Министерство науки и высшего образования Российской Федерации ФГБОУ ВО РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ
(РГГМУ)

Институт Информационных систем и геотехнологий
КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ

БАКАЛАВРСКАЯ РАБОТА

На тему «Разработка интерактивной ГИС ВУЗа»

Исполнитель Шабашов Денис Игоревич

Руководитель кандидат технических наук, доцент – Петров Ярослав Андреевич

«К защите допускаю»

Заведующий кафедрой _____

(подпись)

_____

(ученая степень, ученое звание)

_____

(фамилия, имя, отчество)

«___»_____2023 г.

Санкт–Петербург

2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Концепция интерактивных геоинформационных систем (ГИС) широко применяется в различных областях, включая управление расписанием вузов. Такие системы позволяют:

- Эффективно управлять расписанием занятий
- Контролировать доступность аудиторий
- Контролировать количество ресурсов
- Облегчить процесс регистрации на курсы
- Автоматизировать сдачу экзаменов

Выпускная квалификационная работа рассматривает разработку интерактивной ГИС для управления расписанием вуза, которая позволит автоматизировать процесс планирования занятий и упростить работу преподавателей и студентов.

Объектом исследования является автоматизированная система управления учебной и не учебной деятельностью ВУЗа.

Предметом исследования является система управления расписанием с ГИС возможностями.

Цель работы: проектирование и разработка автоматизированной системы расписания ВУЗа с ГИС возможностями. Система позволит удобно и быстро редактировать и добавлять новые записи в расписании, работать с пользователями, управлять аудиториями и хранить географические данные о расположении корпусов, сохранять и получать данные о погоде. А также иметь множество интеграций с существующими программными решениями.

После постановки цели работы необходимо определить список решаемых задач, методы достижения целей и инструменты, используемые в работе.

Для достижения цели поставлены следующие задачи:

1. Провести предпроектный анализ
2. Определить целевую аудиторию проекта
3. Определить требования
4. Провести экономическое исследование проекта и обосновать затраты
5. Спроектировать информационную систему
6. Провести анализ бизнес-процессов
7. Построить функциональные модели и модели данных
8. На основании моделей и функциональных требований разработать информационную систему
9. На основании существующего серверного проекта реализовать графический интерфейс пользователя
10. Произвести пусконаладочные работы

Методика выполнения задач:

1. Проведение SWOT и VCM анализа
2. Сбор требований и их описание
3. Составление сметы на затраты на ВКР
4. Составление IDEF0, UML, DFD, IDEF1X моделей
5. Разработка информационной системы на проектном языке программирования
6. Выбор облачного провайдера. Запуск информационной системы на виртуальном сервере.

Используемые инструменты:

Проектирование: Visual Paradigm, ERwin Data Modeler, draw.io.

Реализация: Java, Maven, Spring Boot, Spring JPA, Spring Web, PostgreSQL, Thymeleaf, Git, GitHub, IntelliJIDEA.

ГЛАВА 1. АНАЛИТИЧЕСКАЯ ЧАСТЬ

1.1. Определение предметной области

Предметная область — автоматизированная система управления расписанием в учебных заведениях.

Данные системы включают автоматизацию следующих задач:

- Планирование расписания образовательного процесса групп учащихся
- Планирование расписания работы преподавательского состава
- Организация взаимодействия групп учащихся и преподавательского состава
- Формирование отчетности о занятости и результатах работы преподавателя, успеваемости студентов и т.д.

В настоящее время существуют готовые коммерческие проекты, которые сосредоточены на автоматизации расписания организаций любых профилей.

Многие образовательные учреждения вынуждены самостоятельно разрабатывать или заказывать программное обеспечение для управления расписанием.

Например:

- CRM «Система планирования расписанием» НИУ ВШЭ (РУЗ)
- Magellan
- Апекс-ВУЗ
- 1С: Автоматизированное составление расписания. Университет

Все вышеперечисленные системы являются CRM.

1.2. Сравнение существующих систем

Для выявления сильных и слабых сторон проекта проведем функциональный сравнительный анализ:

Таблица 1.1 — Функциональный сравнительный анализ

Функциональность	Интерактивная ГИС ВУЗа	НИУ ВШЭ (РУЗ)	Magellan	1С «Университет»
ГИС возможности	+	-	-	-
Расписание занятий	+	+	+	+
Система уведомлений	+	+	+	-
Управление аудиториями	+	+	+	+
Электронный журнал	+	+	-	+
Личный кабинет	+	+	-	-

Таким образом «Интерактивная ГИС ВУЗа» покрывает все основные функциональные требования, предъявляемые к системе ВУЗа, а так же привносит ГИС возможности, которые являются в данном случае новшеством и автоматизацией потребностей пользователей.

Кроме того система реализует полный список требований выявленных в ходе предпроектного анализа.

Студенты и преподаватели получают расписание из различных источников:

- Мобильное приложение (IOS или Android)
- Веб-приложение
- PDF, DOCX, XLSX, CSV файлы
- Desktop-приложение
- Растровые файлы (Изображения или фотографии)

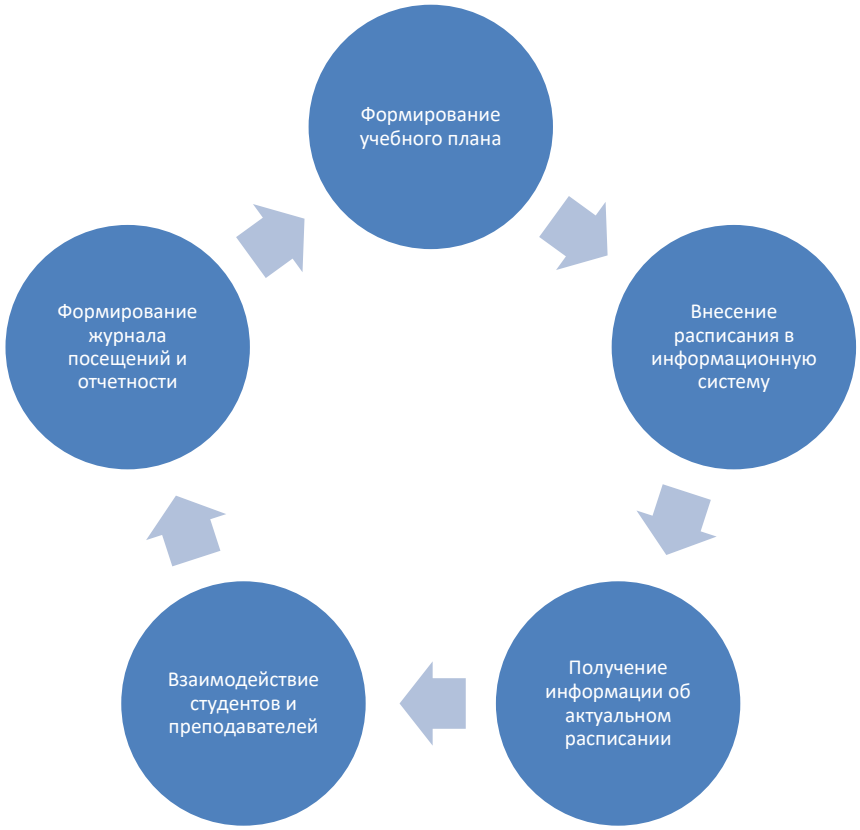

Рисунок 1.1, Диаграмма Автоматизации учебных процессов

На диаграмме выше изображен цикл автоматизации учебных процессов, улучшающий взаимодействие всех уровней иерархии предприятия, научной организации или учебного заведения.

1.3. SWOT и VCM анализ

Таблица 1.2— SWOT-анализ

Положительное влияние	Отрицательное влияние
S – сильные стороны 1. Масштабируемость 2. Безопасность личных данных 3. Интеграции 4. Модульность 5. Высокая скорость доставки функциональности	W – слабые стороны 1. Ограниченное количество HR 2. Ограниченные финансы 3. Продукт новый
O – возможности 1. Быстрый рост пользователей 2. Выход на рынок 3. Работающий сервис на этапе разработки 4. Сертификация по ГОСТ Р ИСО/МЭК 25001	T – угрозы 1. Повышение налогов 2. Увеличение стоимости аренды облачных провайдеров

В ходе SWOT-анализа были выявленные широкие возможности и сильные стороны такие как: Масштабируемость, Сертификация по ГОСТ Р ИСО/МЭК 25001, Высокая скорость доставки функциональности, Безопасность личных данных.

Слабые стороны и Угрозы — несущественны так как слабые стороны можно решить внутри проекта введением в работу большего числа сотрудников и быстрыми доходами за счет высокой доставки функциональности и рабочего сервиса на стадии разработки. Внешние угрозы зависят только от общей экономической и политической ситуации в России, а проект никак не связан с остальными потенциальными угрозами.

На основе анализа всей цепочки видов деятельности в организации (от получения сырья до конечной продукции), которая продается и поставляется потребителю, модель показывает: какое слабое звено приводит к разрыву всего звена. Альтернативный метод - VCM – позволяет выявить наиболее выгодно выступающие элементы цепи ценности.

Рисунок 1.2 VCM основные ценности

VCM анализ показывает, что основную часть создания стоимости приносят следующие ценности:

- Работа в соответствии с манифестом Agile
- Связанный цикл применения лучших практик в разработке и обеспечения качества программного обеспечения
- Высокий сервис поддержки пользователей

1.4. Целевая аудитория проекта

Целевой аудиторией проекта являются как студенты и преподаватели любых учебных заведений, так и административный персонал. С одной стороны непосредственно пользователи, а с другой потенциальный заказчик, целью которого является автоматизация существующих процессов, сокращение издержек на составлении расписания и улучшение образовательных процессов.

Кроме того, проект планируется модульным и сможет распространяться как SDK (software development kit), в качестве базовой реализации требуемого функционала, для последующего масштабирования и добавления новой функциональности силами научно-технических, информационно-аналитических центров образовательных учреждений или отдельных разработчиков программного обеспечения. В данном случае цель проекта – это сокращение сроков разработки и, соответственно, затрат на производство информационных систем данного типа.

Автоматизация процессов и сокращение издержек на составление расписания может значительно улучшить образовательные процессы.

Таким образом, идея модульности и распространения проекта в виде SDK позволит масштабировать функциональность и ускорить процесс разработки.

Проект может быть очень полезен для различных учебных заведений, как для студентов и преподавателей, так и для административного персонала. Это может помочь им значительно сэкономить время и ресурсы, которые могут быть использованы для более важных задач.

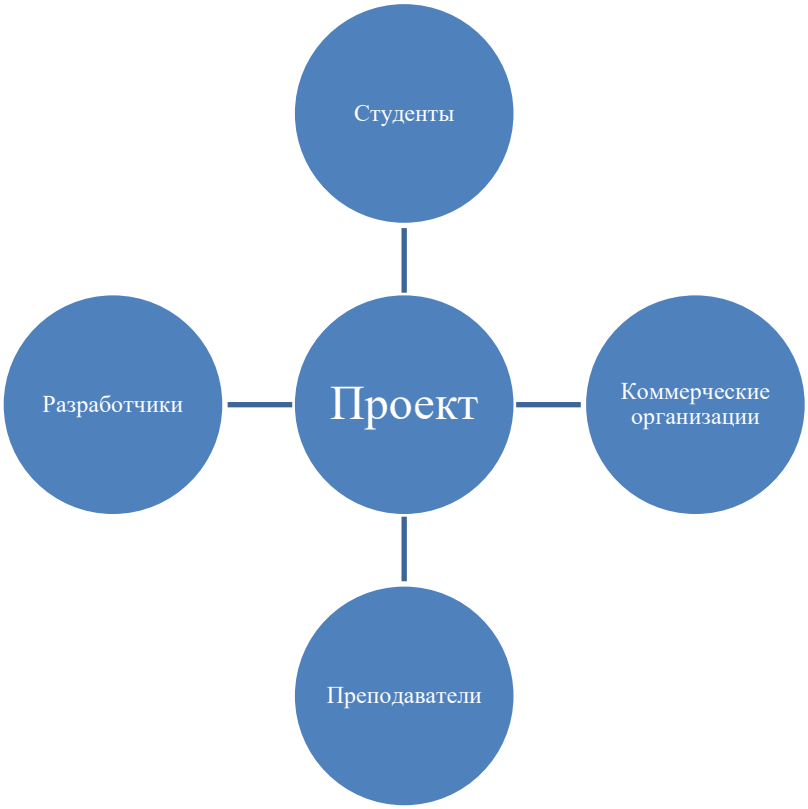

Рисунок 1.3 Заинтересованные лица проекта

Кроме того, такой проект может быть очень интересен для различных научно-технических и информационно-аналитических центров образовательных учреждений, а также для отдельных разработчиков программного обеспечения. Они могут использовать SDK для разработки собственных приложений, которые будут соответствовать их конкретным потребностям.

В целом, есть уверенность, что такой проект может иметь большой потенциал и принести много пользы образовательной сфере. Таким образом, с помощью правильного маркетинга и продвижения, он может стать очень популярным и востребованным продуктом на рынке.

1.5. Требования к системе

Система состоит из следующих приложений:

1. Приложение клиент (Студент)
2. Приложение клиент (Преподаватель)
3. Панель администратора

Функциональные и системные требования к перечисленным выше приложениям вынесены в Приложение А, Таблица 1.0.

Помимо указанных выше системных требований реализуемая информационная система должная соответствовать следующим нефункциональным требования.

Таблица 1.3 — Нефункциональные требования

Нефункциональные требования		
№	**Наименование**	**Описание**
1	Отказоустойчивость	Система работает в режиме 24/7, все подсистемы не выходят из строя
2	Производительность	Система обрабатывает 1000 запросов в секунду
3	Масштабируемость	Информационная система поддерживает вертикальное масштабирование
4	Безопасность	Защита от несанкционированного доступа, SSL сертификация
5	Доступность	Система всегда доступна при необходимости
6	Использование	Система должна быть удобной и простой для пользователя

1.6. Технико-экономическое обоснование

Для оценки ожидаемых затрат реализации проекта на стадии предпроектного анализа требуется рассчитать длительность каждого этапа в человеко-часах. Метод оценивания – экспертные оценки:

Таблица 1.4 — Оценки ожидаемых временных затрат на реализацию проекта

№	Наименование работ	Длительность работ (t, чел. ч.)		
		t_{min}	t_{max}	t_o
1	Предпроектный анализ	24	56	36,8
2	Бизнес-моделирование	50	70	58
3	Проектирование	100	150	120
4	Реализация	250	500	350
5	Тестирование	100	120	108
6	Пусконаладочные работы	8	8	8
Итого:				680.8

Табл.1 Расчет ожидаемой длительности выполнения каждого из этапов реализации проекта

Формула используемая для расчётов:

$$t_j^o = \frac{(3t_{min} + 2t_{max})}{5} \quad (1.1)$$

t_j^o — ожидаемая продолжительность j-того этапа;

t_{min} — наименьшая по длительности оценка эксперта;

t_{max} — наибольшая по длительности оценка эксперта.

Оценочная модель является аппроксимированной, каждый из этапов можно декомпозировать и рассчитать более детально.

Итого проект требует 680.8 человеко-часов.

Оценочную стоимость работ по оценочной продолжительности работ, следует рассчитать относительно заработной платы исполнителя. Для студента примем размер заработной платы равный заработной плате исполнителя.

Формула вычисления часовой ставки:

$$I_j = \frac{P}{K} = \frac{P}{168} \quad (1.2)$$

I_j — часовая ставка

P — размер заработной платы

K — количество часов (в данном случае 168 часов в месяц в среднем)

Примем в качестве заработной платы исполнителя после вычета норматива социальных, страховых взносов и НДФЛ сумму в 70 000 рублей в месяц.

Таблица 1.5 — Расчет ожидаемых затрат на выплаты исполнителю

№	**Наименование работ**	**t_o, чел. ч.**	**I_j, руб. в час**
1	Предпроектный анализ	36,8	15 333,33
2	Бизнес-моделирование	58	24 166,66
3	Проектирование	120	50 000,00
4	Реализация	350	145 833,33
5	Тестирование	108	45 000,00
6	Пусконаладочные работы	8	3 333,33
Итого:		680.8	283 666,66

Необходимо учесть в общей оценочной стоимости проекта норматив отчислений на страховые взносы на обязательное социальное, пенсионное и медицинское страхование на 01.01.2023 — 30%. Ставка НДФЛ на 01.01.2023 — 13%. МРОТ на 01.01.2023 — 16 242 рублей в месяц.

Расчетные значения норматива отчислений НДФЛ за месяц вычисляются по формуле:

$$Н_{ндфл} = \frac{100*Q}{87} - Q = Q * \left(\frac{100}{87} - 1\right) \quad (1.3)$$

Расчетные значения норматива отчислений на страхование вычисляются по формуле:

$$Н_{ст} = М \times Н_{соц} + (Q - М) \times Н_{л} \quad (1.4)$$

Н — итоговая сумма отчислений за норматив

$Н_{соц}$ — базовый норматив (в процентах) (30%)

$Н_{л}$ — льготный норматив (в процентах) (15%)

М — МРОТ

Q — заработная плата

Таблица 1.6 — Расчет затрат на социальные и налоговые выплаты

№	Наименование норматива	Сумма по нормативу, руб.
1	НДФЛ	42 386,97
2	Страховые взносы (медицинское, социальное, пенсионное)	12 936,30
Итого:		55 323,27

Таблица 1.7 — Расчет затрат на оборудование

№	Наименование затраты на оборудование	Тариф, руб.	Сумма, руб.
1	Выделенный сервер	3059 в месяц	9177
2	SSL сертификат	1 599 в год	399,75
Итого:			9576,75

Сведение всех затрат в смете затрат на ВКР:

Таблица 1.8 — Смета затрат на ВКР

№	Наименование статьи	Сумма, руб.
1	Заработная плата сотрудников	283 666,66
2	Отчисления на социальные нужды	55 323,27
3	Расходы на содержание и эксплуатацию оборудования	-
4	Материалы	-
5	Затраты по работам, выполняемым сторонними организациями	-
6	Амортизационные отчисления	-
7	Накладные расходы	-
8	Спецоборудование	9576,75
Итого:		348 566,68

Общая стоимость затрат на ВКР составляет 348 566,68 рублей. Любой реализуемый проект имеет дорожную карту и календарный план. Исходя из этого, следует составить диаграмму Ганта как визуализацию планируемых этапов.

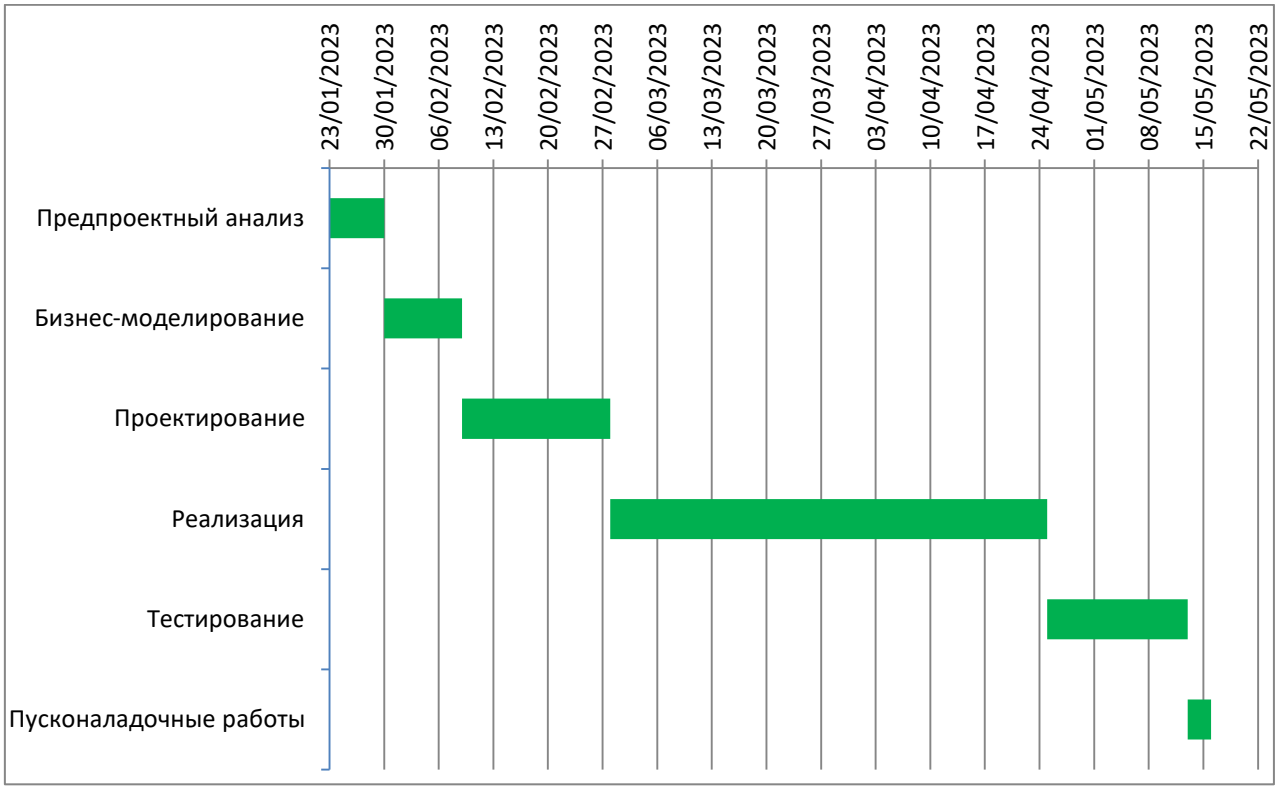

Рисунок 1.4 Диаграмма Ганта этапов выполнения проекта

На диаграмме изображены все этапы необходимые для реализации проекта:

- Предпроектный анализ
- Бизнес-моделирование
- Проектирование
- Реализация
- Тестирование
- Пусконаладочные работы

Длительность этапов выражена в календарных днях с учетом выходных дней, на основе данных полученных путем экспертных оценок.

Как видно из диаграммы, для успешной реализации проекта необходимо пройти через несколько этапов. В начале проекта проводится предпроектный анализ, который позволяет определить цели и задачи проекта, а также риски и возможности, связанные с его реализацией. Затем проводится бизнес-моделирование, на основе которого разрабатывается бизнес-план и определяется финансовая модель проекта.

На следующем этапе — проектировании — создается детальный план реализации проекта, включая разработку технической документации, выбор необходимых ресурсов и определение сроков выполнения работ.

После этого начинается реализация проекта, на этом этапе происходит непосредственное создание продукта или услуги, а также проведение всех необходимых работ и мероприятий.

После завершения реализации проекта начинается тестирование, на этом этапе происходит проверка работоспособности и соответствия продукта или услуги заявленным требованиям. В случае обнаружения ошибок или недостатков проводятся исправления и доработки.

Последний этап — пусконаладочные работы — позволяет убедиться в корректной работе продукта или услуги в реальных условиях эксплуатации. На этом этапе проводятся все необходимые настройки и оптимизации, а также обучение персонала. Длительность каждого этапа определяется на основе экспертных оценок и выражается в календарных днях с учетом выходных дней. Важно понимать, что каждый этап является необходимым и не может быть пропущен или сокращен без ущерба для качества и результативности проекта.

Таким образом, следуя всем этапам, можно добиться успешной реализации проекта и достижения поставленных целей. Важно также помнить о необходимости контроля и управления проектом на каждом этапе, чтобы

своевременно реагировать на возможные риски и изменения в условиях реализации.

Исходя из результатов предпроектного анализа, есть смысл реализовывать практическую часть проекта.

ГЛАВА 2. ПРОЕКТИРОВАНИЕ ИС

2.1. Выбор методологий проектирования

Цель проектирования информационной системы заключается в создании эффективной и функциональной системы, которая может обрабатывать, хранить и передавать информацию в соответствии с требованиями пользователей и бизнес-процессов.

Проектирование информационной системы включает в себя:

- анализ бизнес-потребностей
- определение требований к системе
- разработку архитектуры
- разработку дизайна системы
- выбор инструментов тестирования

Когда информационная система разработана и реализована правильно, она может значительно повысить эффективность и производительность бизнес-процессов, улучшить качество работы и упростить доступ к информации для пользователей.

Перейдем к выбору методологий проектирования:

Необходимо выбрать подходящие методологии для декомпозиции и анализа бизнес-процессов, описания функциональных требований, описания информационных потоков и данных, описания взаимодействия между объектами, описания архитектуры проекта.

Существует множество методологий моделирования бизнес-процессов, некоторые из них более популярны и широко используются, чем другие. Выбор конкретной методологии зависит от индивидуальных потребностей и характеристик проекта. Вот некоторые из них: «BPMN», «DFD», «IDEF», «UML», «Lean Six Sigma», «Event-driven Process Chain (EPC)».

Рассмотрим IDEF, BPMN, UML, DFD:

IDEF — это методология моделирования процессов и систем, которая позволяет описать их работу на разных уровнях детализации. Для этого используются различные типы IDEF-диаграмм, например, IDEF0 для анализа бизнес-процессов на высоком уровне абстракции, IDEF3 и IDEF4 для моделирования информационных потоков и взаимодействия компонентов системы, а IDEF5 для описания функциональных требований к системе.

UML (Unified Modeling Language) — это язык графического описания, используемый для моделирования объектов и процессов в разработке программного обеспечения, а также для моделирования бизнес-процессов и организационных структур. UML является стандартом, который использует графические обозначения для создания абстрактной модели системы, называемой UML-моделью.

BPMN (Business Process Model and Notation) — это стандартная нотация для моделирования бизнес-процессов, которая позволяет создавать графические диаграммы, отображающие последовательность действий, ролей и потоков данных в рамках процесса.

DFD (диаграммы потоков данных) — это методология, которая используется для описания системы в графическом виде. Она помогает определить источники и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ.

Рассмотрим преимущества:

Построение IDEF-диаграмм позволяет увидеть проект в целом, выявить проблемы и узкие места, оптимизировать процессы и улучшить взаимодействие между компонентами системы. Кроме того, IDEF-диаграммы могут быть использованы для обучения новых сотрудников и документирования проекта и

его изменений в будущем, определить, какие процессы нужно улучшить, чтобы повысить эффективность и оптимизировать бизнес-процессы.

UML не является языком программирования, но на основании UML-моделей можно генерировать код. Он обеспечивает стандартизированный подход к моделированию, что позволяет разработчикам и бизнес-аналитикам легче понимать и коммуницировать между собой. UML-модели могут использоваться для создания диаграмм, которые помогают визуализировать различные аспекты системы, такие как ее структура, поведение и взаимодействие между объектами. Это может помочь разработчикам лучше понимать систему и выявлять потенциальные проблемы или улучшения.

DFD может использоваться в различных областях, таких как бизнес, наука, инженерия и т.д. DFD может иметь несколько уровней детализации, каждый из которых описывает систему на разных уровнях. На верхнем уровне DFD обычно описывается весь процесс, а на нижних уровнях - более детально.

В итоге для решения задач проектирования выбраны следующие методологии:

Таблица 2.1 — Выбранные методологии проектирования

Задача	**Выбранная методология**
Анализ бизнес-процессов	IDEF0
Описание функциональных требований к системе	UML: UseCase диаграммы
Описание информационных потоков и данных	DFD: контекст, декомпозиция
Описание взаимодействия между объектами	UML: диаграммы последовательности
Модель данных	IDEF1X

2.2. Анализ бизнес-процессов

Цель — моделирование функциональной диаграммы в нотации IDEF0.

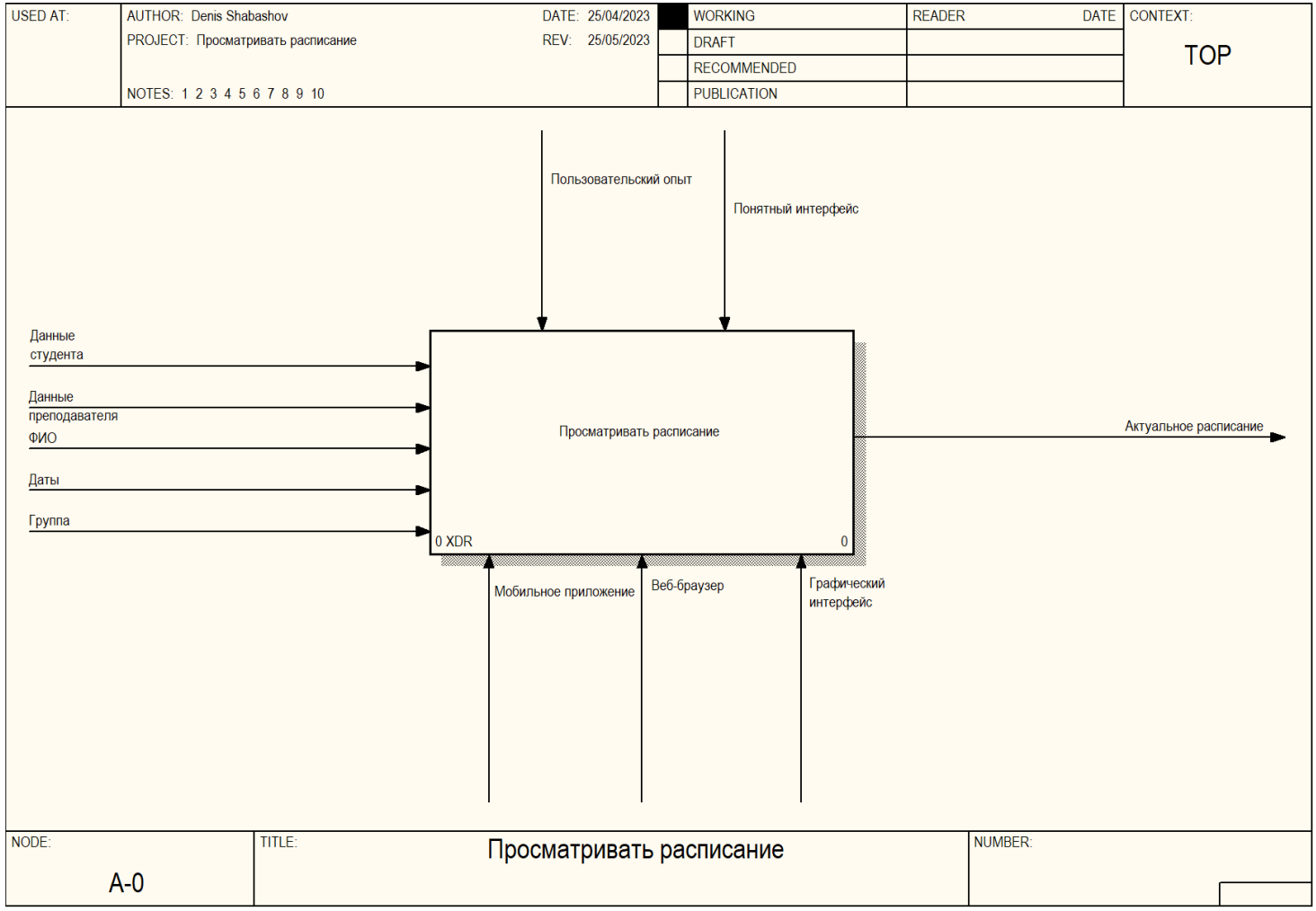

Рисунок 2.1 Диаграмма верхнего уровня, Просмотр расписания, нотация IDEF0

Далее декомпозиция бизнес-процесса «Просмотр расписания»:

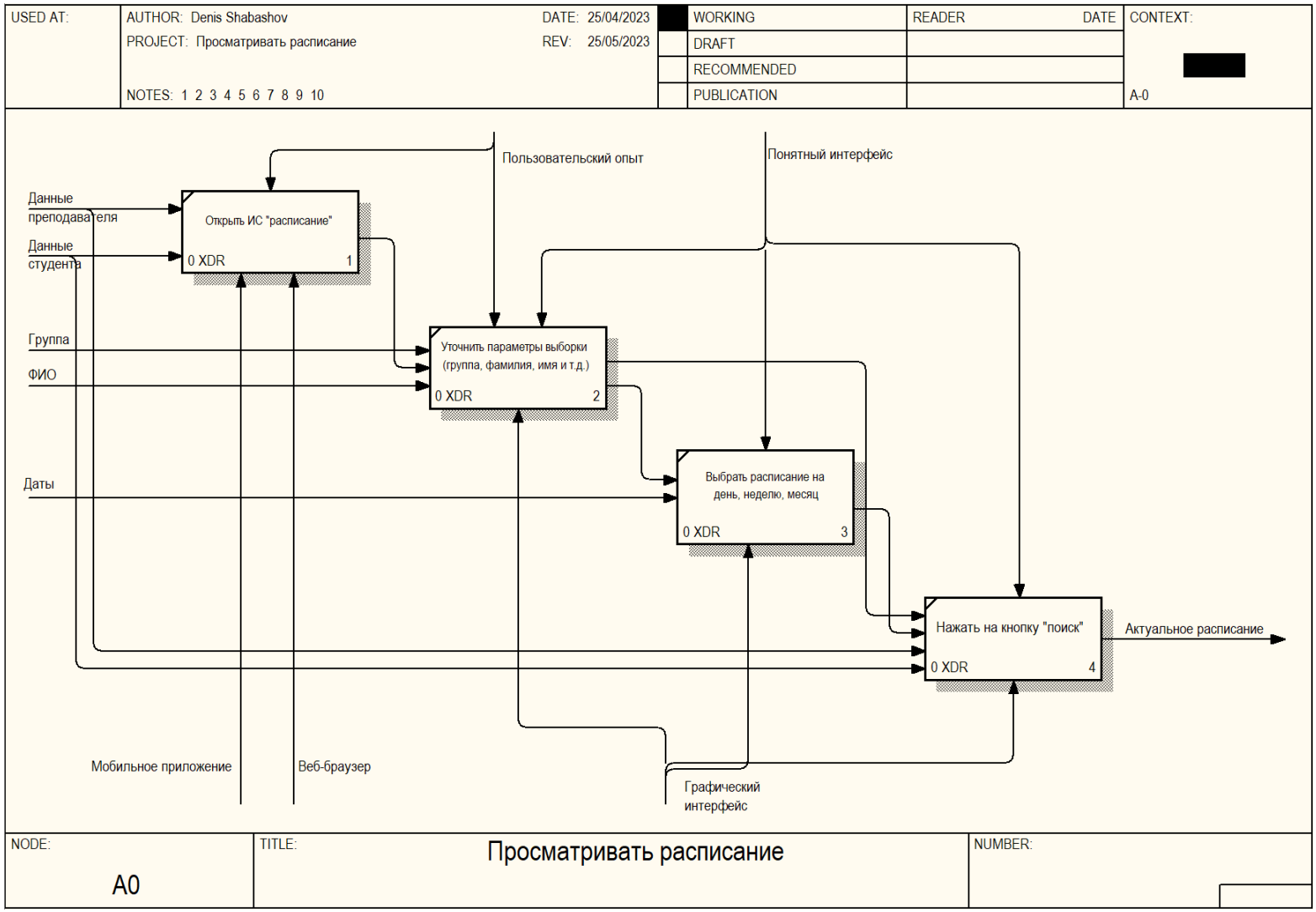

Рисунок 2.2 Декомпозиция бизнес-процесса, Просмотр расписания, нотация IDEF0

Второй основной бизнес-процесс — «Составление учебного процесса»

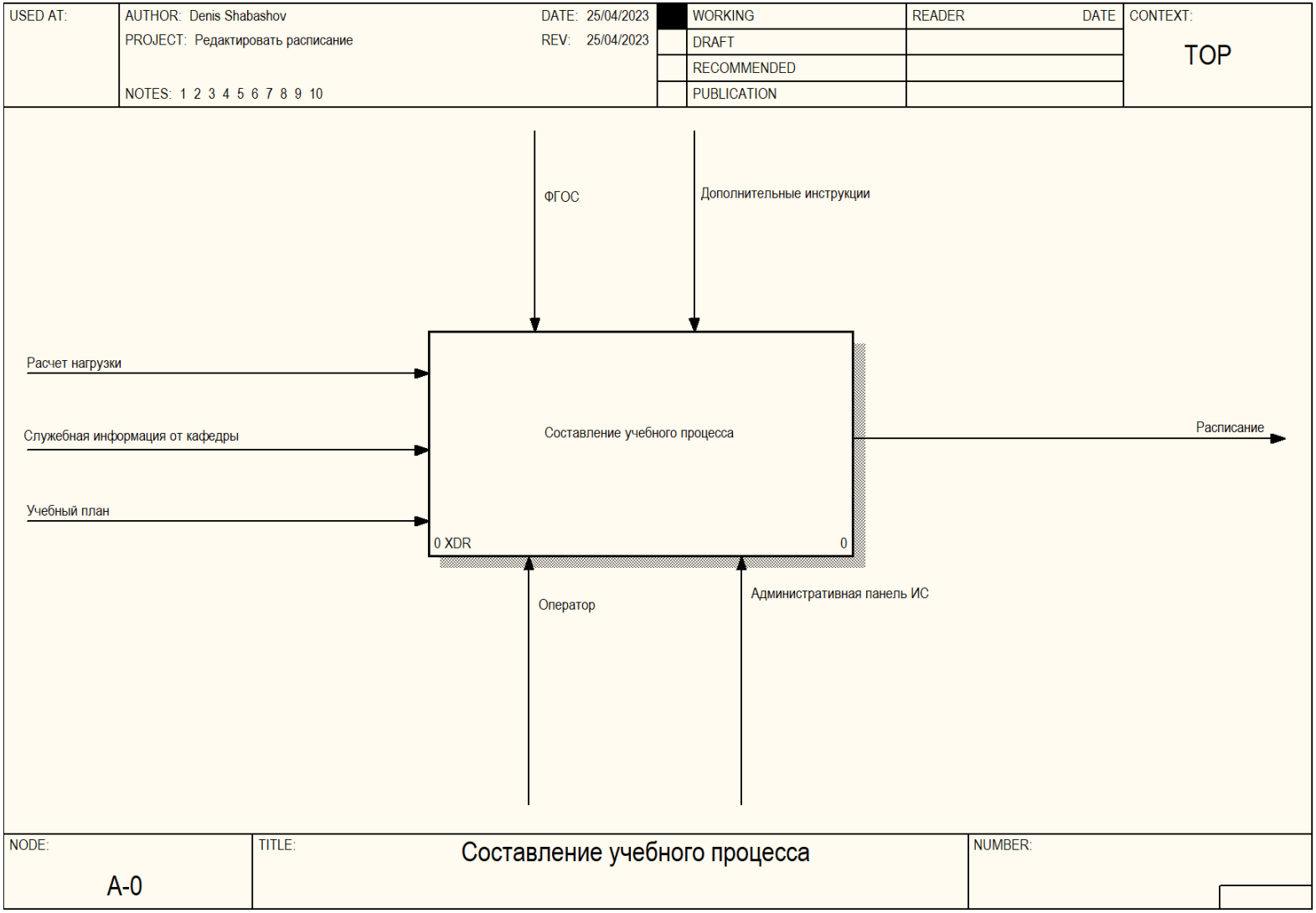

Рисунок 2.3 Контекстная диаграмма, составление учебного процесса, нотация IDEF0

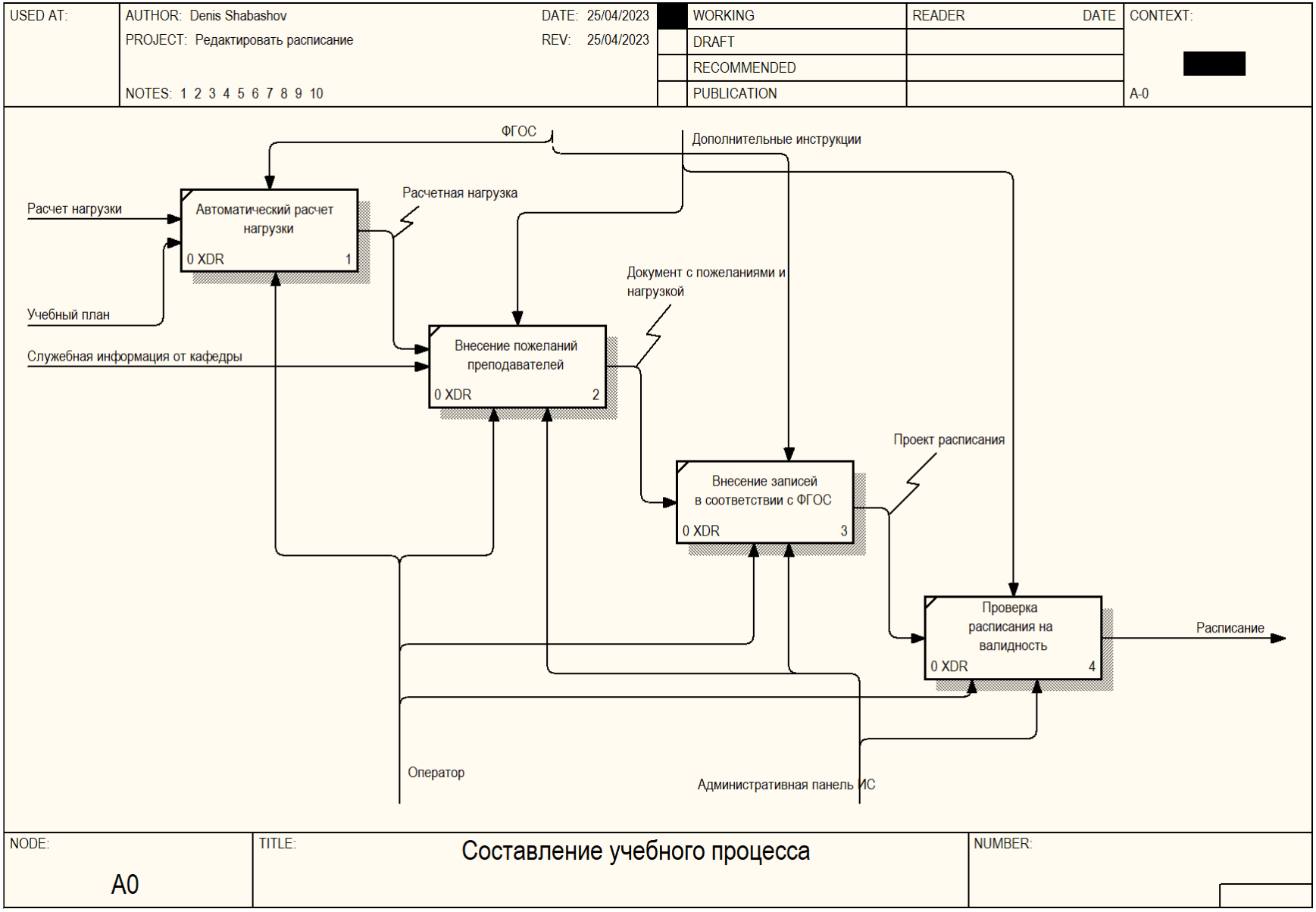

Рисунок 2.4 Декомпозиция, составление учебного процесса, нотация IDEF0

Выделено 2 (два) основных бизнес-процесса:

- составление учебного процесса
- просмотр расписания

Просмотр расписания — это пользовательский запрос, включающий в себя ФИО, диапазон дат, группу студентов и другие параметры. Студенты и преподаватели могут просматривать расписание занятий с помощью ПК или мобильных устройств, используя браузерное или мобильное приложение. Просмотр расписания является не только удобным инструментом для пользователей, но и необходимостью для эффективной работы учебного заведения. Приложение для просмотра расписания должно учитывать потребности пользователей, такие как возможность просмотра расписания разных групп и преподавателей, легкость использования и безопасность приложения. Расписание может содержать конфиденциальную информацию, поэтому безопасность приложения является важным аспектом. Цель разработки приложения для просмотра расписания - это создание мощного и гибкого инструмента, который поможет студентам и преподавателям эффективно использовать свое время и улучшить качество образования.

Составление учебного процесса — это процесс составления расписания в автоматизированной системе, учитывающий учебный план, ФГОСы, пожелания преподавателей, служебные записки кафедры, дополнительные инструкции. Модуль приложения, который отвечает за составление расписания, должен соответствовать бизнес-требованиям и содержать несколько функций, чтобы оператор мог работать эффективно. Когда составление расписания завершено, автоматизированная система должна позволить студентам и преподавателям получить доступ к расписанию в удобном для них формате. Это может быть онлайн-календарь, который можно синхронизировать со своими мобильными устройствами, или печатная версия, которую можно разместить на доске объявлений. Кроме того, система должна позволять операторам вносить

изменения в расписание при необходимости. Например, если преподаватель заболел и не может провести занятие, оператор должен иметь возможность быстро найти замену и внести изменения в расписание. Также важно, чтобы система предоставляла отчеты о том, как используются аудитории и как распределяются преподаватели. Это поможет улучшить эффективность использования ресурсов и оптимизировать расписание для всех участников учебного процесса. Наконец, система должна быть легко настраиваемой и адаптируемой к изменениям в учебном плане или ФГОСах. Это обеспечит гибкость и возможность быстро реагировать на изменения в учебном процессе. Составление расписания - это важный процесс в учебном процессе, который требует множества функций и возможностей от автоматизированной системы. Она должна быть легко настраиваемой и адаптируемой к изменениям в учебном плане, чтобы быстро реагировать на изменения в учебном процессе. Оптимизация расписания поможет улучшить эффективность использования ресурсов и упростить работу операторов.

Таким образом для успешной разработки автоматизированной системы расписания необходимо построить диаграммы прецедентов. Это поможет определить, какие функциональные возможности должны быть включены в систему, чтобы она была наиболее полезной для пользователей. Диаграммы вариантов использования также помогут лучше понять потребности пользователей и создать интуитивно понятный интерфейс. Однако, важно помнить, что диаграммы вариантов использования не являются конечной целью разработки системы. После их построения необходимо создать дизайн системы, написать код, провести тестирование и наладить запуск. Поэтому, чтобы успешно завершить проект, необходимо иметь хорошо продуманный план действий и определенную методологию разработки.

2.3. Описание функциональных требований к системе

Приступим к моделированию:

Рисунок 2.5 Диаграмма прецедентов системы расписания

На изображении выше показаны различные варианты использования системы планирования занятий. Администратор системы имеет доступ ко всем основным модулям, в то время как пользователи — студенты и преподаватели — имеют ограниченный доступ к панели мониторинга, загрузке преподавательского состава, учебной программе и модулю отчетов. Система планирования занятий — это программное обеспечение, которое позволяет учебным заведениям автоматизировать процесс планирования учебных занятий и управления учебным процессом. Необходимо декомпозировать каждый прецедент. Ниже приведены диаграммы и таблицы, описывающие, как пользователь использует систему для достижения конкретной цели.

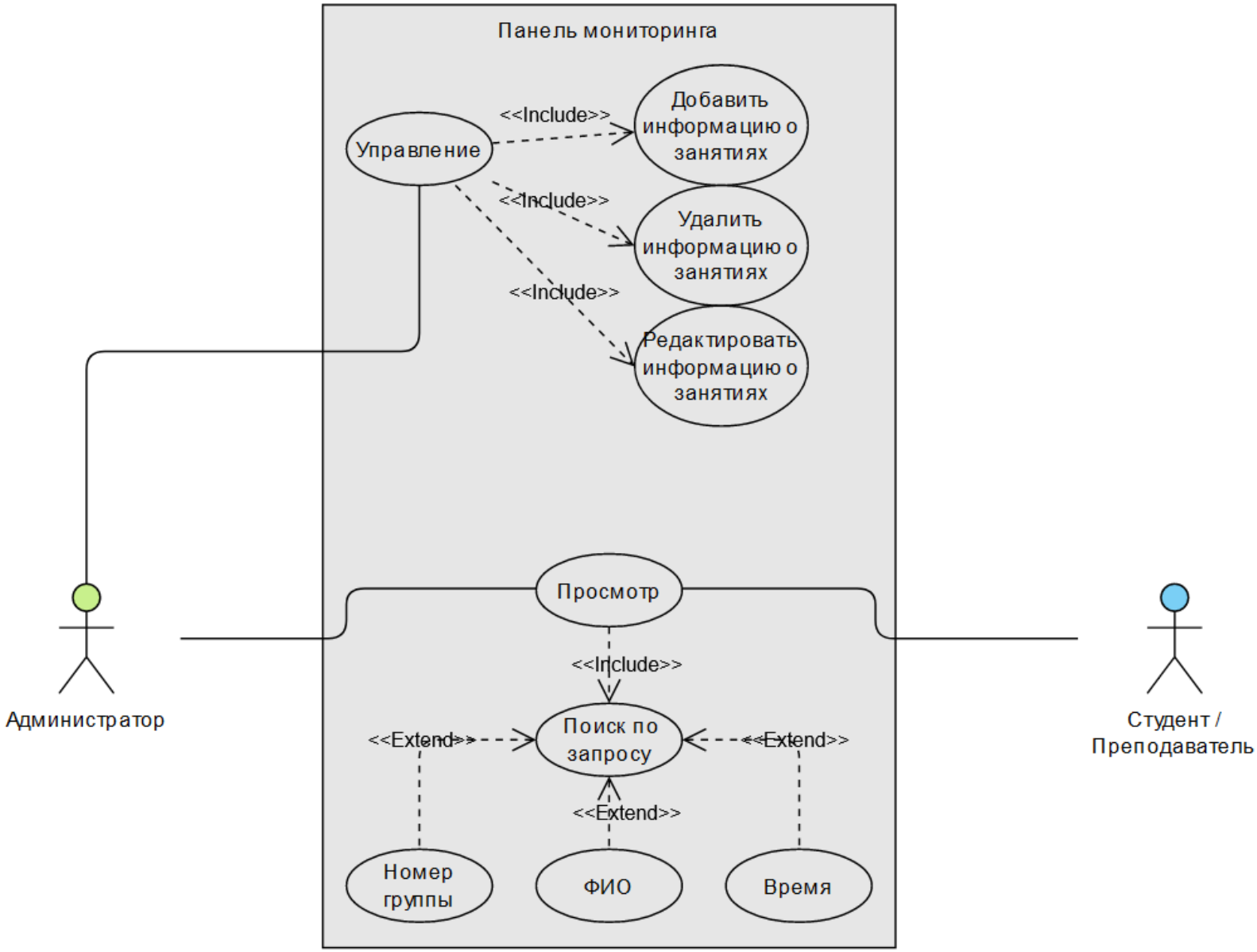

Рисунок 2.6 Панель мониторинга, нотация UML, Use-Case диаграмма

Ниже приведена описательная таблица:

Таблица 2.2 — Описательная таблица варианта «Панель мониторинга»

Наименование	**Описание**
Прецедент	Использование панели мониторинга
Субъект (-ы)	Администратор, студент, преподаватель
Описание	Функция используется для управления информацией, отображаемой на панели мониторинга системы планирования занятий
Успешное завершение	Учащиеся и преподаватели могут просматривать информацию, отображаемую на панели мониторинга, используя эту функцию. Администратор может выполнять поиск, добавлять, обновлять и удалять информацию, которая будет отображаться на панели мониторинга
Альтернатива	Учащиеся и преподаватели могут просматривать только панель мониторинга; администратор может получать доступ ко всей информации панели мониторинга и управлять ею
Предусловие	Чтобы получить доступ к панели мониторинга, администратор, студенты и преподаватели должны быть авторизованы
Постусловие	Информация на панели мониторинга обновлена и доступна

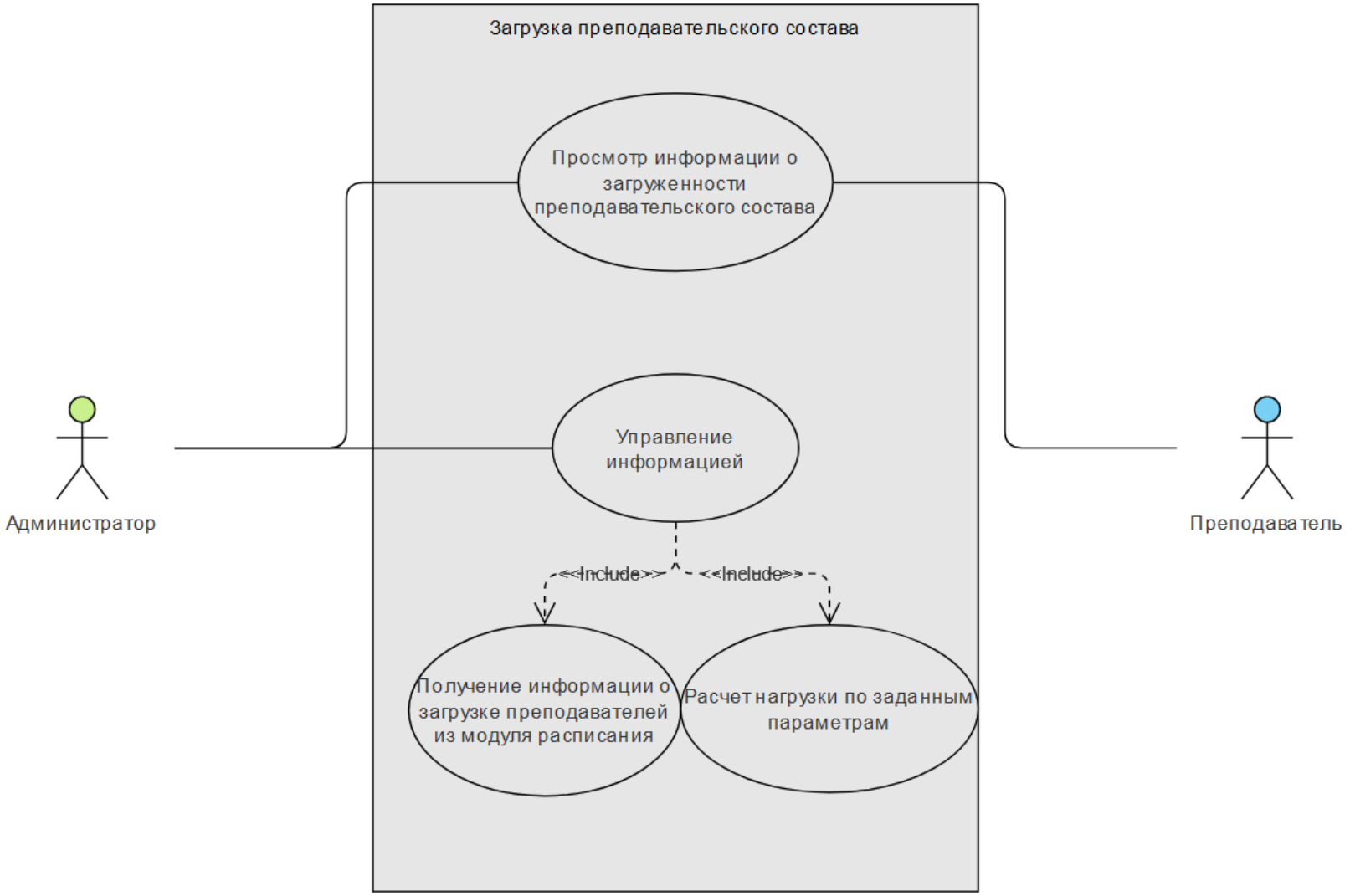

Рисунок 2.7 Загрузка преподавательского состава, нотация UML, Use-Case диаграмма

Таблица 2.3 — Описательная таблица варианта «Загрузка преподавательского состава»

Наименование	**Описание**
Прецедент	Загрузка преподавательского состава
Субъект (-ы)	Администратор, преподаватель
Описание	Функция используется для управления нагрузкой преподавателей в системе по их расписанию.
Успешное завершение	С помощью этой функции преподаватели могут просматривать свое расписание и предметную нагрузку. Администратор может искать, добавлять, обновлять и удалять информацию в модуле загрузки преподавателей.
Альтернатива	Преподаватели могут только просматривать свою загрузку; Администратор может получить доступ и управлять загрузкой всех факультетов.
Предусловие	Чтобы получить доступ к загрузке преподавательского состава, администратор и преподаватели должны быть авторизованы
Постусловие	Информация о загрузке факультета обновлена

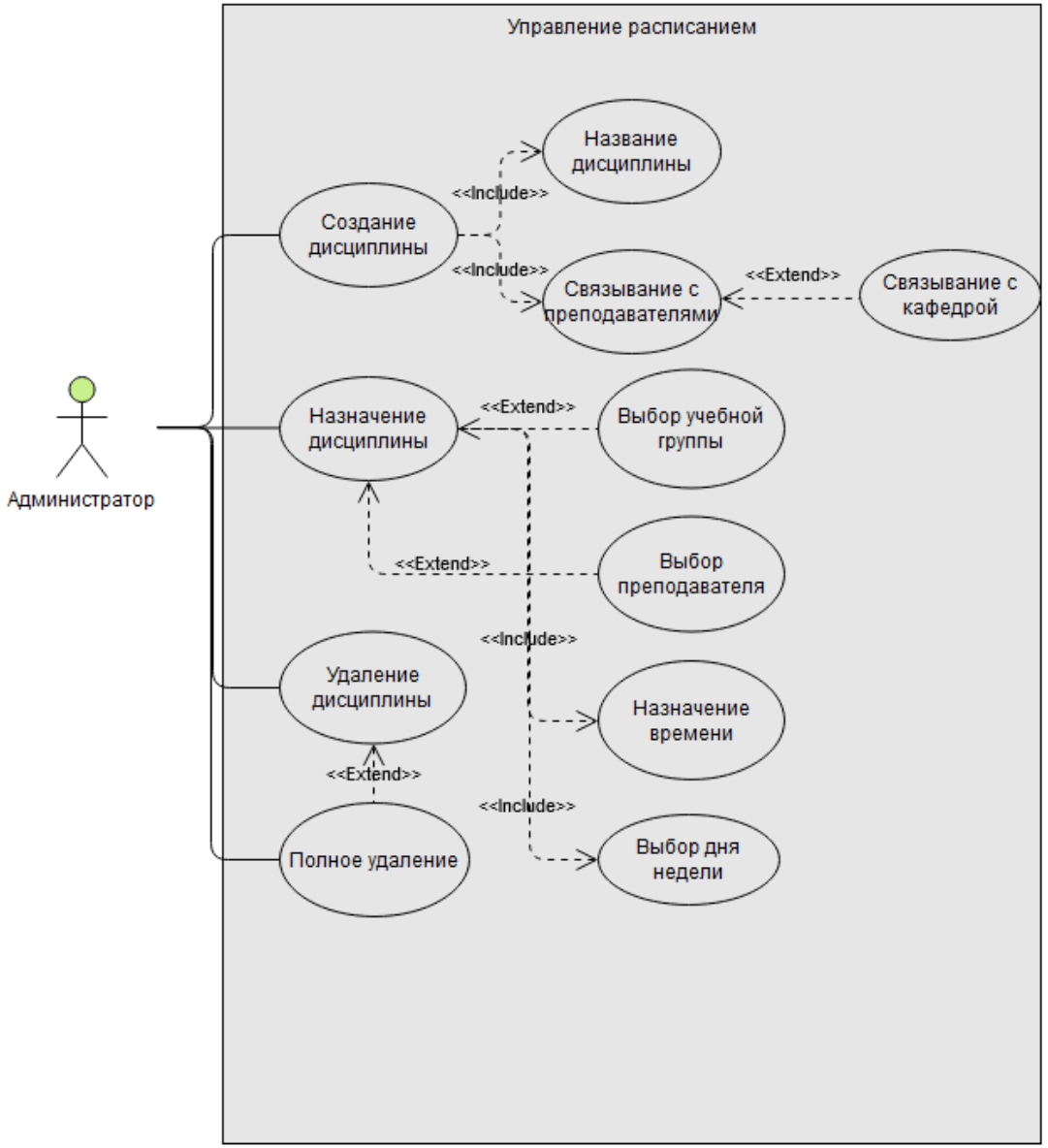

Рисунок 2.8 Модуль управления расписанием, нотация UML, Use-Case диаграмма

Таблица 2.4 — Описательная таблица варианта «Управление расписанием»

Наименование	**Описание**
Прецедент	Управление расписанием
Субъект (-ы)	Администратор
Описание	Функция используется для управления деталями и функциями, которые должны быть в модуле планирования.
Успешное завершение	С помощью этой функции администратор может искать, добавлять, обновлять и удалять сведения о расписании в системе
Альтернатива	администратор может получить доступ ко всей информации о расписании и управлять ею.
Предусловие	Чтобы получить доступ к панели мониторинга, администратор должен быть авторизован
Постусловие	Информация о расписании обновлена и доступна

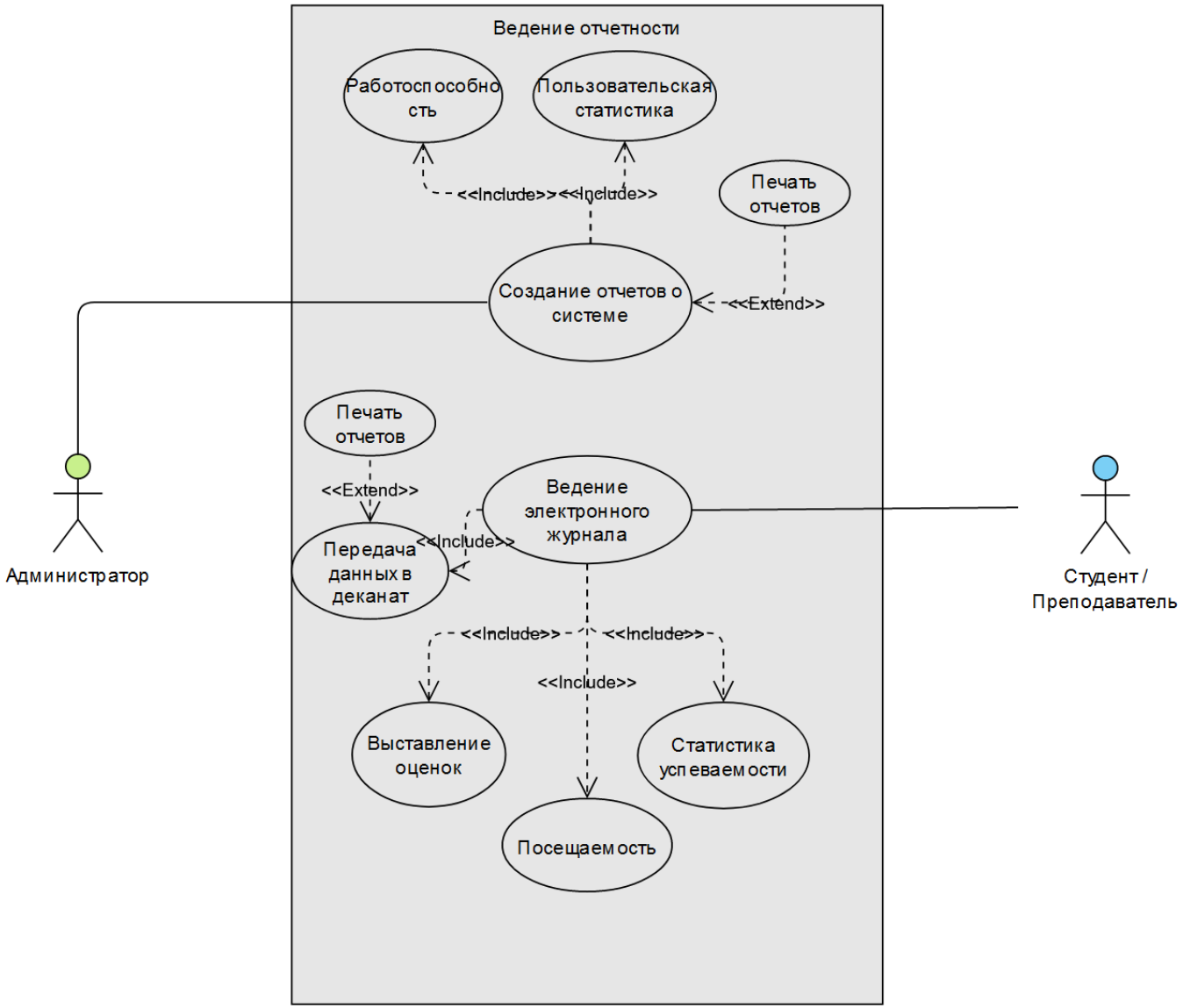

Рисунок 2.9 Модуль ведения отчетности, нотация UML, Use-Case диаграмма

Таблица 2.5 — Описательная таблица варианта «Ведение отчетности»

Наименование	**Описание**
Прецедент	Ведение отчетности
Субъект (-ы)	Администратор, студент, преподаватель
Описание	Функция используется для просмотра и печати отчетов в системе
Успешное завершение	С помощью этой функции учащиеся и учителя могут просматривать и распечатывать отчеты о расписании занятий, вести электронный журнал. Администратор может просматривать, распечатывать и экспортировать отчет системы
Альтернатива	-
Предусловие	Чтобы получить доступ к панели мониторинга, администратор, студенты и преподаватели должны быть авторизованы
Постусловие	Бумажная и электронная копии отчета системы доступны для просмотра

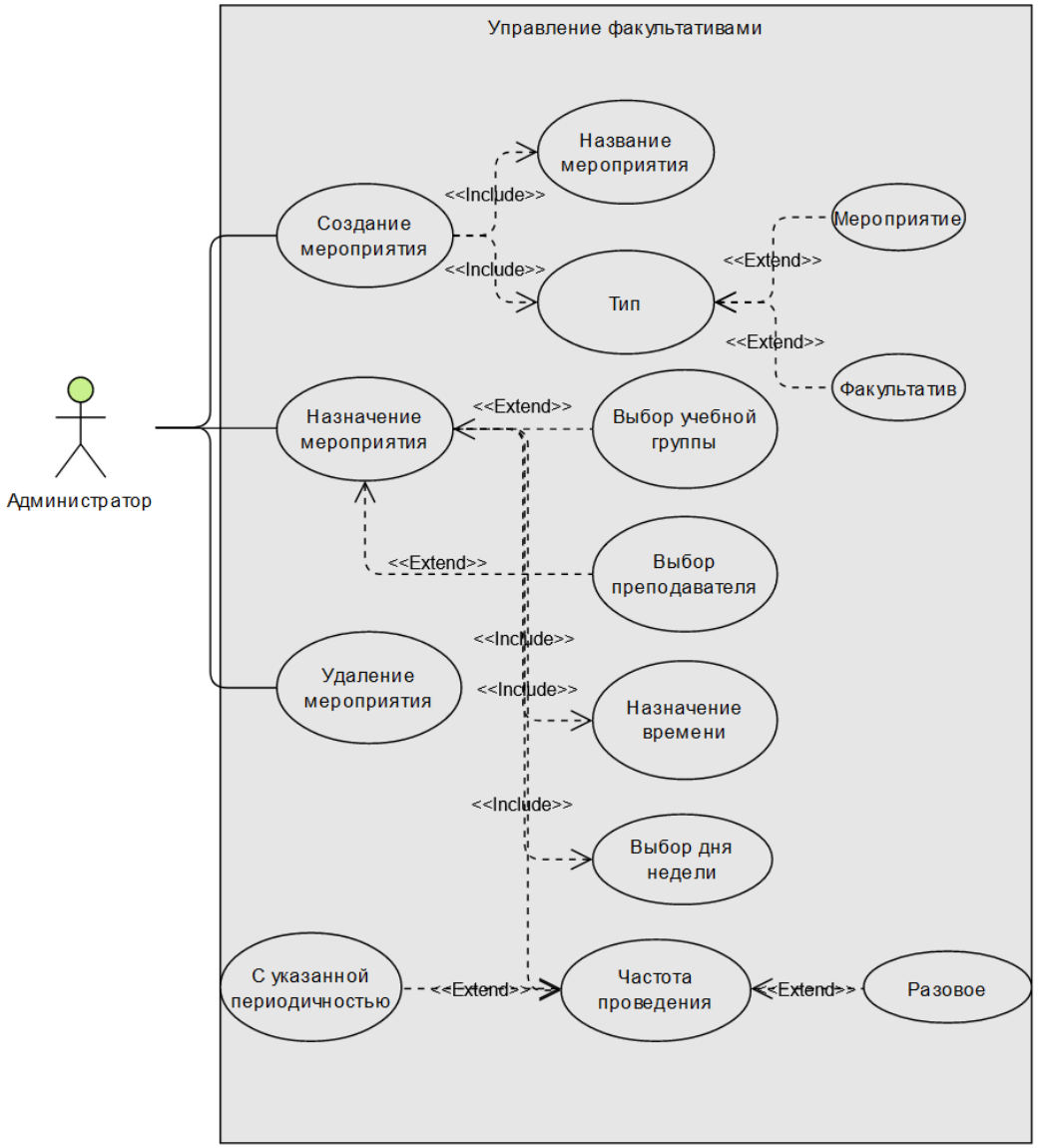

Рисунок 2.10 Управление факультативами и мероприятиями, нотация UML, Use-Case диаграмма

Таблица 2.6 описательная таблица варианта «Управление факультативами»

Наименование	**Описание**
Прецедент	Управление факультативами
Субъект (-ы)	Администратор
Описание	Функция используется для управления расписанием факультативов или мероприятий
Успешное завершение	С помощью этой функции администратор может просматривать, добавлять и удалять факультативные занятия. Студенты и преподаватели могут просматривать актуальное расписание
Альтернатива	-
Предусловие	Чтобы получить доступ к управлению факультативами, администратор должен быть авторизован
Постусловие	Актуальное расписание факультативов и дополнительных (необязательных) занятий, встреч и мероприятий доступно для просмотра

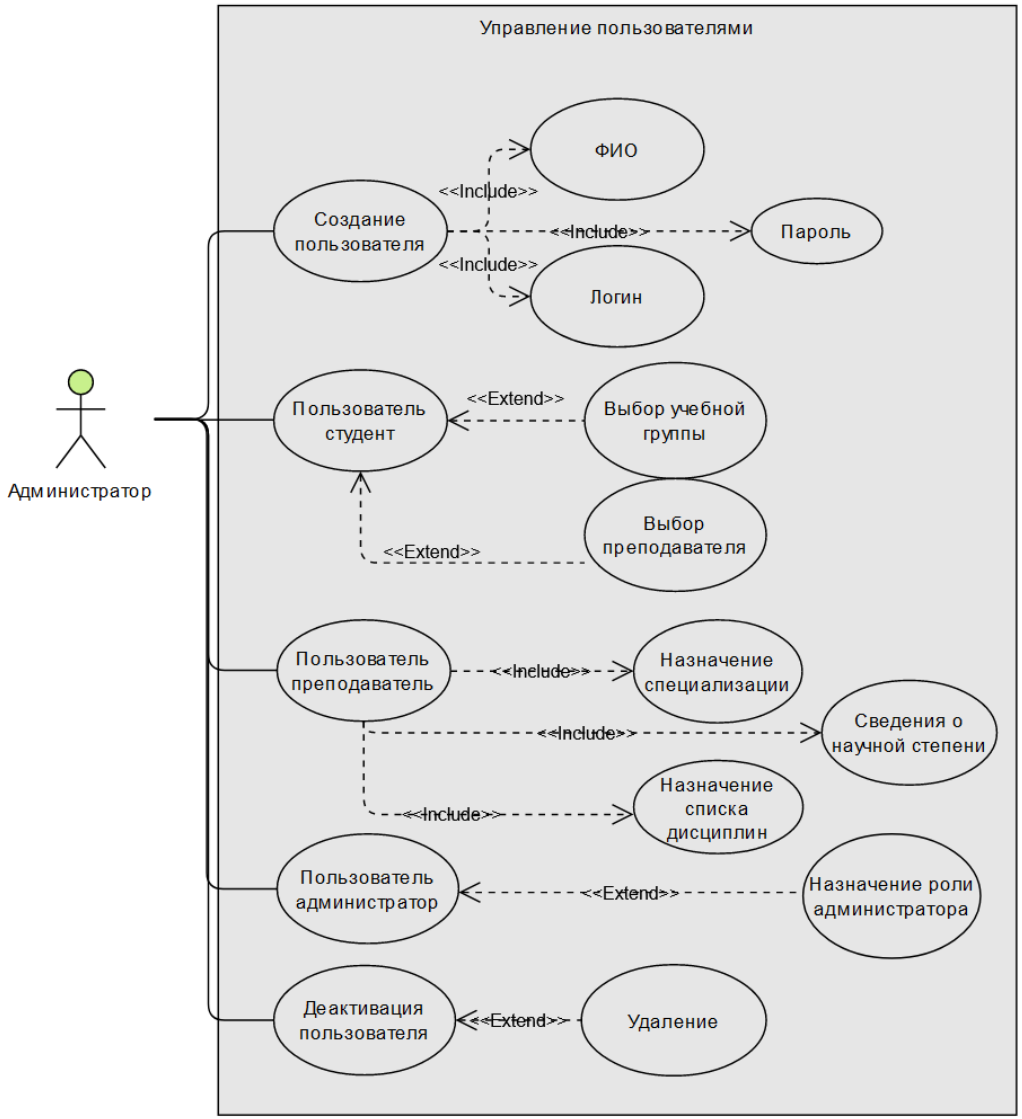

Рисунок 2.11 Управление пользователями, нотация UML, Use-Case диаграмма

Таблица 2.7 описательная таблица варианта «Управление пользователями»

Наименование	**Описание**
Прецедент	Управление пользователями
Субъект (-ы)	Администратор
Описание	Функция используется для управления пользователями системы
Успешное завершение	С помощью этой функции администратор может искать, добавлять, обновлять и удалять данные пользователя в системе, подтверждать самостоятельно созданные пользовательские аккаунты и их роли
Альтернатива	Администратор может получить доступ ко всем профилям пользователей и управлять ими.
Предусловие	Чтобы получить доступ к управлению пользователями, администратор должен быть авторизован
Постусловие	Обновлен профиль пользователя системы и доступен для просмотра.

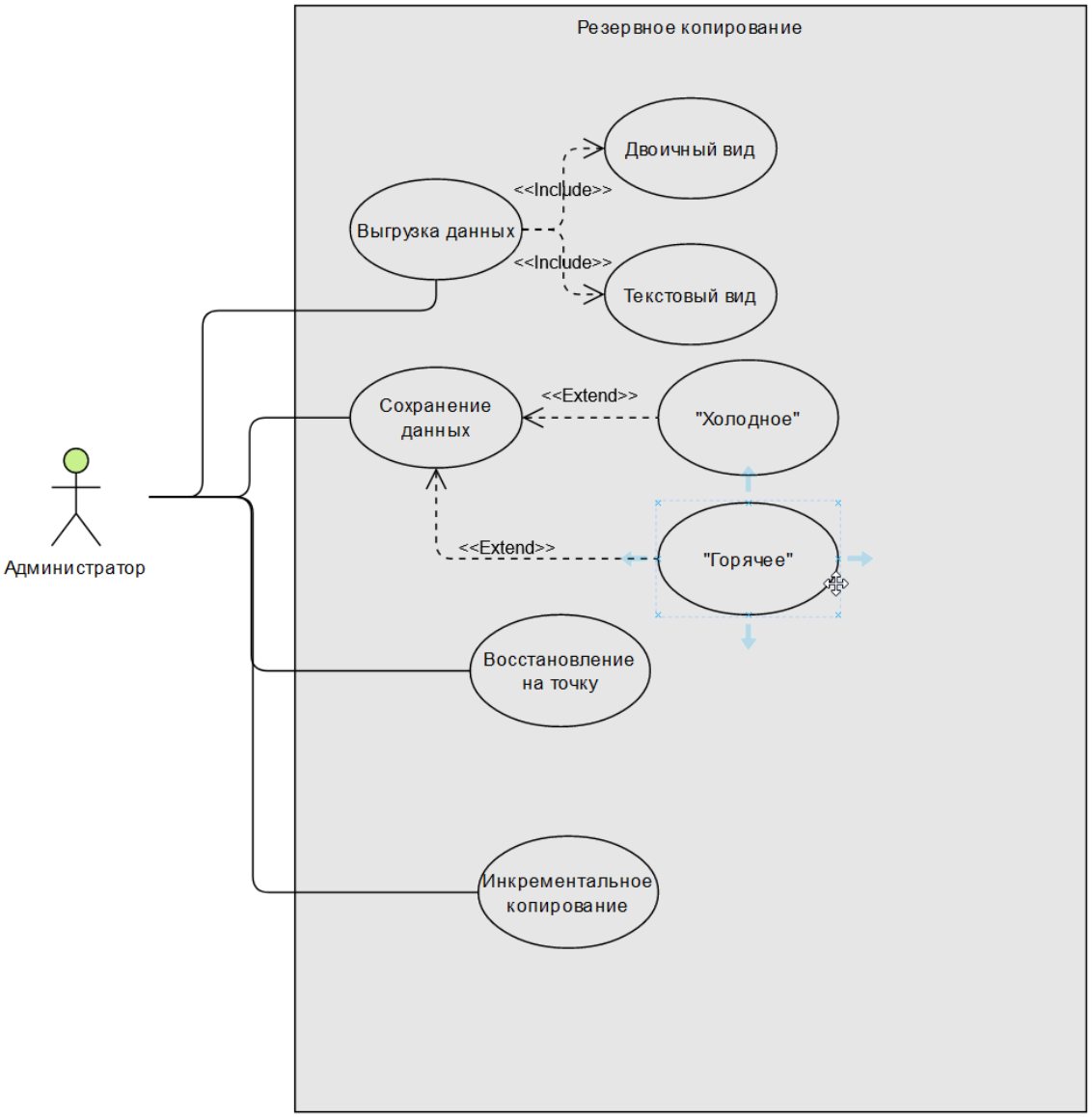

Рисунок 2.12 Резервное копирование, нотация UML, Use-Case диаграмма

Таблица 2.8 описательная таблица варианта «Резервное копирование»

Наименование	**Описание**
Прецедент	Резервное копирование
Субъект (-ы)	Администратор
Описание	Функция используется для управления резервной базой данных системы
Успешное завершение	Администратор может добавлять, редактировать и обновлять информацию о резервной копии базы данных
Альтернатива	-
Предусловие	Администратор создаст и подключит резервную базу данных.
Постусловие	Новая резервная база данных готова к восстановлению в случае повреждения основной рабочей БД

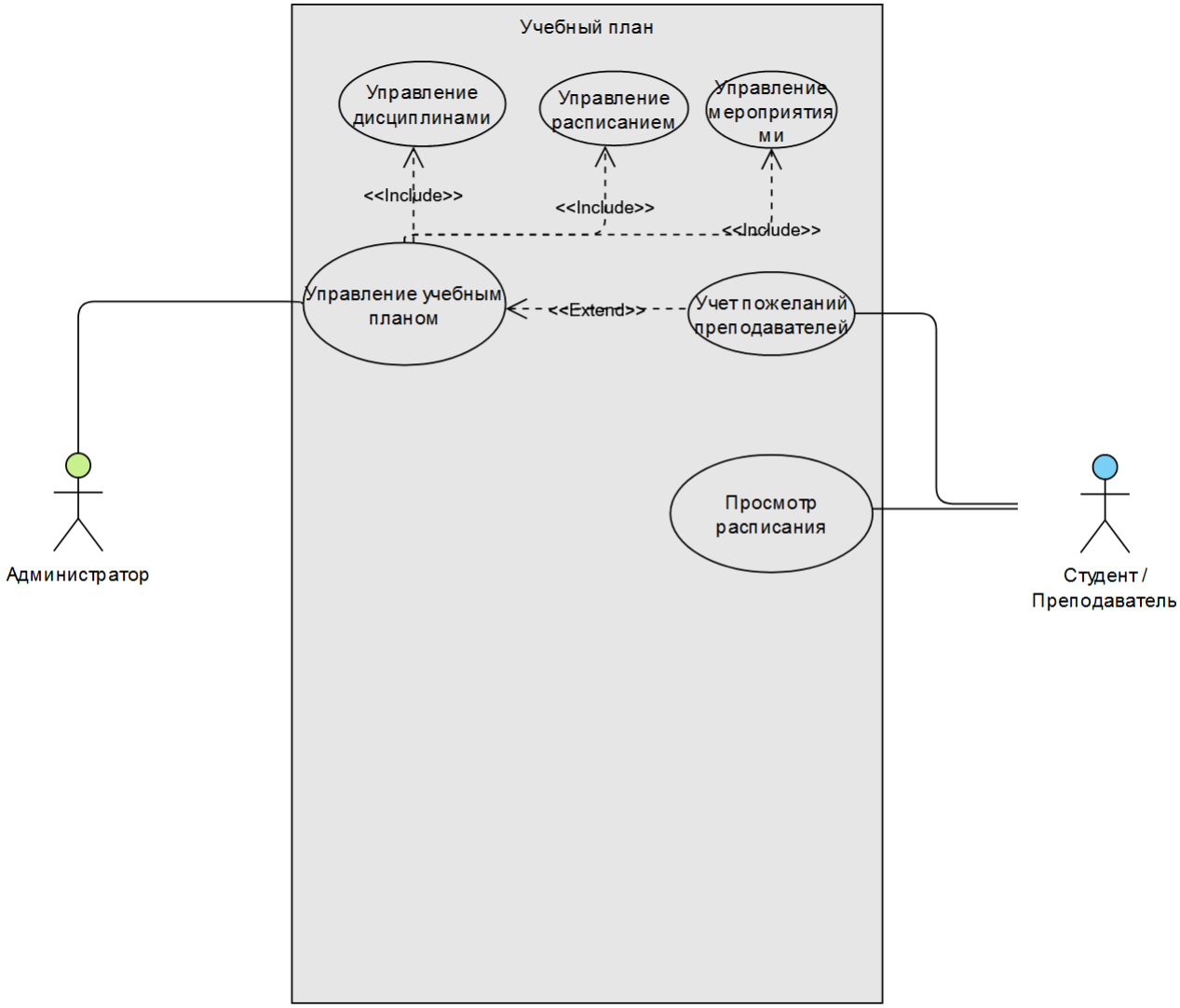

Рисунок 2.13 Учебный план, нотация UML, Use-Case диаграмма

Таблица 2.9 описательная таблица варианта «Учебный план»

Наименование	**Описание**
Прецедент	Учебный план
Субъект (-ы)	Администратор, студент, преподаватель
Описание	Функция используется для просмотра и управления учебным планом и предметами учителей и учеников в системе
Успешное завершение	С помощью этой функции учащиеся и учителя могут просматривать учебную программу и предметы. Администратор может искать, добавлять, обновлять и удалять информацию об учебном плане и предметах
Альтернатива	Студенты и преподаватели могут просматривать учебную программу и предметы; администратор может получить доступ и управлять всей информацией об учебном плане/предмете
Предусловие	Чтобы получить доступ к панели мониторинга, администратор, студенты и преподаватели должны быть авторизованы
Постусловие	Информация об учебной программе и предметах обновлена и доступна для просмотра

Таким образом, были построены UseCase диаграммы для модулей интерактивной системы ВУЗ:

- Панель мониторинга
- Загрузка преподавательского состава
- Управление расписанием
- Ведение отчетности
- Управление факультативами
- Управление пользователями
- Резервное копирование БД
- Учебный план

Перед тем, как описывать необходимую функциональность системы, важно построить диаграмму потоков данных для более полного представления ее структуры данных. Это поможет понять, какие функции имеет смысл реализовывать в первую очередь.

Построение диаграммы потоков данных является важным шагом в разработке интерактивной системы ВУЗа.

Она поможет разработчикам более полно понять структуру данных системы и определить, какие функции следует реализовывать в первую очередь. После построения диаграммы потоков данных разработка может начать работу над реализацией необходимой функциональности. Кроме того, интерактивная система ВУЗа может включать в себя множество других функций, которые могут быть полезны для студентов, преподавателей и администраторов. Например, система может предоставлять возможность онлайн-обучения, управления библиотекой, а также предоставлять информацию о различных мероприятиях, проходящих в ВУЗе. Разработка интерактивной системы ВУЗа является сложным и многогранным процессом, который требует тщательного планирования и анализа.

2.4. Описание информационных потоков и данных

Моделирование процессов бизнеса с помощью диаграмм потоков данных (DFD) является одним из ключевых инструментов при проектировании информационных систем.

DFD-модели представляют собой графические диаграммы, которые показывают потоки данных в системе, состоящие из блоков, представляющих процессы, и стрелок, указывающих направление потоков данных.

Создание DFD-моделей позволяет определить, какие данные нужны для каждого процесса в системе, выявить потенциальные узкие места и оптимизировать процессы.

Одним из ключевых принципов создания DFD-моделей является определение границ системы, которые определяют, какие данные входят и выходят из системы, и какие процессы входят и выходят за ее пределы.

Другим важным принципом является определение уровней DFD-моделей, где на верхнем уровне показываются общие потоки данных, а на более низких уровнях показываются более детальные процессы и потоки данных.

Также важно использовать правильные символы и обозначения в DFD-моделях, где каждый блок должен иметь уникальное имя, каждый поток данных должен иметь уникальный идентификатор, и каждый блок должен иметь описание, показывающее, что он делает.

DFD-модели помогают разработать более эффективную информационную систему, которая будет обрабатывать данные быстрее и точнее. Они также могут использоваться для определения требований к системе, для оценки ее производительности и для обучения пользователей системы. Важно понимать, что DFD-модели не являются окончательными и могут изменяться в процессе разработки системы, поэтому важно вести их актуализацию и обновление.

Разработаны модели потока данных для управления расписанием и получения метеоинформации.

DFD диаграммы вынесены в приложение Б, рисунок 2.14, 2.15, 2.16, 2.17.

Администратор должен выбрать группу, временной слот, день недели и преподавателя для создания расписания.

Для получения метеоданных приложения обращаются к микросервису "метеосервер", который получает данные из сервисов ГРМЦ, сохраняет их в базу данных и высылает ответ на запрошенные параметры. Если данные уже записаны, повторный запрос к сервисам ГРМЦ не требуется.

Модели потока данных упрощают процесс управления расписанием и получения метеоинформации, снижают нагрузку на серверы и ускоряют обработку запросов. Благодаря им процесс обучения и управления учебным процессом становится более эффективным и удобным для всех участников.

2.5. Описание взаимодействия между объектами

В данном разделе рассматривается создание диаграмм последовательности для информационной системы управления расписанием и интерактивной ГИС ВУЗа.

Диаграммы последовательности являются важным инструментом для моделирования взаимодействия между объектами в системе и позволяют более наглядно представить процессы, происходящие в системе.

Для создания диаграмм последовательности в информационной системе управления расписанием и интерактивной ГИС ВУЗа необходимо выполнить несколько шагов.

Во-первых, необходимо определить объекты, между которыми будут происходить взаимодействия в системе. Это могут быть, например, студенты, преподаватели, администрация ВУЗа, учебные кабинеты и т.д.

Затем необходимо определить, какие действия будут происходить между этими объектами. Студент может зарегистрироваться на определенный курс, преподаватель может создать расписание занятий, администрация ВУЗа может назначить аудитории для проведения занятий и т.д.

После определения объектов и действий необходимо создать диаграмму последовательности, которая будет отображать взаимодействие между объектами и действиями в системе.

Для этого необходимо использовать специальные элементы диаграммы, такие как актеры, объекты, сообщения и т.д. Создание диаграмм последовательности позволяет более наглядно представить процессы, происходящие в системе, и оптимизировать работу информационных систем.

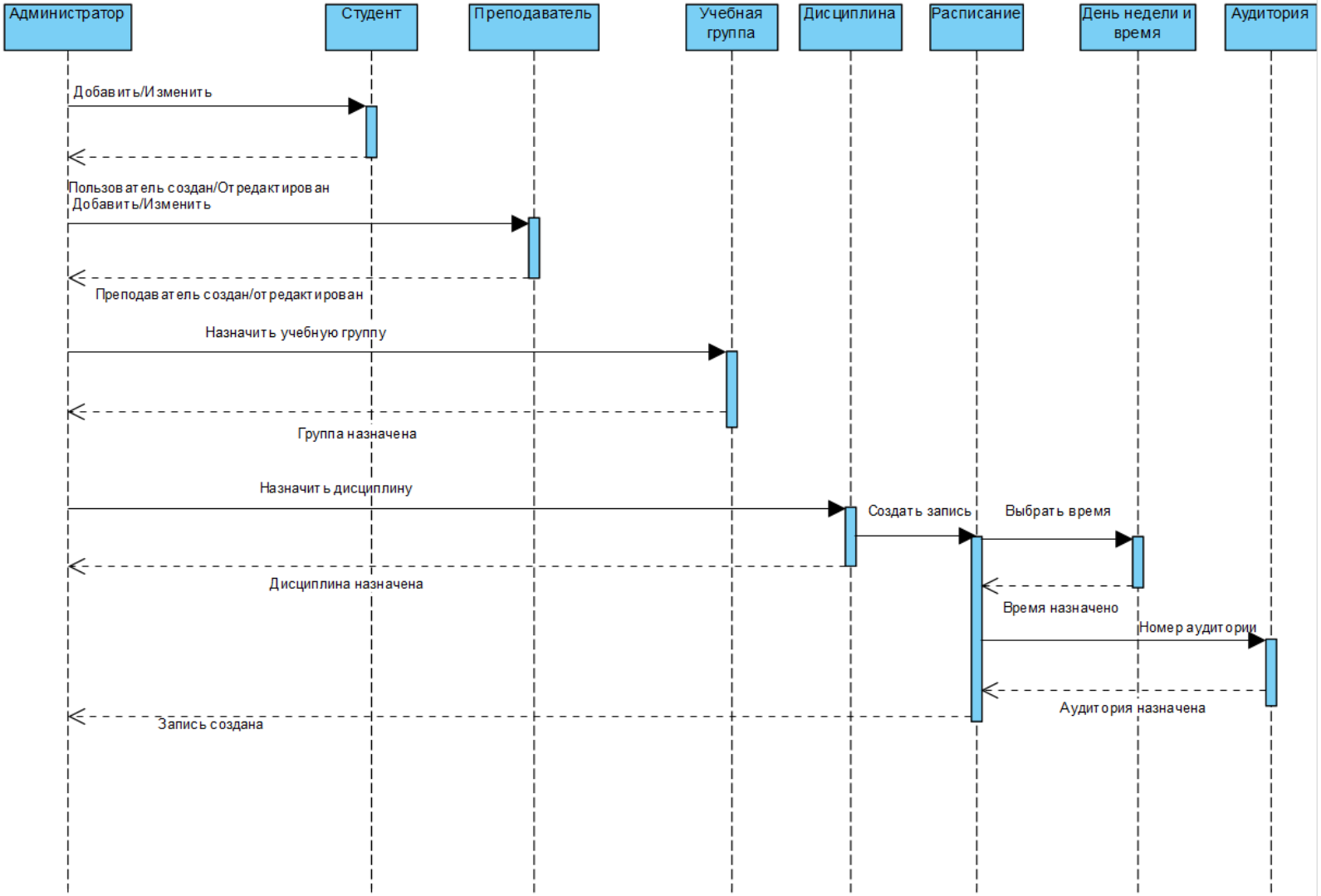

Рисунок 2.14 Управление расписанием, нотация UML, диаграмма последовательности

В результате разработана диаграмма последовательности, отвечающая за создание записи в расписании.

Модель отражает следующую последовательность действий администратора:

1. Добавление или редактирование студента
2. Добавление или редактирование преподавателя
3. Назначить учебную группу преподавателю
4. Назначить учебную группу студенту, если он в ней не состоит
5. Добавить и назначить изучаемую дисциплину
6. Создать запись
7. Назначить аудиторию и время или отложить момент назначения допускается после выполнения предыдущих шагов.
8. Назначить временной интервал (создать, редактировать)
9. Назначить аудиторию (создать, редакактировать)

После воспроизведения всех шагов бизнес-процесс считается завершенным.

Так же есть возможность редактирования и удаления сущностей:

- Пользователь
- Учебная группа
- Дисциплина
- Запись расписания
- Аудитория
- Временной интервал
- День недели

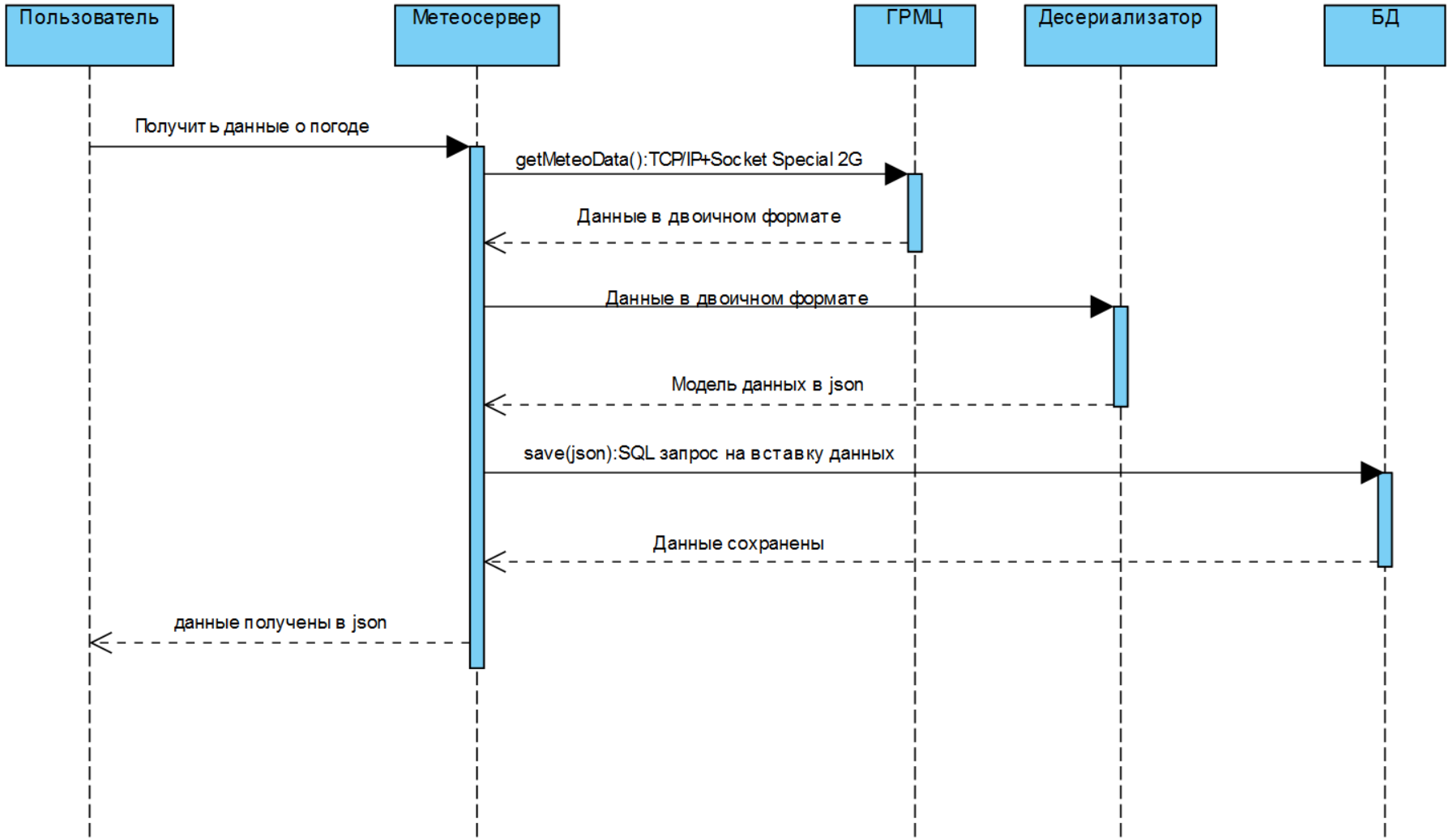

Рисунок 2.15 Получение данных о погоде, нотация UML, диаграмма последовательности

Разработана диаграмма последовательности, отвечающая за получение данных от метеосервера.

Модель отражает следующую последовательность действий пользователя и метеосервера:

1. Отправление запроса на получение метеоданных.
2. Метеосервер по протоколу Socket Special 2G, который работает через протокол TCP/IP, устанавливает соединение с сервером ГРМЦ. Начинается прием данных.
3. Метеосервер принимает данные в двоичном виде.
4. Метеосервер вызывает методы десериализатора и получает данные в json виде.
5. Метеосервер отправляет данные на сервер с базой данных и выполняет запрос на вставку данных.
6. Метеосервер отвечает клиенту; Ответ содержит десериализованные данные.
7. Клиент принимает данные в виде json.

В итоге были разработаны диаграммы последовательности, что означает, что проектирование и описание взаимодействия объектов завершено.

Следующий этап — модель данных.

2.6. Модель данных

При создании баз данных необходимо учитывать множество аспектов, включая структуру данных и связи между ними. Одной из наиболее распространенных нотаций для проектирования баз данных является IDEF1X, которая основана на научном подходе и предоставляет мощный инструментарий для описания структуры данных и их связей. Проектирование баз данных в нотации IDEF1X начинается с определения всех сущностей, которые будут храниться в базе данных. Затем определяются атрибуты каждой сущности и их типы данных. После этого необходимо определить связи между сущностями, которые могут быть однозначными или многозначными, и каждая связь должна иметь определенный тип, такой как один к одному, один ко многим или многие ко многим.

При проектировании баз данных также необходимо учитывать нормализацию, которая позволяет разделить таблицы на более мелкие, чтобы уменьшить повторение данных и улучшить производительность базы данных. После определения всех сущностей, атрибутов и связей необходимо создать диаграмму базы данных в нотации IDEF1X, которая должна быть понятной и легко читаемой для других разработчиков. При правильном выполнении процесса моделирования, можно создать мощную и эффективную базу данных, которая будет служить основой для любого приложения.

Важно отметить, что при проектировании баз данных необходимо также учитывать требования к безопасности и защите данных, а также возможность масштабирования базы данных в будущем. Кроме того, проектирование баз данных должно быть гибким, чтобы можно было вносить изменения в структуру данных и связи между ними в случае необходимости. Базы данных используются для хранения и управления большим объемом информации, которая может быть использована для принятия решений и оптимизации бизнес-процессов.

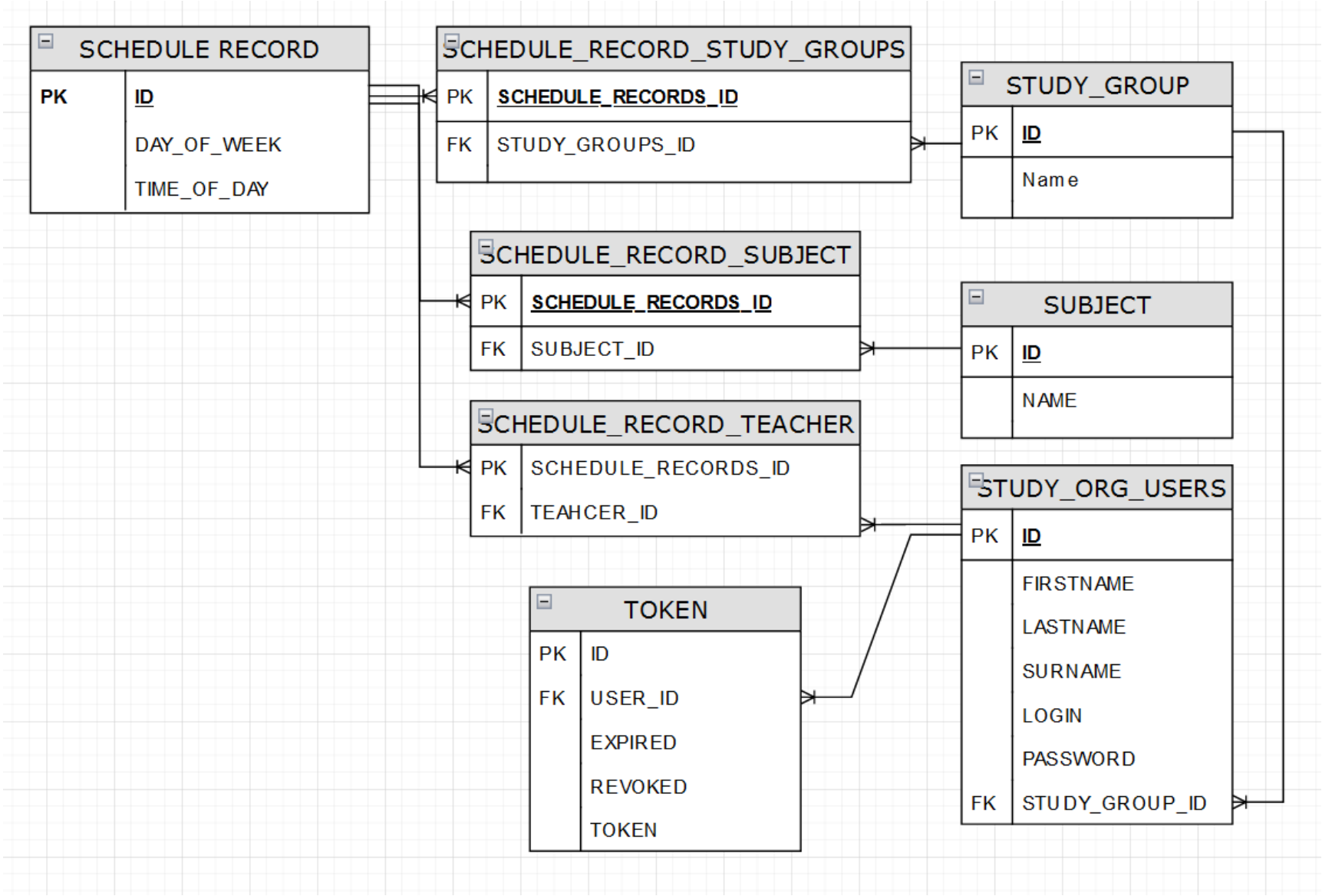

Рисунок 2.16 Модель данных, нотация IDEF1X

В итоге получена модель данных, которая содержит основные сущности необходимые для хранения расписания.

- Запись расписания
- Учебная группа
- Дисциплина
- Пользователи организации
- «Токены» авторизации

Модель прошла нормализацию и имеет 3 нормальную форму.

Достигается 3 нормальная форма базы данных декомпозицию, добавление первичных ключей и зависимость всех атрибутов от первичного ключа. В модели присутствуют связи «Один-ко-многим» и «Многие-ко-многим».

В итоге проект интерактивной ГИС ВУЗа включает в себя:

- Анализ бизнес-процессов
- Описание функциональных требований
- Описание информационных потоков данных
- Описание взаимодействия между объектами
- Модель данных

На основании проектных моделей и диаграмм можно переходить к реализации проекта.

ГЛАВА 3. РЕАЛИЗАЦИЯ ИС

3.1. Выбор инструментов разработки

В таблице 1.3 в системных требованиях присутствует Java, а значит можно рассмотреть ее в качестве проектного языка программирования.

Рассмотрим несколько причин, почему выбор Java может быть хорошим решением для реализации проекта:

1. Кроссплатформенность: Java работает на разных операционных системах, таких как Windows, Linux, MacOS и др. Это означает, что разработчики могут создавать приложения, которые будут работать на разных устройствах, без необходимости переписывать код для каждой платформы.
2. Безопасность: Java имеет встроенные механизмы безопасности, которые защищают приложения от вирусов, злоумышленников и других угроз. Это делает Java одним из самых безопасных языков программирования.
3. Широкое сообщество: Java имеет большое сообщество разработчиков, которые создают различные библиотеки, фреймворки и инструменты. Это может ускорить разработку и улучшить качество кода.
4. Простота: Java имеет простой синтаксис, который легко читать и понимать. Это может упростить работу над проектом и ускорить его разработку.
5. Высокая производительность: Java является компилируемым языком программирования, что означает, что код компилируется в машинный код, что улучшает производительность приложения.

Выбор языка программирования можно считать завершенным, так как Java соответствует всем нефункциональным и системным требованиям.

Выбор реляционной системы управления базами данных (РСУБД) зависит от многих факторов, таких как требования проекта, бюджет и опыт разработчиков. PostgreSQL может быть хорошим выбором для реализации проекта, особенно если в проекте используется язык программирования Java.

1. PostgreSQL является одной из самых надежных и стабильных РСУБД. Он имеет долгую историю разработки и тестирования, а также широкое сообщество пользователей и разработчиков, которые активно работают над его улучшением.
2. PostgreSQL поддерживает множество функций, включая транзакции, хранимые процедуры, триггеры, полнотекстовый поиск и геоданные. Это позволяет разработчикам создавать мощные и гибкие приложения, особенно если в проекте требуется работа с геоданными.
3. PostgreSQL является проектом с открытым исходным кодом. Любой может просмотреть и изменить его код, что делает PostgreSQL очень гибким и адаптивным к нуждам проекта. Это также означает, что PostgreSQL может быть бесплатно использован в проектах любого размера и бюджета.
4. PostgreSQL имеет отличную поддержку для языка программирования Java, включая JDBC-драйвер и множество других инструментов и библиотек. Это делает PostgreSQL очень удобным для использования в проектах, где используется Java.
5. PostgreSQL является бесплатным и доступным для использования, что делает его привлекательным выбором для многих проектов, особенно для небольших и средних компаний, которые не могут позволить себе дорогостоящие решения баз данных.

Выбор РСУБД завершен, в данном случае требуется работа с геоданными. PostgreSQL является оптимальным выбором. Он обеспечивает высокую надежность и стабильность, поддерживает множество функций, имеет

открытый исходный код, отличную поддержку для языка программирования Java, и является бесплатным и доступным для использования.

Git - это система контроля версий, которая используется для эффективного управления изменениями в коде проекта. Он является одним из наиболее популярных инструментов для работы с кодом и широко используется в различных проектах.

Git позволяет эффективно работать с ветвлением кода, что означает, что вы можете создавать отдельные ветки для различных функций или исправлений ошибок, не затрагивая основной код проекта. Это позволяет избежать конфликтов при работе нескольких человек над одним проектом.

Git является открытым и бесплатным инструментом, который имеет большое сообщество пользователей и множество ресурсов для обучения и поддержки. Это делает Git отличным выбором для любого проекта, включая проекты на Java.

Git также позволяет пользователям легко отслеживать изменения в коде, включая историю изменений, комментарии и авторов изменений. Кроме того, Git имеет множество функций, которые делают его идеальным выбором для разработчиков Java. Например,

Git позволяет пользователям создавать отдельные ветки для каждой функции, что делает процесс разработки более эффективным и удобным. Git также позволяет пользователям работать с удаленными репозиториями, что делает возможным совместную работу над проектами с другими разработчиками. В целом, Git - это мощный и эффективный инструмент для управления изменениями в коде проекта. Он является открытым и бесплатным, что делает его идеальным выбором для любого проекта, включая проекты на Java.

Таблица 3.1 — Выбранные инструменты для реализации проекта

№	Наименование инструмента	Версия
1	Java	OpenJDK 18
2	PostgreSQL	PostgreSQL 15.3
3	Git	Git 2.40.1
4	IntelliJ IDEA	IntelliJ IDEA 2023.1
5	Maven	Apache-maven-3.9.2
6	Spring Boot	Spring Boot 3.1.0

3.2. Реализация сущностей

После создания проекта необходимо реализовать сущности смоделированные в главе 2 пункте 2.6 «Модель данных».

Для реализации будем использовать модуль Spring Boot — Spring JPA (Java Persistence API).

Spring JPA - это модуль, который предоставляет абстракцию для работы с базами данных, основанный на JDBC и обеспечивающий более простой и удобный интерфейс для работы с базами данных. Spring JPA — ORM, что означает, что он предоставляет объектно-реляционное отображение, что упрощает работу с базами данных через объекты Java вместо SQL-запросов.

Spring JPA позволяет писать меньше кода, чем при использовании JDBC, так как JPA предоставляет абстракцию над JDBC, что позволяет скрыть детали реализации. Это уменьшает количество кода и делает работу с базами данных проще и понятнее.

Spring JPA использует кэширование, чтобы снизить количество запросов к базе данных и улучшить производительность. Это позволяет ускорить работу приложений, особенно тех, которые работают с большим объемом данных. Программа Spring JPA позволяет работать с различными базами данных, такими как MySQL, PostgreSQL и Oracle, что делает ее универсальной и удобной для использования в различных проектах.

В целом, Spring JPA - это мощный и удобный инструмент для работы с базами данных, который упрощает разработку приложений и повышает их производительность. Он позволяет сократить количество кода, улучшить производительность, работать с различными базами данных и упростить тестирование приложений.

Ниже приведен пример сущности JPA:

```
@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
@Builder
public class StudyGroup {

    @Id
    @SequenceGenerator(name = "gr_gen",
            sequenceName = "group_generator",
            initialValue = 4, allocationSize = 20
            )
    @GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "gr_gen")
    private long id;
    private String name;

    @ManyToMany(mappedBy = "studyGroups", cascade = CascadeType.ALL)
    private List<ScheduleRecord> scheduleRecords = new ArrayList<>();

    @OneToMany(mappedBy = "group")
    private List<User> users;

    @Override
    public String toString() {
        return "StudyGroup{" +
                "name='" + name + '\'' +
                '}';
    }
}
```

Рисунок 3.1 Сущность учебная группа, Java код

Как видно на рисунке обычный Java класс, помеченный аннотацией «@Entity», что сообщает Spring JPA о том, что этот класс является таблицей. У класса есть поле id и поле name, при сохранении в базу данных, эти поля являются столбцами в таблице.

Так же на рисунке представлены аннотации «@OneToMany» и «@ManyToMany», которые являются связями с сущностями User и ScheduleRecords.

«@OneToMany» — это связь один-ко-многим, Spring JPA в данном случае создаст внешний ключ в таблице.

«@ManyToMany» — это связь один-ко-многим, Spring JPA в данном случае создаст третью таблицу для связывания с сущностями.

Таким образом были реализованы все сущности.

3.3. Реализация компонентов системы

Для того чтобы написать приложение на Spring необходимо пользоваться стандартным паттерном MVC.

Паттерн содержит 3 понятия:

- Model — Модель данных, сущности созданные в главе 3.2
- View — Представление. Сущности, отвечающие за отображение пользователю.
- Controller — Контроллер, отвечает за взаимодействие с пользователем.

Всего предполагается 2 интерфейса взаимодействия с пользователем:

1. REST API — необходим для работы мобильных и веб-приложений приложений.
2. Браузер — панель администратора.

Для реализации компонентов в Spring Framework существуют соответствующие аннотации:

«@Service» — сервис, реализует бизнес-логику, зависит от репозитория, в зависимости у контроллера.

«@RestController» и «@Controller» — соответственно контроллеры, использующие бизнес-логику сервиса, непосредственно взаимодействуют с пользователем.

«@Repository» — репозиторий, используется для управления сущностями в БД. В зависимости у сервиса.

Далее необходимо реализовать «@Service», «@RestController», «@Controller», «@Repository».

Приступим к реализации «@RestController» и «@Controller»:

```
@RestController
@RequestMapping("/api/groups")
public class GroupController {

    @Autowired
    private StudentsAndGroupService studentsService;

    @GetMapping(produces = MediaType.APPLICATION_JSON_VALUE)
    public ResponseEntity<?> getGroups(){
        return new ResponseEntity<Collection<StudyGroup>>(studentsService.getGroups(), HttpStatus.OK);
    }

    @PostMapping(produces = MediaType.APPLICATION_JSON_VALUE)
    public ResponseEntity<?> addGroup(@RequestParam String group) {
        try {
            studentsService.createGroup(group);
            return new ResponseEntity(HttpStatus.CREATED);
        } catch (Exception e) {
            return ResponseEntity.badRequest().body("Something went wrong");
        }
    }

    @PutMapping(produces = MediaType.APPLICATION_JSON_VALUE)
    public ResponseEntity<?> addStudentAtGroup(@RequestBody GroupStudentDto studentToGroup){
        try {
            Boolean isMapped = studentsService.mapStudentAndGroup(studentToGroup);
            return new ResponseEntity(HttpStatus.OK);
        } catch (EntityNotFoundException e) {
            return ResponseEntity.badRequest().body(e.getMessage());
        }
    }
}
```

Рисунок 3.2 RestController представляющий учебную группу, Java код

На рисунке 3.2 показан код RestController, этот контроллер осуществляет взаимодействие с пользовательским HTTP запросом, обрабатывает его и передает данные на слой бизнес логики.

Реализованы:

- Get-запросы, которые отправляют данные по учебным группам
- Post-запросы, принимающие и создающие данные об учебных группах
- Put-запросы, которые позволяют редактировать данные об учебной группе

Таким образом, были построены все @RestController в проекте.

Класс StudentsAndGroupService является сервисом. На рисунке 3.3.1 в Приложении В изображен код этого класса сервиса, реализующий бизнес-логику учебной группы.

Класс сервис имеет следующие поля:

- UserRepository
- GroupRepository

Эти поля являются репозиториями, они предназначены для работы с сущностями User и Group при подходе ORM. Spring JPA самостоятельно создает таблицы для базы данных, и все сущности сохраняются в БД при помощи метода save(T entity).

Так же есть возможность создавать свои собственные методы для работы с репозиторием, используя SQL (аннотация «@Query») или при помощи имен производных методов. Имена производных методов состоят из двух основных частей, разделенных первым ключевым словом By. Например, findByName найдет сущность в базе данных по имени.

В данном случае реализованы методы:

- Получить всех пользователей
- Получить все учебные группы
- Создать учебную группу
- Создать студента
- Удалить студента
- Удалить группу
- Редактировать студента
- Редактировать группу

По аналогии были разработаны остальные сервисы и репозитории в проекте. Ниже приведена диаграмма разработанных компонентов.

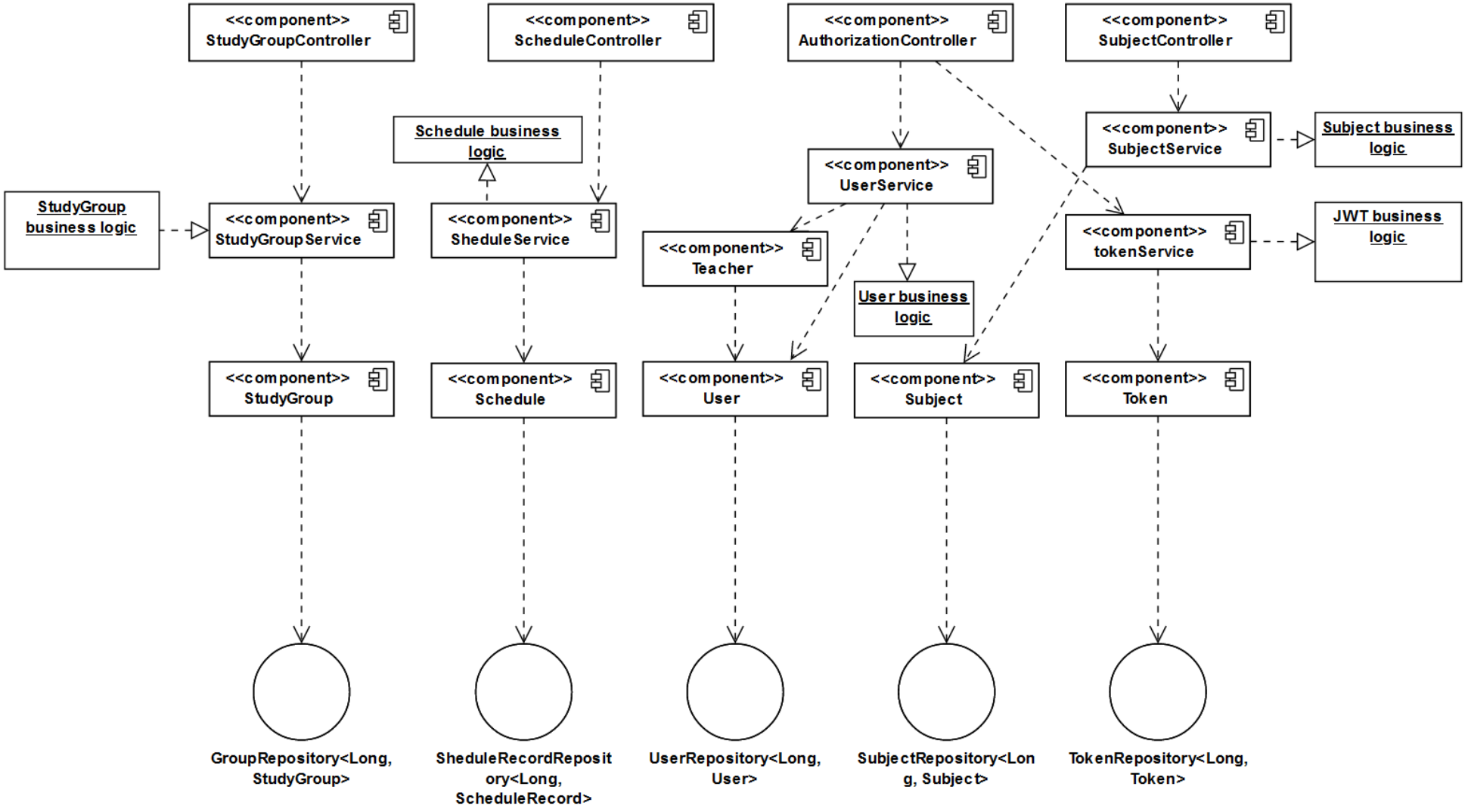

Рисунок 3.3 Диаграмма компонентов в реализованной системе. Нотация UML.

3.4. Развертывание системы

В качестве платформы для развертывания выступает платформа timeweb cloud. Эта компания предоставляет PaaS услуги. В частности выбор пал на VDS.

VDS - это виртуальный сервер, который позволяет запускать проекты с требуемым уровнем производительности. Как и обычный сервер, к нему можно подключиться по протоколу SSH и произвести все пусконаладочные работы.

В ходе реализации ВКР, после ручного развертывания системы, был написан скрипт для автоматического развертывания на сервере:

```
#!/usr/bin/env bash

mvn clean package

echo 'Send files to server'

scp -i ~/.server/vkr_key \
        target/schedule-0.0.1-SNAPSHOT.jar \
        shabashov@188.225.56.133:/shabashov/

echo 'Restart server'

ssh -i ~/.server/vkr_key shabashov@188.225.56.133 << EOF

pgrep java | xargs kill -9
nohup java -jar schedule-0.0.1-SNAPSHOT.jar > log.txt &

EOF

echo 'deploy ended!'
```

Рисунок 3.4 Скрипт для развертывания на виртуальном сервере

Необходимо описать все шаги скрипта.

Описание скрипта:

1. mvn clean package — собирает проект локально.
2. scp — копирует jar файл на сервер.
3. ssh — команда выполняет подключение к серверу
4. pgrep java | xargs kill -9 — останавливает процесс выполнения приложения
5. nohup java –jar schedule-0.0.1-SNAPSHOT.jar > log.txt — выполняет запуск приложения и перенаправляет стандартный вывод в файл.

Скрипт написан на bash и имеет некоторые преимущества перед ручным развертыванием:

- Нет необходимости вводить пароль, так как авторизация на сервере пользователя выполняется с помощью ранее сохраненных ssh ключей.
- Скрипт повышает надежность развертывания, так как все действия стабильно повторяемы и предсказуемы.
- Скрипт повышает продуктивность работы и автоматизирует рутинные действия разработчика.

ЗАКЛЮЧЕНИЕ

В ходе работы были выполнены следующие задачи:

1. Определена предметная область
2. Проведен анализ существующих систем и их аналогов
3. Проведен SWOT и VCM анализ
4. Определена целевая аудитория проекта
5. Определены функциональные, нефункциональные и системные требования
6. Произведен экономический расчет и сведена смета затрат
7. Информационная система спроектирована с помощью UML, IDEF0, IDEF1X, DFD диаграмм
8. Информационная система разработана и запущена на виртуальном сервере

Данный список выполненных задач говорит о достижении поставленной изначально цели проектирование и разработка автоматизированной системы расписания ВУЗа с ГИС возможностями.

Определение предметной области и проведение анализа существующих систем и аналогов позволило лучше понять требования и потребности пользователей.

SWOT и VCM анализы помогли определить сильные и слабые стороны проекта, а также возможности и угрозы связанные с его реализацией.

Определение целевой аудитории и функциональных, нефункциональных и системных требований стало важным шагом в разработке информационной системы. Это позволило определить, какие функции и возможности должны быть включены в систему, чтобы она наилучшим образом соответствовала потребностям пользователей.

Экономический расчет и сведение сметы затрат помогли определить, какие ресурсы будут необходимы для реализации проекта. Это позволило оценить финансовые риски и принять обоснованные решения о том, какие ресурсы необходимо выделить для проекта.

Проектирование информационной системы с помощью IDEF0, IDEF1X, DFD диаграмм было важным шагом в создании понятной и эффективной

системы. Это позволило визуализировать структуру системы и понять, как она будет функционировать.

Разработка и запуск информационной системы на виртуальном сервере были последними шагами в проекте. Создана функциональная и эффективная система, которая поможет улучшить организацию учебного процесса в ВУЗе.

Проект по проектированию и разработке автоматизированной системы расписания для вузов с ГИС возможностями был успешно выполнен.

СПИСОК ИСПОЛЬЗОВАНЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. Смирнов М. UML - быстрый старт / М. Смирнов. – 2-е издание, оформленное. – Москва : Наука, 2016. – 47 с.
2. Рамбо Дж. UML 2.0. Объектно-ориентированное моделирование и разработка, 2-е изд. / Дж. Рамбо, М. Блаха. – СПб : Питер, 2021. – 544 с. – ISBN 978-5-4461-9428-5
3. Волошин В.А. Инфологическая модель данных: пример построения er-диаграммы / В.А. Волошин, В.Д Шляхов, С.О. Барышевский. – Санкт-Петербург : Лань, 2020. – 20 с.
4. DFD (Data Flow Diagram) Диаграммы — зачем они нужны и какие бывают // habr : сайт. – URL: https://habr.com/ru/post/668684/ (дата обращения: 14.01.2023)
5. ГОСТ 34.321-96 Информационные технологии (ИТ). Система стандартов по базам данных. Эталонная модель управления данными : дата введения 2001-07-01. – Москва : Издательство стандартов, 2001. – 24 с.
6. Шёнбергер С., Кемпфер А. Реинжиниринг бизнес-процессов: точки принятия решений по автоматизации в процессе реинжиниринга. — Berlin; Heidelberg: Springer-Verlag, 1996. — 238 с.
7. Шилдт Герберт Java полное руководство / Герберт Шилдт. – М : ООО "И.Д. Вильяме",, 2015. – 1376 с.
8. Эккель Б. Философия Java / Б. Эккель. – 4-е изд. – СПб. : Питер, 2018. – 1159 с.
9. Head First. Паттерны проектирования. Обновленное юбилейное издание / Э. Фримен, Э. Робсон, К. Сьерра, Б. Бейтс. – Санкт Петербург : "Питер", 2021. – 656 с.
10. Седжвик Р. Computer Science. Основы программирования на Java, ООП, алгоритмы и структуры данных / Р. Седжвик, К. Уэйн. – Санкт-Петербург : Питер, 2018. – 1072 с.

11. Лафоре Р. Структуры данных и алгоритмы Java / Р. Лафоре. – Санкт-Петербург : Питер, 2011. – 704 с.
12. Spring Framework Documentation // Spring.IO : сайт. – URL: https://docs.spring.io/spring-framework/docs/current/reference/html/ (дата обращения: 26.02.2023)
13. Web MVC framework // Spring : сайт. – URL: https://docs.spring.io/spring-framework/docs/3.2.x/spring-framework-reference/html/mvc.html (дата обращения: 01.04.2023)
14. Spring Framework Documentation : сайт. – URL: https://docs.spring.io/spring-framework/reference/ (дата обращения: 14.05.2023)
15. Spring Data JPA - Reference Documentation : сайт. – URL: https://docs.spring.io/spring-data/jpa/docs/current/reference/html/ (дата обращения: 30.03.2023)
16. Моргунов Е. П. PostgreSQL. Основы языка SQL: учебное пособие / Е. П. Моргунов. – СПб. : БХВ-Петербург, 2018. – 336 с. – ISBN 978-5-9775-4022-3
17. Бьюли А. Изучаем SQL / А. Бьюли. – Москва : Символ-Плюс, 2019. – 308 с. – ISBN 978-5-932860-5-19
18. Уорд Б Внутреннее устройство Linux / Б Уорд. – Питер : СПб, 2016. – 384 с.
19. Карлинг М. Системное администрирование Linux / М. Карлинг, С. Деглер, Д. Деннис. – Санкт-Петербург : Питер, 2004. – 592 с.
20. Стахов А.А. Сетевое администрирование Linux / А.А. Стахов. – Санкт-Петербург : БХВ-Петербург, 2004. – 480 с.

ПРИЛОЖЕНИЕ А

Таблица 1.0 — Функциональные требования

Приложение клиент (Студент)		
№	**Наименование требования**	**Приоритет**
1	Лист расписания занятий для студента на неделю	Высокий
2	Расписание занятий студента на текущий день	Высокий
3	Пометка о дистанционном занятии в расписании учебной группы	Высокий
4	Личный кабинет студента	Средний
5	Добавление контактов студента	Средний
6	Удаление контактов студента	Средний
7	Редактирование контактов студента	Средний
8	Самостоятельная регистрация студента	Высокий
9	Авторизация студента	Высокий
10	Настройка периодичности уведомлений о занятиях	Средний
11	Настройка периодичности уведомлений события ВУЗа	Средний
12	Просмотр публичных карточек студентов	Средний
13	Информация о состоянии студента	Низкий

14	Возможность просмотра листа расписания на неделю без авторизации	Высокий
Приложение клиент (Преподаватель)		
№	**Наименование требования**	**Приоритет**
15	Расписание занятий для преподавателя на день	Высокий
16	Ведение журнала посещений	Высокий
17	Расписание занятий для преподавателя на неделю	Высокий
18	Личный кабинет преподавателя	Высокий
19	Авторизация преподавателя	Высокий
20	Редактирование журнала посещений	Высокий
21	Создание журнала посещений	Высокий
22	Автоматическое формирование журнала со списком студентов по изучаемой дисциплине	Высокий
23	Просмотр публичных карточек студентов	Низкий
24	Публичная карточка преподавателя	Средний
25	Редактирование информации о преподавателе	Средний
26	Добавление контактов преподавателя	Средний
27	Удаление контактов преподавателя	Средний

Панель администратора		
№	**Наименование требования**	**Приоритет**
28	Пометка о дистанционном занятии в расписании преподавателя	Высокий
29	Место проведения дистанционного занятия	Высокий
30	Конфигурирование чередования учебных недель	Высокий
31	Добавление занятий	Высокий
32	Удаление занятий	Высокий
33	Изменение занятий	Высокий
34	Добавление событий ВУЗа	Высокий
35	Добавление списка аудиторий	Высокий
36	Изменение списка аудиторий	Высокий
37	Удаление списка аудиторий	Высокий
38	Добавление аудитории в список	Высокий
39	Удаление аудитории из списка	Высокий
40	Изменение аудитории в списке	Высокий
41	Информация об аудитории (Расположение, корпус, адрес корпуса)	Средний
42	Добавление учебной группы	Высокий
43	Удаление учебной группы	Высокий
44	Изменение учебной группы	Высокий
45	Добавление студентов в учебную группу	Высокий
46	Удаление студентов из учебной группы	Высокий
47	Добавление студентов	Высокий

48	Редактирование студентов	Высокий
49	Удаление студентов	Высокий
50	Добавление старосты группы	Низкий
51	Добавление куратора для учебной группы	Низкий
52	Добавление преподавателей	Высокий
53	Удаление преподавателей	Высокий
54	Редактирование контактов преподавателя	Высокий
55	Уведомления о начале занятия	Высокий
56	Уведомления о событиях ВУЗа	Высокий
57	Добавление событий ВУЗа	Высокий
58	Удаление событий ВУЗа	Высокий
59	Настройка уведомлений о событии ВУЗа	Высокий
60	Добавление корпуса ВУЗа	Высокий
61	Изменение корпуса ВУЗа	Высокий
62	Удаление корпуса ВУЗа	Высокий
63	Формирование расписания	Высокий
64	Удаление записи в расписании	Высокий
65	Изменение записи в расписании	Высокий
66	Замечания о состоянии аудитории	Низкий
67	Интеграция с сервисами 1С	Высокий
68	Загрузка информации о студентах из 1С	Высокий
69	Загрузка информации о преподавателях из 1С	Высокий

Далее необходимо рассмотреть выдвигаемые нефункциональные требования.

Таблица 0.2 — нефункциональные системные требования.

Системные требования к серверу		
№	**Наименование**	**Описание**
1	Минимальные системные требования к серверу	1. ОЗУ: от 4 ГБ. 2. CPU: от 4 вычислительных ядер 3. OS: Linux 4. ROM: от 1 ГБ свободного пространства 5. Исходящее сетевое соединение от 10 МБИТ/С
2	Реляционная БД	PostgreSQL 6.15 и выше
3	Компиляторы, среды выполнения, платформы	java 19.0.1 2022-10-18 Java(TM) SE Runtime Environment (build 19.0.1+10-21) Java HotSpot(TM) 64-Bit Server VM (build 19.0.1+10-21, mixed mode, sharing)
Системные требования к веб-клиенту		
№	**Наименование**	**Описание**
1	Минимальные системные требования к клиенту	1. ОЗУ: от 2 ГБ. 2. CPU: от 2 вычислительных ядер 3. OS: Windows/Linux/macOS 4. Исходящее/Входящее соединение от 1024 КБИТ/С
2	Браузеры	Google Chrome, Mozilla Firefox,

		Microsoft Edge, Yandex Browser
3	Компиляторы, среды выполнения, платформы	java 19.0.1 2022-10-18 Java(TM) SE Runtime Environment (build 19.0.1+10-21) Java HotSpot(TM) 64-Bit Server VM (build 19.0.1+10-21, mixed mode, sharing)
Минимальные системные требования к мобильным устройствам		
№	**Наименование**	**Описание**
1	Операционная система	Минимальная версия ОС Android 9.0
2	ОЗУ	2 ГБ
3	Хранилище	16 ГБ
4	Частота процессора	1,2 ГГц
5	Архитектура процессора	32-разрядная
6	Защита от пыли и влаги	IP54
7	Ударопрочность	MIL-STD-810G или IEC 62-2-32
8	Системные приложения по умолчанию	• Камера • Контакты • Загрузки • Google • Сообщения • Телефон • Google Play • Настройки

ПРИЛОЖЕНИЕ Б

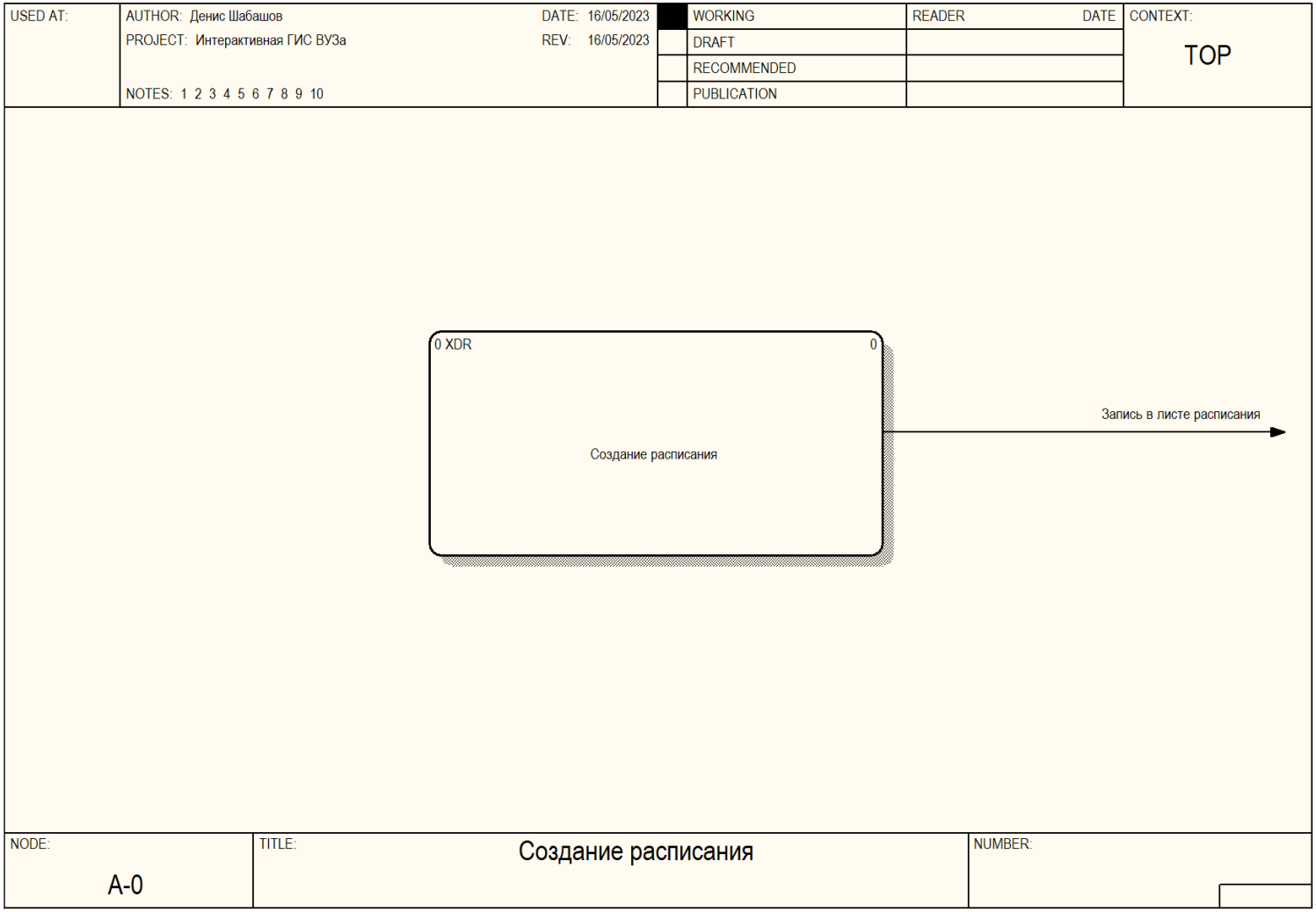

Рисунок 2.14 Процесс создания расписания, нотация DFD, контекстная диаграмма

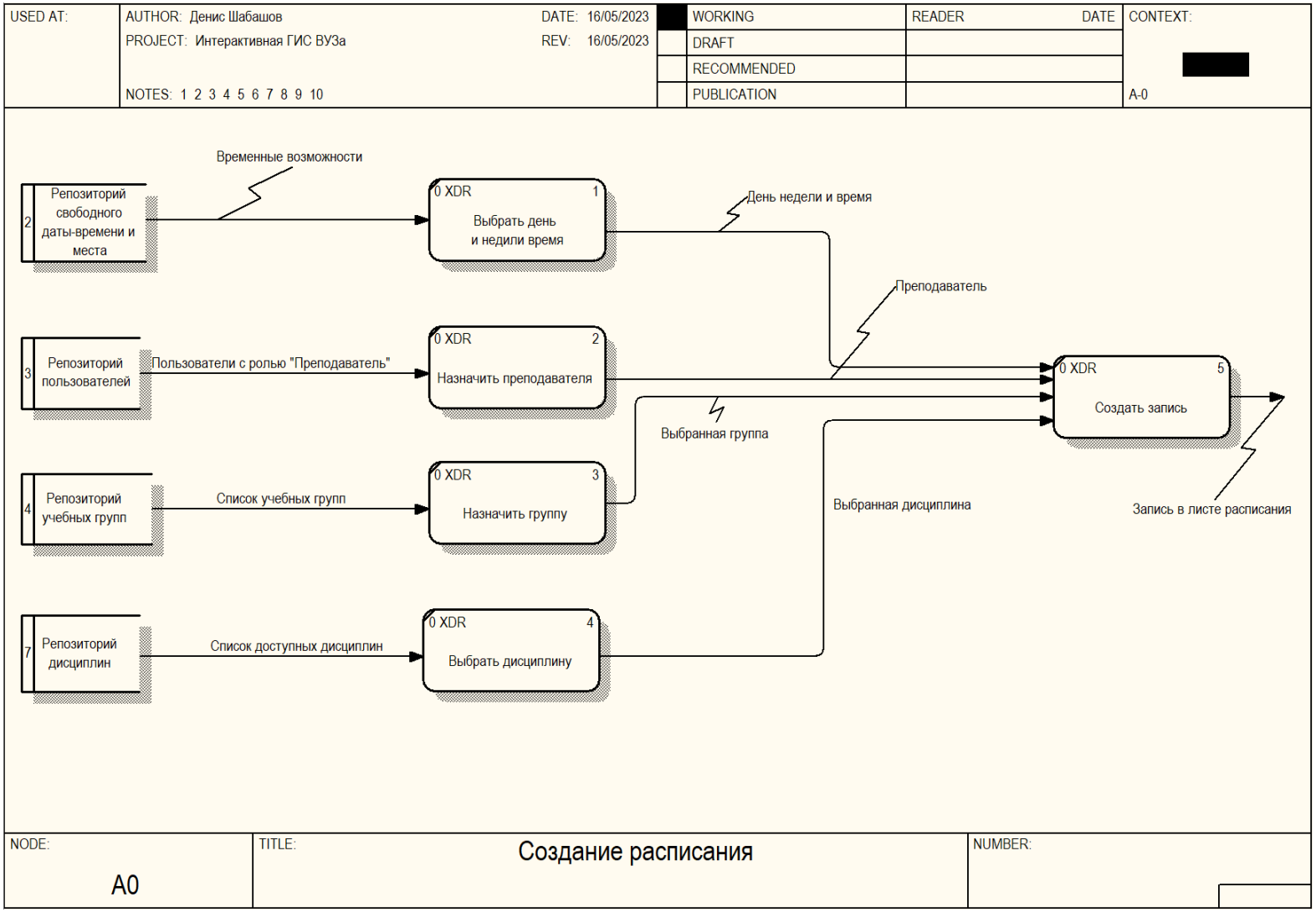

Рисунок 2.15 Процесс создания расписания, нотация DFD, декомпозиция

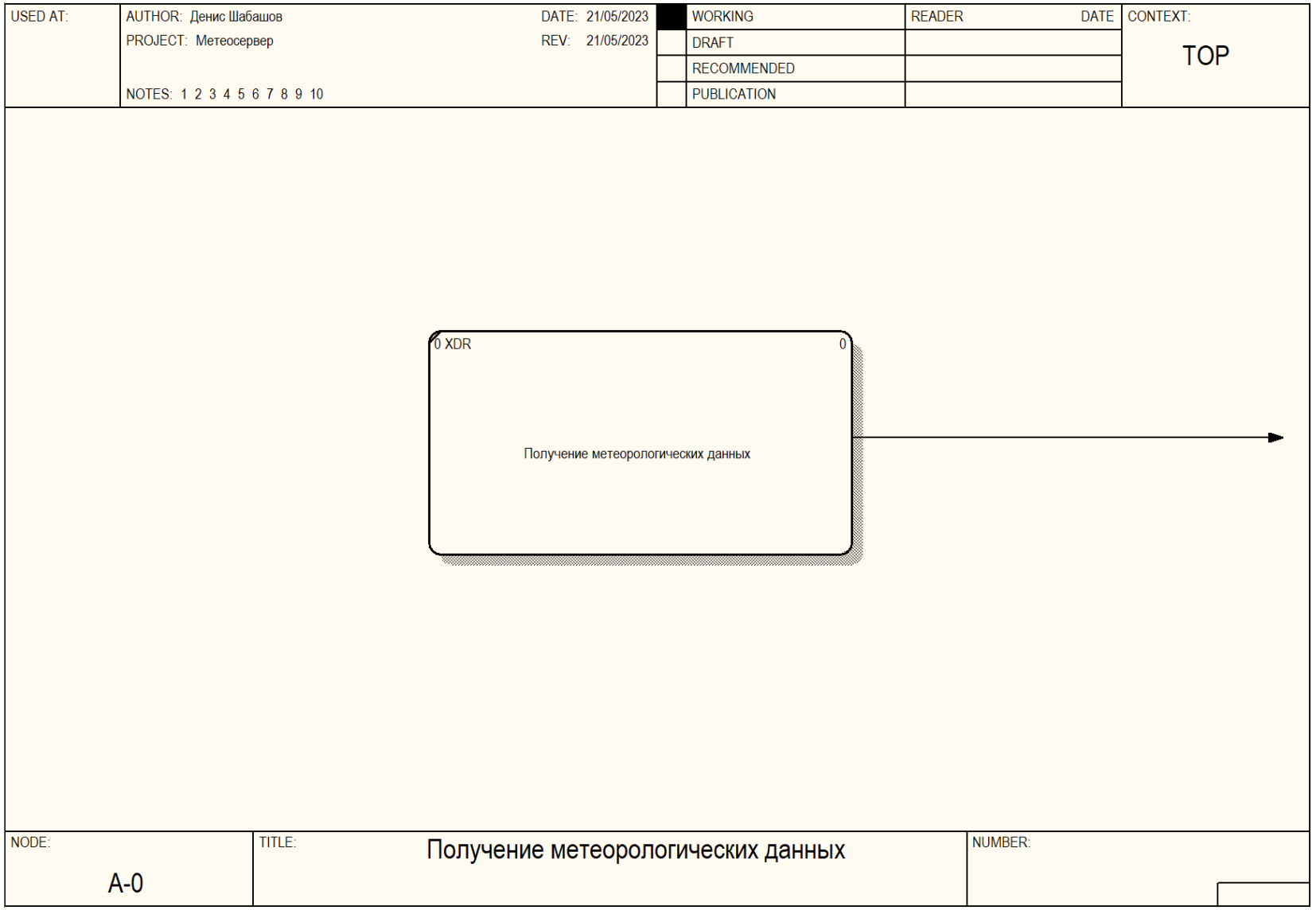

Рисунок 2.16 Получение метеорологических данных, нотация DFD, контекстная диаграмма

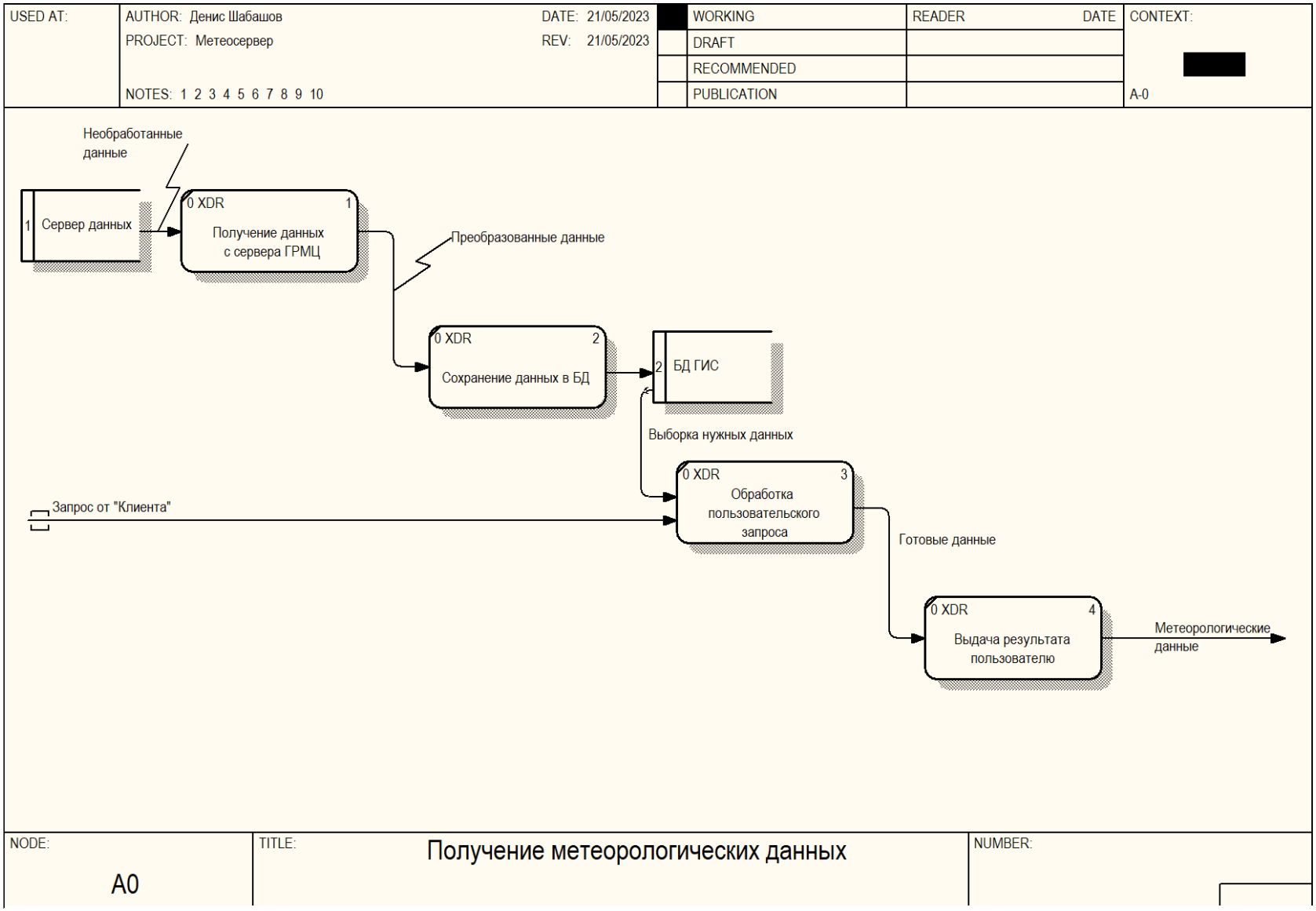

Рисунок 2.17 Получение метеорологических данных, нотация DFD, декомпозиция

ПРИЛОЖЕНИЕ В

```
@Service
@RequiredArgsConstructor
public class StudentsAndGroupService {
    private final UserDetailsService userDetailsService;
    private final UserRepository userRepository;
    private final GroupRepository groupRepository;

    public boolean createStudent(String firstName, String secondName, String surname){
        User user = new User(firstName, secondName, surname);
        userRepository.save(user);
        return true;
    }

    public Collection<User> getStudents(){
        return userRepository.findAll();
    }

    public Page<StudyGroup> getGroups(Pageable pageable){
        return groupRepository.findAll(PageRequest.of(pageable.getPageNumber(),
pageable.getPageSize()));
    }

    public List<StudyGroup> getGroups(){
        return groupRepository.findAll();
    }

    public boolean createGroup(String groupName){
        groupRepository.save(StudyGroup.builder().name(groupName).build());
        return true;
    }

    public boolean mapStudentAndGroup(GroupStudentDto studentDto) throws EntityNotFoundException {
        User existingUser = this.userRepository.findUser(
                studentDto.getFirstname(),
                studentDto.getLastname(),
                studentDto.getLogin(),
                Role.STUDENT
        );
        var group = groupRepository.findByName(studentDto.getGroup());
        if (group.isEmpty()){
            throw new StudyGroupNotFoundException("");
        }
        existingUser.setGroup(group.get());
        userRepository.save(existingUser);
        return true;
    }

    public boolean deleteStudent(User user) throws EntityNotFoundException  {
        User foundUser = this.findStudent(user.getFirstname(), user.getLastname(),
user.getSurname());
        userRepository.delete(foundUser);
        return true;
    }

    public User findStudent(String firstname, String lastname, String surname) throws
StudentNotFoundException {
        User user = userRepository.findUser(firstname, lastname, surname, Role.STUDENT);
        if (user == null) {
            throw new StudentNotFoundException("");
        }
        return userRepository.findUser(firstname, lastname, surname, Role.STUDENT);
    }

    public StudyGroup findGroup(String name) {
        StudyGroup group = groupRepository.findByName(name).orElseThrow(() -> {
            new StudyGroupNotFoundException("");
            return null;
        });
        return group;
    }

    public void deleteGroup(Long id) {
        groupRepository.deleteById(id);
    }
```

Рисунок 3.3.1 Код сервиса, реализующий бизнес-логику учебной группы