Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»

Институт экономики и управления

Кафедра анализа систем и принятия решений

ДОПУСТИТЬ К ЗАЩИТЕ ПЕРЕД ГЭК

Директор ШЭМ ИнЭУ
_____ ___Тургель И.Д.__
(подпись) (Ф.И.О.)

«01» июня 2024 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ)

Развитие инструментария цифровизации проектного управления на уровне сервисной архитектуры

Руководитель: Тарасьев Александр Александрович

Руководитель: Турыгина Виктория Федоровна

Студент группы ЭУМ-220001, Сычева Алена Андреевна

Екатеринбург
2024

РЕФЕРАТ

РАЗВИТИЕ ИНСТРУМЕНТАРИЯ ЦИФРОВИЗАЦИИ ПРОЕКТНОГО УПРАВЛЕНИЯ НА УРОВНЕ СЕРВИСНОЙ АРХИТЕКТУРЫ

ВКР (магистерская диссертация) состоит из введения, трех глав, заключения, библиографического списка, включающего 78 наименований, двух приложений. Работа включает 4 таблицы и 49 рисунков. Общий объем ВКР (магистерской диссертации) – 122 страницы.

Ключевые слова: цифровизация, сервисная архитектура, проектное управление, TOGAF 10, разработка ПО.

Цель исследования – повышение эффективности основных бизнес-процессов в организациях-разработчиках программного обеспечения.

Объектом исследования являются информационно-коммуникационные технологии проектного управления в организациях-разработчиках программного обеспечения.

Научная новизна заключается в разработке авторского алгоритма управления внесением изменений в исходный код проектов и автоматизации учета изменений для повышения эффективности бизнес-процессов проектного управления.

Практическая значимость работы заключается в разработке и внедрении инструментов цифровизации, которые могут быть применены в организациях-разработчиках ПО для повышения эффективности управления проектами, улучшения качества исходного кода и экономии ресурсов.

Эффективность рекомендаций – предложенный авторский алгоритм позволяет сократить временные издержки на сопровождение информационных систем, а также повышает эффективность бизнес-процессов за счет сокращение количества ошибок при работе с проектами. Экономический эффект составляет 710 452 рубля.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Тема развития инструментария цифровизации проектного управления особенно актуальна для российских компаний-разработчиков ПО. В России в данном секторе рынка сейчас наблюдается высокая конкуренция и быстрое развитие.

Пандемия COVID-19 и санкции ускорили процессы цифровизации во всех отраслях, что привело к увеличению спроса на IT-услуги и программное обеспечение. Компании стремятся адаптироваться к новым условиям, внедряя удаленные рабочие процессы, электронные сервисы и автоматизируя бизнес-процессы. Благодаря такой тенденции, на рынке создается дополнительный спрос на услуги организаций-разработчиков программного обеспечения.

Значительную роль в стимулировании развития цифровых технологий играет государственная программа «Цифровая экономика Российской Федерации», утвержденная постановлением Правительства РФ от 28 июля 2017 г. № 1632-р [40]. Программа направлена на создание условий для эффективного развития цифровой среды и внедрения современных технологий в различные отрасли экономики.

На данный момент одно из основных направлений развития – поддержка национальных IT-компаний с помощью различных программ субсидирования, налоговых льгот и других форм государственной поддержки. Например, Постановление Правительства РФ от 31 марта 2022 г. № 522 снижает требования к условиям, предъявляемым компаниям для получения субсидий и дотаций, если их деятельность направлена на разработку высокотехнологичного импортозамещающего товара [39].

Данные стратегические направления как никогда актуальны в условиях кризиса, санкций и большой неопределенности в высокотехнологической отрасли.

Стоит отметить, что данная тема актуальна как на уровне экономического развития страны, так и на уровне потенциального развития фирмы. В

сегодняшней гиперконкурентной среде компании стремятся производить новые и более качественные продукты, чтобы оставаться конкурентоспособными. Стратегия развития инструментария цифровизации проектного управления позволяет фирмам оставаться конкурентоспособными в своих сегментах рынка в краткосрочной перспективе, а также расширяет их возможности по созданию инновационных продуктов и процессов в долгосрочной перспективе.

Все вышеперечисленные факторы подтверждают высокую актуальность исследования. Таким образом, основная цель данной работы состоит в повышении эффективности основных бизнес-процессов в организациях-разработчиках программного обеспечения (далее ПО) за счет внедрения новых инструментов проектного управления.

Для достижения цели исследования были поставлены следующие задачи:

– анализ текущего состояния, разработка и внедрение изменений в исходном коде и процессах проектного управления в организациях-разработчиках ПО;

– проектирование сервисной архитектуры, соответствующей принципам TOGAF 10 для внедрения нового исходного кода;

– создание интерфейса и механизмов сервисной архитектуры, тестирование нового функционала и оценка его эффективности;

– анализ экономической эффективности изменений.

Объектом данного исследования являются информационно-коммуникационные технологии проектного управления в организациях-разработчиках программного обеспечения. Предметом являются процессы разработки и внедрения инструментария цифровизации проектного управления на основе сервисной архитектуры.

В ходе исследования использовались следующие методы:

– анализ и синтез существующих методологий и подходов к цифровизации процессов управления проектами;

– проектирование и моделирование архитектуры компании-разработчика ПО с использованием принципов TOGAF 10;

– программная реализация и тестирование разработанных решений;

– экономический анализ эффективности внедренных изменений.

Тема цифровизации процессов проектного управления широко освещена в научной литературе и практических руководствах. Однако применение модели TOGAF 10 в сочетании с разработкой авторских инструментов управления для организаций-разработчиков ПО является новаторским подходом, требующим дополнительных исследований и разработок. Научная новизна заключается в разработке авторского алгоритма управления внесением изменений в исходный код проектов и автоматизации учета изменений для повышения эффективности бизнес-процессов проектного управления.

Основные составляющие научной новизны:

– исходный код на C#, используемый для реализации проектов в рассматриваемой организации-разработчике ПО;

– процесс изменения исходного кода на C# путем выделения общих методов в отдельные классы;

– внедрение интерфейса «Свойство проекта» для управления изменениями и автоматического внесения их в код;

– разработка и внедрение системы аналитической отчетности на основе прямого взаимодействия с базой данных MySQL и автоматизации выгрузки данных с помощью макросов на C#.

Практическая значимость работы заключается в разработке и внедрении инструментов цифровизации, которые могут быть применены в организациях-разработчиках ПО для повышения эффективности управления проектами, улучшения качества исходного кода и экономии ресурсов. Полученные результаты могут быть использованы для улучшения процессов проектного управления в различных компаниях, что будет способствовать их конкурентоспособности на рынке.

Исследование базируется на анализе существующих методологий и практик цифровизации проектного управления, данных о текущем состоянии процессов управления в исследуемой организации, а также на эмпирических данных, полученных в ходе реализации и тестирования разработанных решений.

Данная работа включает в себя три главы. В первой главе проведён обзор теоретических основ цифровизации процессов проектного управления. Вторая глава включает в себя разработку и внедрение инструментария цифровизации проектного управления на основе сервисной архитектуры. В третьей главе экономически оценены результаты исследования. Заключение содержит выводы на основе полученных результатов.

1 ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ЦИФРОВИЗАЦИИ ПРОЦЕССОВ ПРОЕКТНОГО УПРАВЛЕНИЯ

1.1 Принципы и методологии проектного управления в сфере разработки ПО

Данный раздел посвящен анализу основных принципов и методов управления проектами в области разработки программного обеспечения. Гибкие и классические модели являются двумя основными типами методологий, используемых в современной практике.

Такие модели разработки ПО, как Waterfall и V-модели классифицируются как классические подходы. Они возникли в контексте более ранних практик и основываются на строгих, последовательных этапах разработки. Инкрементная, спиральная, итеративная и RAD-модели представляют собой комбинацию инкрементных и итеративных методов, которые отличаются от классических последовательных подходов, но не полностью соответствуют принципам гибких моделей Agile [21].

В быстро меняющихся условиях разработки программного обеспечения гибкие методологии появились как ответ на ограничения классических подходов. Общее признание они получили в начале 2000-х годов после публикации «Манифеста гибкой разработки программного обеспечения», который был создан группой разработчиков, стремившихся улучшить процессы разработки ПО [26, с.29]. Манифест содержит следующие основополагающие принципы:

– люди и взаимодействие важнее процессов и инструментов;

– работающее программное обеспечение важнее исчерпывающей документации;

– сотрудничество с заказчиком важнее согласования условий контракта;

– готовность к изменениям важнее следования плану [59].

Методология Agile имеет большое количество вариаций, одни из самых известных – Scrum, Kanban и экстремальное программирование (далее XP).

Фреймворки для управления проектами интегрируют принципы проектного управления в свою структуру и предлагают набор инструментов, задач и процессов, необходимых для организации и сопровождения проекта от начала до конца. Фреймворк создан с целью предоставить ясное и последовательное описание проекта, которое обеспечит надежное и повторяемое выполнение задач различными командами и организациями. Он представляет собой документацию, содержащую лучшие практики и рекомендации, доступные для широкого использования. Фреймворки также способствуют установлению общих стандартов внутри компаний и в целых отраслях [17].

Основные принципы и подходы к управлению проектами определяются методологией, а фреймворк предоставляет инструменты и рекомендации для реализации этих принципов на практике. Иными словами, методология объясняет, что нужно сделать, а фреймворк определяет, как это сделать [47].

Рассмотрим наиболее распространенные модели и фреймворки:

– Модель Code and Fix считается самым простым и первым методом разработки ПО. Эта модель предполагает непосредственное написание кода программы без предварительного планирования или разработки конкретной архитектуры. Основная идея модели Code and Fix заключается в том, что разработчики начинают писать код сразу после того, как поступают требования к программе. Они пишут код, исправляют ошибки, тестируют его и затем запускают. При обнаружении ошибок они вносят исправления и продолжают процесс до тех пор, пока система не будет соответствовать требованиям заказчика. Модель является очень простой в применении, но также обладает рядом серьезных недостатков. Во-первых, она не предусматривает никакого предварительного анализа требований или

планирования, что часто приводит к нерациональному использованию времени и ресурсов. Во-вторых, отсутствие структуры и архитектурного дизайна делает программный код сложным для понимания и поддержки, особенно на более поздних этапах разработки. Модель Code and Fix редко используется в современной практике разработки программного обеспечения. Однако она может быть полезной в небольших проектах, где требования ясны и стабильны, а также когда нужно быстро создать прототип или демонстрационную версию программы [65].

– Waterfall («водопад») – это одна из наиболее известных и классических моделей разработки программного обеспечения. Она представляет собой линейный и последовательный подход к разработке, в котором каждая фаза проекта следует за предыдущей, пропуск отдельного этапа и возврат на предыдущие стадии не предусмотрен. Каждый этап проекта должен быть завершен полностью, прежде чем начнется следующий. Каскадная модель применяется в разработке ПО для средних и больших проектов [61, с.39]. Процесс разработки при использовании каскадной водопадной модели выглядит как поток, последовательно проходящий фазы, указанные на Рисунок 1:

1) Анализ и определение требований проекта: этот этап начинается с определения требований к разрабатываемому программному продукту. Важно понять, что именно должен делать продукт и какие функции он должен включать. Требования формулируются в виде спецификаций, которые могут включать в себя функциональные, нефункциональные и технические требования. Далее осуществляется анализ требований. Цель – понять, каким образом требования будут реализованы в программном продукте.

2) Проектирование: в этой фазе определяется архитектура системы и ее компоненты. Принимаются решения о том, как будут взаимодействовать эти компоненты друг с другом. Проектирование включает создание диаграмм, схем и других документов, которые описывают структуру и детали системы.

3) Реализация: на этом этапе программисты начинают писать код, основываясь на проектировании и спецификациях. Они создают отдельные компоненты системы и интегрируют их в единый продукт. Этот этап включает в себя написание кода, тестирование отдельных модулей и их интеграцию в работающую систему.

4) Тестирование продукта: Программное обеспечение подвергается различным видам тестирования, чтобы убедиться, что оно работает правильно и соответствует требованиям. Тестирование включает в себя функциональное тестирование, тестирование производительности, тестирование безопасности и другие виды тестов.

5) Эксплуатация и поддержка: После успешного завершения тестирования программное обеспечение готово к внедрению. Оно устанавливается и запускается в рабочей среде. Внедрение может включать в себя обучение пользователей, настройку системы и переход к новой системе. После внедрения программное обеспечение продолжает поддерживаться и сопровождаться, что включает в себя исправление ошибок, внесение изменений и обновление системы. Цель – обеспечить длительное и стабильное функционирование программного продукта.

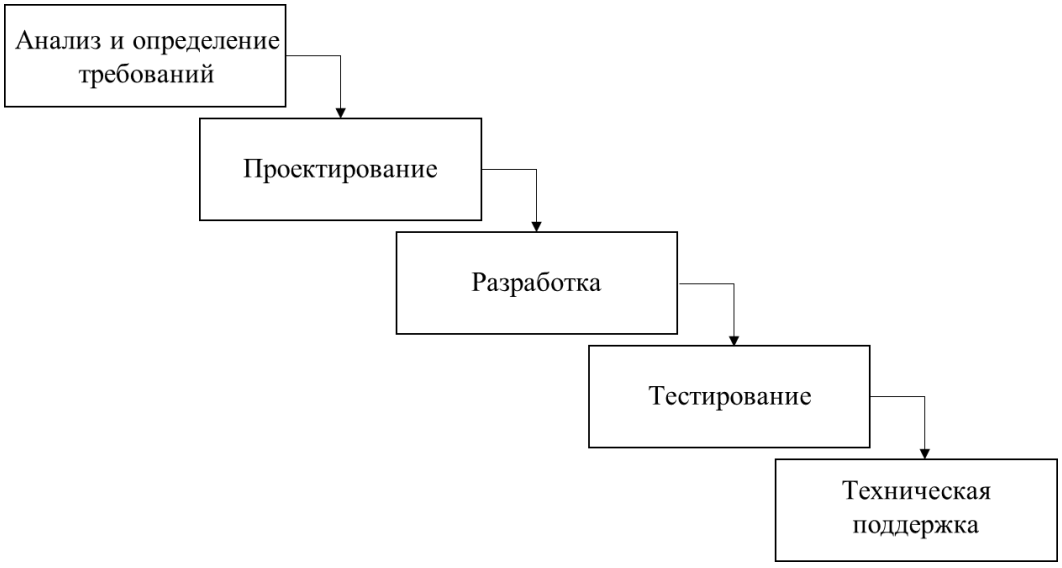


Рисунок 1 – Модель Waterfall[1]

[1] Составлено автором по: [61, с.40]

Основным недостатком модели Waterfall является то, что она не позволяет легко адаптироваться к изменениям в требованиях. В современных проектах изменения требований весьма распространены: в течение реализации они изменяются на 25-50% от первоначального вида. А водопадный подход предполагает, что все требования должны быть определены с самого начала и сохраняться в течение всего проекта. Основные преимущества и недостатки модели представлены в таблице 1.

Таблица 1 – Преимущества и недостатки модели Waterfall[2]

Преимущества	Недостатки
Легко понимаемая и простая модель	Плохо адаптируется к изменениям требований
Строгий контроль над процессом разработки: каждая фаза имеет четкие цели и роли.	Все требования должны быть известны и определены с самого начала.
Хорошо подходит для проектов с жесткими требованиями и стабильными условиями.	Не позволяет команде вернуться назад и повторить этап (итерацию).
Не требуется наличие опытного ИТ-персонала	Не просто адаптировать модель к различным типам проектов.

– Модель эволюционного прототипирования – это ещё одна вариация каскада. Пока проект проходит через традиционные фазы, прототип продукта пошагово дорабатывается на основе отзывов клиентов. Как правило, первый прототип не проходит приемочный тест, поэтому модель прототипирования включает в себя несколько прототипов [71]. Такой подход позволяет разработчикам извлекать уроки из каждого прототипа и применять полученные знания в будущих версиях. Схема модели представлена на Рисунок 2.

[2] Составлено автором по: [61, с.39]

Рисунок 2 – Модель эволюционного прототипирования[3]

Эволюционный подход к созданию прототипов – отличный выбор для исследовательских и опытно-конструкторских проектов, поскольку для таких проектов характерно наличие множества неизвестных требований. Кроме того, быстрое создание макетов компонентов системы для ознакомления пользователей позволяет получать своевременную обратную связь, которая может быть включена в следующий проект или прототип. Основные преимущества и недостатки модели представлены в таблице 2.

Таблица 2 – Преимущества и недостатки модели эволюционного прототипирования[4]

Преимущества	Недостатки
Заказчики могут видеть постоянный прогресс	В начале проекта невозможно определить точные временные затраты
Хорошо подходит для ситуаций, когда требования меняются быстро	Невозможно определить точное количество итераций
Раннее обнаружение проблем и недочетов позволяет снизить риски	Трудно составить точную смету затрат

[3] Составлено автором по: [61, с.40]
[4] Составлено автором по: [61, с.40]

– Спиральная модель жизненного цикла основана на классической каскадной модели с добавлением анализа рисков и итераций [61, с.41]. Спиральная модель подчеркивает необходимость возвращаться к более ранним этапам и повторять их несколько раз по мере продвижения проекта. Это серия коротких последовательных циклов, в ходе каждого из которых создается ранний прототип, представляющий собой часть всего проекта. Такой подход помогает продемонстрировать эффективность концепции на ранней стадии проекта и точно отразить сложность постоянно меняющихся технологий. Модель состоит из четырех основных частей (Рисунок 3) и изображает процесс в виде непрерывной петли, идущей изнутри наружу. Размер спирали отражает увеличивающийся объем выполняемой работы. Радиус спирали указывает на затраты.

Модель состоит из следующих этапов:

1) Определение целей (Identification of Objectives): на этом этапе определяются цели проекта, риски и ограничения. Ключевая цель - определить, что должно быть достигнуто и какие риски могут возникнуть при разработке.

2) Анализ и оценка (Risk Analysis and Evaluation): на этапе анализа и оценки рассматриваются альтернативные подходы к решению задачи, а также анализируются риски, связанные с каждым подходом. Производится оценка рисков и принятие решения о том, какие шаги предпринять для их смягчения.

3) Разработка и тестирование (Development and Testing): на этапе разработки и тестирования создается прототип или часть системы, основываясь на анализе и оценке рисков. Этот прототип тестируется, чтобы проверить его работоспособность и соответствие требованиям.

4) Оценка и планирование следующей итерации (Review and Planning of the Next Iteration): после завершения разработки и тестирования прототипа проводится оценка и анализ результатов. Затем определяются задачи для следующей итерации разработки, учитывая полученный опыт и обратную связь. Планируются дальнейшие шаги и стратегия разработки [20].

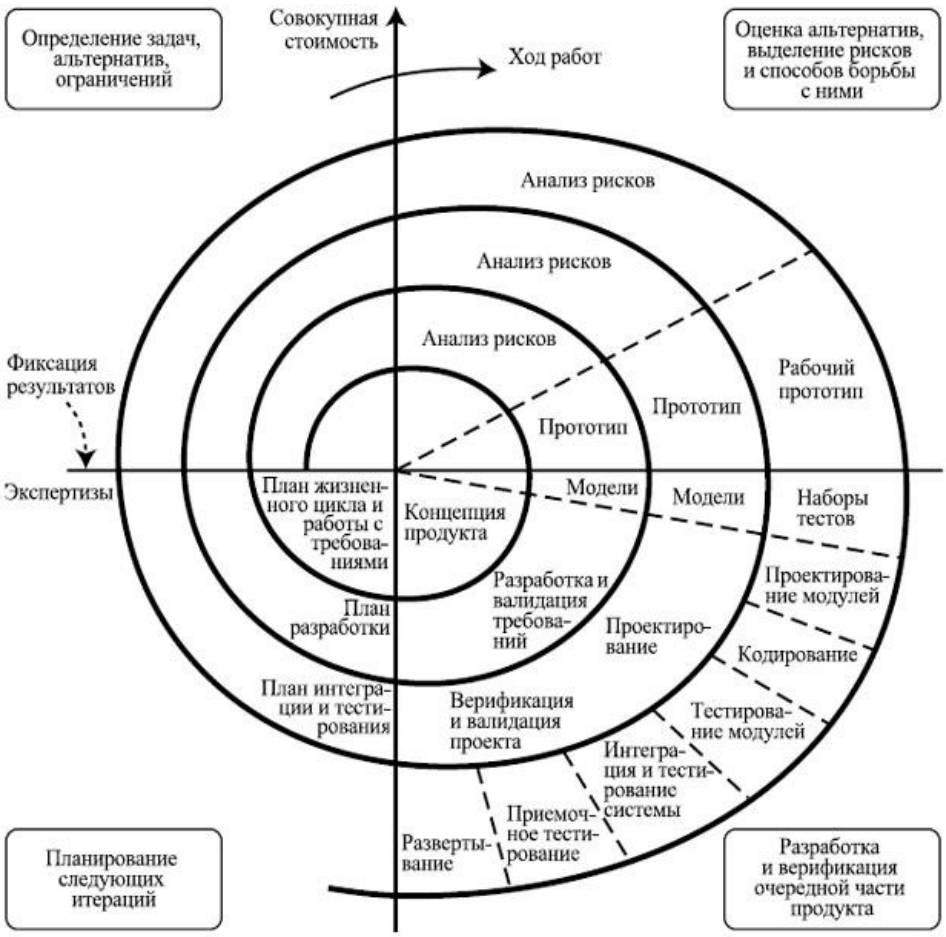


Рисунок 3 – Спиральная модель [69]

Поскольку спиральная модель обеспечивает более гибкий и адаптируемый подход к разработке ПО, она часто используется для сложных и масштабных проектов [48]. Модель также подходит для проектов, которым характерен высокий уровень неопределенности и риска. Можно выделить следующие достоинства спиральной модели:

1) спиральная модель позволяет быстрее показать пользователям системы работоспособный продукт, тем самым активизируя процесс уточнения и дополнения требований;

2) допускает изменение требований при разработке системы;

3) обеспечивает большую гибкость в управлении проектом;

4) позволяет получить более надежную и устойчивую систему, т. к. по мере развития системы обнаруживаются слабые места и ошибки исправляются на каждой итерации;

5) позволяет совершенствовать процесс разработки – анализ, проводимый в каждой итерации, позволяет провести оценку того, что должно быть изменено в организации разработки, и улучшить ее на следующей итерации;

6) уменьшаются риски заказчика, т. к. заказчик может с минимальными для себя финансовыми потерями завершить развитие не перспективного проекта [34].

У этой модели также есть несколько существенных недостатков:

1) увеличивается неопределенность в перспективах развития проекта (этот недостаток вытекает из предыдущего достоинства модели);

2) затруднены операции временного и ресурсного планирования всего проекта в целом (для решения этой проблемы необходимо ввести временные ограничения на каждую из стадий жизненного цикла; план составляется на основе статистических данных, полученных в предыдущих проектах, и личного опыта разработчиков) [70, с.47].

– Итеративные и инкрементальные модели разработки программного обеспечения представляют собой подходы, при которых процесс разработки осуществляется поэтапно, с повторяющимися циклами разработки. В этих моделях проект разбивается на небольшие части, называемые итерациями или инкрементами, которые поочередно разрабатываются, тестируются и внедряются. Каждая итерация или инкремент добавляет новый функционал или улучшает существующий. Разработка продукта происходит эволюционно, с постепенным улучшением и дополнением функциональности. В процессе разработки обеспечивается постоянная обратная связь с заказчиком или пользователями, что позволяет корректировать и улучшать продукт на следующих этапах. Итеративный процесс предполагает, что различные этапы работы не ограничены определенной последовательностью и могут повторяться в течение процесса разработки по мере необходимости, чтобы достичь желаемого результата (рРисунок 4).

Рисунок 4 – Итеративные или инкрементальные модели [72]

Вместе с гибкостью и возможностью быстро реагировать на изменения, итеративные модели привносят дополнительные сложности в управление проектом. Необходимо тщательно контролировать каждую итерацию, чтобы обеспечить выполнение требований и соблюдение графика. Повторение итераций может привести к увеличению общей стоимости проекта. Даже небольшие изменения в требованиях или дизайне могут потребовать дополнительного времени и ресурсов [57]. Поскольку итеративные модели предполагают постоянные изменения и дополнения к требованиям, существует риск увеличения объема работ и задержек в графике, особенно если изменения вносятся на поздних стадиях разработки.

– V-модель (также известная как «V-образная модель») является методологией разработки программного обеспечения, которая представляет собой альтернативу к классической модели Waterfall. Эта модель подразумевает последовательное выполнение шагов разработки, начиная с анализа и заканчивая тестированием, с каждым этапом, имеющим свой симметричный этап на стороне тестирования [73]. Следующая фаза начинается только после завершения предыдущей. В отличие от водопадной модели, которая имеет линейную структуру, V-модель подчеркивает параллельность между различными этапами разработки и тестирования. Такой

подход способствует раннему обнаружению и исправлению ошибок, улучшает контроль качества и уменьшает риски в ходе проекта.

Основные этапы V-модели включают:

- Спецификация требований: на этом этапе определяются и документируются требования к программному продукту.
- Анализ: требования анализируются и конкретизируются, определяются основные характеристики и функции продукта.
- Проектирование: проектируется архитектура системы и ее компонентов на основе выделенных требований.
- Кодирование: на этом этапе разработчики создают исходный код на основе проектных решений, сделанных на предыдущем этапе.
- Модульное тестирование: каждый модуль или компонент программы тестируется отдельно, чтобы убедиться в его правильной работе.
- Интеграция: модули собираются в единое целое и тестируются на предмет взаимодействия и совместимости.
- Системное тестирование: полная система тестируется в соответствии с требованиями, чтобы убедиться в ее соответствии спецификации.
- Валидация и верификация: этот этап включает в себя подтверждение того, что разработанный продукт соответствует исходным требованиям и ожиданиям заказчика [25].

Схематически V-модель представлена на Рисунок 5.

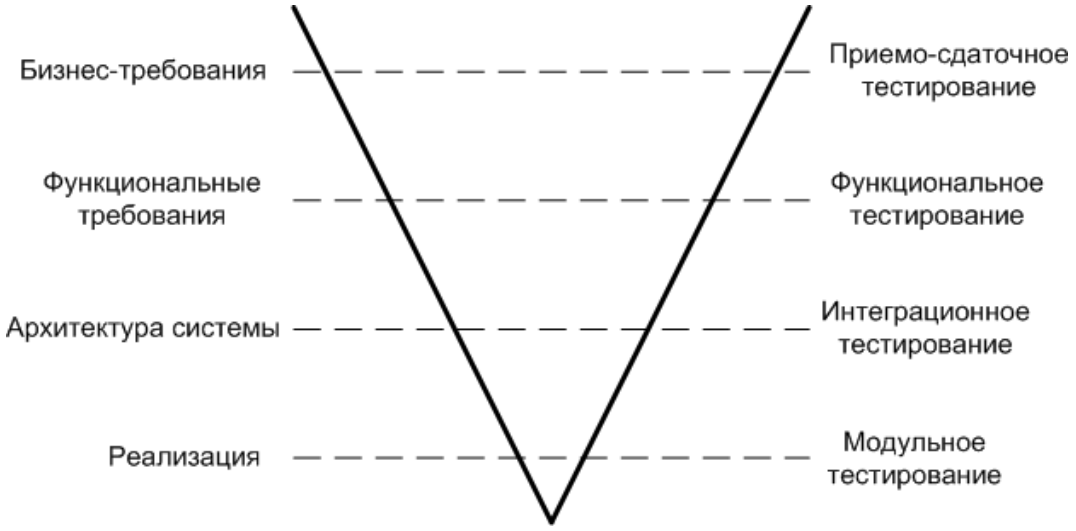


Рисунок 5 – V-модель процесса разработки ПО [73]

В настоящее время гибкие подходы являются более популярными в сфере разработки программного обеспечения. Это связано с рядом преимуществ, которые они предоставляют, таких как готовность к изменениям, акцент на коммуникацию и сотрудничество, а также возможность быстрой итеративной разработки. Кроме того, гибкие подходы позволяют быстрее реагировать на изменения требований клиента и рыночные условия, что делает их более привлекательными для современных компаний.

Agile – это тип процесса управления проектами, в основном используемый для разработки ПО, где требования и решения развиваются благодаря совместным усилиям самоорганизующихся и кросс-функциональных команд и их клиентов [6]. Методологии Agile, такие как Scrum, Kanban, Extreme Programming (XP) и др., предоставляют гибкие рамки для управления проектами. Они обеспечивают регулярные циклы разработки (итерации), частую обратную связь и корректировку процессов в зависимости от изменяющихся требований и условий. Agile также способствует повышению мотивации команды, улучшению качества продукта и увеличению клиентской удовлетворенности [65].

Рассмотрим более подробно гибкие методологии разработки:

– Scrum-фреймворк был разработан в 1990-х Кеном Швабером (Ken Schwaber) и Джефом Сазерлендом (Jeff Sutherland). Впервые он был формализован в 1995 году для решения проблем, связанных с комплексностью, присущей продуктовой разработке и разработке программного обеспечения. На данный момент Scrum успешно применяется для решения комплексных проблем в самых разнообразных сферах [77].

Процесс разработки разбивается на отдельные этапы, результатом каждого из которых является готовый продукт (Рисунок 6). В конце каждого этапа (в терминологии Scrum – спринта) готовый продукт предоставляется заказчику [46]. Полученный от заказчика отзыв позволяет выявить возможные проблемы или пересмотреть некоторые аспекты первоначального плана. Таким образом, Scrum позволяет наилучшим образом следовать принципам Agile-разработки. Главным принципом является выполнение задач за счет совместных усилий команды [18]. Однако нужно понимать, что для успешной реализации Scrum требуется не только глубокое понимание методологии самой командой, но и поддержка со стороны руководства и изменение корпоративной культуры.

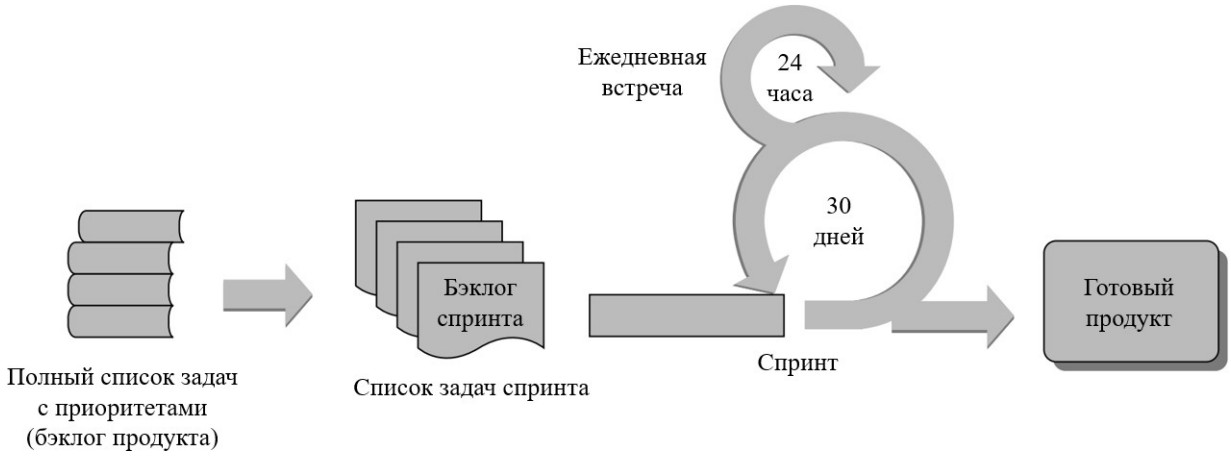


Рисунок 6 – Scrum-методология[5]

5 Составлено автором по: [61, с.45]

– Kanban является еще одним из возможных способов реализации гибкой методологии разработки. Данный метод управления процессами пришел из производства, а затем был адаптирован для разработки программного обеспечения. Основной идеей Kanban является визуализация рабочего процесса с помощью доски Kanban, на которой задачи представлены в виде карточек и перемещаются по колонкам в зависимости от статуса выполнения (Рисунок 7). Kanban-доска позволяет команде видеть текущее состояние работы над проектом, от задач, находящихся в очереди, до выполненных. Доску можно настроить под цели и структуру любых задач и команды [61, с.47]. Kanban не требует радикальных изменений в организации или процессах. Он может быть внедрен постепенно, позволяя команде приспосабливаться к новой системе.

В отличие от Scrum, методология Kanban не требует явных правил или ролей, что может привести к недостаточной структурированности процесса. Также Kanban может быть менее эффективен для проектов с высокой степенью неопределенности или срочности, где требуется более строгий контроль и планирование [29].

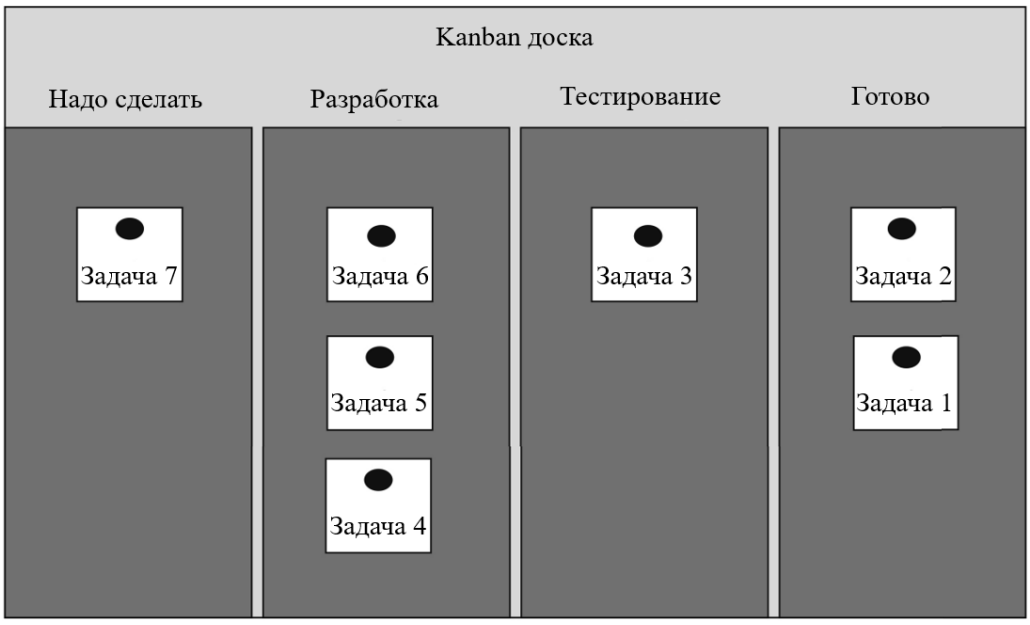


Рисунок 7 – Пример Kanban доски[6]

[6] Составлено автором по: [61, с.47]

– Экстремальное программирование (XP) – это методология разработки программного обеспечения, ориентированная на гибкость, скорость и обеспечение высокого качества продукта. XP было разработано в 1990-х годах и акцентирует внимание на командной работе, постоянной обратной связи и быстром реагировании на изменения в требованиях. Данная методология хорошо применима для маленьких и средних команд (от 2 до 10 человек), в процессе разработки которых присутствует неопределенность или потребность в решении быстро меняющихся требований [2]. XP уделяет большое внимание тестированию. Фактически, программисты должны писать тесты одновременно с написанием производственного кода. Поскольку тестирование интегрировано в процесс сборки, в результате получается высокостабильный и расширяемый продукт. Важное требование данной модели: оперативная качественная командная работа при написании программы и высокий профессионализм сотрудников [63].

– Методология Rational Unified Process (RUP) – это методология разработки программного обеспечения, разработанная компанией Rational Software. RUP обеспечивает набор итеративных процессов разработки ПО, ориентированных на управление изменениями и рисками в проектах. Разработка ведется поэтапно, с четкими итерациями, каждая из которых представляет собой полный цикл разработки и внедрения (Рисунок 8). Особое внимание уделяется архитектуре системы на ранних этапах процесса разработки. RUP состоит из набора предопределенных ролей, задач и артефактов, которые могут быть адаптированы под конкретные потребности проекта. RUP определяет четкие роли внутри команды разработки, такие как аналитик, разработчик, тестировщик и т.д., и определяет их обязанности и взаимодействие. Преимущества RUP включают в себя структурированный подход к разработке, эффективное управление изменениями и рисками, а также акцент на архитектуре и качестве продукта. Однако, недостатками

могут быть сложность и высокая стоимость внедрения полного набора процессов и ролей RUP [3, 4].

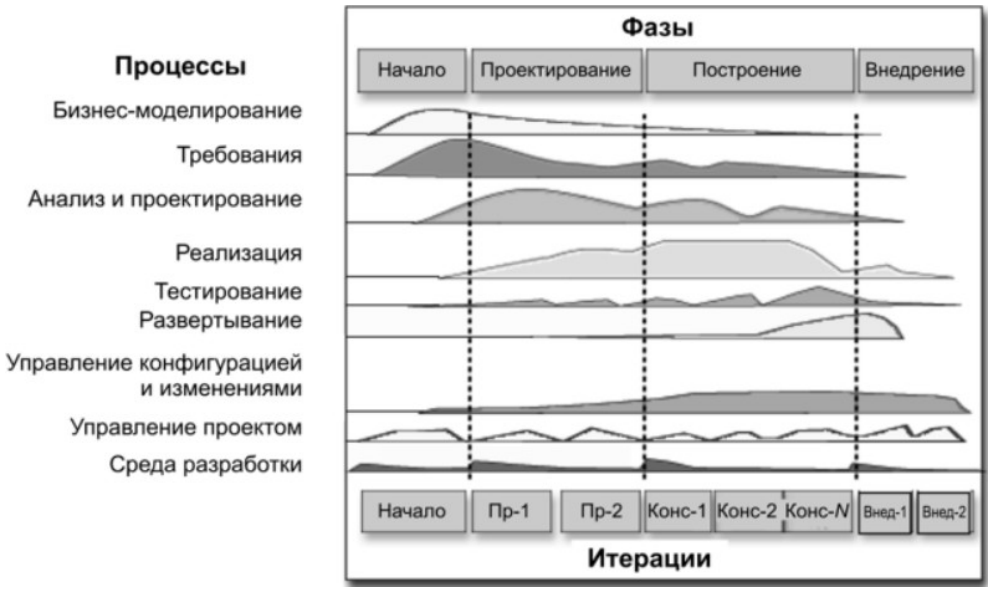


Рисунок 8 – Методология RUP [5]

Проанализировав основные принципы и методологии проектного управления в сфере разработки ПО, можно сделать вывод, что гибкие методологии способствуют более быстрой доставке ценности для заказчика, благодаря коротким итерациям разработки и постоянной обратной связи. Они позволяют лучше адаптироваться к изменениям в требованиях и внешней среде. Гибкие методологии способствуют улучшению коммуникации в команде и с заказчиком, что снижает риск возникновения проблем на поздних этапах разработки.

Однако выбор методологии разработки зависит от конкретных требований проекта, его контекста и особенностей команды. Гибкие методологии обычно предпочтительны для проектов, где требуется высокая гибкость и быстрая реакция на изменения, в то время как классические методологии могут быть более подходящими для проектов с четко определенными и стабильными требованиями.

1.2 Обзор существующих инструментов и подходов к цифровизации в проектном управлении

После изучения принципов и методологий проектного управления в разработке программного обеспечения становится ясно, что успешная реализация проекта требует не только понимания основных концепций управления, но и эффективного использования инструментов и подходов, способствующих его цифровизации. В современном мире компании стремятся не только соблюдать стандарты проектного управления, но и внедрять инновационные методы, направленные на оптимизацию процессов и повышение эффективности.

Цифровизация экономики становится все более значимым аспектом современного мира, оказывая существенное влияние на различные сферы жизни общества и бизнеса. Существует множество подходов к определению цифровой экономики в широком смысле. Наиболее обобщенным определением цифровизации является замещение реальных оценок происходящих событий – виртуальными, сводящимися к числовым значениям различных сфер общественной жизни [6]. Этот процесс направлен на оптимизацию процессов, улучшение качества услуг, повышение производительности и обеспечения конкурентоспособности как отдельных компаний, так и стран в целом.

Цифровизация охватывает различные аспекты экономики, включая производство, образование, здравоохранение, государственное управление, финансовый сектор, транспорт и торговлю [7]. Она изменяет способы взаимодействия между людьми, предприятиями и государственными учреждениями, а также формирует новые бизнес-модели и стратегии развития.

Процесс цифровизации можно разделить на три основных компонента: движущие силы цифровизации, объекты цифровизации и влияние цифровизации. На Рисунок 9 представлена взаимосвязь между технологическими тенденциями, цифровизацией и управлением проектами.

Рисунок 9 – Взаимосвязь между технологическими тенденциями, цифровизацией и управлением проектами[7]

Основными драйверами цифровизации являются:

– интернет и цифровые технологии;

– большие данные и аналитика;

– искусственный интеллект и машинное обучение;

– интернет вещей (IoT);

– цифровые платформы и рынки [68].

Данные технологические достижения и их применение все чаще становятся неотъемлемой частью продуктов и процессов организации. Растущая зависимость от технологий приводит к значительным социальным и корпоративным изменениям.

Также к движущим силам цифровизации можно отнести:

– изменение поведения, взглядов и ожиданий людей (чем чаще люди используют технологии, тем больше они ожидают от них, что меняет их взгляды на технологические тренды);

– низкие барьеры входа (в виду быстрых темпов развития, технологии становятся все дешевле и доступнее) [55].

Что касается влияния цифровизации, то оно весьма разнообразно как на уровне государства, так и на уровне организации.

[7] Составлено автором по: [68]

На сегодняшний день, цифровизация в России играет ключевую роль в развитии экономики, общества и управления [36]. Рассмотрим динамику позиции страны во всемирном рейтинге цифровой конкурентоспособности (Рисунок 10).

Рисунок 10 – Динамика позиции России во всемирном рейтинге цифровой конкурентоспособности по индексу и субиндексам за 2017 – 2021 года [12].

Индекс во всемирном рейтинге цифровой конкурентоспособности – это инструмент, используемый для оценки уровня цифровой развитости страны или региона относительно других стран. Он представляет собой комплексный показатель, учитывающий различные аспекты цифровизации, такие как доступ к цифровым технологиям, качество цифровой инфраструктуры, уровень цифровой грамотности населения, уровень цифровой экономики и другие [58]. Индекс обычно составляется на основе данных и показателей, собранных из различных источников, таких как статистические отчеты, опросы населения, данные о цифровых инвестициях и т. д. Затем эти данные анализируются и используются для расчета комплексного показателя, отражающего уровень цифровой конкурентоспособности.

Как можно увидеть по графику, позиция России в рейтинге за пять лет не изменилась, несмотря на колебания за этот промежуток времени. Как в 2017, так и в 2021 году страна занимает 42 место среди 60 стран, хоть и значение самого индекса уменьшилось (индекс России в 2017 году – 62,854, в 2021 году – 60,271). Позиция по субиндексу «Знания в области ИКТ» не изменилась. Также страна поднялась на пять позиций по субиндексу «Готовность к будущему» и опустилась на четыре позиции по субиндексу «Условия развития технологий».

Подпункт «д» пункта 2 Указа Президента РФ от 21 июля 2020 г. № 474 содержит основные целевые показатели цифровой трансформации, которые позволяют успешно достичь соответствующей национальной цели к 2030году. Одними из основных целевых показателей являются:

– достижение цифровой зрелости ключевых отраслей экономики и социальной сферы, в том числе здравоохранения и образования, а также государственного управления;

– доля массовых социально значимых услуг, доступных в электронном виде, равна 95%;

– доля домохозяйств, имеющих широкополосный доступ к Интернету, равна 97%;

– размер вложений в российские IT-решения должен быть в четыре раза больше по сравнению с их размером в 2019 году [11].

Ввиду текущей внешней политики, одним из основных направлений цифровизации стало обеспечение технологического суверенитета путем импортозамещения ключевых цифровых решений. Постановление Правительства РФ от 22 августа 2022 г. № 1478 требует, чтобы государственные органы, государственные учреждения, российские юридические лица и индивидуальные предприниматели, которым принадлежат информационные системы (КИИ) на праве собственности, аренды или на ином законном основании, перешли на использование

отечественного ПО. Этот процесс должен быть завершен к 1 января 2025 года. [12].

Плановые показатели программы были перевыполнены к концу 2022 года. Доля массовых социально значимых услуг, доступных в электронном виде, достигла 99,97%, что превышает план на 34,97 процентных пункта (план: 65%). Процент домохозяйств с доступом к широкополосному Интернету составил 86,1%, что на 6,1 процентного пункта больше, чем запланированное значение в 80%. Объем вложений в отечественные IT-решения составил 521,9 млрд рублей, что превышает плановый уровень на 1,4% (план: 156%). Уровень достижения цифровой зрелости составил 65,8%, что на 9,6 процентных пункта превышает запланированный уровень в 56,2%. Такое опережение планов стало возможным благодаря существенному расширению видов и объемов государственной поддержки IT-отрасли в 2021 году, что говорит о сильном влиянии цифровизации на внутреннюю и внешнюю экономику страны [14].

На уровне организации цифровизация влияет на способы проектного управления; на продукты и услуги, которые предлагает организация и на ее бизнес-модели. Цифровизация проектного управления включает в себя применение современных методологий, инструментов и подходов для улучшения процессов управления проектами.

Наиболее часто в современной практике стали использовать гибридные модели управления проектами. Гибридные методики представляют собой комбинацию различных подходов и методологий к управлению проектами. Они объединяют в себе элементы как классических, так и гибких подходов с целью адаптации к конкретным требованиям и условиям проекта [5]. Гибридные методологии адаптируются к конкретным потребностям и характеристикам проекта. В зависимости от типа проекта, его сложности, требований заказчика и других факторов, команды могут выбирать сочетание методов, которое наилучшим образом подходит для достижения поставленных целей. Данные методологии в контексте цифровой трансформаций являются наиболее эффективным в отраслях с высокой

степенью бюрократизации. С одной стороны, такой подход позволяет интегрировать проекты в существующие бизнес-процессы компании, а с другой стороны, обеспечивает гибкость в реализации проектов, учитывая необходимость адаптации к изменениям. Некоторые из популярных гибридных методологий: Scrumban, PRINCE2 Agile, P3.express, Scaled Agile Framework (SAFe) [72]. Также часто гибридные методологии создаются отдельными организациями «под себя», в том числе и за счет соединения моделей проектного управления с моделями управления стартапами, такими как бережливый стартап, развитие потребителей, трекинг [52].

Помимо гибридных методологий также набирают популярность и инновационные подходы к управлению проектами, такие как:

– Design Thinking (дизайн-мышление) – это методология, которая акцентирует внимание на понимании потребностей пользователей и на процессе поиска креативных инновационных решений, которые станут конкурентным преимуществом компании [56]. В контексте проектного управления данная методология может использоваться для решения различных задач, начиная от определения стратегии проекта и анализа его целей и задач, и заканчивая созданием инновационных решений и разработкой конкретных продуктов или услуг [38]. Процесс дизайн-мышления обычно состоит из пяти этапов, таких как определение проблемы, исследование и понимание потребностей пользователей, генерация идей, прототипирование и тестирование решений (Рисунок 11). Гибкость и итеративность – ключевые принципы методологии, что позволяет быстро адаптироваться к изменяющимся условиям и требованиям.

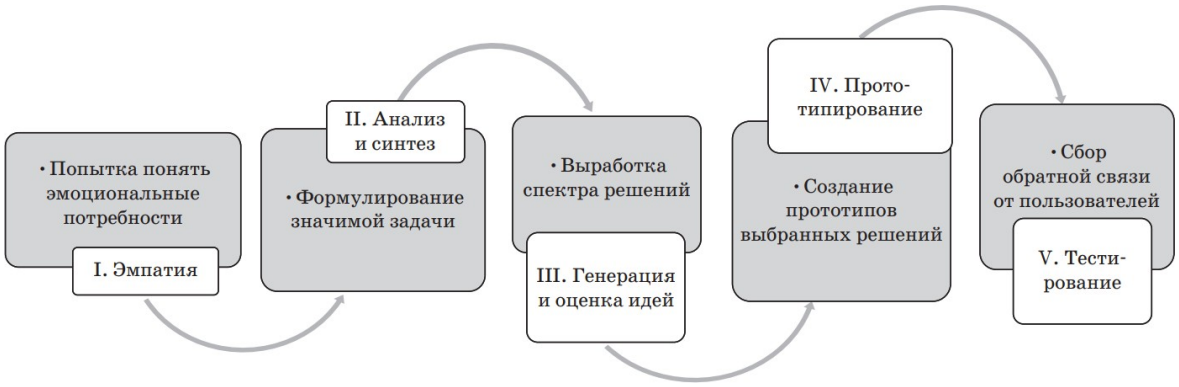


Рисунок 11 – Этапы дизайн-мышления [38, с.67]

– Lean – это философский принцип управления, ориентированный на создание максимальной ценности для клиента при минимальных затратах ресурсов (принцип бережливого производства). Подход был изобретен японской компанией Toyota Production System. Основная цель Lean – это предоставление продуктов или услуг, которые действительно нужны клиенту. Любая деятельность, которая не несет ценности клиенту, является потерей: избыточные запасы продукции, время простоя оборудования, излишние логистические перемещения и т. д. Основная идея заключается в том, чтобы максимально сократить или устранить эти потери. Эта цель достигается с помощью минимизации остановок в производственном процессе, постоянной оптимизации процессов, обеспечения управленцев качественной информацией, изменений в образе мышления и поведении сотрудников [31].

– DevOps – технология (методология) активного взаимодействия разработчиков со специалистами по информационно-технологическому обслуживанию для повышения эффективности, скорости и безопасности разработки программного обеспечения по сравнению с традиционными практиками [19]. Термин появился из слияния двух слов: development (Dev) и operations (Ops), то есть «разработка» и «администрирование». Помимо прямого общения специалистов по разработке и специалистов по эксплуатации, DevOps подразумевает общую ответственность, единые показатели успешности разработки и максимальную автоматизацию с помощью новых инструментов и особых подходов к работе. На рисунке 12

показано, как этапы жизненного цикла DevOps связаны друг с другом. Бесконечность символизирует необходимость постоянного сотрудничества и итеративного улучшения на протяжении всего жизненного цикла. Жизненный цикл DevOps состоит из восьми этапов, представляющих процессы, необходимые для разработки (выделены розовым) и администрирования (выделены желтым). На каждом этапе команды сотрудничают и общаются, чтобы поддерживать согласованность, скорость и качество.

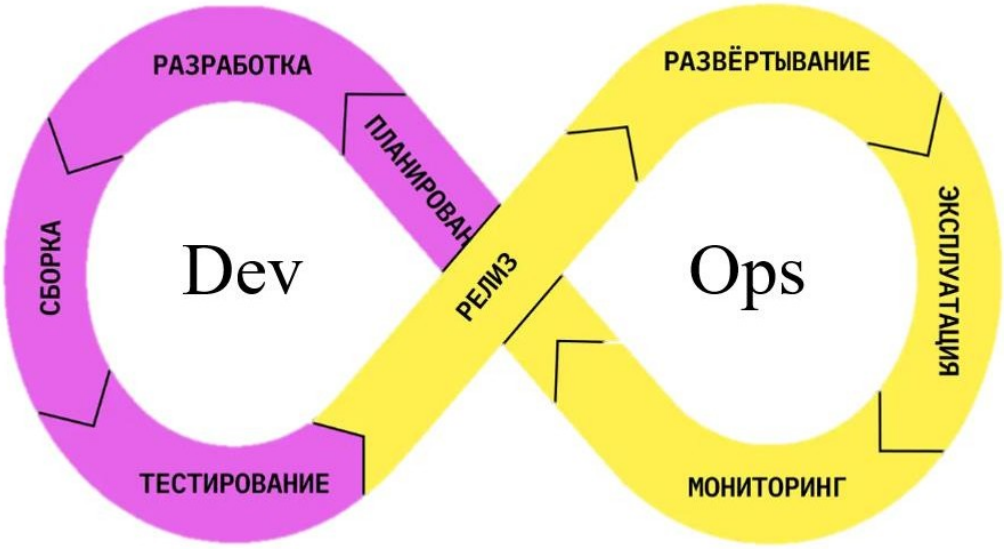


Рисунок 12 – DevOps pipeline[8]

– Design Sprint – это инновационный подход к разработке продуктов, который помогает командам быстро прототипировать и проверять новые идеи перед тем, как вложить большие ресурсы в их разработку. Данный подход был адаптирован командами UX-разработчиков под компанию Google Ventures, он базируется на методологии Agile. Процесс Design Sprint обычно продолжается пять дней и включает в себя последовательные этапы: определение проблемы, исследование и понимание потребностей пользователей, генерация идей, создание прототипов и тестирование с пользователем. Этот метод позволяет сэкономить время и ресурсы,

[8] Составлено автором по: [30]

предотвратить ошибки на ранних этапах разработки и обеспечить создание продукта, который действительно соответствует потребностям пользователей [64].

Цифровая трансформация изменила привычные методы и инструменты управления проектами, открывая новые возможности для компаний. Еще одним ключевым инструментом трансформации современных бизнес-процессов стало использование искусственного интеллекта (ИИ) и анализа данных. Автоматизация рутинных задач (обработка обращений клиентов, планирование ресурсов, анализ конкурентов и т.д.) с помощью искусственного интеллекта освобождает время управленцев для решения более стратегически важных задач. Прогнозирование результатов проектов на основе анализа больших объемов данных помогает компаниям предвидеть возможные риски и принимать взвешенные решения.

Рассмотренный обзор существующих инструментов и подходов к цифровизации в проектном управлении позволяет сделать вывод о значительном влиянии цифровой трансформации на эффективность и результативность управления проектами на уровне организации.

Усиление автоматизации процессов, внедрение современных методологий, интеграция цифровых инструментов и применение аналитики данных становятся неотъемлемой частью современной практики проектного управления, что способствует повышению эффективности, сокращению временных издержек, уменьшению количества ошибок и улучшению коммуникации внутри компании.

Более того, влияние цифровой трансформации на уровне страны способствует развитию экономики, улучшению условий для бизнеса и привлечению инвестиций. Внедрение современных цифровых технологий и подходов в проектное управление создает основу для инноваций,способствует принятию обоснованных решений и повышает общую конкурентоспособность страны на мировой арене. Однако успешная цифровизация требует не только технических изменений, но и культурных.

Организации должны быть готовы к изменениям в образе мышления, внедрению новых методологий и усилению цифровой компетенции сотрудников.

1.3 Модель TOGAF 10: особенности, преимущества и возможности применения в проектном управлении

Чтобы эффективно реализовать цифровые инициативы и обеспечить их успешное внедрение внутри компании, необходимо иметь четкую архитектурную основу, которая обеспечит согласованность и стабильность в проектной деятельности. Архитектура предприятия (Enterprise Architecture) – это концептуальная структура и набор методов, которые описывают текущее и желаемое состояние организации в рамках ее бизнес-стратегии, процессов, информационных потоков и технологической инфраструктуры. Она представляет собой целостную модель, объединяющую бизнес-аспекты (как организация функционирует), информационные аспекты (как данные используются и передаются) и технологические аспекты (какие системы и ресурсы используются) [67].

Существуют различные методики построения и оценки архитектуры предприятия, они задают классификацию основных областей архитектуры, описание используемых стандартов, моделей и процессов [16]. В качестве примеров можно указать следующие методологии:

- Zachman Framework;
- COBIT;
- методика Garther и др.

В данном разделе рассматривается модель TOGAF 10 (The Open Group Architecture Framework), она представляет собой комплексный набор методов и средств для разработки, описания, внедрения и управления архитектурой информационных систем. Первоначальный фреймворк был разработан в 1995

году международной ассоциацией The Open Group, которая объединяет ведущие компании и организации со всего мира. По данным ассоциации, в 2016 году этот фреймворк использовали 80% компаний из списка Global 50 и 60% компаний из списка Fortune 500, такие как Boeing, Procter & Gamble, Bank of America, Shell, Walmart и др. [78].

История развития TOGAF насчитывает несколько версий, привнесших значительные улучшения и расширения функциональности. TOGAF 10 является последней версией, выпущенной в 2022 году. Она предоставляет пользователям обширный набор инструментов и методов для эффективной работы с архитектурой информационных систем [74]. В десятой версии TOGAF особое внимание уделяется цифровой трансформации и гибким средам, документация расширяется и приобретает модульный формат для упрощения внедрения фреймворка в бизнес. The Open Group также в данной версии удалила весь избыточный и устаревший контент. TOGAF 10 привносит не сложные обновления, чтобы изменения можно было внедрять по мере необходимости, не нарушая передовой опыт.

Теперь структура разбита на два основных модуля: «TOGAF Fundamental Content» и «TOGAF Series Guides».

Модуль «TOGAF Fundamental Content» включает документацию по следующим темам:

- введение и основные понятия;
- методы разработки архитектуры (ADM);
- технологии ADM;
- применение ADM;
- архитектурный контент;
- возможности и управление корпоративной архитектурой.

Модуль «TOGAF Series Guides» охватывает такие темы, как:

- создание команды по архитектуре предприятия;
- архитектура безопасности;
- бизнес-архитектура;

- архитектура данных и информации;
- гибкие корпоративные цифровые технологии;
- микросервисная и сервис-ориентированная архитектура;
- адаптация ADM.

TOGAF определяет несколько архитектурных видов (Рисунок 13) для создания полной и всесторонней архитектуры предприятия, такие как:

- архитектурное видение (этот компонент определяет ключевые заинтересованные стороны и основные проблемы, которые нужно решить);
- бизнес-архитектура (описывает бизнес-стратегию, организационную структуру и ключевые бизнес-процессы);
- архитектура данных (описание структуры и организации данных предприятия, а также методы и стандарты управления данными);
- техническая архитектура (аппаратное и программное обеспечение, а также сетевые ресурсы и их взаимосвязи для поддержки бизнес-процессов);
- архитектура приложений (описывает структуру и взаимосвязи между приложениями, включая их функции, интерфейсы и взаимодействие с другими компонентами);
- возможности и решения (определение возможностей для реализации архитектуры и создание дорожной карты для перехода от текущего состояния к целевому состоянию);
- план миграции (разработка детальных планов и графиков для внедрения архитектурных решений);
- архитектура безопасности (включает в себя политики, процедуры и механизмы безопасности для защиты информации и ресурсов предприятия).

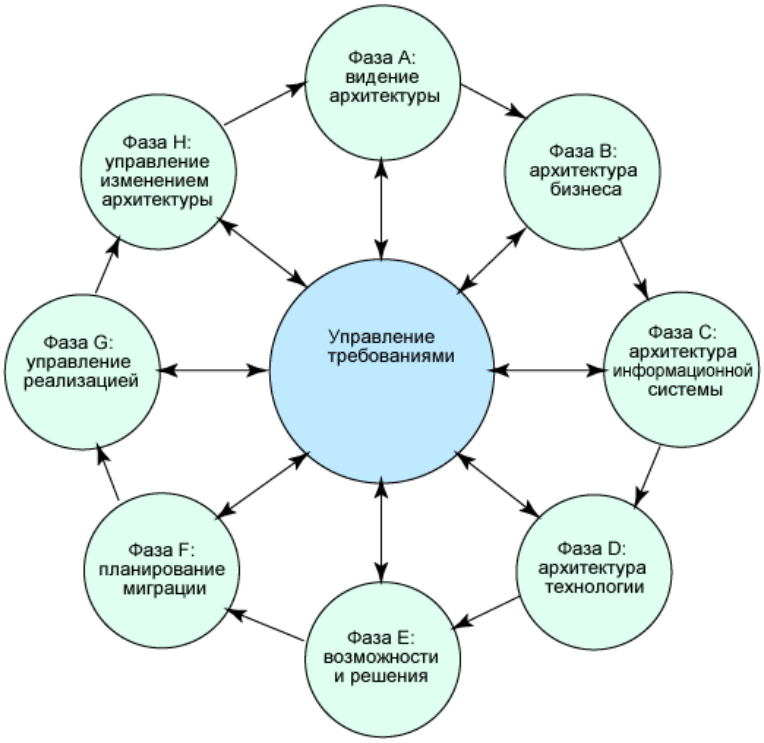


Рисунок 13 – Компоненты архитектуры предприятия по TOGAF 10 [49]

Каждый из этих видов архитектуры поддерживается соответствующими артефактами, такими как диаграммы, модели, документация и т. д. Основными принципами TOGAF, которые должны быть соблюдены при разработке архитектуры предприятия, являются: открытость, поддержка бизнес-целей, модульность, доступность, сопровождаемость и стандартизация [74].

Применение TOGAF 10 в проектном управлении обеспечивает ряд значительных преимуществ:

- стандартизация процессов архитектурного проектирования, что позволяет улучшить согласованность и совместимость между различными проектами в организации;
- улучшенное планирование и управление благодаря подробным рекомендациям и инструментам для оценки текущего состояния архитектуры, определения потребностей бизнеса и разработки планов архитектурного развития;
- интеграция с другими широко распространенными стандартами, такими как ArchiMate, BPMN и ITIL, что обеспечивает совместимость фреймворка с другими методологиями;

– TOGAF 10 позволяет описывать архитектуру на разных уровнях абстракции, что обеспечивает гибкость в анализе и проектировании, а также позволяет архитекторам создавать модели на уровне бизнес-архитектуры, информационной архитектуры, технической архитектуры и других;

– применение TOGAF 10 позволяет организациям создавать подробные модели и документацию, которые помогают повысить прозрачность проекта и обеспечить более эффективный контроль над его выполнением;

– снижение рисков и издержек, оптимизация расходов на разработку и внедрение архитектурных решений, благодаря инструментам для анализа и управления рисками архитектурных проектов;

– ориентация на поддержку цифровой трансформации организаций, предоставление инструментов и методик для адаптации архитектуры к быстро меняющимся потребностям и возможностям бизнеса.

TOGAF 10 предоставляет структурированный подход к разработке архитектуры предприятия, который может быть использован при планировании проектов. Фреймворк позволяет проектировать архитектуру предприятия с учетом бизнес-требований и стратегических целей организации. Его гибкость и комплексный подход позволяют эффективно планировать, разрабатывать и управлять архитектурными аспектами проектов, обеспечивая высокий уровень координации и согласованности.

1.4 Сервисная архитектура как основа для цифровизации процессов: роль и место в системе управления проектами

Одним из ключевых аспектов успешной цифровизации является не только выбор правильной методологии, но и правильное построение архитектуры информационной системы, которая будет поддерживать эти методологии и обеспечивать эффективное взаимодействие между различными компонентами проекта.

С каждым годом все большее число компаний, в том числе российских, переходит на использование сервисного подхода и сервисно-ориентированной архитектуры. Среди этих компаний Amazon Web Services (AWS), Netflix, Яндекс, Сбер и др. Они достигли такого высокого уровня капитализации благодаря созданию уникальной экосистемы и эффективному использованию разнообразных цифровых активов, которые объединены в единую цифровую платформу.

Сервисная архитектура представляет собой методологию проектирования информационных систем, основанную на концепции о создании независимых и взаимодействующих сервисов [8]. В отличие от традиционных монолитных систем, сервисная архитектура строится вокруг множества малых, автономных компонентов, которые называются сервисами или микросервисами (Рисунок 14). Сервисная и микросервисная архитектуры представляют собой подходы к построению и организации информационных систем, которые разделяют функциональность на небольшие, независимые и взаимодействующие сервисы. Однако есть различия между этими двумя подходами. Сервисная архитектура обычно описывает систему, состоящую из отдельных сервисов, каждый из которых может быть крупным и содержать множество функций. Эти сервисы часто разрабатываются и масштабируются как единое целое [13]. Микросервисная архитектура подразумевает создание множества небольших и легковесных микросервисов, каждый из которых отвечает только за определенную функциональность. Микросервисы более гибкие и могут быть масштабированы независимо друг от друга [10].

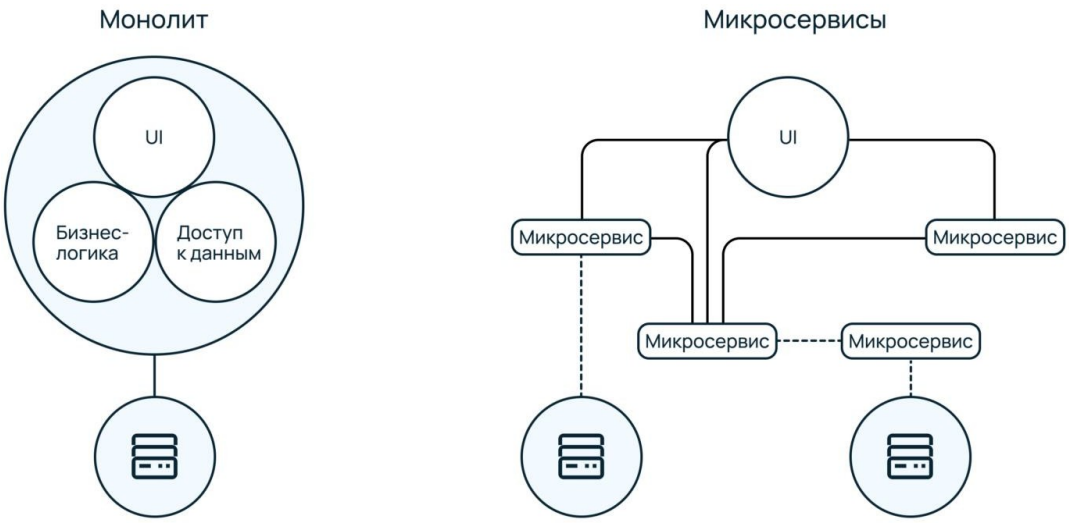


Рисунок 14 – Схематическое представление сервисной архитектуры и монолитной системы [54]

Каждый сервис выполняет ограниченное количество функций и предоставляет их через четко определенные интерфейсы. Сервисы могут быть развернуты и масштабированы независимо друг от друга, что обеспечивает гибкость и масштабируемость всей системы. Основные принципы сервисной архитектуры включают [60]:

– Модульность: система разбивается на независимые сервисы, что позволяет изолировать функциональность и упрощает разработку, тестирование и обновление.

– Стандартизация интерфейсов: каждый сервис предоставляет унифицированный интерфейс для взаимодействия с другими компонентами системы. Это позволяет обеспечить совместимость и легкость интеграции.

– Автономность: каждый сервис может быть развернут и масштабирован независимо от других, что повышает гибкость и надежность системы.

– Использование стандартных протоколов и форматов данных: для обеспечения совместимости и упрощения интеграции сервисы взаимодействуют с использованием широко распространенных протоколов и форматов данных, таких как HTTP, JSON или XML.

– Распределенность: сервисы могут быть размещены на различных серверах и даже в разных географических областях, что позволяет создавать гибкие и отказоустойчивые системы.

– Легкая интеграция: стандартизированные интерфейсы и модульная структура сервисной архитектуры облегчают интеграцию новых компонентов и взаимодействие с внешними системами.

Благодаря стандартизированным интерфейсам и протоколам связи, сервисы могут взаимодействовать между собой без внесений больших изменений в систему. Это особенно важно в контексте цифровизации проектного управления, где часто приходится интегрировать системы управления проектами, облачные сервисы, аналитические инструменты. Кроме того, сервисная архитектура способствует повышению гибкости информационной системы. За счет модульной структуры, новые функции и сервисы могут быть легко добавлены или изменены без необходимости полного пересмотра всей системы. Такая структура позволяет быстро реагировать на изменяющиеся потребности бизнеса и рынка, а также внедрять инновационные решения с минимальными затратами на разработку и внедрение.

Еще одним важным аспектом сервисной архитектуры является ее способность обеспечивать высокую доступность и надежность системы. За счет распределения функционала между независимыми сервисами и использования механизмов обеспечения отказоустойчивости, система может продолжать работать даже при сбое отдельных компонентов. Это особенно важно для проектного управления, где даже небольшие перерывы в работе системы могут привести к серьезным проблемам и временным издержкам.

Сервисно-ресурсная модель – новое направление развития сервисного подхода, представляет собой концепцию, описывающую взаимосвязь ресурсов при их предоставлении конечному потребителю [8]. Она учитывает

различные слои архитектуры предприятия и архитектурные объекты цифровой платформы.

Ключевым аспектом этой модели является соглашение об уровне сервиса (SLA), которое выступает связующим звеном между различными слоями архитектуры предприятия и объектами цифровой платформы. SLA – это формальное соглашение между поставщиком услуг и потребителем, определяющее ожидания и обязательства поставщика услуг перед заказчиком, включая качество обслуживания, доступность, производительность и другие параметры. Данное соглашение позволяет установить четкие критерии и ожидания по обслуживанию и использованию ресурсов [27].

Сервисные архитектуры, управляемые соглашениями, позволяют:

– гарантировать качество обслуживания (SLA определяюткключевые метрики производительности и качества сервисов, что позволяет обеспечить соответствие ожиданиям пользователей и бизнес-потребностям);

– оптимизировать ресурсы (SLA позволяют достичь оптимальной производительности и эффективности при минимальных затратах);

– обеспечивать надежность и отказоустойчивость за счет распределенной архитектуры и механизмов обеспечения отказоустойчивости;

– повышать гибкость и масштабируемость за счет модульной структуры сервисной архитектуры [35].

Однако нужно учитывать, что с увеличением числа сервисов в системе возрастает сложность их управления и мониторинга. Компаниям необходимо разрабатывать эффективные инструменты для управления конфигурацией, мониторинга производительности и выявления проблем. Также внедрение сервисной архитектуры требует изменения организационной культуры и процессов работы. Необходимо обеспечить поддержку как основному персоналу, так и руководству, а также провести необходимую обучение и адаптацию в компании.

1.5 Проблематика в системе управления проектами в научно-производственном объединении «Сапфир»

Переход к цифровой трансформации неизбежно встает перед многими современными организациями, стремящимися к автоматизации своих бизнес-процессов и повышению конкурентоспособности. В рамках данного исследования анализируется проектная деятельность научно-производственного объединения (далее – НПО) «Сапфир».

НПО «САПФИР» является ведущим разработчиком программного обеспечения в области информационных технологий для государственных и муниципальных органов власти Российской Федерации [42].

Ключевыми направлениями деятельности НПО «Сапфир» являются:

– собственные программные решения для государственных и муниципальных органов власти;

– лицензированные учебный и удостоверяющий центры;

– комплекс услуг по проектированию, монтажу и внедрению систем антитеррористической защищенности зданий и сооружений [4].

В связи с разными направлениями деятельности НПО «Сапфир» имеет несколько дочерних компаний, включая ООО «Сапфир-Интеграция», ООО «Сапфир-Эксперт» и ООО «Сапфир-Инжиниринг». Каждая из этих компаний соответствует своему направлению деятельности. В данном исследовании рассматривается деятельность дочернего предприятия ООО «Сапфир-Интеграция» в сфере разработки ПО.

На данный момент компания продолжает активно развиваться, что отражается в различных показателях эффективности её работы:

– доходы компании в 2023 году выросли на 10% по сравнению с 2020 годом и на 7% по сравнению с 2022 годом [11];

– количество заключенных контрактов в 2023 году увеличилось на 26,7% по сравнению с 2022 годом [11];

– расширение географии проектов: заключены контракты на сопровождение информационных систем Свердловской, Мурманской, Кировской, Челябинской областей;

– расширение количества и направлений проектной деятельности [50].

Разработкой компании является программный комплекс под названием «Информационная система управления финансами» (ПК «ИСУФ»). Данный программный комплекс является кроссплатформенным продуктом, который может работать с различными системами управления базами данных (СУБД) и офисными пакетами, в том числе с отечественным ПО. ПК «ИСУФ» предназначен для автоматизации основных процессов, функций и задач на уровне субъектов и муниципальных образований Российской Федерации. Он состоит из множества интегрированных модулей или подсистем, каждый из которых предназначен для автоматизации конкретных задач в различных областях, таких как финансы, экономика, образование, транспорт и другие [4].

При продолжающемся динамичном росте компания ощущает усиление конкурентной борьбы в своем секторе рынка. На рынке разработки программного обеспечения в России сейчас наблюдается высокая конкуренция и быстрое развитие. Компании постоянно стремятсяк инновациям и улучшению качества своих продуктов и услуг. С развитием цифровых технологий и появлением новых тенденций, таких как искусственный интеллект, облачные вычисления, интернет вещей и блокчейн, компании активно адаптируют свои продукты и услуги к изменяющимся потребностям рынка. Крупные игроки уделяют особое внимание инновациям и цифровой трансформации, чтобы оставаться конкурентоспособными. В то же время, на рынке присутствует множество малых и средних компаний, которые также активно участвуют в конкурентной гонке, предлагая свои авторские продукты и услуги.

На данный момент оценить точную емкость корпоративного сегмента разработки ПО представляется сложной задачей, учитывая, что на российском рынке происходит активная замена участников. Статистика обновляется каждый день, поскольку обстановка постоянно меняется. По данным Digital-агентства Notamedia [1] емкость сектора разработки ПО для российских разработчиков на конец 2023 года – 20 млрд. рублей (размер рынка определяется на основе данных внутренних продаж всех игроков рынка за расчетный период).

На Рисунок 15 отражена динамика доходов ООО «Сапфир-Интеграция» за 2014-2023 года. В 2023 году сумма внутренних продаж составила 132 127 000 рублей. На основе данных о емкости рынка разработки ПО в 2023году и данных по объему доходов компании за этот же период была рассчитана рыночная доля фирмы в отрасли, как отношение доходов ООО «Сапфир-Интеграция» к объему рынка за соответствующий год по формуле (1):

$$\text{Доля рынка} = \frac{\text{Доходы компании}}{\text{Общие доходы рынка}} * 100\% \quad (1)$$

ООО «Сапфир-Интеграция» занимает примерно 0,7% рынка программного обеспечения среди российских разработчиков в 2023 году, что является сравнительно небольшой долей.

Рисунок 15 – Динамика доходов ООО «Сапфир-Интеграция» от в сфере разработки ПО за 2014-2023 гг.[9]

Также стоит отметить, что около 70% российских компаний, как и ООО «Сапфир-Интеграция», предоставляют IT-решения с помощью low-code платформ – т.е. составляют системы из готовых компонентов с минимумом программирования [24], что обостряет конкуренцию на рынке для компании.

Участники рынка стремятся адаптироваться к изменяющимся рыночным условиям и технологическим инновациям, чтобы оставаться конкурентоспособными в сегодняшней высококонкурентной производственной среде [66]. Стратегия развития инструментария цифровизации проектного управления позволяет фирмам оставаться конкурентоспособными в своих сегментах рынка в краткосрочной перспективе, а также расширяет их возможности по созданию инновационных продуктов и процессов в долгосрочной перспективе.

Внедрение цифровых процессов в бизнесе приводит к росту организации, что способствует ее будущему успеху и является движущей силой для поддержания своей жизнеспособности в условиях современной экономики. Однако такие небольшие компании, как ООО «Сапфир-

[9] Составлено автором по: [1111]

Интеграция» сталкиваются с ограничениями в бюджетных и кадровых ресурсах, что затрудняет ее способность к инвестированию в развитие и технологические инновации.

В данном исследовании анализируется текущее состояние сервисной архитектуры компании и разрабатывается стратегия по ее улучшению и автоматизации бизнес-процессов с минимальными затратами. Целью работы является повышение эффективности основных бизнес-процессов в ООО «Сапфир-Интеграция» для поддержки конкурентоспособности фирмы.

1.6 Результаты и выводы по разделу 1

В первом разделе были рассмотрены теоретические основы цифровизации процессов проектного управления, а также изучена специфика деятельности и проблематика в проектном управлении ООО «Сапфир-Интеграция».

Были рассмотрены модели и фреймворки для управления проектами. Основными заказчиками ООО «Сапфир-Интеграция» являются органы государственной и муниципальной власти Российской Федерации, поэтому проекты данной организации характеризуются изменяющимися требованиями. Чаще всего это связано с недостаточной четкостью и структурированностью первоначальных запросов или изменениями в законодательстве и политике. В связи с этим, классические методологии разработки ПО, которые предполагают строгие и предварительно определенные требования к проектам, являются недостаточно гибкими для удовлетворения потребностей таких заказчиков. Применение гибких методологий разработки позволяет эффективно управлять динамичными требованиями и поддерживать активное участие заказчика в процессе разработки, что способствует созданию продукта, отвечающего его потребностям.

В ходе исследования были проанализированы особенности применения модели TOGAF 10 для построения и оценки сервисной архитектуры предприятия и ее интеграции с другими методологиями управления проектами. Выбор TOGAF 10 обусловлен его универсальностью и применимостью в широком спектре организаций. Также стандарт является бесплатным и расположен во множестве открытых ресурсов, что приоритетно для небольшой компании с ограниченными ресурсами. Учитывая сложность и масштабность деятельности ООО «Сапфир-Интеграция», данный фреймворк представляется наиболее подходящим инструментом для анализа иповышения эффективности архитектурных процессов. Применение модели TOGAF 10 позволит более системно и эффективно организовать процессы управления проектами в компании, обеспечивая целостность, гибкость и высокую адаптивность сервисных архитектурных решений к изменяющимся потребностям и вызовам бизнеса.

2 РАЗРАБОТКА И ВНЕДРЕНИЕ ИНСТРУМЕНТАРИЯ ЦИФРОВИЗАЦИИ ПРОЕКТНОГО УПРАВЛЕНИЯ НА ОСНОВЕ СЕРВИСНОЙ АРХИТЕКТУРЫ

2.1 Общая характеристика ООО «Сапфир-Интеграция»

В данном исследовании рассматривается общество с ограниченной ответственностью «Сапфир-Интеграция». Основной вид экономической деятельности компании – разработка компьютерного программного обеспечения. Дополнительными видами экономической деятельности ООО «Сапфир-Интеграция» являются:

– деятельность консультативная и работы в области компьютерных технологий;

– деятельность, связанная с использованием вычислительной техники и информационных технологий;

– деятельность по обработке данных, предоставление услуг по размещению информации и связанная с этим деятельность;

– деятельность по созданию и использованию баз данных и информационных ресурсов;

– научные исследования и разработки в области естественных и технических наук;

– торговля оптовая компьютерами, периферийными устройствами к компьютерам и программным обеспечением [43].

Программный комплекс «Информационная система управления финансами» (ПК «ИСУФ») является авторской разработкой компании, зарегистрирован в Роспатенте и в едином реестре российских программ для ЭВМ (электронных вычислительных машин) и БД (баз данных).

Главной целью программного комплекса является автоматизация бюджетного процесса и смежных областей. Основными клиентами являются органы государственной и муниципальной власти. Бизнес-логика

информационной системы и работа с базой данных выполняются на стороне сервера. Работа пользователя с системой происходит через клиентское приложение – веб-браузер. ПК «ИСУФ» является кроссплатформенным многопользовательским решением, построенным на основе микросервисной архитектуры.

В данной работе рассматривается блок проектов, который включает в себя следующие программные модули (Рисунок 16):

– Государственные задания: данный модуль предназначен для управления и мониторинга выполнения государственных заданий, назначаемых получателям бюджетных средств органами государственной власти. Включает в себя функции генерирования, планирования, отслеживания исполнения государственных заданий, формирования отчетов по выполненным заданиям, а также управление документами и информацией, связанной с государственными заказами;

– Муниципальные задания: этот модуль предназначен для учета и управления муниципальными заданиями, поставленными перед компанией со стороны муниципальных органов. Включает функции аналогичные государственным заданиям, но адаптированные к муниципальным потребностям и требованиям;

– Реестр услуг: данный модуль предоставляет возможность учета и систематизации всех предоставляемых услуг получателей бюджетных средств;

– Калькулятор: предназначен для автоматизации расчетов и анализа финансовых показателей, связанных с выполнением различных проектов и задач организации. Включает в себя инструменты для проведения финансовых расчетов, анализа бюджета, прогнозирования финансовых потоков и формирования финансовых отчетов;

– Иные цели: этот модуль предоставляет возможность учета и управления другими проектами и задачами организаций, которые не вписываются в категории государственных или муниципальных заданий;

– Сбор консолидированной отчетности: модуль предназначен для сбора, анализа и формирования консолидированной отчетности по различным направлениям деятельности компании;

– ПФХД (план финансово-хозяйственной деятельности), Штатное расписание: данный модуль предназначен для планирования финансовых и хозяйственных показателей компании, а также учета штатного расписания сотрудников. Позволяет формировать бюджеты, планировать расходы и доходы, управлять штатным расписанием, проводить анализ эффективности использования ресурсов;

– Сбор отчетности департамента молодежной политики: проект предназначен для сбора и анализа отчетности по реализации проектов и программ, связанных с молодежной политикой;

– НУГ (новый учебный год) и антитеррор: модуль предназначен для организации и управления мероприятиями, связанными с подготовкой к новому учебному году и обеспечением безопасности в образовательных учреждениях;

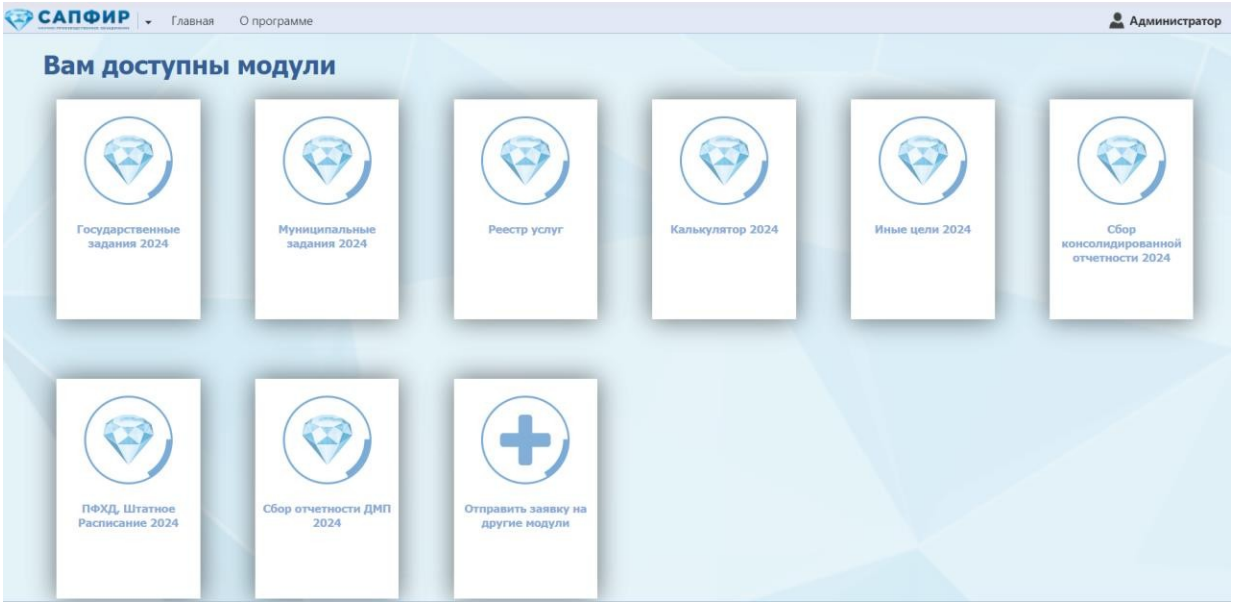


Рисунок 16 – Перечень программных модулей [42]

В качестве пользователей данного блока проектов выступают работники министерств образования, финансов, культуры, здравоохранения Свердловской области, а также работники администраций муниципальных образований Свердловской области и организаций-получателей бюджетных средств [42].

Гибкий инструментарий для настройки ИС обеспечивает:

– организацию процессов сбора, хранения и обработки необходимой информации о социально-экономическом развитии Свердловской области;

– унификацию форм сбора данных на всех уровнях процесса мониторинга социально–экономического развития;

– минимизацию ошибок при формировании консолидированных отчетов;

– автоматизированное формирование типовых отчетов, возможность настройки новых видов отчетов;

– широкий выбор средств для визуального представления данных и их анализа (настраиваемые отчеты, графики, диаграммы, картография, OLAP – кубы).

ООО «Сапфир-Интеграция» предоставляет данное IT-решение с помощью Low-code платформы. Low-code платформа – это программная среда, которая позволяет разрабатывать приложения с минимальным использованием традиционного программирования. С ее помощью разработчики могут создавать приложения, используя визуальные интерфейсы и графические инструменты, а также готовые компоненты и модули [32].

Основными принципами и возможностями Low-code платформ являются [2]:

– визуальное программирование через графический интерфейс, а не путем написания кода;

– использование готовых компонентов и шаблонов для быстрой разработки информационной системы;

– простая интеграция с различными внешними системами и сервисами с помощью предварительно созданных коннекторов и API;

– легкая настройка и расширение функциональности системы.

Несмотря на преимущества использования Low-code платформ, они могут иметь ограничения в гибкости и расширяемости, что может затруднить адаптацию к уникальным требованиям клиентов компании. Также, учитывая специфику заказчика и важность сохранения конфиденциальности данных, создается зависимость от поставщика сервера, то есть от технического оборудования министерства финансов Свердловской области. Зависимость от поставщика сервера означает, что техническая поддержка и обслуживание сервера будут осуществляться со стороны заказчика или уполномоченных им специалистов. Это важно учитывать для оперативного реагирования на любые проблемы или сбои в работе сервера. При использовании сервера заказчика важно также планировать его масштабирование и возможность увеличения вычислительных ресурсов в соответствии с ростом объема данных или потребностей системы. Также использование Low-code платформы может создать дополнительные риски в области безопасности, особенно если платформа не обеспечивает должного уровня защиты от угроз и атак.

Еще одним существенным минусом такого решения являются временные затраты на обучение новых сотрудников. В отличие от традиционного программирования, где программисты работают с общепринятыми языками и инструментами разработки, использование Low-code платформ может потребовать от сотрудников изучения дополнительных концепций, связанных с работой на платформе, таких как использование готовых компонентов, конфигурация интерфейсов и т. д.

2.2 Специфика архитектуры ООО «Сапфир-Интеграция» по TOGAF 10

В данном разделе рассмотрена архитектура ООО «Сапфир-Интеграция» по модели TOGAF 10. В работе акцент сделан на гибком подходе к управлению сервисной архитектурой, что обусловлено специфическими требованиями и целями проекта. TOGAF 10 предоставляет полный и всеобъемлющий набор инструментов для управления архитектурой предприятия, включающий восемь основных компонентов. Однако, в контексте данного проекта не все компоненты оказываются равноценными с точки зрения их актуальности и пользы. В связи с этим было принято решение сосредоточиться на пяти ключевых компонентах, наиболее критичных для достижения поставленных целей:

- бизнес-архитектура;
- архитектура данных;
- техническая архитектура;
- архитектура приложений;
- архитектура безопасности.

Архитектура бизнеса в рамках TOGAF определяет стратегию, миссию и цели предприятия, а также описывает его структуру управления, ключевые бизнес-процессы и взаимосвязи между ними. В основе этой архитектуры лежит понимание бизнес-модели компании и ее стратегии развития [9].

Миссия ООО «Сапфир-Интеграция»: предоставить клиентам высококачественные IT-решения и услуги для повышения эффективности государственного и муниципального управления в Российской Федерации.

Стратегия компании предусматривает постоянное стремление к внедрению новых технологий и методологий разработки для обеспечения конкурентоспособности своих IT-решений, что включает в себя следование трендам в области цифровизации, использование передовых практик разработки и постоянное развитие своих продуктов и услуг.

Стратегические цели ООО «Сапфир-Интеграция»:

– сокращение времени разработки новых продуктов;
– повышение конкурентоспособности на рынке разработки ПО;
– увеличение уровня удовлетворенности клиентов;
– формирование и развитие высококвалифицированных кадров;
– повышение качества продуктов и услуг.

Миссия, цели и стратегия предприятия помогают создать ясное видение того, чего компания хочет достичь в будущем. Они определяют конечные цели и результаты, конкурентные преимущества и уникальные возможности на рынке, целевую аудиторию и потребности клиентов, мотивацию сотрудников [15]. Связи между миссией компании, ее целями и задачами отражены на Рисунок 17.

Рисунок 17 – Миссия, цели и задачи ООО «Сапфир-Интеграция»[10]

Ключевые факторы успеха (КФУ) представляют собой основные аспекты или факторы, которые существенны для достижения целей организации или проекта [41]. Они обычно определяются в начале

[10] Составлено автором по: [51]

стратегического планирования и используются для определения того, что необходимо сделать для успешного завершения проекта.

Для оценки эффективности деятельности организации или конкретных процессов в ней используются показатели эффективности (KPI), конкретные численные или качественные показатели. Данные метрики помогают бизнесу оценить, насколько успешно организация или проект выполняют свои стратегические цели и в каких областях нужно сосредоточить усилия для улучшения результатов. KPI должны быть актуальными и отражать текущие приоритеты и стратегические цели компании. Также они должны быть поддающимися воздействию сотрудников или бизнес-процессов, чтобы компания могла активно управлять своими результатами.

На Рисунок 18 отображены ключевые факторы успеха, стратегические цели и требования, а также ключевые показатели эффективности ООО «Сапфир-Интеграция».

Рисунок 18 – Ключевые факторы успеха и показатели эффективности ООО «Сапфир-Интеграция»[11]

11 Составлено автором по: [51]

Далее рассмотрим организационную структуру ООО «Сапфир-Интеграция». Организационная структура предприятия определяет способ управления, распределения обязанностей и ответственности, а также взаимосвязи между различными подразделениями и уровнями управления. Среднесписочная численность работников в компании на 2023 год – 17 человек. ООО «Сапфир-Интеграция» имеет типичную для многих небольших IT-компаний линейную организационную структуру, включающую следующие ключевые элементы (Рисунок 19):

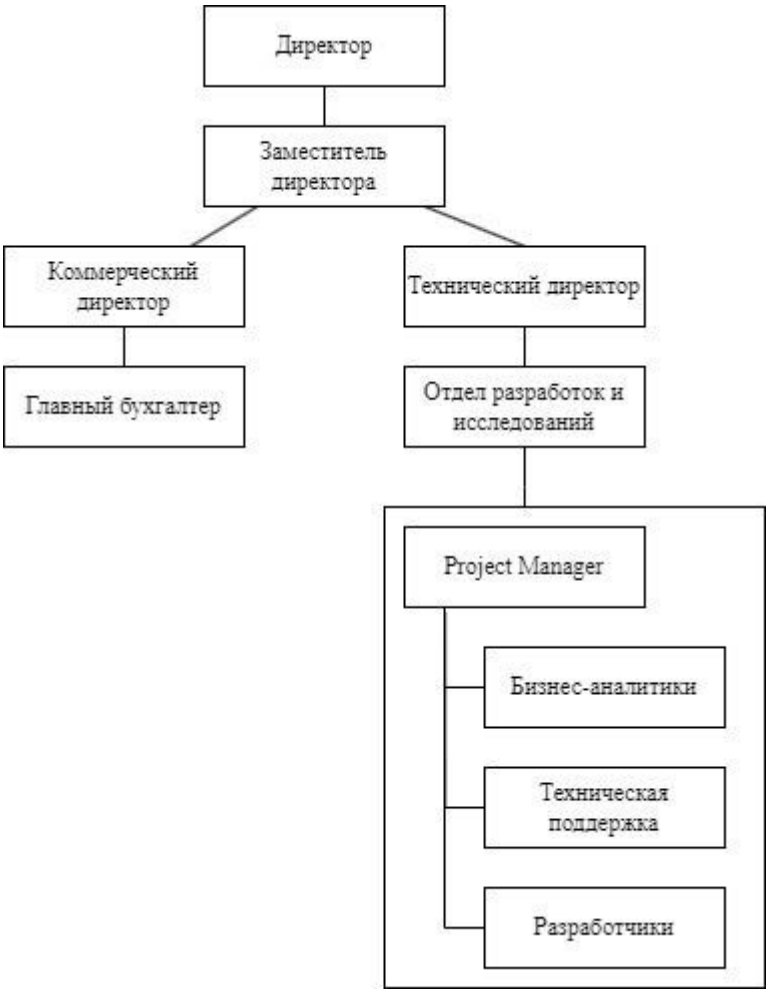


Рисунок 19 – Организационная структура ООО «Сапфир-Интеграция»[12]

Далее перейдем к ключевым бизнес-процессам компании и взаимосвязи между ними. Бизнес-процессы представляют собой последовательность действий или операций, которые выполняются в рамках организации для

[12] Составлено автором по: [51]

достижения определенных целей или результатов. Они описывают, каким образом бизнес-деятельность должна быть организована и выполнена, чтобы эффективно достигать поставленных целей [23]. Бизнес-процессы ООО «Сапфир-Интеграция» классифицированы по следующим критериям:

– Процессы управления (менеджмента) – процессы, которые относятся к управлению организацией в целом. Они включают в себя разработку стратегии, планирование, организацию, координацию, контроль и управление ресурсами предприятия.

– Основные бизнес-процессы, которые прямо связаны с основной деятельностью предприятия и определяют его конкурентоспособность на рынке. Эти процессы могут включать в себя производство товаров, оказание услуг, маркетинг, продажи, обслуживание клиентов и другие.

– Процессы обеспечения (поддержки). Эти процессы направлены на обеспечение поддержки основных бизнес-процессов и управления предприятием. Они включают в себя такие функции, как управление качеством, управление рисками, управление информационными технологиями, закупки и поставки, финансовое управление и др.

К основным бизнес-процессам в ООО «Сапфир-Интеграция» относятся:

- прием заявки на новый контракт;
- сбор требований к ИС;
- разработка ИС;
- тестирование системы;
- сопровождение системы и техническая поддержка.

Полная схема всех бизнес-процессов компании на разных уровнях представлена на Рисунок 20:

Рисунок 20 – Бизнес-процессы ООО «Сапфир-Интеграция»[13]

Чтобы определить, кто именно отвечает за выполнение конкретных процессов нужно построить матрицу распределения ответственности.

RACI – это инструмент управления проектами, используемый для определения ролей и ответственностей различных участников в рамках выполнения конкретной задачи или проекта. Матрица распределения ответственности в ООО «Сапфир-Интеграция» представлена в таблице 3.

В такой матрице по вертикали отображены все бизнес-процессы, а по горизонтали выявленные роли. Аббревиатура RACI расшифровывается следующим образом [33]:

1. R (Responsible) – тот, кто непосредственно выполняет задание.
2. A (Accountable) – тот, кто принимает работу и несёт ответственность за результат.
3. C (Consulted) – тот, кто оказывает консультативную помощь.
4. I (Informed) – тот, кто в курсе принимаемых решений и хода выполнения задачи.

[13] Составлено автором по: [51]

Таблица 3 – Матрица распределения ответственности (RACI) [51]

№	Процесс	Краткое описание процесса	CEO	Заместитель директора	Коммерческий директор	Технический директор	Главный бухгалтер	Руководитель проекта	Бизнес-аналитик	Сотрудник тех.поддержки	Разработчик
1	Управление маркетингом	Прогнозирование успешности фирмы в будущем, оценка возможных путей развития продуктов или услуг на рынке	I	C	RA	I		I			
2	Управление финансами	Управление финансовыми ресурсами организации	RA	I	I	I	C	I			
3	Управление штатом	Процессы найма, увольнения, обучения и развития персонала, а также управление кадровой политикой	I	A			I	R			
4	Стратегическое планирование	Основной процесс определения долгосрочных целей и стратегий развития компании	RA	I	I	I		I			
5	Управление разработкой	Включает в себя планирование, организацию и контроль процесса разработки продуктов или услуг компании	I	I	I	RA					
6	Прием заявки на новый контракт	Прием и обработка заявок от клиентов или партнеров на заключение новых контрактов или соглашений	I	I	I	I	CI	RA	I	I	I
7	Сбор требований к ИС	Процесс определения требований к ИС						AC	R	I	I
8	Разработка ИС	Процесс создания и внедрения информационной системы на основе собранных требований						AC	I	I	R
9	Тестирование системы	Включает в себя проверку функциональности, производительности, безопасности						A	R	I	IC
10	Сопровождение системы и техническая поддержка	Процессы обеспечения непрерывной работы ИС, адаптация к изменениям требований и обеспечение технической поддержки пользователям						A	I	R	C
11	Взаимодействие со стейкхолдерами	Процесс установления и поддержания отношений с заинтересованными сторонами (стейкхолдерами)						RA	I	I	I
12	Учет денежных средств	Процесс учета и контроля финансовых операций	I	I	IC		RA				
13	Хозяйственный учет	Учет товаров, материалов, оборудования	I	I	IC	IC	RA				
14	IT-обеспечение	Управление аппаратным и программным обеспечением, сетевая инфраструктура, безопасность информации и др.	I	I	I	A	I	R			

Для более глубокого и точного понимания процессов в компании визуально представим процессы каждого уровня в нотации Process Landscape. На Рисунок 20 представлена диаграмма, описывающая процессы менеджмента в ООО «Сапфир-Интеграция».

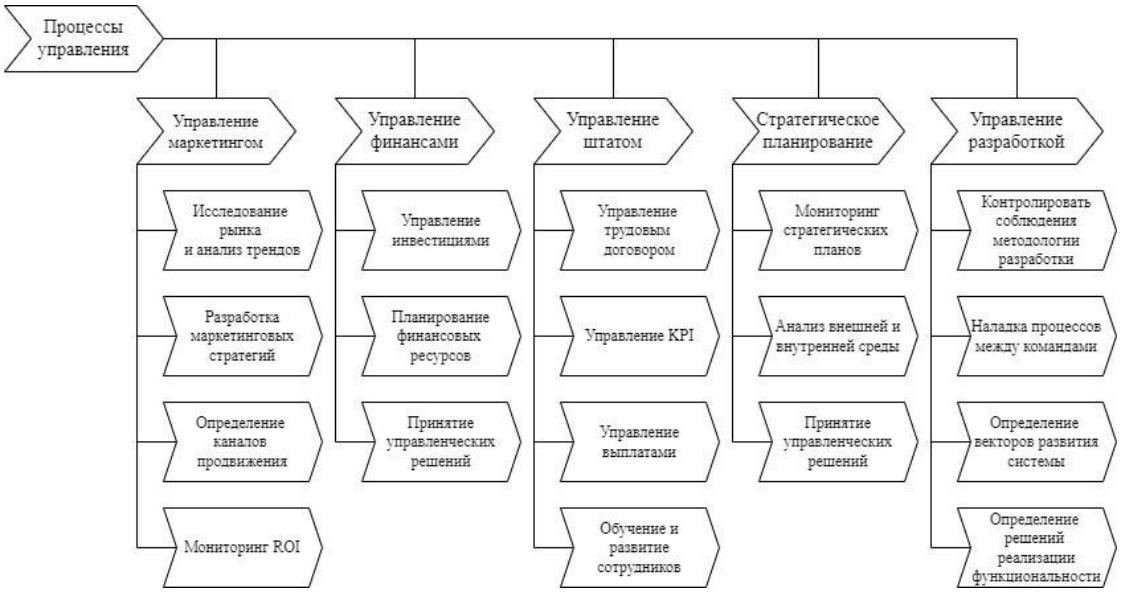


Рисунок 20 – Процессы менеджмента в ООО «Сапфир-Интеграция» в нотации Process Landscape[14]

Далее подробно рассмотрим основные бизнес-процессы (Рисунок 21).

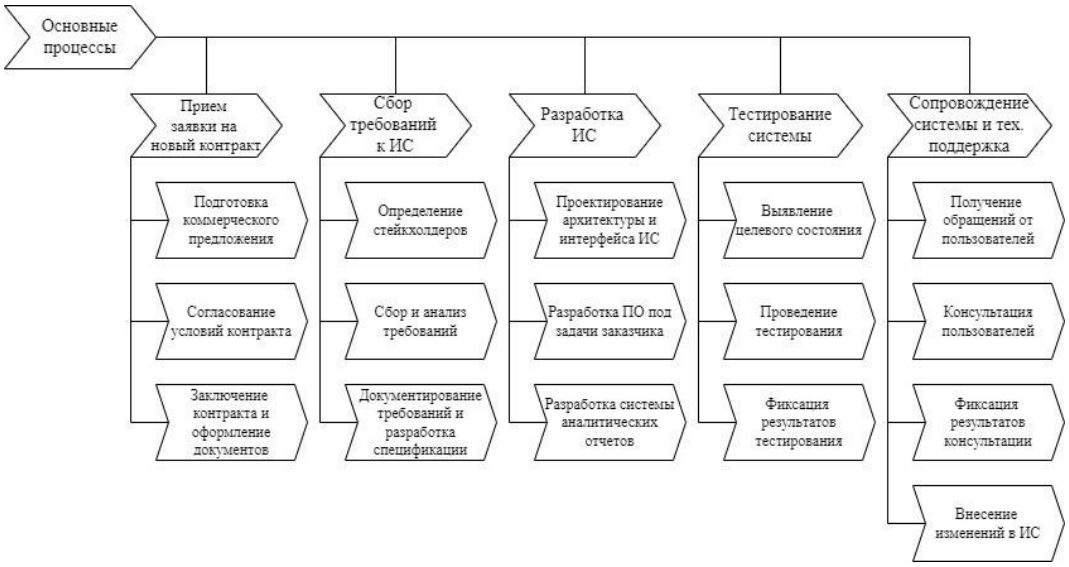


Рисунок 21 – Основные процессы в ООО «Сапфир-Интеграция» в нотации Process Landscape[15]

[14] Составлено автором по: [51]

[15] Составлено автором по: [51]

На рисунке 22 представлены обеспечивающие процессы компании.

Рисунок 22 – Обеспечивающие процессы в ООО «Сапфир-Интеграция» в нотации Process Landscape[16]

Что касается архитектуры приложений ООО «Сапфир-Интеграция», то компания использует как общепринятые инструменты (Microsoft Visual Studio, Telegram, Discord, Ewa-Wiki, Space, MS Office, SQL Server MS, Star UML) так и приложения собственной разработки для реализации своих проектов (Carbon, Model Designer, ETL Designer). Приложения и выполняемые ими функции представлены на Рисунок 23.

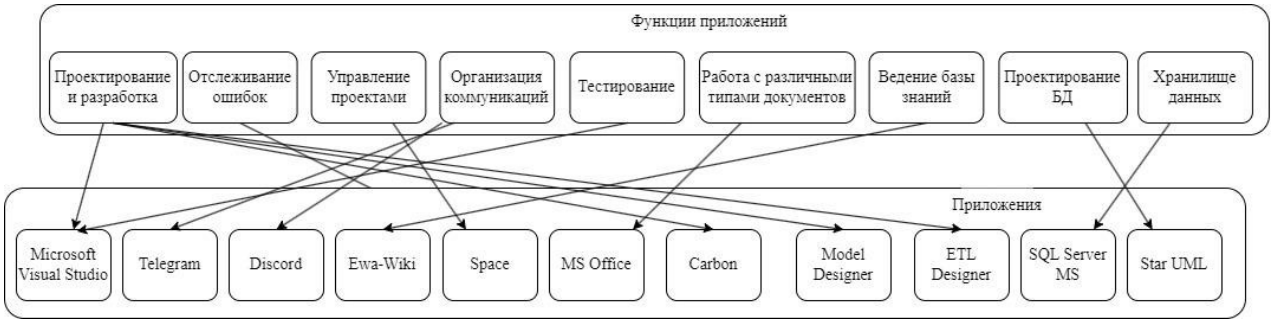


Рисунок 23 – Связь программных продуктов и функций[17]

[16] Составлено автором по: [51]

[17] Составлено автором по: [51]

Техническая архитектура – это составляющая архитектуры информационных систем (ИС), которая описывает структуру и организацию аппаратного и программного обеспечения, необходимого для реализации бизнес-процессов и функциональности системы [15]. Чтобы гарантировать непрерывную работу сотрудников, необходимо создать и поддерживать надежную сетевую инфраструктуру. Схема информационной инфраструктуры ООО «Сапфир-Интеграция» изображена ниже (Рисунок 24).

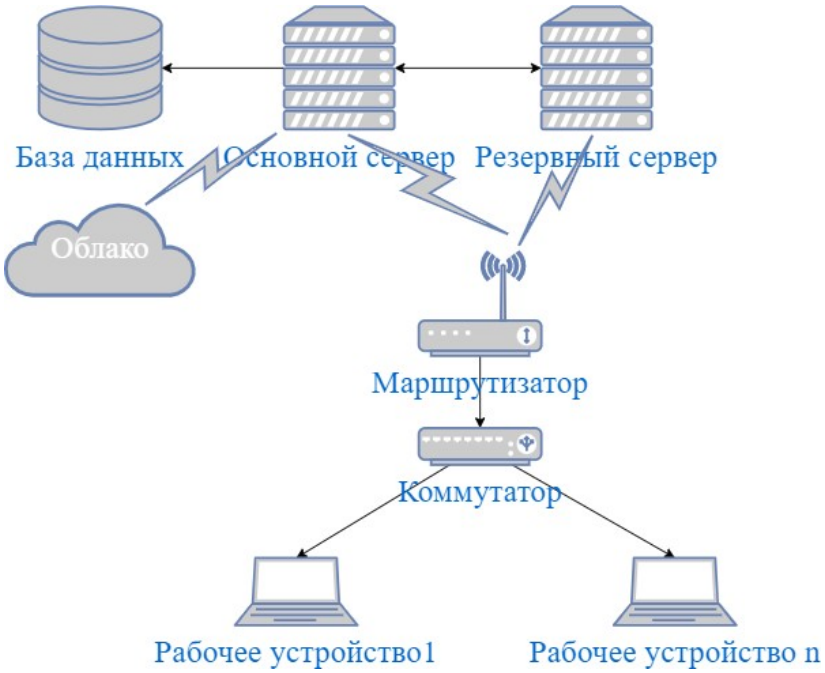


Рисунок 24 – Схема информационной инфраструктуры[18]

Каждый проект в рассматриваемом программном блоке имеет свою собственную базу данных с уникальными сущностями, соответствующими его функциональным требованиям и бизнес-процессам. Такая структура позволяет эффективно управлять данными по каждому проекту, сохранять их изолированность и обеспечивать надежность и производительность работы системы. Структура базы данных каждой информационной системы проектируется в приложении StarUml. Атрибуты таблицы и связи между сущностями создаются в собственном приложении компании Model Designer. Работа с базой данных проходит в приложении SQL Server Management Studio.

[18] Составлено автором по: [51]

Такая организация баз данных позволяет проектам вести свою работу независимо друг от друга, минимизируя возможные конфликты данных и обеспечивая гибкость и масштабируемость системы в целом.

Архитектура безопасности в ООО «Сапфир-Интеграция» играет ключевую роль в обеспечении защиты и сохранности данных, а также в предотвращении угроз со стороны внешних и внутренних источников. В рамках компании разработана комплексная архитектура безопасности, включающая в себя следующие аспекты:

– для обеспечения доступа к системам и данным применяются механизмы идентификации пользователей (определение их личности) и аутентификации (проверка подлинности их учетных данных);

– контроль доступа осуществляется путем установления различных ролей для пользователей и групп пользователей, что позволяет ограничить доступ к конфиденциальной информации только необходимым сотрудникам и предотвратить несанкционированный доступ;

– для защиты конфиденциальных данных применяется шифрование во время их передачи по сети и хранения в базах данных. Используются современные криптографические алгоритмы для обеспечения конфиденциальности и целостности информации;

– внедрены средства мониторинга и аудита, позволяющие отслеживать активность пользователей, обнаруживать подозрительные действия и реагировать на них в реальном времени. Журналы аудита сохраняют информацию о действиях пользователей для последующего анализа инцидентов;

– применяются средства защиты от внешних угроз, такие как фильтрация сетевого трафика, межсетевые экраны (firewalls), системы обнаружения вторжений (IDS/IPS) и антивирусные программы, чтобы предотвратить атаки извне.

В ходе исследования были рассмотрены основные аспекты архитектуры предприятия по TOGAF 10, включая бизнес-архитектуру, информационную архитектуру, архитектуру приложений, архитектуру безопасности и техническую архитектуру. В целом, проведенный анализ предприятия позволяет сформировать базу для разработки и внедрения новых информационных систем, а также оптимизации бизнес-процессов компании.

2.3 Проектирование и реализации системы аналитической отчетности

Разработка системы аналитических отчетов является важной частью основного бизнес-процесса компании – разработки информационных систем. Проектирование и реализация системы аналитической отчетности – это процесс создания инструмента для сбора, анализа и представления данных с целью получения статистической информации и принятия обоснованных решений.

Для построения аналитических отчётов и диаграмм, основанных на данных из различных источников в компании ООО «Сапфир-Интеграция» используется подсистема Карбон (Carbon). Карбон представлен в виде клиентского приложения (thick client), а также имеет web-интерфейс. Интерфейс клиентского приложения представлен на Рисунок 25.

Приложение содержит большую часть бизнес-логики и функциональности, необходимой для его работы, на стороне клиента, а не на сервере. В отличие от thin client, который выполняет большую часть своей работы на сервере, thick client требует более значительных ресурсов на стороне клиента [75].

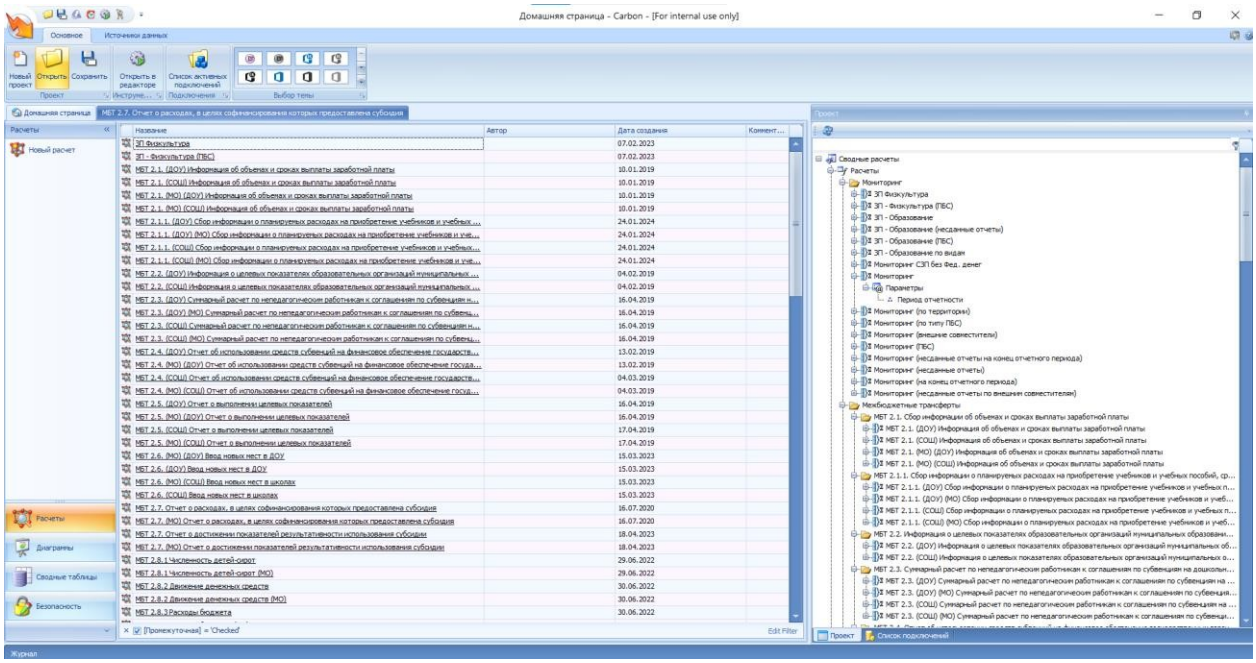


Рисунок 25 – Интерфейс клиентского приложения Карбон[19]

При работе с клиентским приложением были выявлены следующие недостатки данной подсистемы:

– Карбон не выдерживает большой нагрузки на сервер. Если на сайте заполняют и формируют отчетность пиковое количество пользователей, приложение начинает забирать приоритетность процессов на себя. Таким образом, деятельность сервера приостанавливается и сайт приходится перезапускать вручную, при этом понижая приоритетность процессов Карбона.

– Для сложных сводных отчетов, состоящих из данных нескольких таблиц, нужно делать промежуточные расчеты в Карбоне. Источниками данных в проекте служат кубы. Куб – это данные определенной сущности из базы данных. Существуют проекты, в которых нужны отчеты, содержащие в себе данные не из одной сущности, а сразу из нескольких таблиц. Чтобы создать такой отчет, сначала нужно построить промежуточные расчеты по всем таблицам. Примером такого отчета может послужить аналитика в проекте «Сбор консолидированной отчетности» по форме «Межбюджетные

19 Составлено автором по: [51]

трансферты. Отчет о расходах, в целях софинансирования которых предоставлена субсидия» (Рисунок 26). Сама форма состоит из трех таблиц с разными атрибутами. По всем трем формам созданы промежуточные расчеты и только потом из них собран финальный отчет. Время загрузки таких отчетов в web-интерфейсе может занимать от 2 до 15 минут в период пиковой загрузки сайта. Также отчеты не могут загружаться одновременно у разных пользователей, вся аналитика строится в порядке очереди. Поэтому такое количество времени для построения одного отчета является существенными недостатком подсистемы.

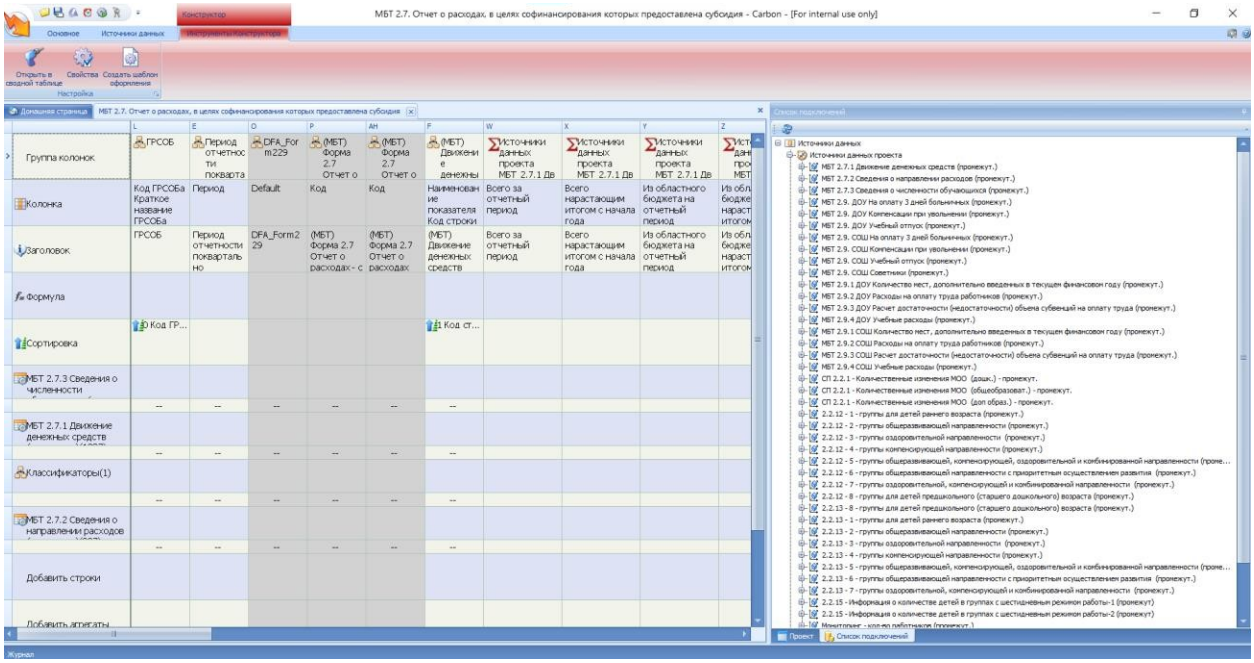


Рисунок 26 – Аналитический отчет, состоящий из данных нескольких сущностей (из промежуточных отчетов)[20]

– Еще одним недостатком является длительное время реагирования на изменения в базе данных. При обновлении данных в базе, информация не моментально подгружается в Карбоне. Это приводит к ситуациям, когда пользователи могут видеть устаревшую информацию в аналитическихотчетах. Более того, некоторые обновления могут занимать до нескольких дней, что замедляет процесс принятия решений и может привести к неправильным или устаревшим аналитическим выводам.

[20] Составлено автором по: [51]

– Кроме того, Карбон обладает ограниченной гибкостью и возможностями настройки. Изменение структуры отчетов или добавление новых аналитических функций требует значительных усилий и времени, так как процесс настройки и внесения изменений в подсистему достаточно сложен и требует специальных знаний в настройке системы.

Данные недостатки негативно влияют на работу с аналитическими отчетами и усложняют процесс принятия управленческих решений представителями государственных органов.

Проектирование и реализация новой системы аналитических отчетов осуществлялась с целью устранения недостатков, связанных с использованием подсистемы Карбон, а также для повышения эффективности и гибкости процесса анализа данных. Новая система аналитических отчетов базируется на прямом взаимодействии с базой данных (SQL) и автоматизации выгрузки данных с помощью макросов на C# для создания готовых отчетов в формате Excel. Тестирование системы производится на выгрузке аналитической отчетности по проекту «НУГ (новый учебный год) и антитеррор». Пользователями данного проекта являются работники Министерства культуры, здравоохранения, образования и молодежной политики, социальной политики, физической культуры и спорта Свердловской области, а также работники администраций муниципальных образований Свердловской области.

Данный проект состоит из 17 форм по НУГу. Заказчику необходима аналитика по всем 17 таблицам с информацией об общем количество учреждений у каждого распорядителя средств областного бюджета (ГРСОБ). Также в аналитике должна содержаться информации о том, сколько учреждений у каждого ГРСОБа заполнило ту или иную форму НУГа.

Такой аналитический отчет заказчику необходимо выгружать ежедневно, особенно важно, чтобы информация была актуальна на момент выгрузки. Также стоит отметить, что отчет будет собирать информацию из более чем 17 таблиц. Если строить такой расчет в Карбоне, то для начала

нужно будет создать как минимум 17 промежуточных отчетов и в дальнейшем собирать по ним единую аналитику. Такой отчет будет создавать дополнительную нагрузку на сервер и долго загружаться в web-интерфейсе. Также существует большая вероятность того, что на момент выгрузки аналитики, отчет будет содержать не актуальную информацию, если пользователи незадолго до выгрузки внесли изменения в данные. При возникновении такой ситуации, информационная система будет не соответствовать требованиям от заказчика.

Все вышеперечисленное подтверждает актуальность проектирования новой системы аналитической отчетности для ООО «Сапфир-Интеграция». Процесс разработки и реализации новой системы включал следующие этапы:

1) Анализ требований пользователей и составление функциональных и нефункциональных требований к новой системе.

Функциональные требования:

– система должна позволять генерировать аналитические отчеты в формате Excel;

– система должна агрегировать данные из различных источников (сущностей базы данных);

– должна быть реализована возможность настройки различных параметров фильтрации и сортировки в отчете.

Нефункциональные требования:

– система должна иметь высокую производительность: время отклика на пользовательские действия не должно превышать 0.5 секунды;

– данные на момент выгрузки аналитики должны быть актуальными;

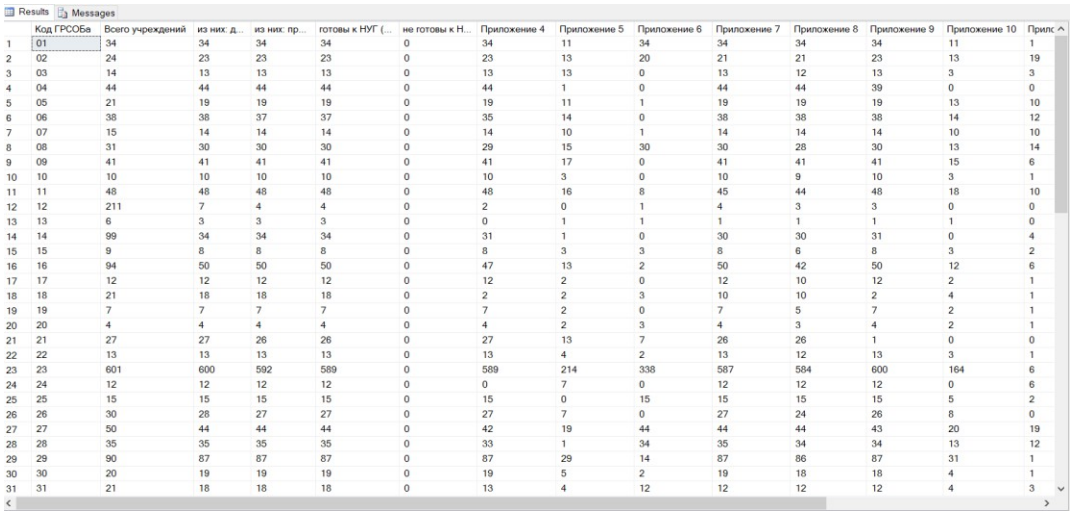
– аналитика должна загружаться за 3-5 секунд в историю при стабильном Интернет-соединении;

– доступ к данным и функционалу системы должен быть защищен с помощью аутентификации и авторизации пользователей;

– аналитика должна быть доступной для пользователей вне зависимости от времени и местоположения;

– интерфейс системы должен быть интуитивно понятным и удобным для пользователей всех уровней навыков.

2) Разработка скрипта на SQL. Для получения необходимой аналитической информации из базы данных был разработан SQL-запрос, который осуществляет агрегацию и обработку данных в соответствии с требованиями к ИС. Запрос направлен на получение аналитической таблицы (Рисунок 27) по различным приложениям НУГа (проект – Новый учебный год и антитеррор). Для каждого приложения НУГа подсчитывается количество учреждений, которое приступило к заполнению определённой формы в проекте. Каждый подзапрос содержит специфические условия и фильтры для определения статуса учреждений по конкретному приложению НУГа.



Results Messages

	Код ГРСОБв	Всего учреждений	из них: д...	из них: пр...	готовы к НУГ (...	не готовы к Н...	Приложение 4	Приложение 5	Приложение 6	Приложение 7	Приложение 8	Приложение 9	Приложение 10	Прилс
1	01	34	34	34	34	0	34	11	34	34	34	34	11	1
2	02	24	23	23	23	0	23	13	20	21	21	23	13	19
3	03	14	13	13	13	0	13	13	0	13	12	13	3	3
4	04	44	44	44	44	0	44	1	0	44	44	39	0	0
5	05	21	19	19	19	0	19	11	1	19	19	19	13	10
6	06	38	38	37	37	0	35	14	0	38	38	38	14	12
7	07	15	14	14	14	0	14	10	1	14	14	14	10	10
8	08	31	30	30	30	0	29	15	30	30	28	30	13	14
9	09	41	41	41	41	0	41	17	0	41	41	41	15	6
10	10	10	10	10	10	0	10	3	0	10	9	10	3	1
11	11	48	48	48	48	0	48	16	8	45	44	48	18	10
12	12	211	7	4	4	0	2	0	1	4	3	3	0	0
13	13	6	3	3	3	0	0	1	1	1	1	1	1	0
14	14	99	34	34	34	0	31	1	0	30	30	31	0	4
15	15	9	8	8	8	0	8	3	3	8	6	8	3	2
16	16	94	50	50	50	0	47	13	2	50	42	50	12	6
17	17	12	12	12	12	0	12	2	0	12	10	12	2	1
18	18	21	18	18	18	0	2	2	3	10	10	2	4	1
19	19	7	7	7	7	0	7	2	0	7	5	7	2	1
20	20	4	4	4	4	0	4	2	3	4	3	4	2	1
21	21	27	27	26	26	0	27	13	7	26	26	1	0	0
22	22	13	13	13	13	0	13	4	2	13	12	13	3	1
23	23	601	600	592	589	0	589	214	338	587	584	600	164	6
24	24	12	12	12	12	0	0	7	0	12	12	12	0	6
25	25	15	15	15	15	0	15	0	15	15	15	15	5	2
26	26	30	28	27	27	0	27	7	0	27	24	26	8	0
27	27	50	44	44	44	0	42	19	44	44	44	43	20	19
28	28	35	35	35	35	0	33	1	34	35	34	34	13	12
29	29	90	87	87	87	0	87	29	14	87	86	87	31	1
30	30	20	19	19	19	0	19	5	2	19	18	18	4	1
31	31	21	18	18	18	0	13	4	12	12	12	12	4	3

Рисунок 27 – Аналитическая таблица в Microsoft SQL Server Management Studio[21]

В запросе используются следующие функции и операторы:

– COUNT(CASE WHEN ... THEN 1 END) для условного подсчета значений в столбце в соответствии с определенным условием. Внутри функции CASE WHEN ... THEN 1 END выполняется условная проверка, и если условие истинно, то возвращается значение 1, в противном случае - NULL.

[21] Составлено автором по: [51]

Функция COUNT затем подсчитывает все ненулевые значения, что позволяет получить количество удовлетворяющих условию строк [76].

– Функция COALESCE применяется для замены NULL на заданное значение. В данном скрипте она используется для обработки случаев, когда количество строк, соответствующих определенным условиям, может быть равно нулю [62].

– Оператор LEFT JOIN используется для соединения таблиц таким образом, чтобы в результирующем наборе данных были включены все строки из левой таблицы (основной) и соответствующие строки из правой таблицы (подзапросы). Если соответствия в правой таблице нет, то для соответствующих строк будут подставлены NULL значения. В данном случае LEFT JOIN используется для объединения результатов подзапросов с основным набором данных, чтобы получить информацию о каждом муниципалитете и агрегированные данные по приложениям НУГа [37].

– Оператор DISTINCT используется для удаления дубликатов из результирующего набора данных. В контексте данного скрипта он применяется в подзапросах для исключения повторяющихся значений [22]. Сам SQL-запрос представлен в Приложении А.

3) Для автоматизации процесса создания отчетов в формате Excel был разработан макрос на языке программирования C#, который срабатывает при нажатии кнопки «Выгрузить аналитику по всем формам» в web-интерфейсе (Рисунок 28). Сам код на C# представлен в Приложении Б.

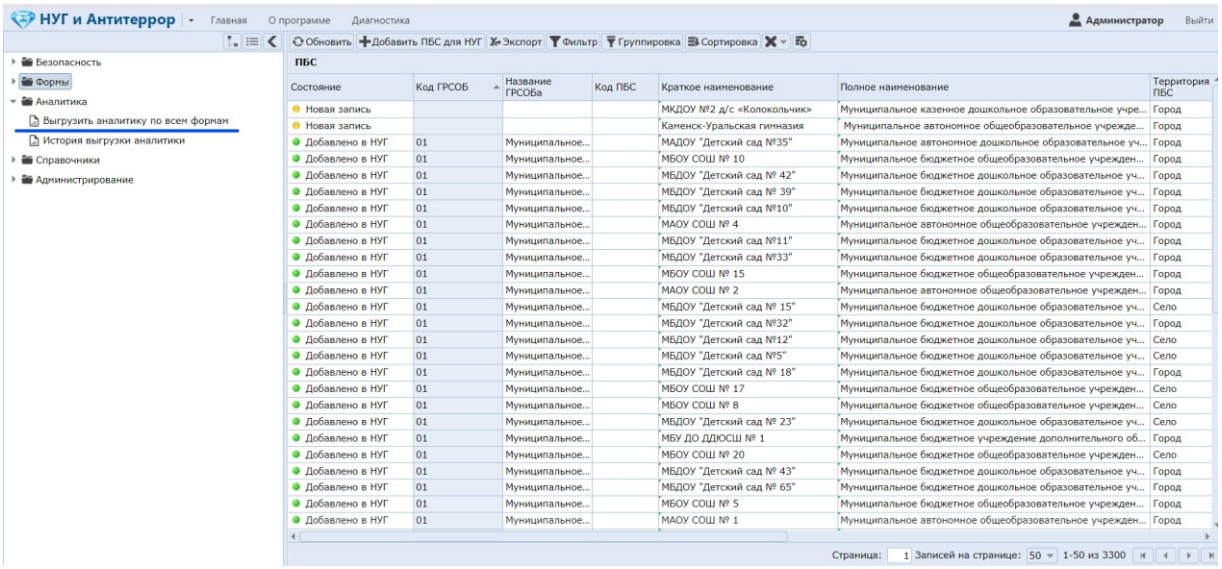


Рисунок 28 – Выгрузка аналитического отчета в web-интерфейсе[22]

В классе Analytics используются специализированные библиотеки для собственной платформы компании – Sector. Также базовая библиотека .NET Framework – System, которая предоставляет основные классы и методы для работы с файлами, потоками данных, строками и коллекциями. Библиотека OfficeOpenXml нужна для работы с файлами Excel, она предоставляет возможность создания, чтения и записи файлов, а также управление их содержимым. Библиотека System.Data.SqlClient относится к технологии ADO.NET и предоставляет средства для работы с базами данных SQL Server. System.IO отвечает за работу с файловой системой, включая чтение, запись, копирование и перемещение файлов и папок (Рисунок 29).

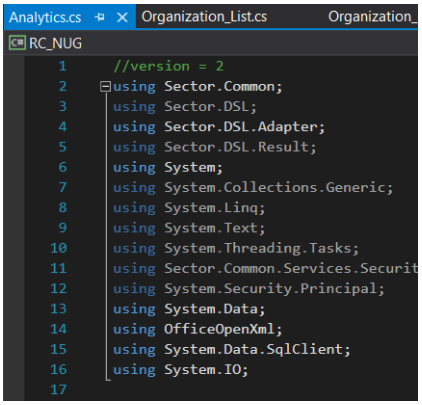


Рисунок 29 – Используемые библиотеки[23]

[22] Составлено автором по: [51]
[23] Составлено автором по: [51]

SQL-скрипт для аналитики загружается из файла и выполняется с помощью SqlCommand (Рисунок 30) для извлечения необходимых данных из базы.

```
SectorEntityAdapter TechicalTable = Entities["TechicalTable"];
string connectionString = "Data Source=gz_test\\mssql2012;Initial Catalog=RC_NUG23;User ID=amber;Pwd=password;"; // Подключение к базе данных
string templateFilePath = @"\\192.168.150.121\ProjectService\MinFin23\RC_NUG\datasector\Upload\Шаблон.xlsx"; // Путь к файлу-шаблону Excel
string outputFolderPath = @"\\192.168.150.121\ProjectService\MinFin23\RC_NUG\datasector\Upload"; // Путь для сохранения нового файла Excel
string outputFileName = "Аналитика НУГа " + DateTime.Now.ToString("yyyy-MM-dd (HH-mm)") + ".xlsx";

// Чтение всего содержимого SQL файла
string fileSQLPath = @"\\192.168.150.121\ProjectService\MinFin23\RC_NUG\Scripts\Аналитика.sql"; // Расположение скрипта для аналитики
string query = File.ReadAllText(fileSQLPath);

using (SqlConnection connection = new SqlConnection(connectionString))
{
    connection.Open();

    using (SqlCommand command = new SqlCommand(query, connection))
    {
        command.ExecuteNonQuery();
    }
}
```

Рисунок 30 – Фрагмент макроса для выполнения SQL-скрипта[24]

Для работы с файлом Excel используется библиотека EPPlus. Она позволяет создавать, открывать и редактировать файлы. В данном макросе это происходит путем итерации по строкам и столбцам DataTable и записи значений в ячейки (Рисунок 31).

```
// Загрузка данных из SQL в DataTable
using (SqlConnection connection = new SqlConnection(connectionString))
{
    connection.Open();

    using (SqlCommand command = new SqlCommand(query, connection))
    {
        using (SqlDataAdapter adapter = new SqlDataAdapter(command))
        {
            DataTable dataTable = new DataTable();
            adapter.Fill(dataTable);

            // Записываем MunicipalityCode в первую колонку, начиная с строки A6
            for (int row = 0; row < dataTable.Rows.Count; row++)
            {
                worksheet.Cells[row + 6, 1].Value = dataTable.Rows[row][0];
            }

            // Записываем остальные данные начиная со второй колонки, начиная с строки C6
            for (int row = 0; row < dataTable.Rows.Count; row++)
            {
                for (int col = 0; col < dataTable.Columns.Count - 1; col++)
                {
                    worksheet.Cells[row + 6, col + 3].Value = dataTable.Rows[row][col + 1];
                }
            }
        }
    }
}
```

Рисунок 31 – Запись данных из БД в шаблон Excel[25]

[24] Составлено автором по: [51]

[25] Составлено автором по: [51]

После того как все данные записаны, файл Excel сохраняется на сервере. Результат выполнения макроса отображается в виде сообщения об успешном выполнении или ошибке. Код содержит комментарии, которые помогают понять его структуру и логику работы. Также предусмотрена обработка исключений для возможных ошибок, таких как отсутствие файла-шаблона Excel или невозможность выполнить SQL-запрос.

Готовый аналитический отчет, с помощью макроса, выгружается в созданную таблицу «История выгрузки аналитики», в которой видно, кто и когда делал последнюю выгрузку отчета, а также сам Excel-файл с аналитикой (Рисунок 32).

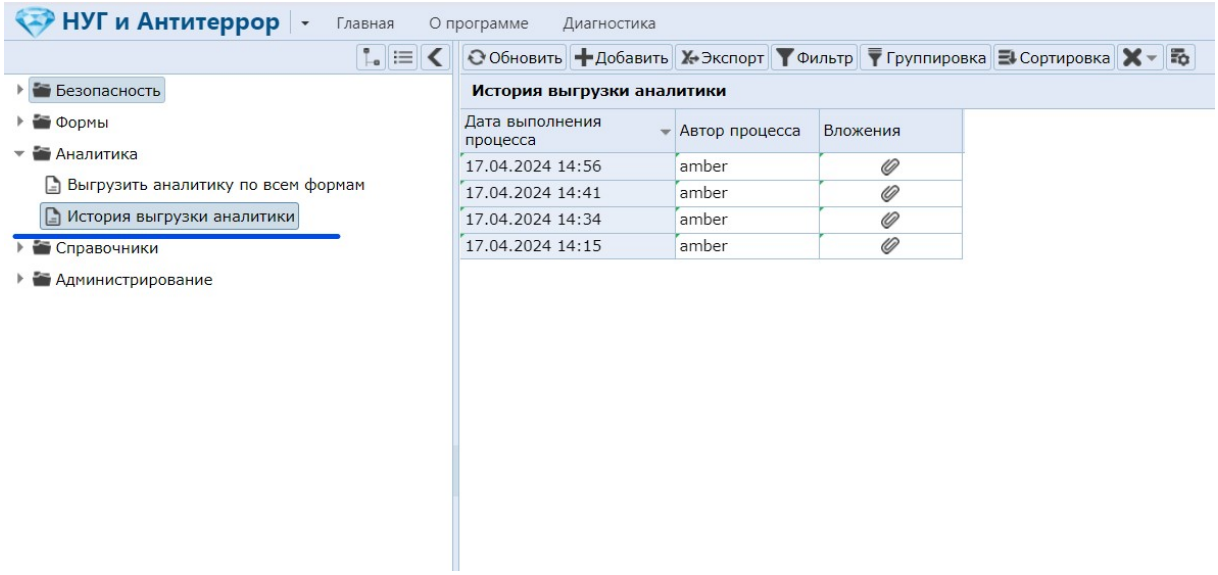


Рисунок 32 – История выгрузки аналитики в web-интерфейсе[26]

После проектирования и реализации новой информационной системы аналитической отчетности можно перейти к ее тестированию и внедрению.

2.4 Внедрение аналитических отчетов

Тестирование системы аналитических отчетов важно для обеспечения ее качества и надежности. Процесс тестирования был разбит на несколько этапов:

26 Составлено автором по: [51]

– Этап функционального тестирования, на котором проводилась проверка каждой функции системы, чтобы убедиться, что она работает в соответствии с требованиями. В данный этап включается проверка правильности данных в отчетах, работоспособность сортировки и фильтров.

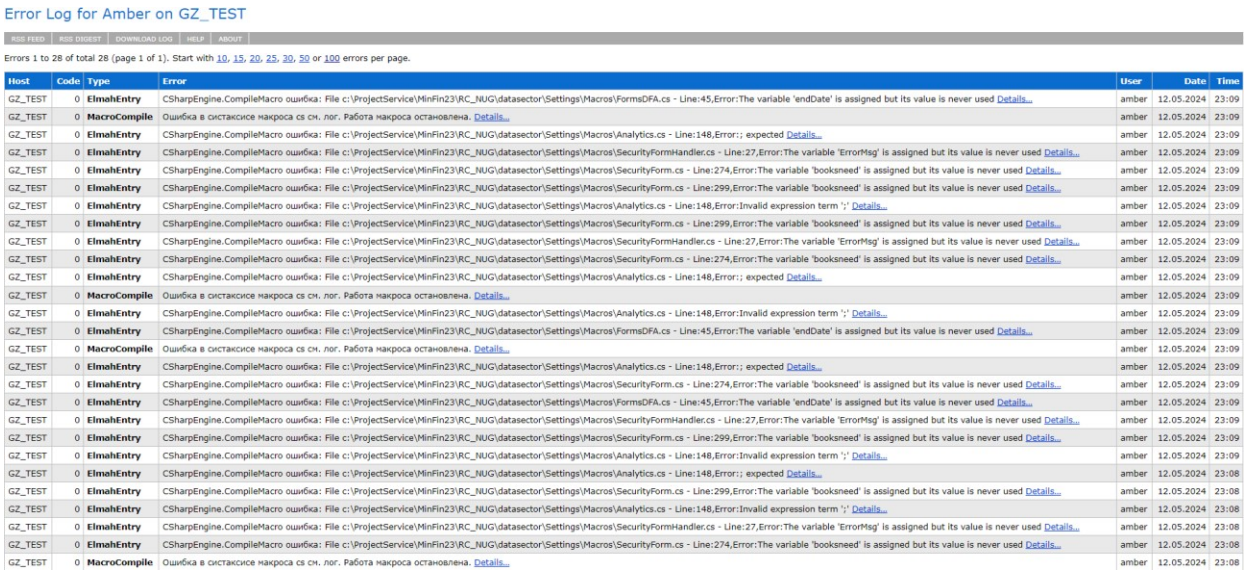
– Далее проверялась производительность системы: как быстро отчеты генерируются и экспортируются, а также скорость обновления данных в отчете после внесения последних изменений в базу данных.

– Тестирование безопасности, для защищенности системы от несанкционированного доступа или утечки данных.

– Тестирование пользовательского интерфейса, работоспособности кнопок и элементов управления веб-интерфейса.

– Тестирование того, как система будет работать при высокой нагрузке.

В процессе тестирования выявленные ошибки в макросе были устранены, а функциональность системы была улучшена. Пример выявленных багов на этапе тестирования представлен на Рисунок 33.



Error Log for Amber on GZ_TEST

RSS FEED | RSS DIGEST | DOWNLOAD LOG | HELP | ABOUT

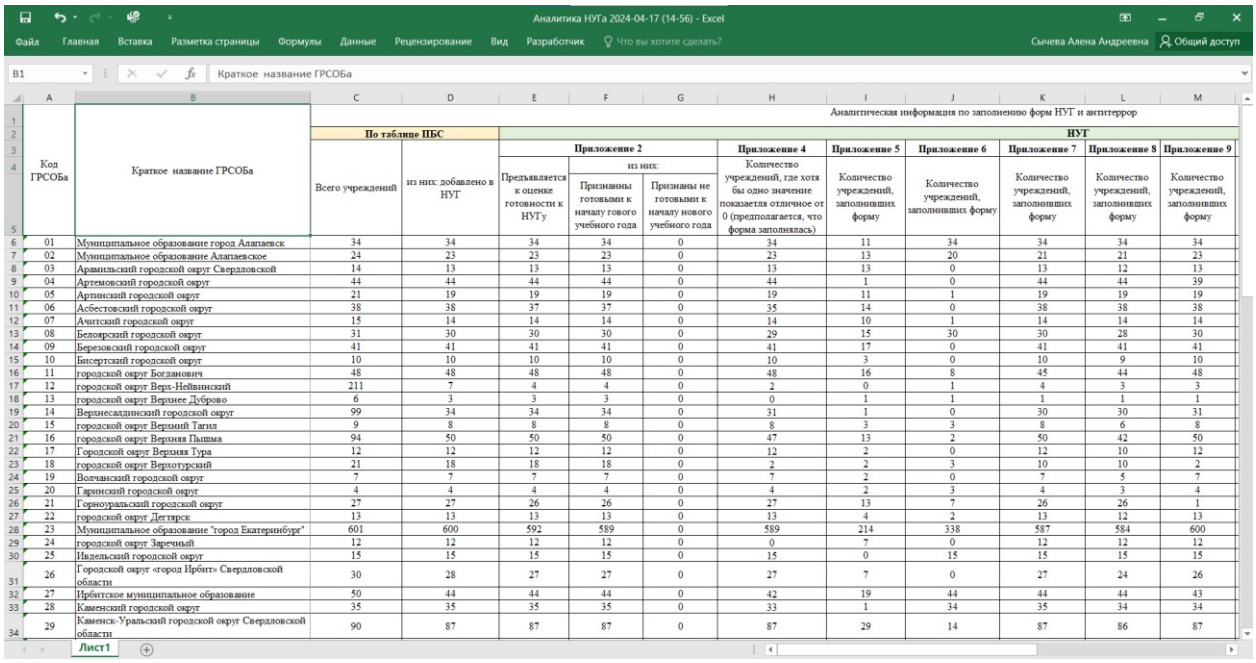
Errors 1 to 28 of total 28 (page 1 of 1). Start with 10, 15, 20, 25, 30, 50 or 100 errors per page.

Host	Code	Type	Error	User	Date	Time
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\FormsDFA.cs - Line:45,Error:The variable 'endDate' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	MacroCompile	Ошибка в систаксисе макроса cs см. лог. Работа макроса остановлена. Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\Analytics.cs - Line:148,Error:; expected Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityFormHandler.cs - Line:27,Error:The variable 'ErrorMsg' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityForm.cs - Line:274,Error:The variable 'booksneed' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityForm.cs - Line:299,Error:The variable 'booksneed' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\Analytics.cs - Line:148,Error:Invalid expression term ';' Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityForm.cs - Line:299,Error:The variable 'booksneed' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityFormHandler.cs - Line:27,Error:The variable 'ErrorMsg' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityForm.cs - Line:274,Error:The variable 'booksneed' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\Analytics.cs - Line:148,Error:; expected Details...	amber	12.05.2024	23:09
GZ_TEST	0	MacroCompile	Ошибка в систаксисе макроса cs см. лог. Работа макроса остановлена. Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\Analytics.cs - Line:148,Error:Invalid expression term ';' Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\FormsDFA.cs - Line:45,Error:The variable 'endDate' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	MacroCompile	Ошибка в систаксисе макроса cs см. лог. Работа макроса остановлена. Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\Analytics.cs - Line:148,Error:; expected Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityForm.cs - Line:274,Error:The variable 'booksneed' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\FormsDFA.cs - Line:45,Error:The variable 'endDate' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityFormHandler.cs - Line:27,Error:The variable 'ErrorMsg' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityForm.cs - Line:299,Error:The variable 'booksneed' is assigned but its value is never used Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\Analytics.cs - Line:148,Error:Invalid expression term ';' Details...	amber	12.05.2024	23:09
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\Analytics.cs - Line:148,Error:; expected Details...	amber	12.05.2024	23:08
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityForm.cs - Line:299,Error:The variable 'booksneed' is assigned but its value is never used Details...	amber	12.05.2024	23:08
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\Analytics.cs - Line:148,Error:Invalid expression term ';' Details...	amber	12.05.2024	23:08
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityFormHandler.cs - Line:27,Error:The variable 'ErrorMsg' is assigned but its value is never used Details...	amber	12.05.2024	23:08
GZ_TEST	0	ElmahEntry	CSharpEngine.CompileMacro ошибка: File c:\ProjectService\MinFin23\RC_NUG\datasector\Settings\Macros\SecurityForm.cs - Line:274,Error:The variable 'booksneed' is assigned but its value is never used Details...	amber	12.05.2024	23:08
GZ_TEST	0	MacroCompile	Ошибка в систаксисе макроса cs см. лог. Работа макроса остановлена. Details...	amber	12.05.2024	23:08

Рисунок 33 – Логи в файле log[27]

27 Составлено автором по: [51]

Написанный макрос обеспечивает выполнение SQL-запроса и выгрузку данных из созданной таблицы в готовый шаблон Excel с необходимой структурой и визуализацией данных (Рисунок 34).



Аналитика НУГа 2024-04-17 (14-56) - Excel

Файл Главная Вставка Разметка страницы Формулы Данные Рецензирование Вид Разработчик Что вы хотите сделать? Сычева Алена Андреевна Общий доступ

B1 Краткое название ГРСОБа

	A	B	C	D	E	F	G	H	I	J	K	L	M
1			Аналитическая информация по заполнению форм НУГ и антитеррор										
2			По таблице ПБС		НУГ								
3					Приложение 2			Приложение 4	Приложение 5	Приложение 6	Приложение 7	Приложение 8	Приложение 9
4	Код ГРСОБа	Краткое название ГРСОБа	Всего учреждений	из них: добавлено в НУГ	Предъявляется к оценке готовности к НУГу	из них: Признанны готовыми к началу гового учёного года	Признаны не готовыми к началу нового учёбного года	Количество учреждений, где хотя бы одно значение показателя отличное от 0 (предполагается, что форма заполнялась)	Количество учреждений, заполнивших форму	Количество учреждений, заполнивших форму	Количество учреждений, заполнивших форму	Количество учреждений, заполнивших форму	Количество учреждений, заполнивших форму
5													
6	01	Муниципальное образование город Алапаевск	34	34	34	34	0	34	11	34	34	34	34
7	02	Муниципальное образование Алапаевское	24	23	23	23	0	23	13	20	21	21	23
8	03	Арамильский городской округ Свердловской	14	13	13	13	0	13	13	0	13	12	13
9	04	Артемовский городской округ	44	44	44	44	0	44	1	0	44	44	39
10	05	Артинский городской округ	21	19	19	19	0	19	11	1	19	19	19
11	06	Асбестовский городской округ	38	38	37	37	0	35	14	0	38	38	38
12	07	Ачитский городской округ	15	14	14	14	0	14	10	1	14	14	14
13	08	Белоярский городской округ	31	30	30	30	0	29	15	30	30	28	30
14	09	Березовский городской округ	41	41	41	41	0	41	17	0	41	41	41
15	10	Бисертский городской округ	10	10	10	10	0	10	3	0	10	9	10
16	11	городской округ Богданович	48	48	48	48	0	48	16	8	45	44	48
17	12	городской округ Верх-Нейвинский	211	7	4	4	0	2	0	1	4	3	3
18	13	городской округ Верхнее Дуброво	6	3	3	3	0	0	1	1	1	1	1
19	14	Верхнесалдинский городской округ	99	34	34	34	0	31	1	0	30	30	31
20	15	городской округ Верхний Тагил	9	8	8	8	0	8	3	3	8	6	8
21	16	городской округ Верхняя Пышма	94	50	50	50	0	47	13	2	50	42	50
22	17	Городской округ Верхняя Тура	12	12	12	12	0	12	2	0	12	10	12
23	18	городской округ Верхотурский	21	18	18	18	0	2	2	3	10	10	2
24	19	Волчанский городской округ	7	7	7	7	0	7	2	0	7	5	7
25	20	Гаринский городской округ	4	4	4	4	0	4	2	3	4	3	4
26	21	Горноуральский городской округ	27	27	26	26	0	27	13	7	26	26	1
27	22	городской округ Дегтярск	13	13	13	13	0	13	4	2	13	12	13
28	23	Муниципальное образование "город Екатеринбург"	601	600	592	589	0	589	214	338	587	584	600
29	24	городской округ Заречный	12	12	12	12	0	0	7	0	12	12	12
30	25	Ивдельский городской округ	15	15	15	15	0	15	0	15	15	15	15
31	26	Городской округ «город Ирбит» Свердловской области	30	28	27	27	0	27	7	0	27	24	26
32	27	Ирбитское муниципальное образование	50	44	44	44	0	42	19	44	44	44	43
33	28	Каменский городской округ	35	35	35	35	0	33	1	34	35	34	34
34	29	Каменск-Уральский городской округ Свердловской области	90	87	87	87	0	87	29	14	87	86	87

Лист1

Рисунок 34 – Аналитический отчет в формате Excel[28]

Блок «История выгрузки аналитики» был введен после тестирования системы. Изначально в функционале новой системы аналитической отчетности предполагалось, что отчеты будут сразу загружаться в локальную папку пользователя (в которую он сам укажет путь).

При тестировании системы была рассмотрена ситуация, когда пользователь выгружает отчет в начале срока заполнения данной формы, затем удаляет его с локального компьютера, а дальше выгружает обновленнуюверсию отчета в конце срока заполнения по данной форме. Если пользователь ошибочно удалил первоначальный отчет с локальной папки, он больше не сможет его восстановить, что может привести к потере данных. А запись истории выгрузки аналитики позволит просматривать и анализировать историю выгрузок непосредственно на сайте, осуществлять сравнительный

[28] Составлено автором по: [51]

анализ данных в отчетах за нужную дату. Также информация о каждой выгрузке будет храниться на сервере.

Перед внедрением новой системы было проведено обучение сотрудников, ответственных за ее управление и сопровождение. Для пользователей также были подготовлены инструкции и руководства. После обучения пользователей система была внедрена в рабочую среду проекта. После внедрения продолжается мониторинг ее работы и сопровождение. Проводится анализ эффективности и качества работы системы.

Внедрённая система отчетности существенно улучшает процесс аналитической деятельности в компании. Сокращение времени загрузки отчетов и обновления данных позволяет сотрудникам быстрее получать актуальную информацию и принимать более обоснованные бизнес-решения. Представители министерств Свердловской области смогут более точно отслеживать ключевые показатели производительности и результаты работы получателей бюджетных средств, что улучшит процесс мониторинга и управления в организации.

Также новая система аналитических отчетов обладает значительным преимуществом при наборе и подготовке новых сотрудников благодаря использованию распространенных технологий, таких как SQL и C#. Эти языки программирования являются базовыми требованиями к аналитическим специалистам, что значительно упрощает процесс обучения новых сотрудников и интеграции их в работу с системой. В отличие от более специализированных платформ, использование стандартных технологий делает процесс адаптации к системе более быстрым и эффективным, а также позволяет организации экономить время и ресурсы на подготовке персонала.

Таким образом система способствует:

1) Ускорению процесса получения отчетов и анализа данных.

2) Увеличению точности представляемой отчетности.

3) Сокращению временных ресурсов, необходимых для создания и обновления отчетов.

4) Предоставлению более полной и своевременной информации для принятия решений на всех уровнях компании.

5) Устранению legacy system (устаревшие информационные системы, программное обеспечение или технологии, которые продолжают использоваться из-за того, что их замена или модернизация является сложной или затратной) в компании.

Также важно продумать направления развития для дальнейшего совершенствованию системы аналитических отчетов для ООО «Сапфир-Интеграция», чтобы она могла эффективно соответствовать потребностям компании в будущем. Вот несколько направлений для улучшения и доработки системы:

1) Расширение функциональности:
   - добавление новых типов отчетов и аналитических инструментов в соответствии с растущими потребностями бизнеса;
   - внедрение возможности создания персонализированных отчетов и дашбордов для пользователей различных уровней и ролей.
2) Улучшение производительности:
   - внедрение технологий и архитектурных решений, позволяющих масштабировать систему и обрабатывать большие объемы данных.
3) Повышение безопасности:
   - разработка новой группы ролей для пользователей для работы с отчетностью;
   - разработка и внедрение новых механизмов мониторинга и обнаружения угроз для предотвращения инцидентов.
4) Улучшение пользовательского интерфейса:

– проведение пользовательских исследований для выявления потребностей и предпочтений пользователей относительно интерфейса и функционала системы.

5) Интеграция с другими системами:

– разработка и внедрение механизмов интеграции с другими информационными системами компании для обмена данными и автоматизации процессов;

– интеграция с внешними источниками данных для расширения возможностей анализа.

2.5 Результаты и выводы по разделу 2

Во втором разделе были рассмотрены ключевые аспекты, связанные с анализом, проектированием и внедрением инструментов цифровизации проектного управления в ООО «Сапфир-Интеграция».

В исследовании рассматривается дочерняя компания НПО «Сапфир» – ООО «Сапфир-Интеграция». Основная деятельности компании – разработка программного обеспечения. Основными заказчиками являются органы государственной и муниципальной власти. Собственная разработка компании – программный комплекс «Информационная система управления финансами» (ПК «ИСУФ»), на базе которого реализуются все проекты с государственными органами.

В работе была построена архитектура ООО «Сапфир-Интеграция» по модели TOGAF 10. Для всестороннего и полного анализа деятельности компании было построено несколько архитектурных видов:

– бизнес-архитектура;
– архитектура данных;
– техническая архитектура;
– архитектура приложений;
– архитектура безопасности.

После подробного ознакомления с бизнес-процессами дочернего предприятия, была реализована и внедрена новая система аналитических отчетов, направленная на улучшение основного бизнес-процесса ООО «Сапфир-Интеграция» – разработку ПО.

Новая система способствует дальнейшему устранению устаревшей информационной системы для аналитической деятельности (Карбон), которую продолжают использовать ввиду того, что ее замена и модернизация являются затратными.

Аналитическая отчетность, построенная с помощью макросов на C# и SQL-запросов, позволяет:

– упростить процесс разработки аналитических сводных расчетов;

– увеличить точность данных в представляемых отчетах;

– повысить скорость загрузки отчетности на web-сайте, что повлияет на лояльность пользователей;

– снизить нагрузку на сервер;

– упростить сопровождение проекта.

3 ЗНАЧИМОСТЬ РЕЗУЛЬТАТОВ ИССЛЕДОВАНИЯ

3.1 Разработка новой структуры исходного кода

В данном разделе рассматривается программный модуль «Сбор консолидированной отчетности».

Пользователями программного комплекса являются специалисты:

– ГКУ СО «Хозяйственно-эксплуатационное управление»;

– Министерства образования и молодёжной политики Свердловской области;

– Подведомственных учреждений Министерства образования и молодёжной политики Свердловской области;

– Органов управления образованием муниципалитетов Свердловской области.

Данная система используется органами исполнительной власти для автоматизации процесса сбора информации с учреждений по следующим блокам [45]:

– модуль «Мониторинг»;

– модуль «Сетевые показатели»;

– модуль «Межбюджетные трансферты»;

– модуль «Оздоровительная компания»;

– модуль «Имущество потребителей бюджетных средств».

На Рисунок 35 представлена стартовая страница web-сайта данного программного комплекса.

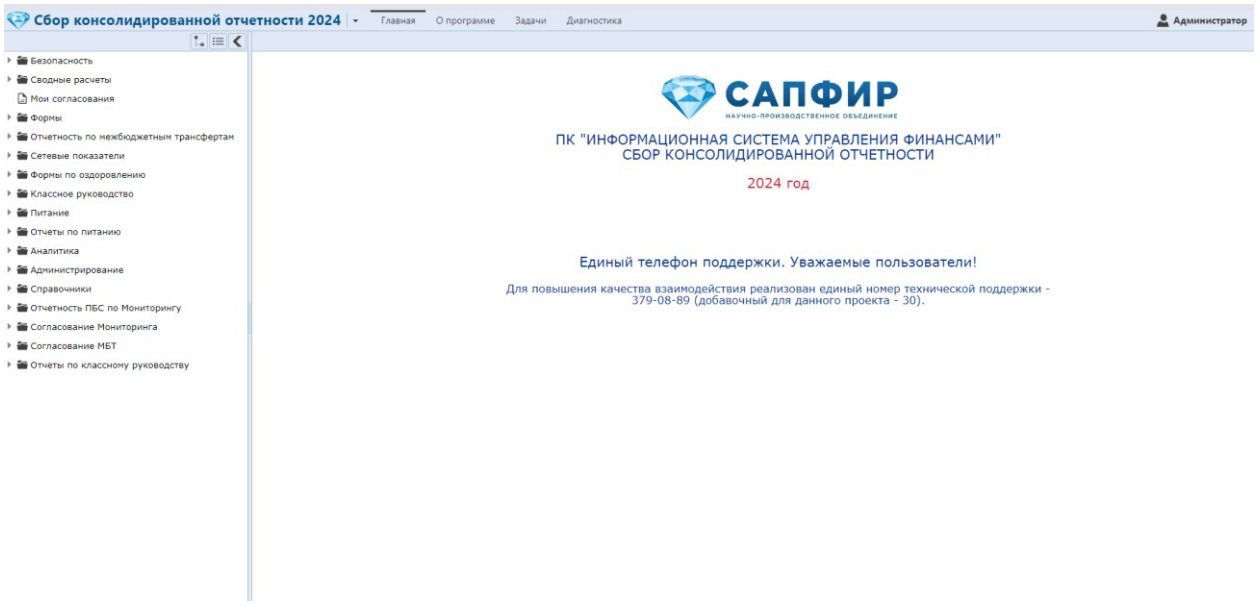


Рисунок 35 – Стартовая страница web-сайта программного комплекса «Сбор консолидированной отчетности»[29]

Проект «Сбор консолидированной отчетности» содержит большое количество внутренних подсистем, поэтому важно эффективно организовать управление и сопровождение всего комплекса. Обеспечение надлежащего функционирования и развития каждого модуля требует системного подхода к управлению проектом и координации деятельности всех участников процесса.

Эффективное сопровождение информационной системы включает в себя ряд ключевых аспектов, таких как:

– регулярное обновление и поддержание работы всех модулей;

– мониторинг и анализ производительности системы;

– реагирование на возникающие проблемы и решение их в кратчайшие сроки;

– внедрение нового функционала и улучшение существующего;

– обеспечение безопасности и защиты данных.

В исследовании рассматривается исходный код проекта на языке программирования C#. C# – это объектно-ориентированный язык программирования, разработанный компанией Microsoft. Он является одним

[29] Составлено автором по: [42]

из наиболее популярных и востребованных языков в современной разработке программного обеспечения. Ключевой особенностью C# является его объектно-ориентированный подход. Это означает, что в основе языка лежит концепция объектов – сущностей, которые содержат данные (свойства) и методы (функции). Объекты взаимодействуют друг с другом, образуя сложные программные системы.

Основные элементы объектно-ориентированного программирования в C# включают:

1) Классы – шаблоны для создания объектов, содержащие свойства и методы.

2) Объекты – экземпляры классов, которые можно создавать, использовать и манипулировать.

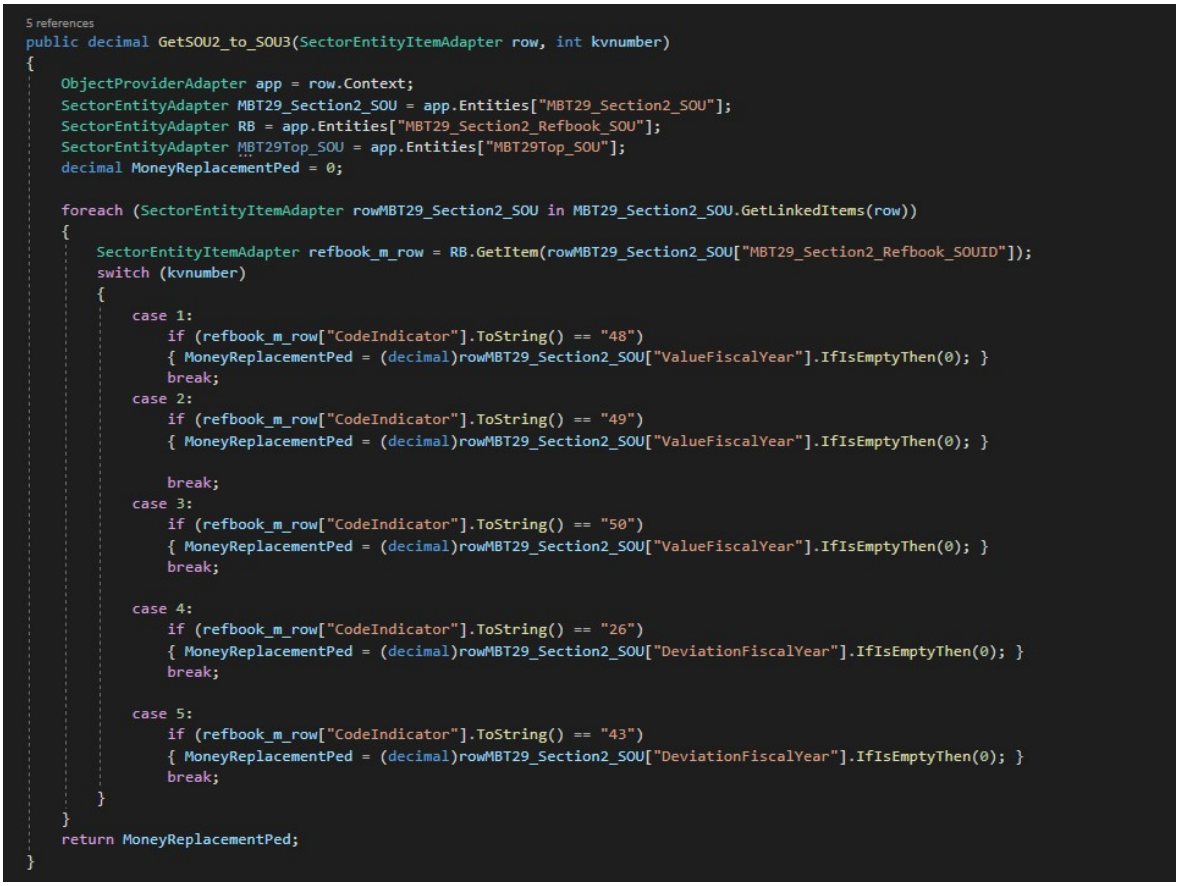
3) Наследование – возможность создавать новые классы на основе существующих, перенимая их свойства и методы.

4) Инкапсуляция – механизм, позволяющий скрывать внутреннюю реализацию объекта и предоставлять доступ к нему только через определенные интерфейсы.

5) Полиморфизм – способность объектов различных классов выполнять одни и те же действия по-разному [3].

В ходе анализа исходного кода проекта были выявлены следующие недостатки:

– Отсутствие комментариев в коде. Неаннотированный код затрудняет понимание его структуры и функциональности, что делает процесс поиска и исправления ошибок более трудозатратным. Примером такого кода могут послужить методы для модуля «Межбюджетные трансферты» (Рисунок 36).



```
5 references
public decimal GetSOU2_to_SOU3(SectorEntityItemAdapter row, int kvnumber)
{
    ObjectProviderAdapter app = row.Context;
    SectorEntityAdapter MBT29_Section2_SOU = app.Entities["MBT29_Section2_SOU"];
    SectorEntityAdapter RB = app.Entities["MBT29_Section2_Refbook_SOU"];
    SectorEntityAdapter MBT29Top_SOU = app.Entities["MBT29Top_SOU"];
    decimal MoneyReplacementPed = 0;

    foreach (SectorEntityItemAdapter rowMBT29_Section2_SOU in MBT29_Section2_SOU.GetLinkedItems(row))
    {
        SectorEntityItemAdapter refbook_m_row = RB.GetItem(rowMBT29_Section2_SOU["MBT29_Section2_Refbook_SOUID"]);
        switch (kvnumber)
        {
            case 1:
                if (refbook_m_row["CodeIndicator"].ToString() == "48")
                { MoneyReplacementPed = (decimal)rowMBT29_Section2_SOU["ValueFiscalYear"].IfIsEmptyThen(0); }
                break;
            case 2:
                if (refbook_m_row["CodeIndicator"].ToString() == "49")
                { MoneyReplacementPed = (decimal)rowMBT29_Section2_SOU["ValueFiscalYear"].IfIsEmptyThen(0); }

                break;
            case 3:
                if (refbook_m_row["CodeIndicator"].ToString() == "50")
                { MoneyReplacementPed = (decimal)rowMBT29_Section2_SOU["ValueFiscalYear"].IfIsEmptyThen(0); }
                break;

            case 4:
                if (refbook_m_row["CodeIndicator"].ToString() == "26")
                { MoneyReplacementPed = (decimal)rowMBT29_Section2_SOU["DeviationFiscalYear"].IfIsEmptyThen(0); }
                break;

            case 5:
                if (refbook_m_row["CodeIndicator"].ToString() == "43")
                { MoneyReplacementPed = (decimal)rowMBT29_Section2_SOU["DeviationFiscalYear"].IfIsEmptyThen(0); }
                break;
        }
    }
    return MoneyReplacementPed;
}
```

Рисунок 36 – Пример неаннотированного метода в модуле «Межбюджетные трансферты»[30]

– В исходном коде проекта наблюдается недостаточное разделение на модули и классы. Вместо этого, для многих подсистем проекта созданы всего 1-2 класса, в которых сосредоточены все методы. Это приводит к тому, что совокупный код класса по проекту выходит на большое количество строк. Такая структура затрудняет понимание кода и усложняет процесс его модификации и сопровождения. Кроме того, в одном классе собраны все методы для каждой таблицы, что может привести к конфликтам при одновременной работе нескольких сотрудников. Например, аналитики, работающие над макросами данного проекта, могут случайно перезаписать изменения друг друга, что снижает эффективность работы и увеличивает риск возникновения ошибок. На Рисунок 37 можно увидеть часть методов в классе MbtForms для модуля «Межбюджетные трансферты».

30 Составлено автором по: [51]

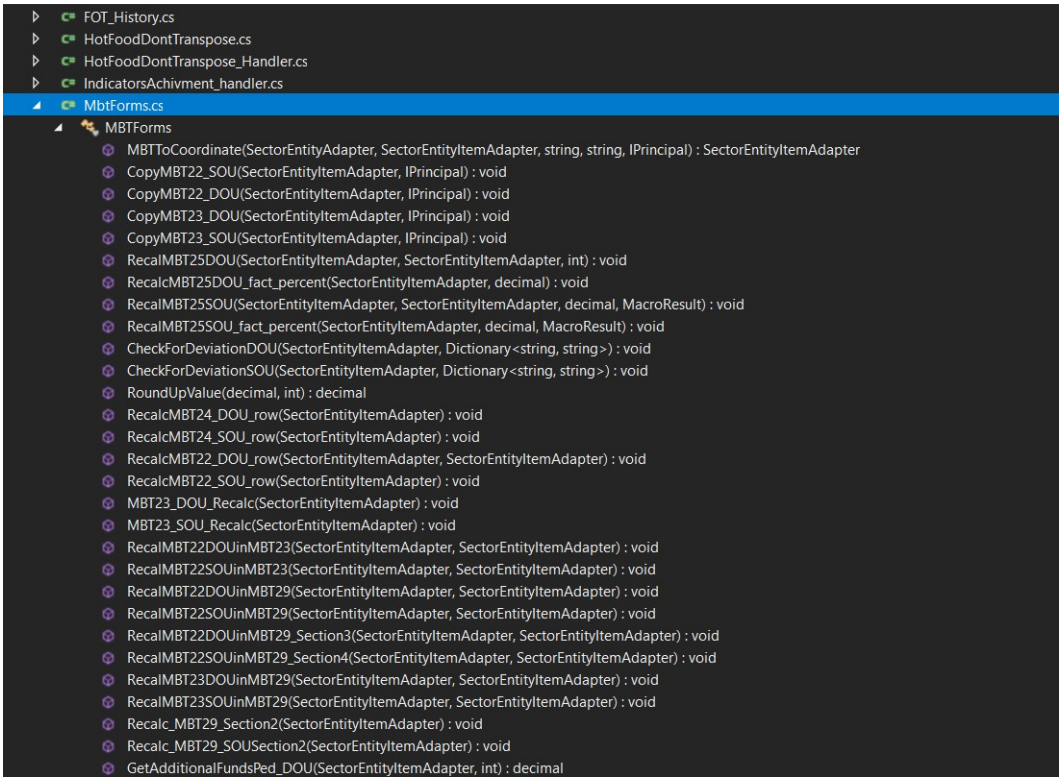


Рисунок 37 – Методы класса MbtForms для модуля «Межбюджетные трансферты»[31]

– Большие классы с множеством методов и недостаточно модульная структура проекта усложняют процесс обнаружения и решения проблем. Изменения в одной части кода могут нежелательно повлиять на другие части системы, что увеличивает риск возникновения ошибок.

– Наличие дублирующего кода, который можно было бы выделить в отдельный метод и в нужных местах просто вызывать его, а не повторять каждый раз.

– Отсутствует стандартизация кодирования. Отсутствие единого стандарта форматирования кода может привести к тому, что разные разработчики используют различные стили кодирования, что затрудняет чтение и понимание кода другими членами команды.

– Отсутствует документация и объяснения структуры проекта, что также замедляет процесс реагирования на проблемы.

31 Составлено автором по: [51]

Все вышеперечисленные пункты негативно влияют на основные бизнес-процессы ООО «Сапфир-Интеграция», что подтверждает актуальность изменений в исходном коде проекта, а также разработки авторского алгоритма управления внесением изменений в исходный код.

3.2 Реализация и тестирование

Для улучшения структуры кода системы, а также для автоматизации процесса сопровождения информационной системы ПК «ИСУФ» в части сбора консолидированной отчетности исходный код проекта был разбит на отдельные классы с методами. В работе рассматривается тестирование системы по модулю «Межбюджетные трансферты» (МБТ).

Данный проект включает в себя 10 форм (Рисунок 38), каждая из которых представляет собой таблицы с данными и требует отдельного макроса для обработки изменений в этих данных. Прежняя структура проекта, в которой все методы для каждой формы находились в одном классе, приводила к значительным трудностям при совместной работе, сопровождении и модификации кода.

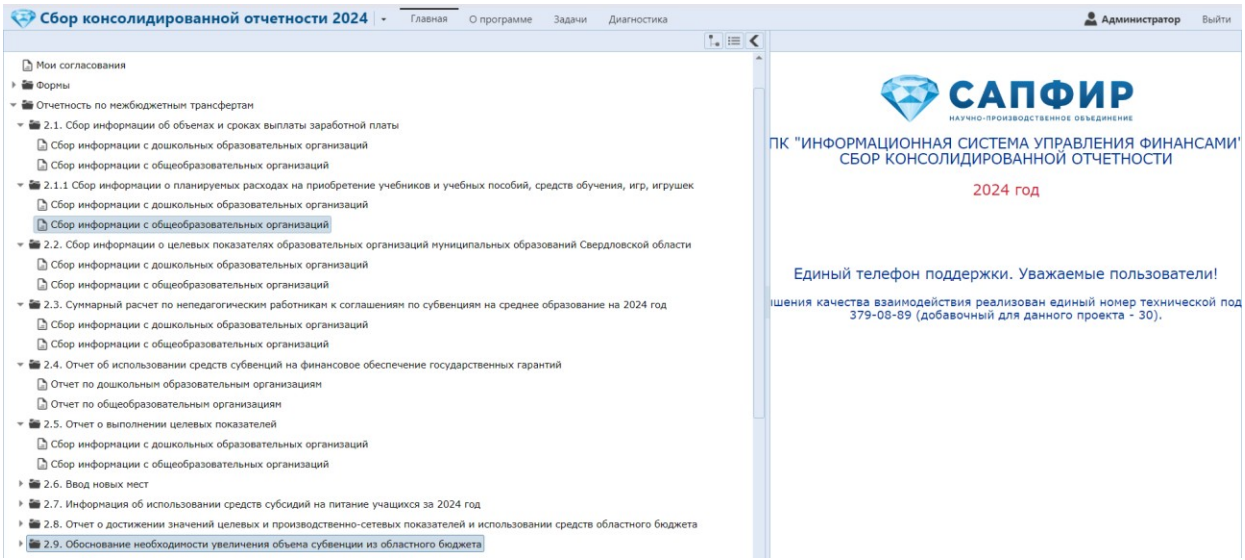


Рисунок 38 – Формы модуля «Межбюджетные трансферты»[32]

32 Составлено автором по: [51]

Новая структура кода предусматривает создание отдельных классов для каждой формы в проекте МБТ, что позволяет лучше организовать код и облегчает его поддержку. Пример структуры классов для форм проекта МБТ выглядит следующим образом (Рисунок 39):

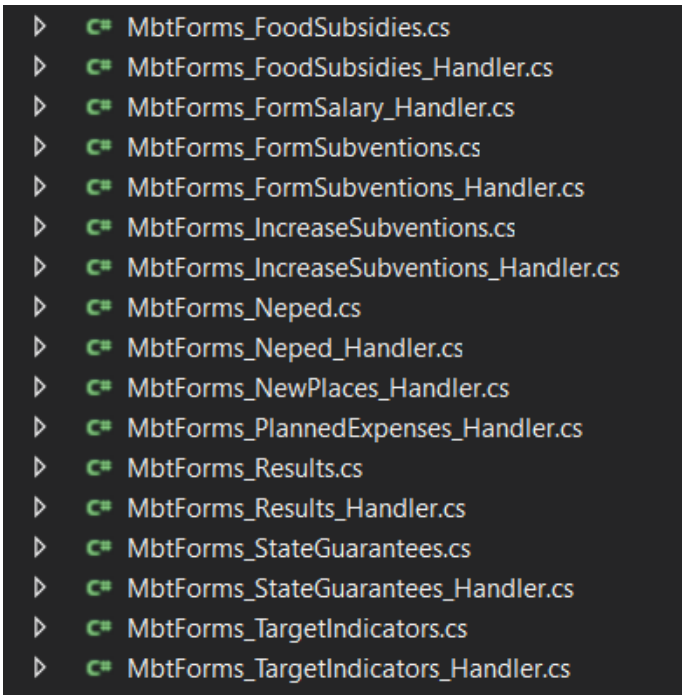


Рисунок 39 – Классы для модуля «Межбюджетные трансферты»[33]

Все наименования классов начинаются с системного имени проекта. После слеша идет системное имя таблицы (сущности). Для каждой формы создается два класса:

– Класс для обработки данных формы (Handler). Класс с суффиксом _Handler отвечает за обработку текущей строки данных в форме (Рисунок 40). Этот класс включает методы, которые вызываются при внесении изменений в данные формы. Основная задача этого класса – подготовка данных и передача параметров в основной класс формы для дальнейшей обработки.

[33] Составлено автором по: [51]

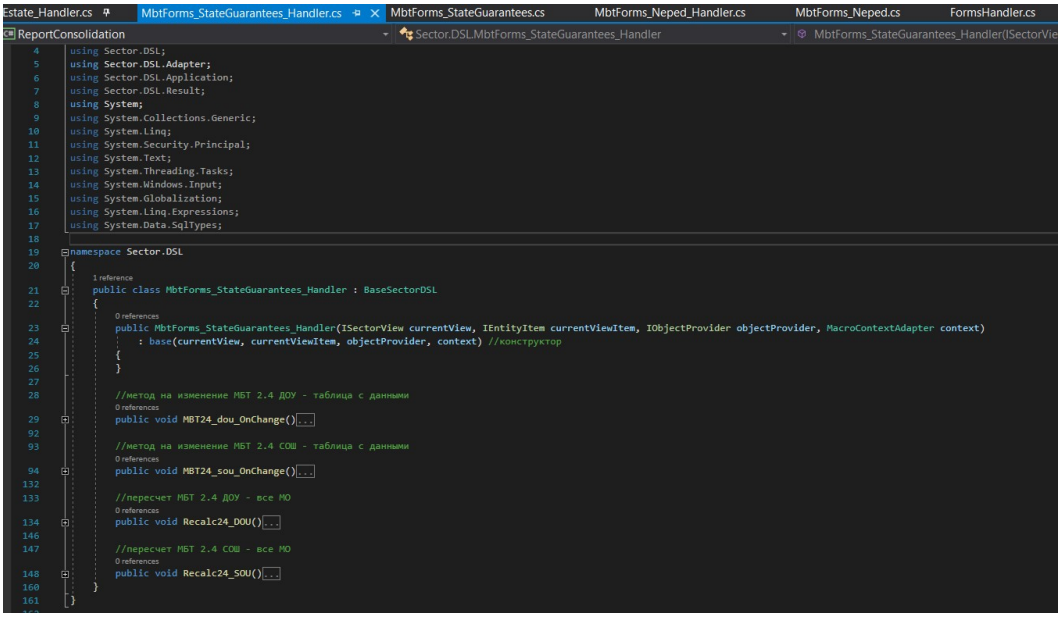


Рисунок 40 – Пример структуры класса MbtForms_StateGuarantees_Handler [34]

– Основной класс для формы, без суффикса _Handler, содержит методы для выполнения всех необходимых расчетов и бизнес-логики, связанных с данной формой (Рисунок 41). В этом классе определяются все необходимые методы для работы с данными формы, что делает код более модульным и легче поддерживаемым.

Рисунок 41 – Пример структуры класса MbtForms_StateGuarantees[35]

[34] Составлено автором по: [51]

[35] Составлено автором по: [51]

Для управления изменениями и автоматического внесения их в код был разработан интерфейс «Свойство проекта» (Рисунок 42). Этот интерфейс позволяет изменять функционал проекта прямо на сайте, без необходимости вмешательства в исходный код.

Свойства проекта

Наименование свойства	Значение свойства (текст)
Закон о бюджете	от 07.12.2023 № 128-ОЗ «Об областном бюджете на 2024 год и плановый период 2025 и 2026 годов»
Год	2024
Документооборот и ЭЦП	
Детализация ЗПК	
Использование листа согласования	
Форма ЗНП	
Детализация ЗНП	
Путь до папки Upload	C:\\ProjectServices\\MinFin24\\GZ\\DataSector\\Upload\\

Рисунок 42 – Раздел в web-сайте «Свойство проекта»[36]

Таблица «Свойство проекта» состоит из следующих атрибутов:

1) Наименование свойства – название параметра, который можно изменять.

2) Значение свойства – текущее значение данного параметра.

Для работы с данной таблицей был написан класс макросов ProjectProperties. У каждого свойства в таблице есть свое представление в web-интерфейсе, которые открывается с помощью метода OpenForm в данном классе в зависимости от наименования свойства проекта (Рисунок 43).

```
46 references
class ProjectProperties
{
    //Метод ищет свойство с наименованием и возвращает его EIID
    6 references
    public SectorEntityItemAdapter GetProperty(ObjectProviderAdapter application, string propertyName)...

    //Метод определяет наименование формы для редактирования по типу свойства
    1 reference
    public string GetPropertyType(SectorEntityItemAdapter itemProjectProperties)
    {
        ObjectProviderAdapter application = itemProjectProperties.Context;

        SectorEntityAdapter PropertyType = application.Entities["PropertyType"];
        SectorEntityItemAdapter rowPropertyType = PropertyType.GetItem(itemProjectProperties["PropertyTypeID"]);
        string formName = "";
        formName = (string)rowPropertyType["FormPropertyType"];
        if (formName.IsEmpty())
        {
            formName = "F_ProjectProperties_New";
        }

        return formName;
    }
```

Рисунок 43 – Метод для определения наименования формы для редактирования по типу свойства[37]

[36] Составлено автором по: [42]
[37] Составлено автором по: [42]

3.3 Анализ результатов тестирования и сравнение с предыдущим состоянием

После реализации новой структуры кода была проведена серия тестов для проверки корректности работы модулей и их взаимодействия. Тестирование включало следующие этапы:

- Разработка и выполнение юнит-тестов для каждого класса, чтобы удостовериться в корректности работы всех методов и функциональности. Это помогло выявить и исправить ошибки на ранних этапах.
- Тестирование взаимодействия между классами для каждой формы, чтобы убедиться в правильной передаче данных и корректной работе всей цепочки обработки и расчетов.
- Оценка производительности новой структуры кода по сравнению с предыдущей, что включало в себя измерение времени выполнения операций и потребления мощностей серверов.
- Проверка всей системы на предмет появления новых ошибок после внесения изменений.

Результаты тестирования подтвердили эффективность и правильность новой структуры исходного кода, показав улучшение в производительности, читаемости и поддерживаемости кода.

Разделение кода на отдельные классы для каждой формы делает проект более понятным и структурированным. Это облегчает разработчикам понимание логики и функциональности каждого модуля. Благодаря разбиению на модули, каждый разработчик может работать над своим классом, не рискуя перезаписать изменения, внесенные другими. Это значительно снижает вероятность конфликтов и ошибок при коллективной работе. При необходимости добавления новой формы или изменения существующей логики достаточно создать или модифицировать соответствующий класс, не затрагивая весь проект. Это упрощает процесс внедрения новых функциональных возможностей и делает проект более

гибким. Также модульная структура и четко определенные обязанности классов позволяют быстрее вносить изменения и исправлять ошибки. Это особенно важно для динамичных проектов, где постоянно меняются требования.

Интерфейс «Свойство проекта» позволяет легко и быстро вносить изменения в код, что существенно экономит время и ресурсы на поддержку проекта.

Одной из наиболее частых задач в сопровождении данной информационной системы является обновление года в названии форм при открытии сайта для следующего года. Ранее для этого требовалось внести изменения для каждой таблицы в нужный макрос, базу данных и в интерфейс формы. Теперь, благодаря интерфейсу «Свойство проекта», достаточно изменить значение свойства «Год» с предыдущего года на текущий, и это изменение автоматически будет применено ко всем формам проекта (Рисунок 44).

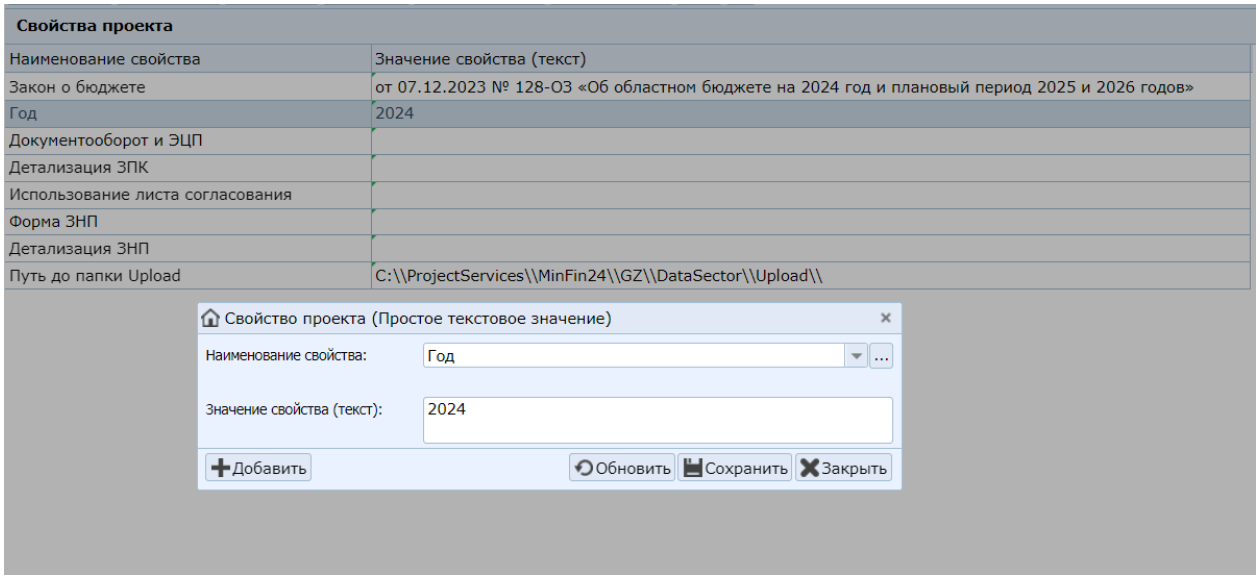


Рисунок 44 – Свойство проекта «Год»[38]

Также частым обновлением в проекте является добавление листа согласования в форму, это необходимо для некоторых таблиц, где требуется согласование несколькими сотрудниками министерства.

[40] Составлено автором по: [51]

Чтобы каждый раз не создавать связи между необходимой сущностью и сущностью с листом согласования был написан класс ProjectProperties_GRBS_Handler с методом GenerationLSInAllGZ для генерации таких листов для сотрудников (Рисунок 45).

```
//Метод генерирует листы согласования по ГРБС, указанному в свойстве, для всех разделов ГЗ, которые были ранее созданы
0 references
public void GenerationLSInAllGZ()
{
    SectorViewItemAdapter AR = (SectorViewItemAdapter)ActiveRow;
    SectorEntityItemAdapter eAR = (SectorEntityItemAdapter)AR.EntityItem;

    SectorEntityAdapter GZ = Entities["Reestr_yslyg"];
    SectorEntityAdapter GRBS = Entities["GRBS"];
    SectorEntityAdapter LS = Entities["Coordination_List"];

    //Определяем код ГРБС
    object gRBSID = eAR["GRBSID"];
    SectorEntityItemAdapter rowGRBS = GRBS.GetItem(gRBSID);
    GRBS objGRBS = new GRBS();
    string codeGRBSfromLS = objGRBS.GetCodeGRBS(rowGRBS);
    if (codeGRBSfromLS == "")
        throw new Exception("Не определён код ГРБС по свойству. Обратитесь к администартору");

    Reestr_yslyg objGZ = new Reestr_yslyg();
    Coordination_List objLS = new Coordination_List();
    int countGZofGRBS = 0;
    int countGZofGRBSWhereAddLS = 0;
    foreach (SectorEntityItemAdapter rowGZ in GZ.GetRecords())
    {
        //Определяем код ГРБС по разделу ГЗ (через услугу)
        string codeGRBS = objGZ.GetCodeGRBS(rowGZ);
        if(!codeGRBS.IsEmpty())
        {
            if (codeGRBS == codeGRBSfromLS)
            {
                countGZofGRBS++;
```

Рисунок 45 – Часть метода для генерации листов согласования[39]

Теперь листы согласования генерируется автоматически на основе значений, указанных в «Свойстве проекта» (Рисунок 46).

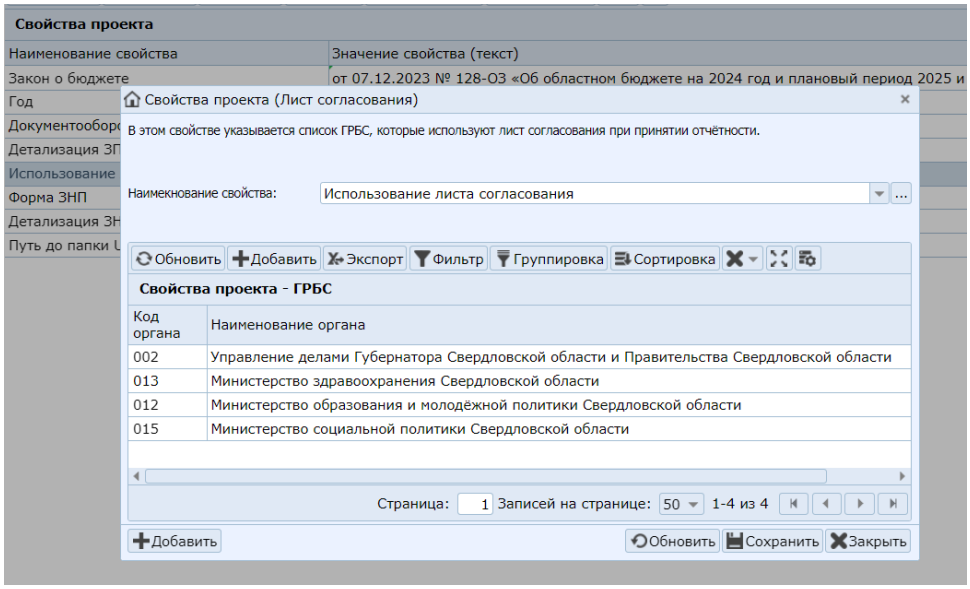


Рисунок 46 – Свойство проекта «Использование листа согласования»[40]

[39] Составлено автором по: [51]

[40] Составлено автором по: [51]

Достаточно указать нужные роли государственных органов в соответствующем свойстве, и система автоматически создаст листы согласования для указанных ролей.

На рисунке 47 можно увидеть, что в разделе государственных (муниципальных) заданий для организаций, относящихся к Министерству образования и молодежной политики Свердловской области (код органа: 012), созданы листы согласования.

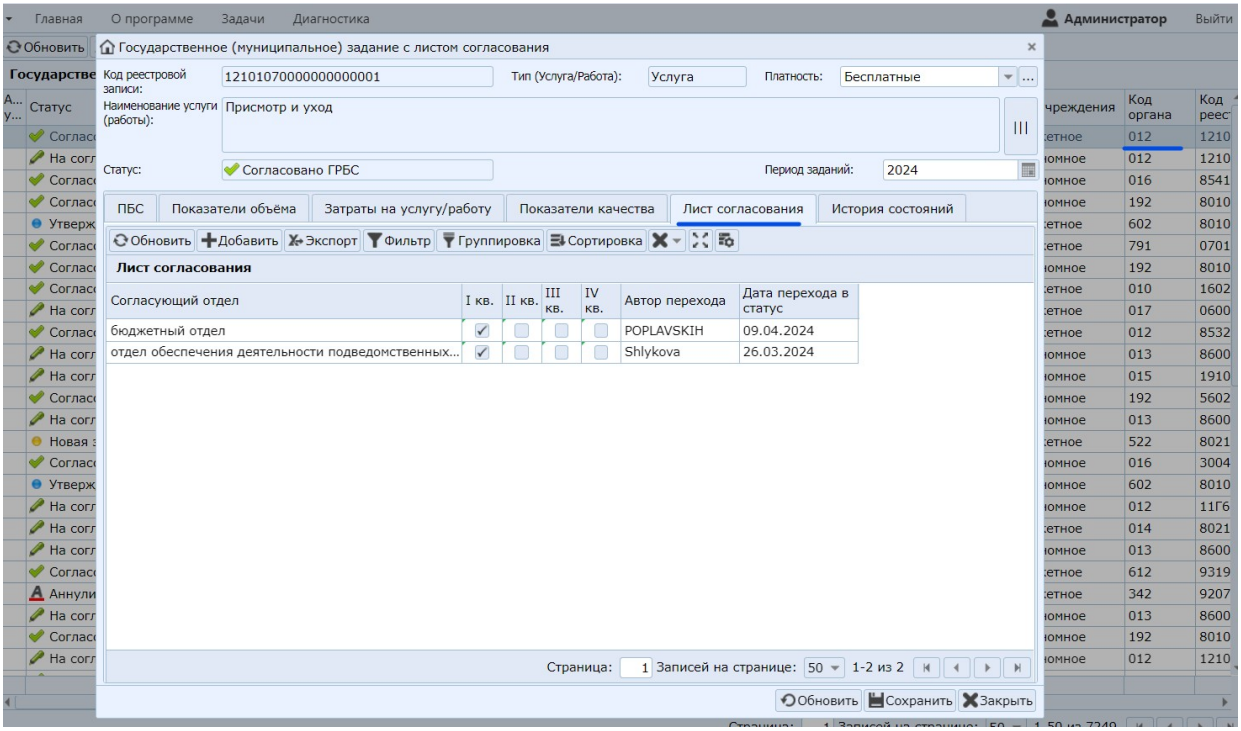


Рисунок 47 – Результат тестирования web-интерфейса «Свойство проекта»[41]

Использование интерфейса «Свойство проекта» упрощает процесс управления изменениями и автоматического внесения их в код. Это сокращает время на сопровождение информационной системы и позволяет сотрудникам сосредоточиться на более важных задачах, таких как разработка нового функционала или оптимизация производительности системы.

Даже сотрудники, не обладающие глубокими знаниями в программировании, теперь могут вносить изменения в систему через интерфейс «Свойство проекта», что повышает общую эффективность работы.

[40] Составлено автором по: [51]

3.4 Оценка экономической эффективности изменений

В данном разделе проводится оценка экономической эффективности изменений, внесенных в бизнес процессы ООО «Сапфир-Интеграция».

Принятие инвестиционных решений является сложной задачей для любой бизнес-структуры, и главным критерием является увеличение стоимости предприятия. Это достигается за счет роста доходов компании, снижения финансовых рисков или производственных затрат, а также за счет повышения эффективности работы компании [69].

Эффективность проекта оценивается в течение временного интервала, в котором планируются и оцениваются финансовые показатели проекта, включая доходы, расходы и инвестиции. Такой временной интервал называется бюджетным периодом. Бюджетный период обычно начинается с момента начала вложения средств в проект, например, с даты начала проектно-изыскательских работ, и продолжается до момента завершения проекта или до установленного срока окупаемости инвестиций. Для более детального анализа и управления финансовыми потоками бюджетный период разбивается на шаги. Шаги могут быть равными временными отрезками, такими как месяцы, кварталы или годы. Разбиение на шаги помогает следить за выполнением плана, проводить промежуточные оценки и вносить корректировки в случае необходимости.

Денежные потоки являются одним из ключевых аспектов при оценке экономической эффективности проекта. Они представляют собой движение денежных средств, связанное с различными операциями в проекте на протяжении всего его жизненного цикла.

Оценка эффективности проектов осуществляется на основе двух критериев: финансовой и экономической оценки. Эти два критерия взаимно дополняют друг друга [44].

Финансовая оценка используется в анализе ликвидности в процессе реализации инвестиционного проекта. Другими словами, задача финансовой

оценки – определить, будут ли у предприятия достаточные финансовые ресурсы для выполнения всех финансовых обязательств в рамках проекта в установленный срок.

Экономическая оценка используется для определения потенциальной способности инвестиционного проекта сохранить и увеличить стоимость вложенных средств. На практике, методы оценки экономической эффективности инвестиционных проектов можно разделить на две большие группы: простые (или статистические) методы и методы оценки на основе дисконтирования (Рисунок 48).

Рисунок 48 – Классификация методов оценки экономической эффективности инвестиционных проектов [28]

К простым методам относятся: срок окупаемости и учетная норма прибыли.

Срок окупаемости (Payback Period, PP) определяет период, в течение которого инвестиции будут возмещены за счет чистых денежных потоков. Формула нахождения срока окупаемости инвестиций выглядит следующим образом (формула (2)):

$$PP = \frac{K_o}{C_{cr}} \quad (2)$$

где K_o – первоначальные инвестиции;

CF_{cr} – среднегодовой приток денежных средств.

Учетная норма прибыли (Accounting Rate of Return, ARR) оценивает прибыльность проекта в процентах от первоначальных инвестиций и рассчитывается по формуле 3.

$$ARR = \frac{C_F}{K_o} \quad (3)$$

К методам на основе дисконтирования относятся: чистая приведенная стоимость, внутренняя норма доходности, индекс рентабельности, дисконтированный срок окупаемости.

Одним из наиболее часто используемых показателей для расчета экономической эффективности проекта является чистая приведенная стоимость (Net Present Value, NPV) – разница между текущей стоимостью денежных потоков и первоначальными инвестициями. Если NPV больше 0, то проект считается экономически эффективным. Если NPV меньше 0, то доходы от предложенных вложений не компенсируют риск, присущий данному проекту. Показатель рассчитывается по формуле 4.

$$NPV = \sum_{n=1}^{m} \frac{C_n}{(1+r)^n} - \sum_{k=0}^{t} \frac{I_k}{(1+r)^k} \quad (4)$$

где C_n – объем генерируемых денежных средств в периоде n;

I_k – инвестиции в проект в периоде k;

r – ставка дисконтирования;

n – момент времени поступления денежных средств;

m – продолжительность периода действия проекта.

Внутренняя норма доходности (Internal Rate of Return, IRR) – ставка дисконтирования, при которой NPV равен нулю. Проект считается

экономически эффективным, если IRR больше ставки дисконтирования. Показатель рассчитывается по формуле 5.

$$IRR = r_1 + \frac{NPV_{(r1)}}{NPV_{(r1)} - NPV_{(r2)}} * (r_2 - r_1) \quad (5)$$

где r_1 – ставка дисконтирования при $NPV_{(r1)} > 0$;

r_2 – ставка дисконтирования при $NPV_{(r1)} < 0$.

Индекс рентабельности (Profitability Index, PI) – отношение текущей стоимости будущих денежных потоков к первоначальным инвестициям. Проект принимается, если PI>1. Показатель рассчитывается по формуле 6.

$$PI = \frac{\sum_{n=1}^{m} \frac{C_n}{(1+r)^n}}{\sum_{k=1}^{t} \frac{I_k}{(1+r)^k}} \quad (6)$$

Дисконтированный срок окупаемости (Discounted Payback Period, DPP) – время, необходимое для возврата инвестиций с учетом дисконтирования. В целом, чем меньше срок окупаемости, тем более эффективным является проект.

Методы, основанные на дисконтировании чистых денежных потоков, чаще применяются в практике оценки экономической эффективности инвестиционных проектов, так как они учитывают временную стоимость денег, а также лучше оценивают риски и неопределенности [28].

В контексте данной работы доходы от внедрения проекта можно оценить в виде экономии времени и повышения эффективности. Время – это ценный ресурс, и его экономия напрямую влияет на производительность и снижение операционных затрат, которые можно оценить в денежном эквиваленте.

Поэтому оценка эффективности будет основываться на следующих аспектах:

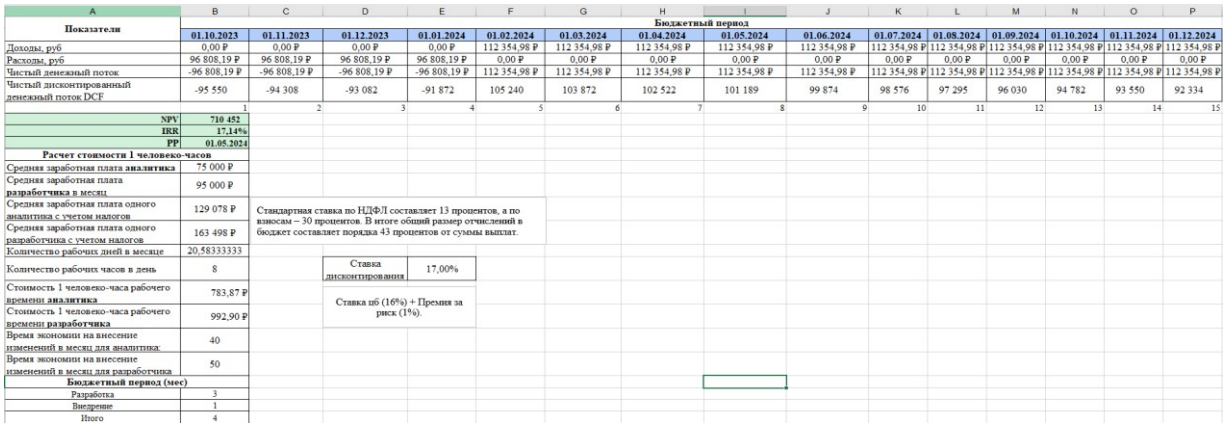
- экономия времени на сопровождение;
- экономия в денежном эквиваленте;
- затраты на разработку и интеграцию системы.

За счет автоматизации и улучшения процессов, экономия может быть оценена в снижении времени на выполнение задач для каждого сотрудника: аналитиков и разработчиков. В команде данных проектов 1 разработчик и 2 аналитика, поэтому доходы будут рассчитываться с них. Доходы пойдут с 5 месяца, после внедрения. Реализацией проекта занимался 1 аналитик на протяжении 4 месяцев (6 часов в день занимала разработка, остальное время - текущие задачи).

Для оценки экономической эффективности изменений была использована ставка дисконтирования, которая составляет 17%. Показатель был рассчитан на основе следующих компонентов:

- ключевая ставка Центрального банка России ЦБ РФ, которая является хорошим индикатором безрисковой ставки, так как она отражает минимальную доходность, которую инвесторы могут получить от государственных облигаций, считающихся безрисковыми (на момент расчета ключевая ставка ЦБ РФ составляет 16%);
- премия за риск для ИТ-проектов составляет 1% (значение учитывает специфические риски, связанные с технологическими изменениями, рыночной волатильностью и возможными изменениями регуляторной среды).

Таблица с расчетами в формате Excel представлена на рисунке 49.



Показатели	Бюджетный период														
	01.10.2023	01.11.2023	01.12.2023	01.01.2024	01.02.2024	01.03.2024	01.04.2024	01.05.2024	01.06.2024	01.07.2024	01.08.2024	01.09.2024	01.10.2024	01.11.2024	01.12.2024
Доходы, руб	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽
Расходы, руб	96 808,19 ₽	96 808,19 ₽	96 808,19 ₽	96 808,19 ₽	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽	0,00 ₽
Чистый денежный поток	-96 808,19 ₽	-96 808,19 ₽	-96 808,19 ₽	-96 808,19 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽	112 354,98 ₽
Чистый дисконтированный денежный поток DCF	-95 550	-94 308	-93 082	-91 872	105 240	103 872	102 522	101 189	99 874	98 576	97 295	96 030	94 782	93 550	92 334
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

NPV	710 452
IRR	17,14%
PP	01.05.2024
Расчет стоимости 1 человеко-часов	
Средняя заработная плата **аналитика**	75 000 ₽
Средняя заработная плата **разработчика** в месяц	95 000 ₽
Средняя заработная плата одного аналитика с учетом налогов	129 078 ₽
Средняя заработная плата одного разработчика с учетом налогов	163 498 ₽
Количество рабочих дней в месяце	20,58333333
Количество рабочих часов в день	8
Стоимость 1 человеко-часа рабочего времени **аналитика**	783,87 ₽
Стоимость 1 человеко-часа рабочего времени **разработчика**	992,90 ₽
Время экономии на внесение изменений в месяц для аналитика:	40
Время экономии на внесение изменений в месяц для разработчика	50
Бюджетный период (мес)	
Разработка	3
Внедрение	1
Итого	4

Стандартная ставка по НДФЛ составляет 13 процентов, а по взносам – 30 процентов. В итоге общий размер отчислений в бюджет составляет порядка 43 процентов от суммы выплат.

Ставка дисконтирования	17,00%

Ставка цб (16%) + Премия за риск (1%).

Рисунок 49 – Расчеты экономической эффективности изменений в Excel[42]

Для оценки экономической эффективности изменений была проведена комплексная оценка, включающая расчеты доходов, расходов и дисконтированных денежных потоков (DCF) в течение 15-месячного бюджетного периода с 01.10.2023 по 01.12.2024. Отдельно результаты расчетов представлены в таблице 4.

Таблица 4 – Результаты расчетов экономической эффективности[43]

Показатель	Значение показателя
Чистая приведённая стоимость, NPV	710 452
Внутренняя норма доходности, IRR	17,14%
Период окупаемости, PP	8 месяцев

Результаты оценки экономической эффективности изменений показывают, что проект является финансово обоснованным и прибыльным. Рассчитанные показатели NPV, IRR и срок окупаемости подтверждают целесообразность внедрения изменений:

– положительное значение NPV указывает на то, что проект приносит чистую прибыль;

– внутренняя норма доходности превышает ставку дисконтирования, что говорит о высокой доходности проекта;

[42] Составлено автором по: [51]
[43] Составлено автором по: [51]

– проект окупается к маю 2024 года, что составляет 8 месяцев с начала реализации и 4 месяца после начала поступления доходов.

3.5 Результаты и выводы по разделу 3

В третьем разделе была выполнена комплексная работа по улучшению структуры исходного кода, внедрению новых функциональных возможностей и проведению оценки экономической эффективности внесенных изменений в бизнес-процессы ООО «Сапфир-Интеграция».

В процессе разработки новой структуры исходного кода была проведена значительная работа по рефакторингу существующих проектов. Основным фокусом стало выделение общих методов в отдельные классы и структурирование кода по модульному принципу. Для каждой формы были созданы отдельные классы с методами и комментариями по ним, что позволило:

– уменьшить объем кода в отдельных классах;

– упростить понимание и поддержку кода;

– снизить вероятность возникновения конфликтов при одновременной работе нескольких сотрудников над проектом.

Внедрен интерфейс «Свойство проекта», позволяющий автоматизировать процесс внесения изменений в код. Это упростило работу для аналитиков и других сотрудников, не владеющих навыками программирования на C#.

Анализ результатов тестирования показал значительное улучшение следующих показателей:

– сокращение времени на внесение изменений в код за счет автоматизации и структурирования;

– уменьшение количества ошибок благодаря более ясной и понятной структуре кода;

– повышение производительности труда сотрудников за счет разделения обязанностей и снижения числа конфликтов при одновременной работе над проектом.

Результаты оценки экономической эффективности изменений показывают, что проект является финансово обоснованным и прибыльным. Рассчитанные показатели NPV, IRR и срок окупаемости подтверждают целесообразность внедренных изменений.

ЗАКЛЮЧЕНИЕ

Основная цель работы состояла в повышении эффективности основных бизнес-процессов в организациях-разработчиках ПО за счет внедрения новых инструментов проектного управления.

Для достижения поставленной цели в начале исследования были проанализированы теоретические основы цифровизации процессов проектного управления, а также изучена специфика деятельности и проблематика в проектном управлении ООО «Сапфир-Интеграция» – дочерней компании НПО «Сапфир» в области разработки ПО. Было установлено, что проекты данной организации характеризуются изменяющимися требованиями. В связи с этим, к проектам компании применимы гибкие методологии разработки. В ходе анализа было решено применить модель TOGAF 10 для построения сервисной архитектуры предприятия. Выбор TOGAF 10 обусловлен его универсальностью и доступностью.

Во втором разделе был дана общая характеристика ООО «Сапфир-Интеграция», рассмотрены основные проектные направления компании, а также изучен разработанной программный комплекс «Информационная система управления финансами» (ПК «ИСУФ»). Был проведен анализ текущего состояния архитектуры ООО «Сапфир-Интеграция» по модели TOGAF 10. В контексте данного проекта были построены:

- бизнес-архитектура;
- архитектура данных;
- техническая архитектура;
- архитектура приложений;
- архитектура безопасности.

После подробного ознакомления с бизнес-процессами предприятия, была реализована и внедрена новая система аналитических отчетов, которая

функционирует на основе прямого взаимодействия с базой данных MySQL и автоматизации выгрузки данных с помощью макросов на C#. Данная система направлена на улучшение основного бизнес-процесса ООО «Сапфир-Интеграция» – разработку ПО, а также на устранение legacy-системы в компании. Данная система позволяет:

– упростить процесс разработки аналитических сводных расчетов;

– увеличить точность данных в представляемых отчетах;

– повысить скорость загрузки отчетности на web-сайте, что повлияет на лояльность пользователей;

– снизить нагрузку на сервер;

– упростить сопровождение проекта.

В третьем разделе была выполнена комплексная работа по улучшению структуры исходного кода, внедрению новых функциональных возможностей и проведению оценки экономической эффективности всех внесенных изменений в бизнес-процессы ООО «Сапфир-Интеграция».

В процессе разработки новой структуры исходного кода была проведена значительная работа по рефакторингу существующих проектов. Основным фокусом стало выделение общих методов в отдельные классы и структурирование кода по модульному принципу. Для каждой формы были созданы отдельные классы с методами и комментариями по ним. Данные изменения способствуют:

– улучшению читаемости и поддержки кода;

– снижению вероятности возникновения ошибки при совместной работе над проектом;

– ускорению процесса разработки.

Также был внедрен новый интерфейс «Свойство проекта», позволяющий автоматизировать процесс внесения изменений в код. Плюсами данной разработки являются:

– автоматизация процесса внесения изменений;

- упрощение управления проектами;
- повышение производительности команды;
- быстрая адаптация к изменениям.

Внедрение изменений способствовало значительной экономии времени сотрудников и повысило эффективность работы команды, что в конечном итоге привело к положительному финансовому результату. По итогам проведенной работы была достигнута цель исследования, а также выполнены задачи, поставленные для ее выполнения.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Аналитика от Notamedia.Integrator. Рынок разработки в РФ 2023: [сайт]. – URL: https://integrator.nota.media/blog/analytics/robot/ (дата обращения: 04.05.2024). – Текст: электронный.

2. Андирякова О. О., Крюкова А. А., Иваев М. И. Применение LOW-CODE технологий для решения бизнес-задач. – Текст: электронный // Индустриальная экономика. – 2023. – №2. – С. 20-24 – URL: https://cyberleninka.ru/article/n/primenenie-low-code-tehnologii-dlya-resheniya-biznes-zadach (дата обращения: 05.05.2024).

3. Андреева В.В., Самохина С.И., Петелин А.Е. Программирование на языке C#: учебное пособие. – Томск: Издательский Дом Томского государственного университета, 2019. – 110 с.

4. АРПП Отечественный софт: [сайт]. – URL: https://arppsoft.ru/members/9227/ (дата обращения: 30.04.2024). – Текст: электронный.

5. Багдыков К.Т., Шевченко Д.А., Муромец Н.Е. Инструменты проектного управления в цифровой трансформации бизнеса. – Текст: электронный // Информатизация в цифровой экономике. – 2023. – С. 325-338. – URL: https://elibrary.ru/item.asp?id=59559060 (дата обращения 21.04.2024)

6. Брусов А. С. КОНЦЕПЦИЯ AGILE: ВОЗМОЖНОСТИ И ПЕРСПЕКТИВЫ ПРИМЕНЕНИЯ В ГОСУДАРСТВЕННОМ УПРАВЛЕНИИ (ОБЗОР ПУБЛИКАЦИЙ). – Текст: электронный // Вопросы государственного и муниципального управления. – 2022. – №2. – С. 134-158. – URL: https://cyberleninka.ru/article/n/kontseptsiya-agile-vozmozhnosti-i-perspektivy-primeneniya-v-gosudarstvennom-upravlenii-obzor-publikatsiy (дата обращения: 01.04.2024).

7. Буч Г., Рамбо Д., Джекобсон А. Язык UML. Руководство пользователя. – пер. с англ. – ДМК Пресс, 2000.

8. Васин Д. В. Проектирование сервисной архитектуры цифровой платформы для предприятия розничной торговли. – Текст: электронный // Научные записки молодых исследователей. – 2020. – №2. – С. 78-86 – URL: https://cyberleninka.ru/article/n/proektirovanie-servisnoy-arhitektury-tsifrovoy-platformy-dlya-predpriyatiya-roznichnoy-torgovli (дата обращения: 29.04.2024).

9. Воронова О.В., Ильин И.В. АДАПТАЦИЯ ADM МЕТОДА СТАНДАРТА TOGAF К РАЗРАБОТКЕ И ВНЕДРЕНИЮ АРХИТЕКТУРНЫХ РЕШЕНИЙ В СФЕРЕ СЕТЕВОГО FMCG-РИТЕЙЛА. – Текст: электронный // Экономика и управление. – 2019. – №7 (165). – С. 97-107 – URL: https://cyberleninka.ru/article/n/adaptatsiya-adm-metoda-standarta-togaf-k-razrabotke-i-vnedreniyu-arhitekturnyh-resheniy-v-sfere-setevogo-fmcg-riteyla (дата обращения: 05.05.2024).

10. Ворсин В. А. МИКРО-СЕРВИСНАЯ АРХИТЕКТУРА БИЗНЕС-ПРИЛОЖЕНИЙ – ПЕРСПЕКТИВЫ И ПРОБЛЕМЫ. – Текст: электронный // Глобус. –2020. – №4 (50). – С. 50-52 – URL: https://cyberleninka.ru/article/n/mikro-servisnaya-arhitektura-biznes-prilozheniy-perspektivy-i-problemy (дата обращения: 29.04.2024).

11. Всероссийская система проверки контрагентов: [сайт]. – URL: https://zachestnyibiznes.ru/company/ul/1136671021772_6671429046_OOO-SAPFIR--INTEGRACIYa (дата обращения: 30.04.2024). – Текст: электронный.

12. Г. И. Абдрахманова, С. А. Васильковский, К. О. Вишневский, Л. М. Гохберг и др. Индикаторы цифровой экономики: 2022 – статистический сборник – Нац. исслед. ун-т «Высшая школа экономики». – М. НИУ ВШЭ, 2023. – 332 с. – URL: https://issek.hse.ru/news/780811313.html (дата обращения 20.04.2024). – Текст: электронный.

13. Гагарина Л. Г., Кокорева Е. В., Виснадул Б. Д. Технология разработки программного обеспечения. – пер. с англ. – ИД «Форум»: ИНФРА-М, 2008.

14. Гарант.ру: официальный сайт. – Москва. – URL: https://www.garant.ru/article/1605871/ (дата обращения 20.04.2024). – Текст: электронный.

15. Григорьева, П. А. Архитектура оценочной компании ООО «Консалт». – Текст: электронный // Мавлютовские чтения: Материалы XIV Всероссийской молодежной научной конференции. – 2020. – Том 7. Часть 2. – С. 12. – URL: https://www.elibrary.ru/item.asp?id=44649626 (дата обращения: 05.05.2024).

16. Грубич Т. Ю., Шролик А. В. Анализ архитектуры предприятия. – Текст: электронный // Научный журнал КубГАУ. – 2014. – №104. – URL: https://cyberleninka.ru/article/n/analiz-arhitektury-predpriyatiya (дата обращения: 30.04.2024).

17. Дерябина Л.В., Немкина А.В. ОСОБЕННОСТИ УПРАВЛЕНИЯ IT ПРОЕКТАМИ. – Текст: электронный // Скиф. – 2022. – №4 (68). – С. 281-285. – URL: https://cyberleninka.ru/article/n/osobennosti-upravleniya-it-proektami (дата обращения: 15.02.2024).

18. Джаппарова Н. Л. Scrum - как эффективная методология управления проектами. – Текст: электронный // Скиф. – 2019. – №11 (39). – С. 681-684. – URL: https://cyberleninka.ru/article/n/scrum-kak-effektivnaya-metodologiya-upravleniya-proektami (дата обращения: 03.04.2024).

19. Ермаков А. С. Перспективное развитие методологии DEVOPS. – Текст: электронный // Вестник НГУЭУ. – 2020 – №4. – С. 174-183 – URL: https://cyberleninka.ru/article/n/perspektivnoe-razvitie-metodologii-devops (дата обращения: 29.04.2024).

20. Иванова И.А., Бардина А.С. СПИРАЛЬНАЯ ДИНАМИКА КАК МОДЕЛЬ РАЗВИТИЯ ОРГАНИЗАЦИИ. – Текст: электронный // ТДР. – 2022. – №2. – С. 88-90. – URL: https://cyberleninka.ru/article/n/spiralnaya-dinamika-kak-model-razvitiya-organizatsii (дата обращения: 01.04.2024).

21. Иванова Т. Н., Иванов Д. В. Классический и гибкие подходы к управлению проектами. – Текст: электронный // Бюллетень науки и практики.

– 2019. – №10. – С. 168-175. – URL: https://cyberleninka.ru/article/n/klassicheskiy-i-gibkie-podhody-k-upravleniyu-proektami (дата обращения: 14.02.2024).

22. Исключение дубликатов, DISTINCT: [сайт]. – URL: https://sql-academy.org/ru/guide/distinct-operator (дата обращения: 09.05.2024). – Текст: электронный.

23. К. Г. Сильченко, Н. А. Кривоносенко. Бизнес-процесс как основа процессного подхода. – Текст: электронный // Молодой ученый. – 2022. – № 20 (415). – С. 500-504. – URL: https://moluch.ru/archive/415/91607/ (дата обращения: 07.05.2024).

24. Как уральские компании замещают цифровые технологии: [сайт]. – URL: https://www.kommersant.ru/doc/6352878 (дата обращения: 04.05.2024). – Текст: электронный.

25. Каширина, О. И. Жизненный цикл тестирования программных систем. V-модель. – Текст: электронный // Новые информационные технологии в научных исследованиях: материалы XXI Всероссийской научно-технической конференции студентов, молодых ученых и специалистов, Рязань,. – 2016. – С. 135-136.– URL: https://www.elibrary.ru/item.asp?id=27312761 (дата обращения 01.04.2024)

26. Козлов С.В., Беляков П.В. Проблемы применения методологии AGILE при разработке программного обеспечения. – Текст: электронный // Сборник избранных статей по материалам научных конференций ГНИИ "НАЦРАЗВИТИЕ". – 2021. – С. 29-31. – URL: https://www.elibrary.ru/item.asp?id=47143664 (дата обращения 14.02.2024)

27. Козлова В. Н. ДОГОВОР ОБ УРОВНЕ ОБСЛУЖИВАНИЯ (SLA-СОГЛАШЕНИЕ): ПРАВОВАЯ ПРИРОДА И СУЩЕСТВЕННЫЕ УСЛОВИЯ. – Текст: электронный // Право и государство: теория и практика. – 2023. – №9 (225). – С. 292-294 – URL: https://cyberleninka.ru/article/n/dogovor-ob-urovne-obsluzhivaniya-sla-soglashenie-pravovaya-priroda-i-suschestvennye-usloviya (дата обращения: 29.04.2024).

28. Кононова Е.Е. МЕТОДЫ ОЦЕНКИ ЭФФЕКТИВНОСТИ ИНВЕСТИЦИОННЫХ ПРОЕКТОВ. – Текст: электронный // Исследование и практика в социально-экономической и гуманитарной сфере: сборник избранных статей Всероссийской (национальной) научно-практической конференции ГНИИ «Нацразвитие». – 2022. – С. 37-43. – URL: https://elibrary.ru/item.asp?id=48435949&pff=1 (дата обращения 14.05.2024)

29. Котляр Е. В., Пушкарева Е. М. Система управления проектами канбан. – Текст: электронный // Бизнес-образование в экономике знаний. – 2020. – №1 (15). – С. 57-59. – URL: https://cyberleninka.ru/article/n/sistema-upravleniya-proektami-kanban (дата обращения: 03.05.2024).

30. Кто такой DevOps-инженер: [сайт]. – URL: https://practicum.yandex.ru/blog/kto-takoy-devops-inzhener/ (дата обращения: 29.04.2024). – Текст: электронный.

31. М. А. Мирошниченко, Е. О. Голобородько, И. Н. Сарычева Методология эффективного управления на основе принципов бережливого производства. – Текст: электронный // Вестник Академии знаний. – 2020. – №2 (37). – С. 178-183 – URL: https://cyberleninka.ru/article/n/metodologiya-effektivnogo-upravleniya-na-osnove-printsipov-berezhlivogo-proizvodstva (дата обращения: 29.04.2024).

32. Магомадов В.С. Платформы LOW-CODE и NO-CODE как способ сделать программирование более доступным для широкой общественности». – Текст: электронный // МНИЖ. – 2021. – №6 (108). – С. 100-103 – URL: https://cyberleninka.ru/article/n/platformy-low-code-i-no-code-kak-sposob-sdelat-programmirovanie-bolee-dostupnym-dlya-shirokoy-obschestvennosti (дата обращения: 05.05.2024).

33. Матрица RACI как инструмент управления ответственностью, пример матрицы raci по распределению ролей: [сайт]. – URL: https://www.itexpert.ru/rus/ITEMS/200809141918/ (дата обращения: 12.05.2024). – Текст: электронный.

34. Модели разработки ПО: [сайт] – URL: https://qa-guide.ru/forums/topic/modeli_razrabotki/ (дата обращения 02.04.2024). – Текст: электронный.

35. Нетес В. А. Что нужно для успешного применения SLA. – Текст: электронный // T-Comm. – 2015. – №7. – С. 16-20 – URL: https://cyberleninka.ru/article/n/chto-nuzhno-dlya-uspeshnogo-primeneniya-sla (дата обращения: 29.04.2024).

36. О. Б. Иваненко, А. О. Степанова, А. И. Ковалев. Тенденции цифровизации экономики России. – Текст: электронный // Вестник Сибирского института бизнеса и информационных технологий. – 2022. – С. 29-31. – URL: https://cyberleninka.ru/article/n/tendentsii-tsifrovizatsii-ekonomiki-rossii (дата обращения 19.02.2024)

37. Операции LEFT JOIN, RIGHT JOIN (Microsoft Access SQL): [сайт]. – URL: https://learn.microsoft.com/ru-ru/office/client-developer/access/desktop-database-reference/left-join-right-join-operations-microsoft-access-sql (дата обращения: 09.05.2024). – Текст: электронный.

38. Попова В. П. Дизайн-мышление для бизнеса как клиентоориентированная бизнес-модель. – Текст: электронный // Актуальные проблемы экономики и управления. – 2021. – № 4. – С. 66-69. – URL: https://fs.guap.ru/emtp/journals/2021-4.pdf#page=66 (дата обращения 23.04.2024)

39. Постановление Правительства РФ от 31.03.2022 N 522 (ред. от 14.04.2022) «О внесении изменений в Правила предоставления субсидии из федерального бюджета автономной некоммерческой организации «Агентство по технологическому развитию» на поддержку проектов, предусматривающих разработку конструкторской документации на комплектующие изделия, необходимые для отраслей промышленности» – Текст: электронный // КонсультантПлюс: [сайт]. – URL: https://www.consultant.ru/document/cons_doc_LAW_413684/ (дата обращения 01.05.2024).

40. Распоряжение от 28 июля 2017 года №1632-р. «Об утверждении программы «Цифровая экономика Российской Федерации» – Текст: электронный // сайт Правительства РФ: [сайт]. – URL: http://government.ru/docs/28653/ (дата обращения 01.05.2024).

41. Сакина Е. Н., Квасова Е. В. Методика определения ключевых факторов успеха принтмедиаорганизации. – Текст: электронный // Вестник МГУП. – 2016. – №3. – С. 74-79 – URL: https://cyberleninka.ru/article/n/metodika-opredeleniya-klyuchevyh-faktorov-uspeha-printmediaorganizatsii (дата обращения: 07.05.2024).

42. Сапфир научно-производственное объединение: [сайт]. – URL: https://nposapfir.ru/ (дата обращения: 30.04.2024). – Текст: электронный.

43. СБИС ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ «САПФИР – ИНТЕГРАЦИЯ»: [сайт]. – URL: https://sbis.ru/contragents/6671429046/667101001 (дата обращения: 04.05.2024). – Текст: электронный.

44. Свиридов К.М., Свиридова Е.Е. ОСНОВНЫЕ МЕТОДЫ ОЦЕНКИ ЭФФЕКТИВНОСТИ ИННОВАЦИОННО-ИНВЕСТИЦИОННЫХ ПРОЕКТОВ. – Текст: электронный // Экономика и социум. – 2020. – №5-2 (72). – С. 587-599 – URL: https://cyberleninka.ru/article/n/osnovnye-metody-otsenki-effektivnosti-innovatsionno-investitsionnyh-proektov (дата обращения: 14.05.2024).

45. СИСТЕМА СБОРА И АНАЛИЗА КОНСОЛИДИРОВАННОЙ ОТЧЕТНОСТИ С УЧРЕЖДЕНИЙ: [сайт]. – URL: https://nposapfir.ru/resheniya-dlya-gosudarstvennoy-i-munitsipalnoy-vlasti/dlya-obrazovaniya-/sbor-i-analiz-konsolidirovannoy-otchetnosti (дата обращения: 09.05.2024). – Текст: электронный.

46. Скрам: Фреймворк для снижения риска и скорейшей доставки ценности: [сайт] – URL: https://theliberators.com/ (дата обращения 03.04.2024). – Текст: электронный.

47. Современные технологии создания программного обеспечения. Обзор: [сайт] – URL: http://citforum.ru/programming/application/program/3.shtml (дата обращения 03.04.2024). – Текст: электронный.

48. Спиральная модель (Spiral model): [сайт] – URL: https://qalight.ua/ru/baza-znaniy/spiralnaya-model-spiral-model/ (дата обращения 01.04.2024). – Текст: электронный.

49. Сынбулатова А.Т. АНАЛИЗ И СРАВНЕНИЕ МЕТОДОЛОГИЙ ПО МОДЕЛИРОВАНИЮ АРХИТЕКТУРЫ ПРЕДПРИЯТИЯ. – Текст: электронный // Бенефициар. – 2021. – №90. – С. 28-32 – URL: https://elibrary.ru/item.asp?id=44709725 (дата обращения: 29.04.2024).

50. Сычева А. А., Корнева Т.К. Влияние расходов на НИОКР на производительность высокотехнологичных российских компаний на примере НПО «Сапфир». – Текст: электронный // XVII международная конференция «Российские регионы в фокусе перемен»: сборник докладов (Екатеринбург, 17–19 ноября 2022 г.). – Екатеринбург: ООО Издательский Дом «Ажур», 2023. – С. 683-686. – URL: https://elar.urfu.ru/handle/10995/121887 (дата обращения 01.05.2024)

51. Тебекин А.В., Тебекин П.А. Стратегический менеджмент. / Учебник / Сер. 60 Бакалавр. Прикладной курс (2-е изд., пер. и доп.). – Москва: Издательство: «ЮРАЙТ» – 2016. (дата обращения: 05.05.2024).

52. Титов С.А., Титова Н.В. Проектное управление цифровой трансформацией компаний. – Текст: электронный // Вестник университета. – 2022. – № 7. – С. 22-29. – URL: https://cyberleninka.ru/article/n/proektnoe-upravlenie-tsifrovoy-transformatsiey-kompaniy (дата обращения 21.04.2024)

53. Центральный Банк Российской Федерации: официальный сайт. – Москва. – URL: https://cbr.ru/ (дата обращения 20.04.2024). – Текст: электронный.

54. Что такое микросервисная архитектура: [сайт]. – URL: https://selectel.ru/blog/what-is-microservice-architecture/ (дата обращения: 30.04.2024). – Текст: электронный.

55. Широкова В.Е. Анализ проблем и перспектив управления инновационными проектами в условиях цифровизации и глобальной нестабильности – Текст: электронный // Россия и Азия. – 2020. – № 5(14). – с. 72-86. – URL: https://www.elibrary.ru/ip_restricted.asp?rpage=https%3A%2F%2Fwww%2Eelibrary%2Eru%2Fitem%2Easp%3Fid%3D46215011 (дата обращения 19.02.2024)

56. Щетинина Е. А. Дизайн-мышление в бизнес-стратегиях корпорация. – Текст: электронный // Научный журнал НИУ ИТМО. Серия «Экономика и экологический менеджмент». – 2021. – № 1. – С. 85-93. – URL: https://cyberleninka.ru/article/n/dizayn-myshlenie-v-biznes-strategiyah-korporatsiy (дата обращения 23.04.2024)

57. Ю. М. Лисецкий. Модели сопровождения информационных систем предприятия по этапам жизненного цикла. – Текст: электронный // Программные продукты и системы. – 2018. – №3. – С. 455-460. – URL: https://cyberleninka.ru/article/n/modeli-soprovozhdeniya-informatsionnyh-sistem-predpriyatiya-po-etapam-zhiznennogo-tsikla (дата обращения: 02.04.2024).

58. Янгульбаева Л. Ш. Цифровая конкурентоспособность стран. – Текст: электронный // Индустриальная экономика. – 2021. – С. 153-158. – URL: https://cyberleninka.ru/article/n/tsifrovaya-konkurentosposobnost-stran (дата обращения 22.02.2024)

59. Agile-манифест разработки программного обеспечения: [сайт]. – URL: https://agilemanifesto.org/iso/ru/manifesto.html (дата обращения 15.02.2024). – Текст: электронный.

60. Brettel M., Friederichhsen N., Keller M., Rosenberg M. How Virtualization, Decentralization and Network Building Change the Manufacturing Landscape: An Industry 4.0 Perspective. – Текст: электронный // International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering. – 2014. – №8(1) – С. 37–44 – URL: https://www.researchgate.net/publication/285495324_How_virtualization_decentra

lization_and_network_building_change_the_manufacturing_landscape_An_Industry_40_Perspective (дата обращения 30.04.2024).

61. Brewer J.L., Dittman K.C. Methods of It Project Management. – 4 изд. – West Lafayette, Indiana: Purdue University Press, 2022. – 555 с.

62. COALESCE (Transact-SQL): [сайт]. – URL: https://learn.microsoft.com/ru-ru/sql/t-sql/language-elements/coalesce-transact-sql?view=sql-server-ver16 (дата обращения: 09.05.2024). – Текст: электронный.

63. eXtreme Programming: Overview: [сайт] – URL: https://burkov-nikita.medium.com/extreme-programming-9ea4a35176c6 (дата обращения 03.04.2024). – Текст: электронный.

64. Huić I, Horvat N, Škec S. DESIGN SPRINT: USE OF DESIGN METHODS AND TECHNOLOGIES. – Текст: электронный // Proceedings of the Design Society. – 2023. – №3 – С. 1317-1326 – URL: https://elibrary.ru/item.asp?id=63266222 (дата обращения 29.04.2024).

65. Jan-Niklas Meckenstock. Shedding light on the dark side – A systematic literature review of the issues in agile software development methodology use. – Текст: электронный // Journal of Systems and Software. – 2024. – № 211. – URL: https://www.sciencedirect.com/science/article/abs/pii/S0164121224000098 (дата обращения 01.04.2024)

66. Jooh, L., Kwon, H., Pati, N. Exploring the relative impact of R&D and operational efficiency on performance: A sequential regression-neural network approach. – Текст: электронный // Expert Systems with Applications. – 2019. – №137 – С. 420-431 – URL: https://www.sciencedirect.com/science/article/abs/pii/S0957417419305056 (дата обращения 04.05.2024).

67. Lankhorst M., Proper E., Jonkers H., Quartel D. Enterprise Architecture at Work: Modelling, Communication, and Analysis. – 3rd ed. – Berlin: Springer, 2017. – 400 p.

68. Marnewick C., Marnewick A. Digitalization of project management: Opportunities in research and practice. – Текст: электронный // Project Leadership

and Society. – 2022. – № 3. – URL: https://www.sciencedirect.com/science/article/pii/S2666721522000217 (дата обращения 20.04.2024)

69. Omonov Bahodir Fayzullo o‘g‘li. METHODS OF EVALUATING THE ECONOMIC EFFICIENCY OF INVESTMENT PROJECTS. – Текст: электронный // World Economics and Finance Bulletin. – 2023. – №20. – С. 18-21 – URL: https://scholarexpress.net/index.php/wefb/article/view/2300 (дата обращения: 10.05.2024).

70. Pressman R.S. Software Engineering: A Practitioner's Approach. – 7 изд. – New York: McGraw – Hill Education, 2022. – 1104 с.

71. Proglib SDLC модели: [сайт] – URL: https://proglib.io/p/sdlc-modeli-kak-vybrat-pravilnyy-podhod-k-razrabotke-i-ne-zavalit-proekt-2021-05-15# (дата обращения 01.04.2024). – Текст: электронный.

72. Reiff, J., Schlegel, D. Hybrid project management – a systematic literature review. – Текст: электронный // International Journal of Information Systems and Project Management. – 2022. – № 10 (2). – URL: https://revistas.uminho.pt/index.php/ijispm/article/view/4084 (дата обращения 22.04.2024)

73. The Code and Fix Model: [сайт] – URL: https://productcoalition.com/the-code-and-fix-model-2cabd4c48166 (дата обращения 01.04.2024). – Текст: электронный.

74. The TOGAF Standart, 10th Edition: [сайт]. – URL: https://www.opengroup.org/togaf/10thedition (дата обращения: 30.04.2024). – Текст: электронный.

75. Thin Client vs. Thick Client: How Do They Compare?: [сайт]. – URL: https://www.eposaudio.com/en/gb/insights/articles/thin-client-or-thick-client-whats-the-difference (дата обращения: 07.05.2024). – Текст: электронный.

76. Using SQL COUNT() with CASE WHEN: A Comprehensive Guide: [сайт]. – URL: https://www.interviewquery.com/p/sql-count-case-when (дата обращения: 09.05.2024). – Текст: электронный.

77. V-модель процесса разработки программного обеспечения: [сайт] – URL: https://janberg.by/v-model (дата обращения 01.04.2024). – Текст: электронный.

78. Was ist TOGAF: [сайт]. – URL: https://www.computerwoche.de/a/was-ist-togaf,3553455 (дата обращения: 30.04.2024). – Текст: электронный.

ПРИЛОЖЕНИЕ А

```
--аналитика по таблице пбс (смотрим, сколько всего пбс в группе и сколько добавлено в
нуг), Далее смотрим по каждому приложению НУГа
--pb - представляет собой подзапрос, который используется для агрегации данных по таблице
Organizations_list с помощью объединения с таблицей GRSOB
--oa - представляет собой подзапрос, который используется для агрегации данных по таблице
SecurityForm3Organization с помощью объединения с таблицей Organizations_list и GRSOB
--pr - является подзапросом, используемым для агрегации данных, pr анализирует данные из
таблицы SafetyObrSystemData по приложению НУГ 4
--ol - Organizations_list

SELECT pb.MunicipalityCode AS 'Код ГРСОБа',
       pb.Total AS 'Всего учреждений',
       pb.InNUG AS 'из них: добавлено в НУГ',
       COALESCE(oa.Total_Pril_2, 0) AS 'из них: предъявляется к оценке (Приложение 2)',
       COALESCE(oa.Pril_2_ready, 0) AS 'готовы к НУГ (Приложение 2)',
          COALESCE(oa.Pril_2_not_ready, 0) AS 'не готовы к НУГ (Приложение 2)',
          COALESCE(pr.Pril_4, 0) AS 'Приложение 4',
          COALESCE(oa.Pril_5, 0) AS 'Приложение 5',
          COALESCE(oa.Pril_6, 0) AS 'Приложение 6',
          COALESCE(oa.Pril_7, 0) AS 'Приложение 7',
          COALESCE(oa.Pril_8, 0) AS 'Приложение 8',
          COALESCE(oa.Pril_9, 0) AS 'Приложение 9',
          COALESCE(oa.Pril_10, 0) AS 'Приложение 10',
          COALESCE(oa.Pril_11, 0) AS 'Приложение 11',
          COALESCE(oa.Pril_12, 0) AS 'Приложение 12',
          COALESCE(oa.Pril_13, 0) AS 'Приложение 13',
          COALESCE(oa.Pril_17, 0) AS 'Приложение 17'
FROM
       (SELECT gb.code AS MunicipalityCode,
                    COUNT(ol.GRSOBID) AS Total,
                    COUNT(CASE WHEN ol.STATENAME = 'DfaStateAdded' THEN 1 END) AS InNUG
       FROM Organizations_list ol
       INNER JOIN GRSOB gb ON ol.GRSOBID = gb.eiid
       GROUP BY gb.code)  -- в данном запросе смотрим, сколько всего учреждений в целом,
и сколько в нуге, также группируем данные по каждому муниципалитету
AS pb

--джойним 4 приложение НУГа
LEFT JOIN
  (SELECT gb.code AS MunicipalityCode,
          COUNT(DISTINCT sf.eiid) AS Pril_4
   FROM SafetyObrSystemData ss
   INNER JOIN SecurityForm3Organization sf ON ss.SecurityForm3OrganizationID = sf.eiid
   INNER JOIN Organizations_list ol ON sf.Organizations_listID = ol.eiid
   INNER JOIN GRSOB gb ON ol.GRSOBID = gb.eiid
   WHERE ss.Value <> 0
   GROUP BY gb.code) AS pr -- в данном запросе смотрим, сколько учреждений в 4 приложении
НУГа, у которых хотя бы одно значение Value не равно 0, также группируем данные по
каждому муниципалитету
ON pb.MunicipalityCode = pr.MunicipalityCode

--джойним остальные приложения НУГа
LEFT JOIN
  (SELECT gb.code AS MunicipalityCode,
          COUNT(CASE WHEN sf.Nug3_31 = 1 THEN 1 END) AS Total_Pril_2,
          COUNT(CASE WHEN sf.Nug3_31 = 1 AND sf.Nug3_4 = 1 THEN 1 END) AS Pril_2_ready,
          COUNT(CASE WHEN sf.Nug3_31 = 1 AND sf.Nug3_5 = 1 THEN 1 END) AS
Pril_2_not_ready,
               COUNT(CASE WHEN sf.Internet_1_1 = 1 OR --аналитика по 5 приложению
                    sf.Internet_1_2 = 1 OR
```

```
        sf.Internet_1_3 = 1 OR
        sf.Internet_1_4 = 1 OR
        sf.Internet_2_1 = 1 OR
        sf.Internet_2_2 = 1 OR
        sf.Internet_2_3 = 1 OR
        sf.Internet_2_4 = 1 OR
                    sf.Internet_3_1 = 1 OR
        sf.Internet_3_2 = 1 OR
        sf.Internet_3_3 = 1 OR
        sf.Internet_3_4 = 1 OR
                    sf.Internet_4_1 = 1 OR
        sf.Internet_4_2 = 1 OR
        sf.Internet_4_3 = 1 OR
        sf.Internet_4_4 = 1 OR
                    sf.Internet_5_1 = 1 OR
        sf.Internet_5_2 = 1 OR
        sf.Internet_5_3 = 1 OR
        sf.Internet_5_4 = 1 THEN 1 END) AS Pril_5,
COUNT(CASE WHEN sf.nug6_1 <> 0 OR       --аналитика по 6 приложению
                        sf.nug6_2 <> 0 OR
                        sf.nug6_3 <> 0 OR
                        sf.nug6_4 <> 0 OR
                        sf.nug6_5 <> 0 OR
                        sf.nug6_6 <> 0 OR
                        sf.nug6_7 <> 0 OR
                        sf.nug6_8 <> 0 OR
                        sf.nug6_9 <> 0 OR
                        sf.nug6_10 <> 0 OR
                        sf.nug6_11 <> 0 OR
                        sf.nug6_12 <> 0 OR
                        sf.nug6_13 <> 0 OR
                        sf.nug6_14 <> 0 OR
                        sf.nug6_15 <> 0 OR
                        sf.nug6_16 <> 0 THEN 1 END) AS Pril_6,
 COUNT(CASE WHEN sf.AlarmSystem <> 0 OR --аналитика по 7 приложению
        sf.CCTV <> 0 OR
        sf.AccessControlSystem <> 0 OR
        sf.PerimeterFencing <> 0 OR
        sf.OutdoorLighting <> 0 OR
        sf.MetalDetector <> 0 OR
                    sf.EvacuationControlSys <> 0 OR
                    sf.Rosgvardia <> 0 OR
                    sf.EnterpriseOkhrana <> 0 OR
                    sf.PrivateSecurityCompany <> 0 OR
                    sf.StaffMembers <> 0 OR
                    sf.NotGuarded <> 0 OR
                    sf.NationalGuardTroops <> 0 OR
                    sf.EmergencyServices <> 0 OR
                    sf.EmergencyServicesDiff <> 0 OR
                    sf.NoEmergencyServices <> 0 OR
                    sf.CategorizedObjects <> 0 THEN 1 END) AS Pril_7,
       COUNT(CASE WHEN sf.nug54_Column1 <> 0 OR --аналитика по 8 приложению
       sf.nug54_Column3 <> 0 OR
        sf.nug54_Column2 <> 0 OR
        sf.nug54_Column4 <> 0 OR
        sf.nug54_Column18 <> 0 OR
        sf.nug54_Column5_2023 <> 0 OR
                    sf.nug54_Column7_2023 <> 0 OR
                    sf.nug54_Column13_2023 <> 0 OR
                    sf.nug54_Column9_2023 <> 0 OR
                    sf.nug54_Column11_2023 <> 0 THEN 1 END) AS Pril_8,
        COUNT(CASE WHEN sf.nug9_1 <> 0 OR       --аналитика по 9 приложению
        sf.nug9_2 <> 0 OR
        sf.nug9_3 <> 0 OR
```

```
                        sf.nug9_4 <> 0 OR
                        sf.nug9_5 <> 0 OR
                        sf.nug9_6 <> 0 OR
                                    sf.nug9_Money <> 0 THEN 1 END) AS Pril_9,
                        COUNT(CASE  WHEN  sf.NUG_10_1  <>  0  OR  --аналитика  по  10  приложению
                        sf.NUG_10_2 <> 0 OR
                        sf.NUG_10_3 <> 0 OR
                        sf.NUG_10_4 <> 0 OR
                        sf.NUG_10_5 <> 0 OR
                        sf.NUG_10_6 <> 0 OR
                                    sf.NUG_10_7 <> 0 OR
                                    sf.NUG_10_8 <> 0 OR
                                    sf.NUG_10_9 <> 0 OR
                                    sf.NUG_10_11 <> 0 OR
                                    sf.NUG_10_12 <> 0 OR
                                    sf.NUG_10_13 <> 0 OR
                                    sf.NUG_10_14 <> 0 OR
                                    sf.NUG_10_15 <> 0 OR
                                    sf.NUG_10_16 <> 0 OR
                                    sf.NUG_10_17 <> 0 OR
                                    sf.NUG_10_19 <> 0 OR
                                    sf.NUG_10_20 <> 0 OR
                                    sf.NUG_10_21 <> 0   THEN 1 END) AS Pril_10,
                       COUNT(CASE  WHEN  sf.NUG_11_1  <>  0  OR  --аналитика  по  11  приложению
                        sf.NUG_11_2 <> 0 OR
                        sf.NUG_11_3 <> 0 OR
                        sf.NUG_11_4 <> 0 OR
                        sf.NUG_11_5 <> 0 OR
                        sf.NUG_11_6 <> 0 OR
                                    sf.NUG_11_7 <> 0 OR
                                    sf.NUG_11_8 <> 0 OR
                                    sf.NUG_11_9 <> 0 OR
                                    sf.NUG_11_11 <> 0 THEN 1 END) AS Pril_11,
                        COUNT(CASE WHEN sf.NUG_12_1 <> 0 OR    --аналитика по 12 приложению
                        sf.NUG_12_3 <> 0 OR
                        sf.NUG_12_4 <> 0 OR
                        sf.NUG_12_5 <> 0 OR
                        sf.NUG_12_6 <> 0 THEN 1 END) AS Pril_12,
                        COUNT(CASE WHEN sf.NUG_13_1 <> 0 OR    --аналитика по 13 приложению
                        sf.NUG_13_3 <> 0 OR
                        sf.NUG_13_4 <> 0 OR
                        sf.NUG_13_5 <> 0 OR
                        sf.NUG_13_6 <> 0 OR
                                    sf.NUG_13_7 <> 0 OR
                                    sf.NUG_13_8 <> 0 OR
                                    sf.NUG_13_9 <> 0 THEN 1 END) AS Pril_13,
                        COUNT(CASE WHEN sf.NUG_17_1 <> 0 OR   --аналитика по 17 приложению
                        sf.NUG_17_3 <> 0 OR
                        sf.NUG_17_4 <> 0 OR
                        sf.NUG_17_5 <> 0 OR
                        sf.NUG_17_6 <> 0 OR
                                    sf.NUG_17_7 <> 0 OR
                                    sf.NUG_17_8 <> 0 OR
                                    sf.NUG_17_9 <> 0 OR
                                    sf.NUG_17_10 <> 0 OR
                                    sf.NUG_17_11 <> 0 OR
                                    sf.NUG_17_12 <> 0 THEN 1 END) AS Pril_17
   FROM SecurityForm3Organization sf
   INNER JOIN Organizations_list ol ON sf.Organizations_listID = ol.eiid
   INNER JOIN GRSOB gb ON ol.GRSOBID = gb.eiid
   GROUP BY gb.code) AS oa
ON pb.MunicipalityCode = oa.MunicipalityCode
ORDER BY
```

```
    CASE WHEN LEN(pb.MunicipalityCode) = 2 THEN 1 ELSE 2 END, -- сначала двузначные,
потом трехзначные
    pb.MunicipalityCode; -- сортировка по коду

--комментарии к функциям в скрипте
--1. COUNT(CASE WHEN ... THEN 1 END) - условный подсчёт, позволяющий считать записи
только при определённых условиях
--2. COALESCE() - функция, используемая для замены NULL значений на 0 (или другое
указанное значение). Итоговый набор данных не будет содержать NULL значения
--3. LEFT JOIN - оператор для соединения таблиц таким образом, что результат включает все
строки левой таблицы (таблицы, указанной перед LEFT JOIN) и соответствующие строки из
правой таблицы. Если соответствия нет, результат будет содержать NULL по столбцам правой
таблицы. Это используется для соединения основной таблицы с подзапросами, агрегирующими
данные по различным приложениям.
--4. DISTINCT - используется для удаления дубликатов из результирующего набора данных
```

ПРИЛОЖЕНИЕ Б

```
//version = 2
using Sector.Common;
using Sector.DSL;
using Sector.DSL.Adapter;
using Sector.DSL.Result;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Sector.Common.Services.Security;
using System.Security.Principal;
using System.Data;
using OfficeOpenXml;
using System.Data.SqlClient;
using System.IO;

namespace Sector.DSL
{
    class Analytics : BaseSectorDSL
    {
        public Analytics(ISectorView currentView, IEntityItem currentViewItem,
IObjectProvider objectProvider, MacroContextAdapter context)
           : base(currentView, currentViewItem, objectProvider, context)
        {
        }

        //метод для вывода аналитической таблицы по всем формам НУГа из SQL в Excel
        public void ExportAnalyticsToExcel()
        {
            SectorEntityAdapter TechicalTable = Entities["TechicalTable"];
            string connectionString = "Data Source=gz_test\\mssql2012;Initial
Catalog=RC_NUG23;User ID=amber;Pwd=password;"; // Подключение к базе данных
            string templateFilePath =
@"\\192.168.150.121\ProjectService\MinFin23\RC_NUG\datasector\Upload\Шаблон.xlsx"; //
Путь к файлу-шаблону Excel
            string outputFolderPath =
@"\\192.168.150.121\ProjectService\MinFin23\RC_NUG\datasector\Upload"; // Путь для
сохранения нового файла Excel
            string outputFileName = "Аналитика НУГа " + DateTime.Now.ToString("yyyy-MM-dd
(HH-mm)") + ".xlsx";

            // Чтение всего содержимого SQL файла
            string fileSQLPath =
@"\\192.168.150.121\ProjectService\MinFin23\RC_NUG\Scripts\Аналитика.sql"; //
Расположение скрипта для аналитики
            string query = File.ReadAllText(fileSQLPath);

            using (SqlConnection connection = new SqlConnection(connectionString))
            {
                connection.Open();

                using (SqlCommand command = new SqlCommand(query, connection))
                {
                    command.ExecuteNonQuery();
                }
            }

            // Получаем путь к папке с файлом на сервере
```

```
            string outputFilePath = Path.Combine(outputFolderPath, outputFileName);

            // Загружаем файл-шаблон Excel
            FileInfo templateFileInfo = new FileInfo(templateFilePath);

            // Убеждаемся, что файл-шаблон существует
            if (!templateFileInfo.Exists)
            {
                throw new FileNotFoundException("Шаблон для выгрузки аналитики не
найден.");
            }

            // Копируем шаблон в новый файл
            templateFileInfo.CopyTo(outputFilePath, true);

            // Открываем новый файл для редактирования
            FileInfo outputFileInfo = new FileInfo(outputFilePath);
            using (ExcelPackage excelPackage = new ExcelPackage(outputFileInfo))
            {
                ExcelWorksheet worksheet = excelPackage.Workbook.Worksheets["Лист1"];

                // Убеждаемся, что лист существует
                if (worksheet == null)
                {
                    throw new Exception("Лист не найден в файле Excel.");
                }

                // Загрузка данных из SQL в DataTable
                using (SqlConnection connection = new SqlConnection(connectionString))
                {
                    connection.Open();

                    using (SqlCommand command = new SqlCommand(query, connection))
                    {
                        using (SqlDataAdapter adapter = new SqlDataAdapter(command))
                        {
                            DataTable dataTable = new DataTable();
                            adapter.Fill(dataTable);

                            // Записываем MunicipalityCode в первую колонку, начиная с
строки A6
                            for (int row = 0; row < dataTable.Rows.Count; row++)
                            {
                                worksheet.Cells[row + 6, 1].Value =
dataTable.Rows[row][0];
                            }

                            // Записываем остальные данные начиная со второй колонки,
начиная с строки C6
                            for (int row = 0; row < dataTable.Rows.Count; row++)
                            {
                                for (int col = 0; col < dataTable.Columns.Count - 1;
col++)
                                {
                                    worksheet.Cells[row + 6, col + 3].Value =
dataTable.Rows[row][col + 1];
                                }
                            }
                        }
                    }
                }


                // Сохраняем файл Excel
```

```
excelPackage.Save();
```

```
                //Сохранение истории выгрузки аналитики (создаем новую строку)
                SectorEntityAdapter HistoryUploadingAnalytics =
Entities["HistoryUploadingAnalytics"];
                SectorEntityItemAdapter newRowHistory =
HistoryUploadingAnalytics.NewRow();

                //Генерируем уникальный идентификатор (GUID) для созданного отчета
                Guid attachmentGuid = Guid.NewGuid();

                newRowHistory["DateCreate"] = System.DateTime.Now; //дата выгрузки отчета
                newRowHistory["Author"] = CurrentUser.Identity.Name; //автор выгрузки
                newRowHistory["Attachment"] = attachmentGuid; //вложен сам отчет

                Result.SetSuccessMessage("Отчет успешно сформирован и вложен в историю
выгрузки аналитики", 9000);
            }
        }
    }
}
```