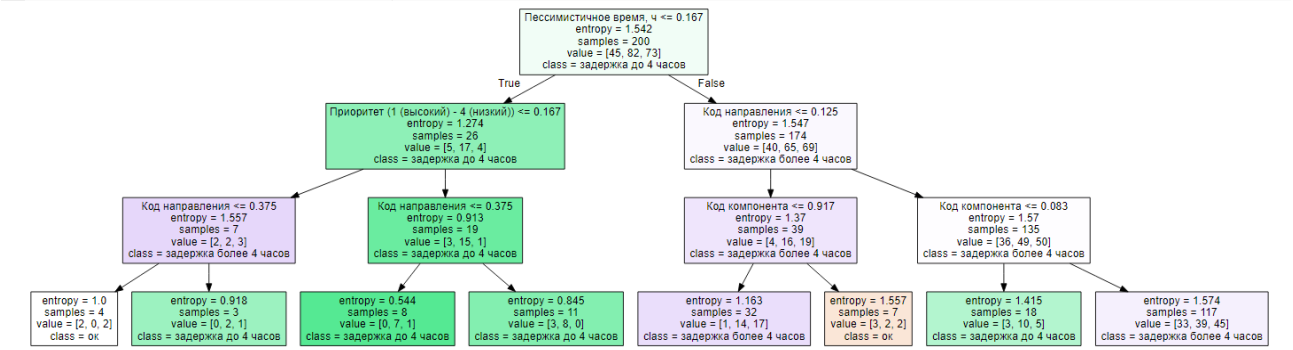

Рисунок 12 – Дерево принятия решений

На рисунке 12 можно увидеть получившиеся дерево принятия решений, присутствует три класса и глубина дерева равная трем.

4.3 Случайный лес

Тестирование проводилось на выборке в 50 записей показало следующие результаты в таблице 5.

Таблица 5 – Оценка эффективности глубины случайного дерева.

Глубина дерева	Точность
2	Примерно 90 %
3	Примерно 100 %
4	Примерно 100 %

Таким образом, выбрана глубина дерева равная 3. Приоритеты по атрибутам выборки представлены на рисунке 13.

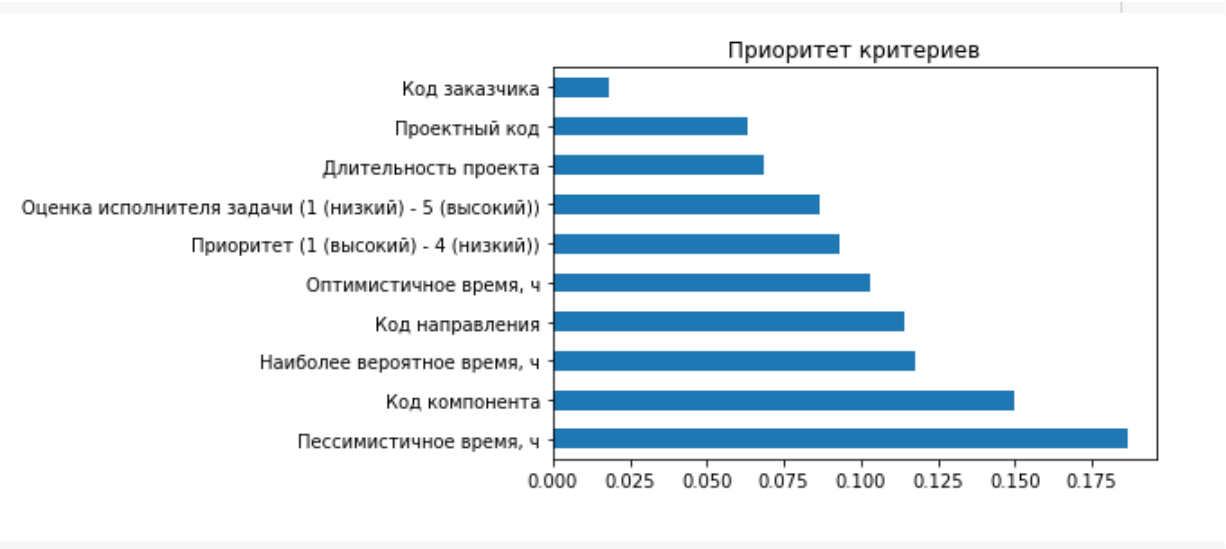

Рисунок 13 – Диаграмма приоритетов критериев случайного леса

В данном случае важным показателем в задаче является пессимистичное время и код компонента. Затем заданное наиболее вероятное время и код направления. Другие критерии модели менее важны.

4.4 Метод k-ближайших соседей (KNN)

Тестирование проводилось на выборке в 50 записей показало следующие результаты в таблице 6.

Таблица 6 – Оценка эффективности числа соседей для KNN.

Число соседей	Точность
1	Примерно 28 %
2	Примерно 28 %
3	Примерно 30 %
5	Примерно 30 %

Таким образом, выбрано число три (соседа), поскольку другие варианты увеличивают время выполнения алгоритма, а меньшее число соседей дает наименьшую точность прогноза.

4.5 Сравнительный анализ алгоритмов

В таблице 7 проведен сравнительный анализ алгоритмов по основным критериям: время обучения, сек; время предсказания, сек; точность предсказания, процент на выборке в 50 записей.

Таблица 7 – Сравнительный анализ алгоритмов

Показатель	ДПР	Случайный лес	KNN
Время обучения, сек	0.0085	0.1429	0.0033
Время предсказания, сек	0.0041	0.0211	0.0061
Точность предсказания, %	Примерно 99 %	Примерно 99 %	Примерно 30 %

Можно говорить о том, что для текущего набора данных по временным показателям лучше себя показал алгоритм KNN. По точности и времени лучше себя показал алгоритм: дерева принятия решений.

4.6 Результаты опроса среди сотрудников ИТ компаний

4.6.1 Результаты опроса по гибридной методологии

Исследователи из Уппсальского университета проводили свое эмпирическое исследование с помощью опроса. В таблице 8 можно найти информацию о респондентах.

Таблица 8 – Информация о респондентах

Должность	Опыт работы	Локация
Проектный менеджер 1 (ПМ1)	15	Индия
Руководитель группы 1 (РГ1)	6	Индия
Член команды 1 (ЧК1)	3	Индия
Проектный менеджер 2 (ПМ2)	11	США
Руководитель группы 2 (РГ2)	7	США
Член команды 2 (ЧК2)	5	США
Проектный менеджер 3 (ПМ3)	9	Швеция
Руководитель группы 3 (РГ3)	6	Швеция
Член команды 3 (ЧК3)	5	Швеция

Все опрошенные заявили, что гибкость является основной причиной для принятия гибридной методологии. ПМ, РГ1, ЧК1 объяснили, что они приняли гибридную методологию из-за гибкости, прозрачности данных, потенциальных преимуществ, сокращения времени, тенденция рынка, требования пользователей.

В организации 2, ПМ2 объяснил, что они приняли гибридную гибкость, следование рыночным тенденциям, соответствие требованиям пользователей. В то время как ПМ3 сообщил что они приняли гибридную методологию для гибкости, потенциальной выгоды, сокращения времени.

Распределение соответствующей работы между членами проектной команды с учетом их сильных сторон может повысить эффективность работы и сократить время итераций. Такие аспекты эффективности, как приверженность, лидерство, точность информации, были повышены благодаря применению гибридного подхода. Таким образом, гибкость и адаптивность были определены как основная причина использования гибридного подхода к управлению проектами.

Существует множество способов сочетания различных этапов для

управления проектной работой. Задействованные фазы могут быть чисто водопадными или Scrum, но не сочетать оба варианта. Ниже приведены некоторые из способов комбинирования различные фазы при смешивании классической методологии и Scrum:

- инициация, планирование – по классической модели, а реализация, исполнение, закрытие в соответствии со Scrum;
- инициация, планирование, закрытие – по классической модели, а выполнение в соответствии со Scrum.

Анализ исследователей показал, что совмещение водопада и Scrum варьируется в зависимости от проекта и от того, как менеджер проекта хочет реализовать методологию в соответствии с требованиями проекта [71].

4.6.2 Результаты опроса по улучшенной гибридной методологии

Был проведен опрос среди проектных руководителей компании «НетКрэкер» в сфере общего руководства, разработке, контролю качества и бизнес анализа по оценке улучшенной гибридной методологии среди групп:

- сотрудники российского офиса компании, 89 % респондентов;
- сотрудники индийского офиса компании, 11 % респондентов;

На рисунке 14 показана диаграмма с процентным соотношением опыта работы в IT сфере респондентов.

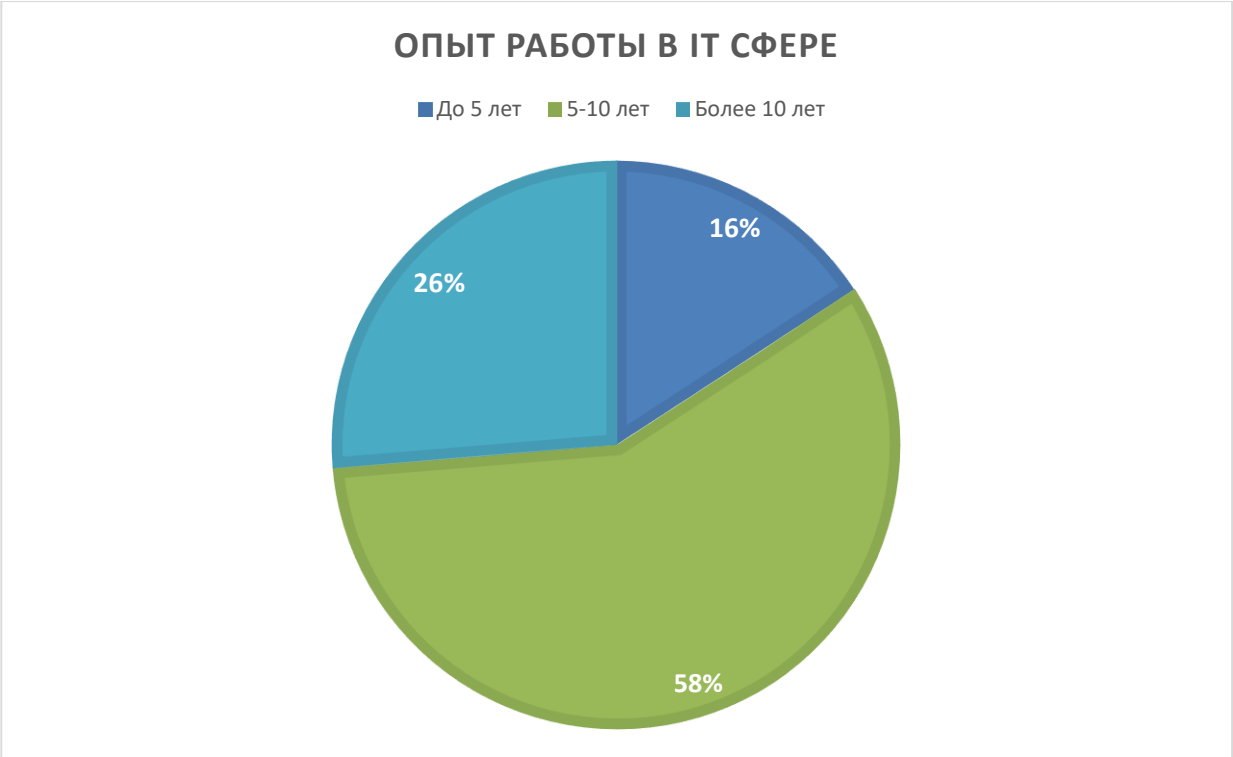

Рисунок 14 – Диаграмма результатов ответов по вопросу 1

После ознакомления с описанием модели улучшенной гибридной методологии были заданы вопросы на оценку улучшенных процессов и модели в целом.

На рисунке 14 можно заметить, что большая часть респондентов работает в индустрии 5-10 лет.

Результаты опроса по вопросу 2 «Согласны ли вы с утверждением: в рамках данной модели подход Lean Prioritization может способствовать успешной миграции ПО на микросервисную архитектуру» представлены на рисунке 15.

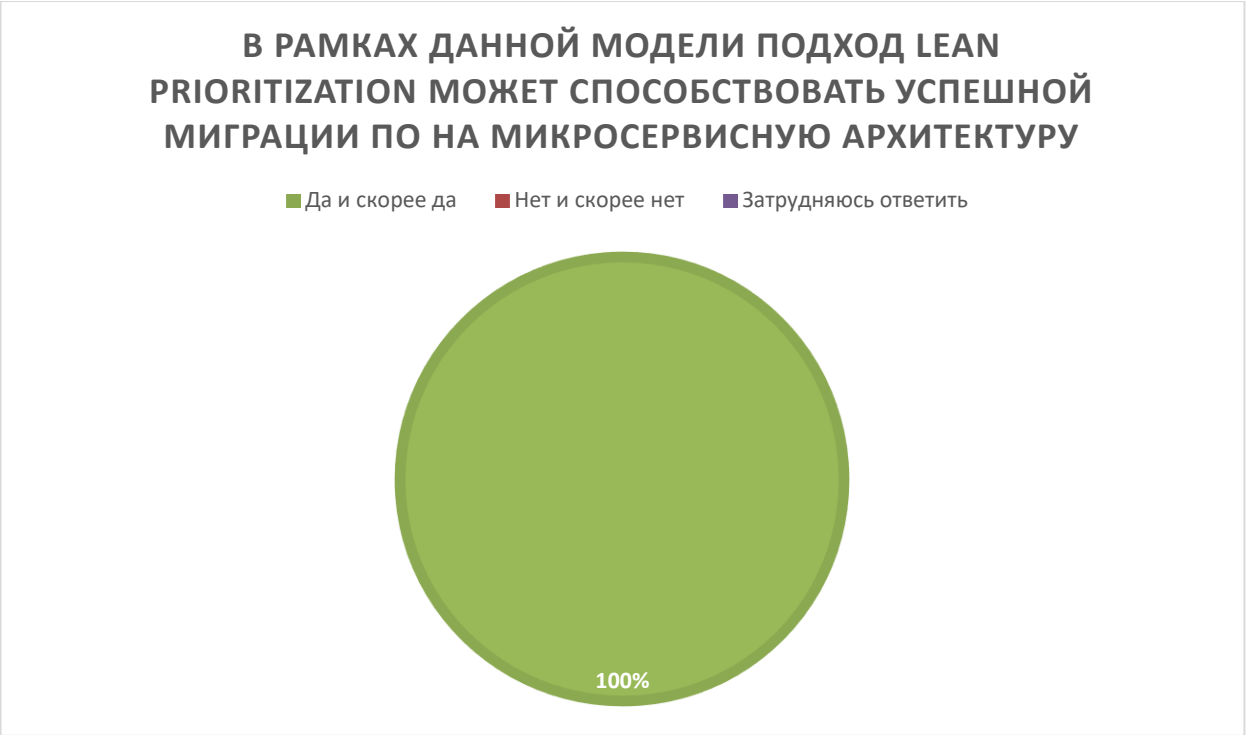

Рисунок 15 – Диаграмма результатов ответов по вопросу 2

На рисунке 15 можно заметить, что все респондентов согласны, что подход Lean Prioritization может способствовать успешной миграции ПО на микросервисную архитектуру.

Результаты опроса по вопросу 3 «Согласны ли вы с утверждением: в рамках данной модели алгоритмы машинного обучения может способствовать успешной миграции ПО на микросервисную архитектуру» представлены на рисунке 16.

Рисунок 16 – Диаграмма результатов ответов по вопросу 3

На рисунке 16 можно заметить, что большая часть респондентов согласны, что алгоритмы машинного обучения могут способствовать успешной миграции ПО на микросервисную архитектуру. Менее 20 % опрошенных ответили иначе.

Результаты опроса по вопросу 4 «Согласны ли вы с утверждением: в рамках данной модели подход бизнес ориентированной разработки через тестирование может способствовать успешной миграции ПО на микросервисную архитектуру» представлены на рисунке 17.

Рисунок 17 – Диаграмма результатов ответов по вопросу 4

На рисунке 17 видно, что большая часть респондентов согласны, что в рамках данной модели подход бизнес ориентированной разработки через тестирование может способствовать успешной миграции ПО на микросервисную архитектуру. Около 10 % опрошенных ответили иначе.

Результаты опроса по вопросу 5 «Согласны ли вы с утверждением: представленная модель может способствовать успешной миграции ПО на микросервисную архитектуру» представлены на рисунке 18.

Рисунок 18 – Диаграмма результатов ответов по вопросу 5

На рисунке 18 можно увидеть, что большая часть респондентов согласны, что Представленная модель может способствовать успешной миграции ПО на микросервисную архитектуру. Около 10 % опрошенных ответили иначе.

В целом респонденты выделяют достоинства данной модели:

– гибкое взаимодействие с заказчиком, быстрое обнаружение дефектов на ранних стадиях и как следствие экономия средств;
– планировать и мониторить проект может быть легче;
– позволяет быстро реагировать на изменения в требованиях и получать отзывы от заказчика;
– гибридная модель всегда работает лучше, поскольку мы берем лучшее из обеих моделей, шансы на успех будут выше.

Потенциальный недостаток: сроки не сдвигаются на первоначальном анализе. Таким образом, большая часть респондентов положительно отвечала на вопросы по оценке модели. Можно сказать, что модель может помочь с успешным переходом программного обеспечения на микросервисную архитектуру.

Заключение

В рамках данной магистерской диссертации проводилось исследование методологий разработки программного обеспечения в проектах миграции ПО телекоммуникационной компаний на микросервисную архитектуру. На основе изученной информации была выдвинута гипотеза о том, что миграция программного обеспечения на микросервисную архитектуру будет успешной, если использовать модель улучшенной гибридной методологии разработки.

В первой части была рассмотрена текущая ситуация на рынке телекоммуникационных компаний. Далее был проведен анализ монолитной архитектуры, выделены основные достоинства и недостатки внедрения предлагаемой микросервисной архитектуры, были рассмотрены результаты опросов компаний по внедрение такой архитектуры, а также рассмотрены конкретные примеры, такой как опыт компании «Мегафон».

После изучения различных источников, можно сделать вывод, что стоит переместить программное обеспечение на микросервисную архитектуру, если в дальнейшем требуется масштабирование системы, добавление новых функций и так далее. Организации, которые собираются перемещать свое решение на микросервисы нуждаются в правильном подходе по управлению таким проектом. В положительном результате компании могут получить простую в развертывании систему, которая будет проста в масштабировании.

Во второй части были проанализированы стандарты по управлению проектом и обеспечению качеством программного продукта, были выделены достоинства и недостатки.

В третьей части представлена модель улучшенной гибридной методологии разработки базировалась на основе классической водопадной модели и Scrum, а также на гибридной модели исследователей Абхинава Чандрабабу и Анушы Муддангула из Уппсальского университета.

Таким образом, была разработана модель улучшенной гибридной методологии со следующими этапами:

– первоначальное планирование;
– бизнес-анализ и проектирование;
– итерации (спринты), где команда проходит стандартные фазы, но работая в параллель. В процессе разработки и обеспечения качества ПО необходимо использовать подход непрерывной поставки и непрерывной интеграции, а также бизнес ориентированную разработку через тестирование. Для определения сроков выполнения задач и потенциальных рисков было предложено использовать дерево принятия решений;
– приемочное тестирование;
– внедрение.

В четвертой части были апробированы алгоритмы машинного обучения и сама модель улучшенной методологии. Были проанализированы варианты алгоритмов машинного обучения, выбрано дерево принятия решений с глубиной в три уровня, так как оно показало себя наиболее быстрым и точным на тестовой выборке.

Также проводился опрос среди проектных руководителей компании ООО «НетКрэкер» по оценке улучшенной гибридной методологии. В ходе анализа ответов этого опроса можно сказать, большая часть респондентов положительно отвечала на вопросы по оценке модели. Она может помочь с успешным переходом программного обеспечения на микросервисную архитектуру. Таким образом, результаты данного опроса подтверждают гипотезу исследования.

Список используемой литературы

1. 4 типа архитектуры программного обеспечения [Электронный ресурс] : Nuances of programming. URL: https://nuancesprog.ru/p/12019/ (дата обращения: 01.10.2021);
2. Веселова А.А. Проект внедрения онлайн-сервиса для обслуживания клиентов // Междисциплинарные исследования: опыт прошлого, возможности настоящего, стратегии будущего 2021. №2. С. 9-21. URL: https://cyberleninka.ru/article/n/proekt-vnedreniya-onlayn-servisa-dlya-obsluzhivaniya-klientov (дата обращения: 01.10.2022);
3. Внедрение TELE2 Россия [Электронный ресурс] : Бастион интегратор 2022. URL: https://bastion-integrator.com/blog/casetele2rus (дата обращения: 01.03.2022);
4. Все говорят о микросервисной архитектуре приложений. Чем она хороша и как на нее перейти? [Электронный ресурс] : ООО «Селдон Новости» 2021. URL: https://news.myseldon.com/ru/news/index/245517080 (дата обращения: 02.10.2021);
5. ГОСТ Р 54869-2011 Проектный менеджмент требования к управлению проектом: национальный стандарт Российской Федерации: дата введения 2012-09-01 / Федеральное агентство по техническому регулированию и метрологии. – Изд. официальное. – Москва: Стандартинформ, 2011. – 9 с;
6. ГОСТ Р 57193-2016 Системная и программная инженерия процессы жизненного цикла систем: национальный стандарт Российской Федерации: дата введения 2017-11-01 / Федеральное агентство по техническому регулированию и метрологии. – Изд. официальное. – Москва: Стандартинформ, 2016. – 98 с;
7. ГОСТ Р ИСО 21500-2014 Руководство по проектному менеджменту. – Москва: Изд-во стандартов, 2015. – 50 с.;

8. ГОСТ Р ИСО/МЭК 25010-2015 Системная и программная инженерия требования и оценка качества систем и программного обеспечения (SQuaRE). Модели качества систем и программных продуктов: национальный стандарт Российской Федерации: дата введения 2016-06-01 / Федеральное агентство по техническому регулированию и метрологии. – Изд. официальное. – Москва: Стандартинформ, 2015. – 36 с.;

9. Гранько О. Экстремальное программирование (XP) не для слабонервных [Электронный ресурс] : Worksection 2021. URL: https://worksection.com/blog/extreme-programming.html (дата обращения: 30.03.2022);

10. Деревья решений: общие принципы [Электронный ресурс] : Loginom 2022. URL: https://loginom.ru/blog/decision-tree-p1 (дата обращения: 14.11.2022);

11. Дроздов С.А., Луканина В.Е. Особенности проектирования серверного и клиентского программного обеспечения web-сайта с использованием rest-архитектуры // Вестник МГУП. №2. URL: https://cyberleninka.ru/article/n/osobennosti-proektirovaniya-servernogo-i-klientskogo-programmnogo-obespecheniya-web-sayta-s-ispolzovaniem-rest-arhitektury (дата обращения: 01.10.2021);

12. Ермаков А.А., Павлов И.А. Сравнительный анализ методов планирования // Наука, техника и образование. 2020. №2 (66). С. 10-16. URL: https://cyberleninka.ru/article/n/sravnitelnyy-analiz-metodov-planirovaniya (дата обращения: 03.10.2021);

13. Завертайлов В. Настольная книга project-менеджера. Что нужно знать, чтобы управлять IT, digital и другими проектами с учетом российских реалий / Завертайлов В.: Изд-во ООО «Эксмо», 2022. – 749 с. – ISBN 978-5-04-467417-2;

14. Как мы управляем проектами развития аналитической отчётности [Электронный ресурс] : ООО Хабр 2022. URL:

https://habr.com/ru/company/rostelecom/blog/558670 (дата обращения: 01.03.2022);

15. Качество [Электронный ресурс] : EMPII Group BV 2022. URL: https://prince2.wiki/ru/temy/kachestvo/ (дата обращения: 01.04.2022);

16. Кириллов Н. И. Преимущества и недостатки CRM систем с открытым исходным кодом // International scientific review. №5 (36). URL: https://cyberleninka.ru/article/n/preimuschestva-i-nedostatki-crm-sistem-s-otkrytym-ishodnym-kodom (дата обращения: 05.10.2021);

17. Классификатор kNN // ООО Хабр 2023. URL: https://habr.com/ru/post/149693/ (дата обращения: 03.03.2023);

18. Кугушева Д.С. Проектирование сложного программного обеспечения с использованием микросервисной архитектуры // Инновации и инвестиции. 2020. №5. С. 188-190. URL: https://cyberleninka.ru/article/n/proektirovanie-slozhnogo-programmnogo-obespecheniya-s-ispolzovaniem-mikroservisnoy-arhitektury (дата обращения: 05.10.2021);

19. Лаврищева Е. Программная инженерия. Парадигмы, технологии и case-средства 2-е изд., испр. и доп. Учебник для вузов / Лаврищева Е.: Изд-во ООО «ЛитРес», 2022. – 281 с. – ISBN 978-5-04-028330-9;

20. Лучшая архитектура для MVP: монолит, SOA, микросервисы или бессерверная? Часть 1 [Электронный ресурс] : ООО Хабр 2022. URL: https://habr.com/ru/company/otus/blog/476024/ (дата обращения: 03.11.2021);

21. Малиновская В.В. Методы управления проектами, актуальные для разработки программного обеспечения // SCIENCE TIME. С. 27-31. URL: https://cyberleninka.ru/article/n/metody-upravleniya-proektami-aktualnye-dlya-razrabotki-programmnogo-obespecheniya (дата обращения: 03.11.2021);

22. Малышева Л.А. Успешность, устойчивость и эффективность проектов [Электронный ресурс] : larisamalysheva.ru 2022. URL: https://larisamalysheva.ru/blog/uspeshnost_ustoichivost_i_effectivnost_proekta (дата обращения: 14.11.2022);

23. Метод PERT [Электронный ресурс] : forPM 2022. URL: https://forpm.ru/%D0%BC%D0%B5%D1%82%D0%BE%D0%B4-pert/ (дата обращения: 14.11.2022);

24. Методологии разработки программного обеспечения [Электронный ресурс] : ООО Хабр 2022. URL: https://habr.com/ru/sandbox/43802 (дата обращения: 01.05.2022);

25. Методология, фреймворк или стандарт проектного управления [Электронный ресурс] : ООО Хабр 2022. URL: https://habr.com/ru/post/547140/ (дата обращения: 01.05.2022);

26. Микросервисы (Microservices) [Электронный ресурс] : ООО Хабр 2006-2021. URL: https://habr.com/ru/post/249183/ (дата обращения: 03.10.2021);

27. Микросервисы: плюсы, минусы, когда и зачем внедрять [Электронный ресурс] : ООО Хабр 2022. URL: https://habr.com/ru/company/southbridge/blog/674600/ (дата обращения: 03.10.2022);

28. Моисеева Е.Н. Анализ причин перехода программного обеспечения телеком-операторов на микросервисную архитектуру // МОЛОДЕЖЬ. НАУКА. ОБЩЕСТВО – 2021. С. 233-237. URL: https://www.elibrary.ru/item.asp?id=50248583 (дата обращения: 11.01.2023);

29. Моисеева Е.Н., Ерофеева Е.А. Анализ методологий управления разработкой программного обеспечения телеком-операторов // Информационные технологии в моделировании и управлении: подходы, методы, решения. С. 370-375. URL: https://www.elibrary.ru/item.asp?id=49755582 (дата обращения: 03.12.2022);

30. Николаенко В.С. Превентивный риск-менеджмент в ИТ-проектах // Государственное управление. Электронный вестник. 2016. №55. URL: https://cyberleninka.ru/article/n/preventivnyy-risk-menedzhment-v-it-proektah (дата обращения: 03.12.2022);

31. Обзоры итогов спринтов в agile [Электронный ресурс] : Atlassian 2023. URL: https://www.atlassian.com/ru/agile/scrum/sprint-reviews (дата обращения: 03.02.2023);

32. Орлов С.А. Программная инженерия. Учебник для вузов. 5-е издание обновленное и дополненное. Стандарт третьего поколения / Орлов С.А.: СПб.: Питер, 2016. – 640 с. – ISBN 978-5-496 -01917-0;

33. Переход на микросервисы: опыт «М.Видео-Эльдорадо» и «МегаФона» [Электронный ресурс] : Mail.ru Cloud Solutions, 2021. URL: https://mcs.mail.ru/blog/perekhod-na-mikroservisy-opyt-m-video-eldorado-i-megafona

34. Планирование спринтов [Электронный ресурс] : Atlassian 2023. URL: https://www.atlassian.com/ru/agile/scrum/sprint-planning (дата обращения: 05.10.2022);

35. Преимущества и недостатки Oracle SQL – Вокруг-Дом – 2021 [Электронный ресурс] : univdesigntechnologies 2021. URL: https://ru.univdesigntechnologies.com/82-advantages-disadvantages-of-oracle-sql-17634 (дата обращения: 05.10.2021);

36. Преимущества и недостатки методологии Scrum в разработке сайтов и программного обеспечения [Электронный ресурс] : sonikelf 2022 URL: https://sonikelf.ru/preimushhestva-i-nedostatki-metodologii-scrum-v-razrabotke-sajtov-i-programmnogo-obespecheniya (дата обращения: 30.04.2022);

37. Простое руководство по UML-диаграммам и моделированию баз данных [Электронный ресурс] : Microsoft 2022. URL: https://www.microsoft.com/ru-ru/microsoft-365/business-insights-ideas/resources/guide-to-uml-diagramming-and-database-modeling (дата обращения: 03.12.2022);

38. Рекопф М. Автоматическое тестирование ПО [Электронный ресурс] : Atlassian 2023. URL: https://www.atlassian.com/ru/continuous-delivery/software-testing/automated-testing (дата обращения: 03.02.2023);

39. Репин В. Моделирование бизнес-процессов в нотации BPMN в Business Studio 5. Практическое руководство / Репин В.: Изд-во ООО «ЛитРес», 2022. – 105 с. – ISBN 978-5-04-435796-9;

40. Ретроспективы agile [Электронный ресурс] : Atlassian 2023. URL: https://www.atlassian.com/ru/agile/scrum/retrospectives (дата обращения: 03.02.2023);

41. Сарри М. TDD, исследования и профессионализм [Электронный ресурс] : Highload. 2021-2022. URL: https://highload.today/blogs/tdd-100/ (дата обращения: 30.10.2022);

42. Сертификация ПМ стандарт [Электронный ресурс] : ГК «Проектная практика» 2022. URL: https://pmpractice.ru/training/sertification/pm-standart/ (дата обращения: 25.03.2022);

43. Сертификация по методу PRINCE2 [Электронный ресурс] : ГК «Проектная практика» 2022. URL: https://pmpractice.ru/training/sertification/prince2/ (дата обращения: 25.03.2022);

44. Сертификация, оценка и развитие проектного управления в России [Электронный ресурс] : АНО «ЦОРПУ». URL: https://www.isopm.ru/ (дата обращения: 09.03.2022);

45. Случайный лес (Random Forest) [Электронный ресурс] : Елена Капаца 2022. URL: https://www.helenkapatsa.ru/sluchainyi-lies/ (дата обращения: 03.03.2023);

46. Специфика проведения тестирования по в kanban [Электронный ресурс] : TestMatick 2022. URL: https://testmatick.com/ru/spetsifika-provedeniya-testirovaniya-po-v-kanban/ (дата обращения: 01.04.2022);

47. Сурков А. Как устроена каскадная модель управления проектами [Электронный ресурс] : ProКачество 2023. URL: https://kachestvo.pro/kachestvo-upravleniya/proektnoe-upravlenie/kak-ustroena-kaskadnaya-model-upravleniya-proektami/ (дата обращения: 01.04.2022);

48. Телеком-компании и коронавирус: три действия для управления кризисом [Электронный ресурс] : РосБизнесКонсалтинг 1995-2021. URL:

https://pro.rbc.ru/demo/5e8dab359a79470cb8c55a65 (дата обращения: 01.10.2021);

49. Торосян Е.К., Шеховцова Е.И., Акопян А.А. Анализ методов сквозной автоматизации базовой архитектуры систем оператора связи // Петербургский экономический журнал. С. 172-177. URL: https://www.elibrary.ru/item.asp?id=38479424 (дата обращения: 01.03.2022);

50. Управление ИТ-проектами – 5 вызовов и их преодоление [Электронный ресурс] : «ИТ Гильдия» 2022. URL: https://it-guild.com/info/blog/upravlenie-it-proektami-5-vyzovov-i-ikh-preodolenie/ (дата обращения: 01.03.2022);

51. Управление ИТ-проектами: методологии, инструменты, специфика преодоление [Электронный ресурс] : ECM-Journal. URL: https://ecm-journal.ru/material/Upravlenie-IT-proektami-metodologii-instrumenty-specifika (дата обращения: 01.03.2022);

52. Управление операционной и проектной ИТ-деятельностью в их взаимосвязи на примере ИТ департамента ОАО «ВымпелКом» (БИЛАЙН) [Электронный ресурс] : FOSTAS Foundation. URL: http://www.fostas.ru/library/show_article.php?id=207 (дата обращения: 01.03.2022);

53. Управление проектами в компании: определение и решение ключевых проблем [Электронный ресурс] : Advanta 2022. URL: https://www.advanta-group.ru/blog/upravlenie-it-proektami/ (дата обращения: 02.03.2022);

54. Цепков М. Завершение спринта в Scrum – демо и ретро [Электронный ресурс] : vc.ru 2022. URL: https://vc.ru/hr/98403-zavershenie-sprinta-v-scrum-demo-i-retro (дата обращения: 29.11.2022);

55. Что такое DevOps? [Электронный ресурс] : Microsoft 2021. URL: https://azure.microsoft.com/ru-ru/overview/what-is-devops/#cloud (дата обращения: 05.10.2021);

56. Что такое Kanban? [Электронный ресурс] : Atlassian 2023. URL: https://www.atlassian.com/ru/agile/kanban (дата обращения: 03.02.2023);

57. Что такое Scrum? [Электронный ресурс] : Atlassian 2023. URL: https://www.atlassian.com/ru/agile/scrum (дата обращения: 03.02.2023);

58. Что такое канбан: принципы и преимущества [Электронный ресурс] : SendPulse 2022. URL: https://sendpulse.com/ru/support/glossary/kanban (дата обращения: 01.05.2022);

59. Шахина И.В., Муллин А. А., Алышев Ю.В. Agile vs Waterfall: разница между методологиями // StudNet. 2020. №6. С.9-14. URL: https://cyberleninka.ru/article/n/agile-vs-waterfall-raznitsa-mezhdu-metodologiyami (дата обращения: 07.10.2021);

60. Яковлева Е.А. Обзор современных систем контроля версий [Электронный ресурс] : Материалы IX Международной студенческой научной конференции «Студенческий научный форум» URL: https://scienceforum.ru/2017/article/2017030261 (дата обращения: 30.10.2022);

61. Agile или Waterfall – какой вариант соответствует вашему бизнесу [Электронный ресурс] : Worksection. URL: https://worksection.com/blog/waterfall-vs-agile.html (дата обращения: 07.10.2021);

62. IT-инфраструктура в контролируемых средах обеспечения [Электронный ресурс] : ООО Хабр 2022. URL: https://habr.com/ru/company/netcracker/blog/652109/ (дата обращения: 02.05.2022);

63. RUP. Общие сведения [Электронный ресурс] : Informicus 2022. URL: http://www.informicus.ru/default.aspx?SECTION=6&id=73&subdivisionid=7 (дата обращения: 19.03.2022);

64. Scrum: что это и зачем нужно [Электронный ресурс] : ScrumTrek 2022. URL: https://clck.ru/gvhBu (дата обращения: 01.04.2022);

65. 10 Steps for a Painless Software Upgrade [Электронный ресурс] : Datix 2022. URL: https://blog.datixinc.com/blog/software-upgrade (дата обращения: 01.05.2022);

66. 3 Top Project Management Methodologies You Should Know [Электронный ресурс] : Northeastern University 2022. URL: https://www.northeastern.edu/graduate/blog/project-management-methodologies/ (дата обращения: 03.05.2022);

67. A comparison of Extreme Programming and the Rational Unified Process from a practical perspective [Электронный ресурс] : Authorea 2022. URL: https://www.authorea.com/users/99017/articles/118850-a-comparison-of-extreme-programming-and-the-rational-unified-process-from-a-practical-perspective (дата обращения: 02.05.2022);

68. Agile Software Development [Электронный ресурс] : Tadviser 2021. URL: https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:Agile_software_development (дата обращения: 08.03.2022);

69. Allweyer T. BPMN 2.0 Introduction to the Standard for Business Process Modeling / Allweyer T.: Books on Demand. 2016. – 172 с. – ISBN 978-3-8370-9331-5;

70. Becker, Gregory M. A practical risk management approach / Becker, Gregory M. [Электронный ресурс] : Project Management Institute, Inc. 2021. URL: https://www.pmi.org/learning/library/practical-risk-management-approach-8248 (дата обращения: 08.10.2021);

71. Chandrababu A., Muddangula A., Adoption of Hybrid Methodology in projects [Электронный ресурс] : Uppsala University. URL: https://uu.diva-portal.org/smash/get/diva2:1393923/FULLTEXT01.pdf (дата обращения: 08.03.2022);

72. Fan, CF, Jindal, A, Gerndt, M, Microservices vs Serverless: A Performance Comparison on a Cloud-native Web Application [Электронный ресурс] : Proceedings of the 10th international conference on cloud computing and

services science (closer). URL: https://www.webofscience.com/wos/woscc/full-record/WOS:000615960500017 (дата обращения: 08.10.2021);

73. Fewell, J., 2017. Hybrid PM Approaches Can Sow Confusion; If Thoughtfully Developed, They Will Deliver Value [Электронный ресурс] : PM Network. URL: https://www.pmi.org/learning/library/hybrid-pm-approaches-can-sow-confusion-but-can-deliver-value-10850 (дата обращения: 18.09.2022);

74. Global Mobile Trends 2021 [Электронный ресурс] : GSMA Association 2021. URL: https://data.gsmaintelligence.com/api-web/v2/research-file-download?file=141220-Global-Mobile-Trends.pdf&id=58621970 (дата обращения: 01.03.2022);

75. How telecom operators can succeed in 2022 [Электронный ресурс] : Telecoms 2022. URL: https://telecoms.com/opinion/how-telecom-operators-can-succeed-in-2022/ (дата обращения: 02.03.2022);

76. Machine Learning [Электронный ресурс] : Cloud 2022. URL: https://sbercloud.ru/ru/services/machine-learning (дата обращения: 03.11.2022);

77. Mike Loukides, Steve Swoyer Microservices Adoption in 2020 / Mike Loukides, Steve Swoyer [Электронный ресурс] : O'Reilly Media, Inc 2021. URL: https://www.oreilly.com/radar/microservices-adoption-in-2020/ (дата обращения: 05.10.2021);

78. Monolithic applications [Электронный ресурс] : Microsoft 2021. URL: https://docs.microsoft.com/en-us/dotnet/architecture/containerized-lifecycle/design-develop-containerized-apps/monolithic-applications (дата обращения: 06.10.2021);

79. Pmp vs prince2: a combat of certifications [Электронный ресурс] : Vinsys 2020. URL: https://www.vinsys.com/blog/pmp-vs-prince2-a-combat-of-certifications/ (дата обращения: 05.03.2022);

80. PRINCE2 PMMM Case Study [Электронный ресурс] : ILX Group 2022. URL: https://www.prince2.com/eur/prince2-pmmm-case-study (дата обращения: 05.03.2022);

81. Project Management Institute A Guide to the Project Management Body of Knowledge (PMBOK Guide) – Seventh Edition and The Standard for Project Management 2021 / Project Management Institute . – 369 c. – ISBN 978-1-62-825680-2;

82. Salah, A., Ramadan, N., Ahmed, H., 2017. Towards a Hybrid Approach for Software Project Management using Ontology Alignment. / Int. J. Comput. Appl. 168, 12–19;

83. Sample Research [Электронный ресурс] : The Standish Group International 2022. URL: https://www.standishgroup.com/sample_research (дата обращения: 01.03.2022);

84. Schwaber K., Sutherland J. The 2020 Scrum GuideTM [Электронный ресурс] : ScrumGuides.org 2022. URL: https://scrumguides.org/scrum-guide.html (дата обращения: 03.11.2022);

85. What is Business-Driven Development | BDD? [Электронный ресурс] : Functionize 2023. URL: https://www.functionize.com/blog/business-driven-development-what-and-how-functionize-helps (дата обращения: 03.11.2022);

86. What is Test-Driven Development? [Электронный ресурс] : TestDriven Labs 2022. URL: https://testdriven.io/test-driven-development/ (дата обращения: 03.11.2022);

87. Why is project management so important for telcos? [Электронный ресурс] : Bilginc 2022. URL: https://bilginc.com/en/blog/5638/why-is-project-management-so-important-for-telcos (дата обращения: 01.03.2022).

Приложение А

Сравнительный анализ международных и российских стандартов управления проектами

Таблица А.1 – Сравнительный анализ стандартов в управлении проектами

Параметр	Стандарт в управлении проектами		
	PMBoK	PRINCE2	ГОСТ Р ИСО 21500-2014
Основное содержание стандарта	10 областей знаний: интеграция; содержание; сроки; стоимость; качество; человеческие ресурсы; коммуникации; риски; поставки; заинтересованные стороны	«7 тем: экономическое обоснование; организация; управление качеством; планы работ; анализ и управление рисками; управление изменениями содержания; принятие решений» [25]	«10 предметных групп: интеграция; заинтересованные стороны; содержание; ресурсы; сроки; стоимость; риски; качество; закупки; коммуникации» [25]
Область применения	проекты в различных отраслях	проекты в контролируемых средах	проекты в различных отраслях независимо от их размера и длительности
Принципы	ориентированность на выполнение требований заказчика; приводятся процессы, инструменты и методы; носит описательный и рекомендательный характер.	разделение проекта на управляемые и контролируемые стадии; управление одной стадией в один период времени; регулярная проверка экономического обоснования по ходу проекта; управление по исключениям; четкое описание и распределение ролей и обязанностей всей команды	по аналогии с PMBoK

Продолжение таблицы А.1

Параметр	Стандарт в управлении проектами		
	PMBoK	PRINCE2	ГОСТ Р ИСО 21500-2014
Принципы		адаптация в соответствии с проектной спецификой и средой; использование предыдущего опыта	
Инструменты управления	предоставляет расширенный список инструментов и методов по каждому процессу управления проектом	примеры не перечислены	примеры не перечислены
Использование в качестве руководства	рекомендации	руководством	руководство
Управление качеством проекта	планирование, обеспечение, контроль качества	определение качества, контроль качества, внешний контроль качества и улучшение качества.	по аналогии с PMBoK
Достоинства	ориентация на процессы; возможность управления проектом, программами и портфелями проектов через процессы; определение вводных ресурсов, инструментов, методик и результатов, в том числе и для подсистемы управления качеством проекта	гарантия ответственности дирекции за обеспечение выполнения работ и их качество; возможность применения для различных проектов; обеспечивает стандартный подход к менеджменту проектов, общую терминологию, контроль в использовании ресурсов и управлении качеством	описаны верхнеуровневые требования к руководителю проектов, его компетенциям и навыкам; идеологически аналогичен PMBoK, лаконичен

Продолжение таблицы А.1

Параметр	Стандарт в управлении проектами		
	PMBoK	PRINCE2	ГОСТ Р ИСО 21500-2014
Недостатки	сложность применения для небольших проектов; потребность в адаптации к области применения, размеру и сфере деятельности проекта, времени, бюджету и ограничениям по качеству	отсутствие регламентирования управления контрактами поставок, участниками проекта	«отсутствует полноценная структура процессов и документов; отсутствуют примеры конкретных инструментов; отсутствует четкое описание жизненного цикла проектов; отсутствует описание возможности интеграции с гибкими методологиями» [25]

Приложение Б

Сравнительный анализ методологий разработки ПО

Таблица Б.1 – Сравнительный анализ методологий разработки

Критерий	Традиционная (Waterfall) [71]	Scrum [36, 64,71]	Kanban [56,71]	eXtreme Programming [9, 67]	Rational Unified Process [63,67]	Гибридная методология исследователей из Уппсальского университета [71]
Развитие	жесткое	гибкое	гибкое	гибкое	гибкое с элементами традиционного подхода	гибкое с элементами традиционного подхода
Процесс	последовательный	итеративный	итеративный	итеративный	итеративный	итеративный
Документация	строгая	не строгая	не строгая	не строгая	строгая	не строгая
Контроль качества проекта	только конечный продукт	после каждого спринта	после готовности задачи [46]	во время разработки	после каждой итерации	после каждой итерации и перед внедрение
Активное участие заказчика (клиента)	не обязательно участвует	участвует	участвует	участвует	не обязательно участвует	не обязательно участвует
Прототип ПО	в конце разработки	после каждого спринта	после готовности задачи	после каждой итерации	после каждой итерации	после каждой итерации

Продолжение таблицы Б.1

Критерий	Традиционная (Waterfall) [71]	Scrum [36, 64,71]	Kanban [56,71]	eXtreme Programming [9, 67]	Rational Unified Process [63,67]	Гибридная методология исследователей из Уппсальского университета [71]
В каких ситуациях лучше применять	заказчик хорошо понимает, что хочет; заказчик не планирует участвовать в проекте после принятия задания; известны точные сроки и результаты каждой фазы; известны инструменты для разработки; команда уже создавала аналогичный проект	гибкое	гибкое	гибкое	гибкое с элементами традиционного подхода	гибкое с элементами традиционного подхода
Достоинства	устойчива к обновлению кадров благодаря очень подробному документированию каждого этапа;	возможность быстрого запуска проекта с наиболее приоритетными функциями и минимально	помогает быстро выявлять слабые места; наглядность продвижения работы	низкие накладным расходы; процесс может показать исключительную эффективность	снижение основных рисков заказчика, и разработчика; улучшение качества по за счет проверок	снижение основных рисков заказчика, и разработчика; улучшение качества по за счет многократных проверок; частые демонстрации

Продолжение таблицы Б.1

Критерий	Традиционная (Waterfall) [71]	Scrum [36, 64,71]	Kanban [56,71]	eXtreme Programming [9, 67]	Rational Unified Process [63,67]	Гибридная методология исследователей из Уппсальского университета [71]
Достоинства	четкого плана и последовательности этапов; заранее понятно, на каком этапе что будет происходить	возможным бюджетом; ежедневный контроль над ходом работ. частые демонстрации проекта заказчику				проекта заказчику
Недостатки	большое количество документов подробнейший план может создать проблемы при реализации и сдвиги в планировании, что негативно повлияет на мнение	присутствуют сложности при заключении договоров; большое количество	не подходит для долгосрочного планирования; приоритетность задач может	рассчитан только на команду опытных разработчиков, не разбитую на несколько частей	несогласованность решений; непродуктивные затраты ресурсов на переработку кода	нет рекомендаций по процессу установления приоритетов и планированию; нет рекомендаций по инженерным практикам

Продолжение таблицы Б.1

Критерий	Традиционная (Waterfall) [71]	Scrum [36, 64,71]	Kanban [56,71]	eXtreme Programming [9, 67]	Rational Unified Process [63,67]	Гибридная методология исследователей из Уппсальского университета [71]
Недостатки	заказчика о команде проекта			может потребовать значительного объема изменений в процесс е разработки; недостаточное документирование		

Приложение В

Блок схема разработанной модели

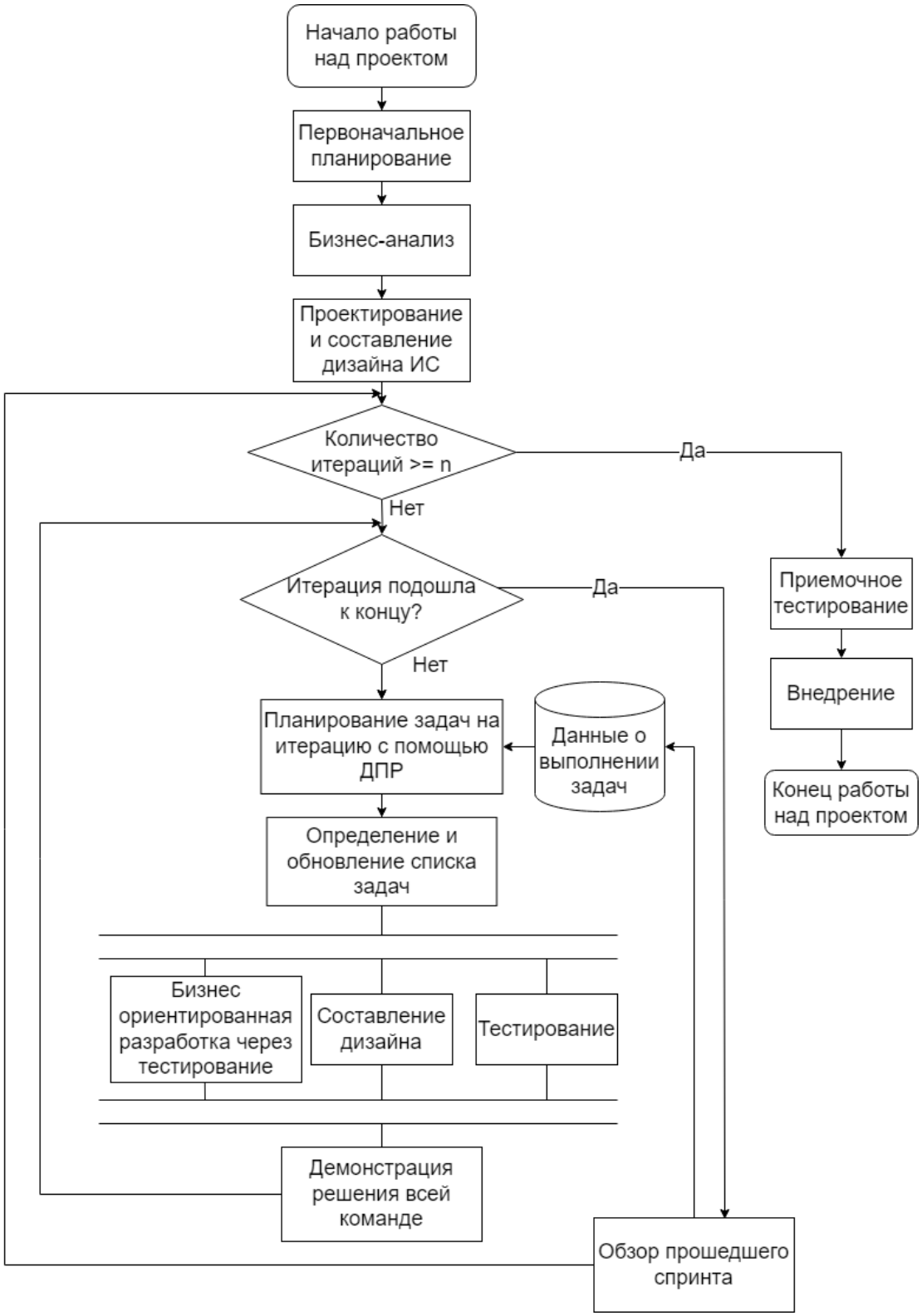

Рисунок В.1 – Блок схема модели улучшенной гибридной методологии разработки программного обеспечения