1. Общее описание

1.1 Цель работы

Изучить теоретические принципы унифицированного процесса разработки СОИУ и составляющих его этапов. Получить практические навыки применения шаблонов при проектировании и разработке СОИУ. Освоить применение CASE средств для разработки СОИУ.

1.2 Описание задания

Выполнить проектирование СОИУ в соответствии с описанием ее функциональности (определяется вариантом). Для проектирования использовать этапы и модели унифицированного процесса. По результатам проектирования получить работающую программу с паттернами (по варианту). Для построения диаграмм использовать среду StarUML

1.3 Задание варианта АСУ/ИС:

Тема: АИС IT фирмы
паттерн бизнес-логики: service layer
паттерн работы с БД: active record
паттерн GOF: наблюдатель
Система предназначена для автоматизации управления задачами разработчиков в IT фирме

2. Этап анализа и планирования требований

2.1 Спецификация основных проектных требований

2.1.1 Функциональные требования

Система должна:

1. Обеспечить возможность управления учетными записями пользователей
2. Обеспечить возможность идентификации пользователя
3. Обеспечить возможность управления задачами пользователей
4. Обеспечить возможность ввести учет задач пользователей
5. Обеспечить возможность распределения задач между пользователями

2.1.2 Нефункциональные требования

Система должна:

1. Иметь удобный и эргономичный интерфейс клиентской части
2. Предоставлять клиентскую часть через веб-интерфейс
3. Обеспечивать постоянное и стабильное соединение с базой данных
4. Обеспечивать одновременную работу с 100 пользователями
5. Масштабироваться и обеспечивать возможность добавления новых типов задач и статусов
6. Обеспечивать ответ в течении 500мс при стабильном соединении
7. Потреблять до 2Gb ОП

2.2 Модель предметной области

Диаграмма классов предметной области приведена на рисунке 2.1.

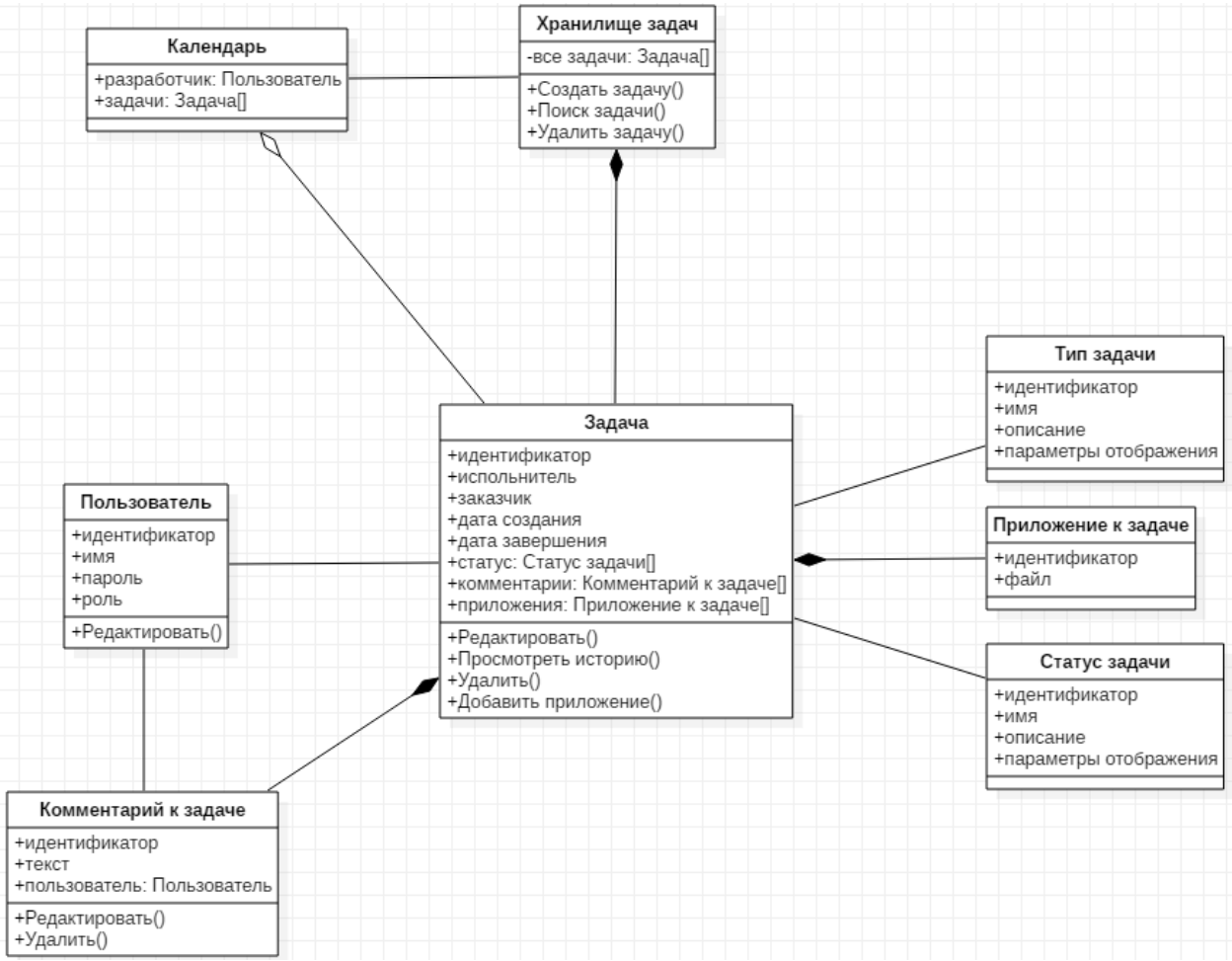

Рис 2.1 – Диаграмма классов предметной области

2.3 Выявленные актеры

В рамках проекты выявлено 2 актера:

1. Пользователь, занимается созданием задач и редактированием задач, к которым он имеет доступ
2. Администратор, может делать все, что и пользователь, только имеет доступ ко всем задачам и имеет право на создание новых типов задач и статусов задач

2.4. Прецеденты

2.5.1 Выявленные прецеденты

Диаграмма прецедентов приведена на рисунке 2.2

По результатам проведенного анализа выявлены следующие основные прецеденты:

1. Создание пользователя
2. Проверка прав доступа пользователя
3. Добавление задачи
4. Редактирование задачи
5. Загрузка приложений к задаче
6. Добавление комментария к задаче
7. Редактирование комментария
8. Добавление новых типов задач
9. Добавление новых статусов задач
10. Построение календаря задач
11. Поиск задач

Прецедент **Создание пользователя** заключается в наполнении БД информацией о новом пользователе (регистрация пользователя)

Прецедент **Проверка прав доступа пользователя** заключается в проверке того имеет ли пользователь право на редактирование данных, которые он пытается изменить

Прецедент **Добавление задачи** заключается в наполнении БД информацией о новой задаче (установка сроков задачи, выбор исполнителя, добавление описания)

Прецедент **Редактирование задачи** заключается в изменении записи задачи в БД (подразумевается, как изменение отдельных полей, так и удаление задачи целиком)

Прецедент **Загрузка приложений к задаче** заключается в загрузке на сервер системы файлов на которые можно будет позже ссылаться в описании задач и комментариях

Прецедент **Добавление комментария к задаче** заключается в наполнении БД данными о комментарии, который будет отображаться вместе с задачей

Прецедент **Редактирование комментария** заключается в редактировании записи комментария в БД (подразумевается, как изменение отдельных полей, так и удаление целиком)

Прецедент **Добавление новых типов задач** заключается наполнение БД информацией о новом типе задач, который будет отображаться при редактировании задачи

Прецедент **Добавление новых статусов задач** заключается наполнение БД информацией о новом статусе задач, который будет отображаться при редактировании задачи

Прецедент **Построение календаря задач** заключается в выборке задач по параметрам и представления их в виде календаря задач, на котором отображается краткая информация о задаче

Прецедент **Поиск задач** заключается в поиске задач по параметрам и представлении их в виде списка

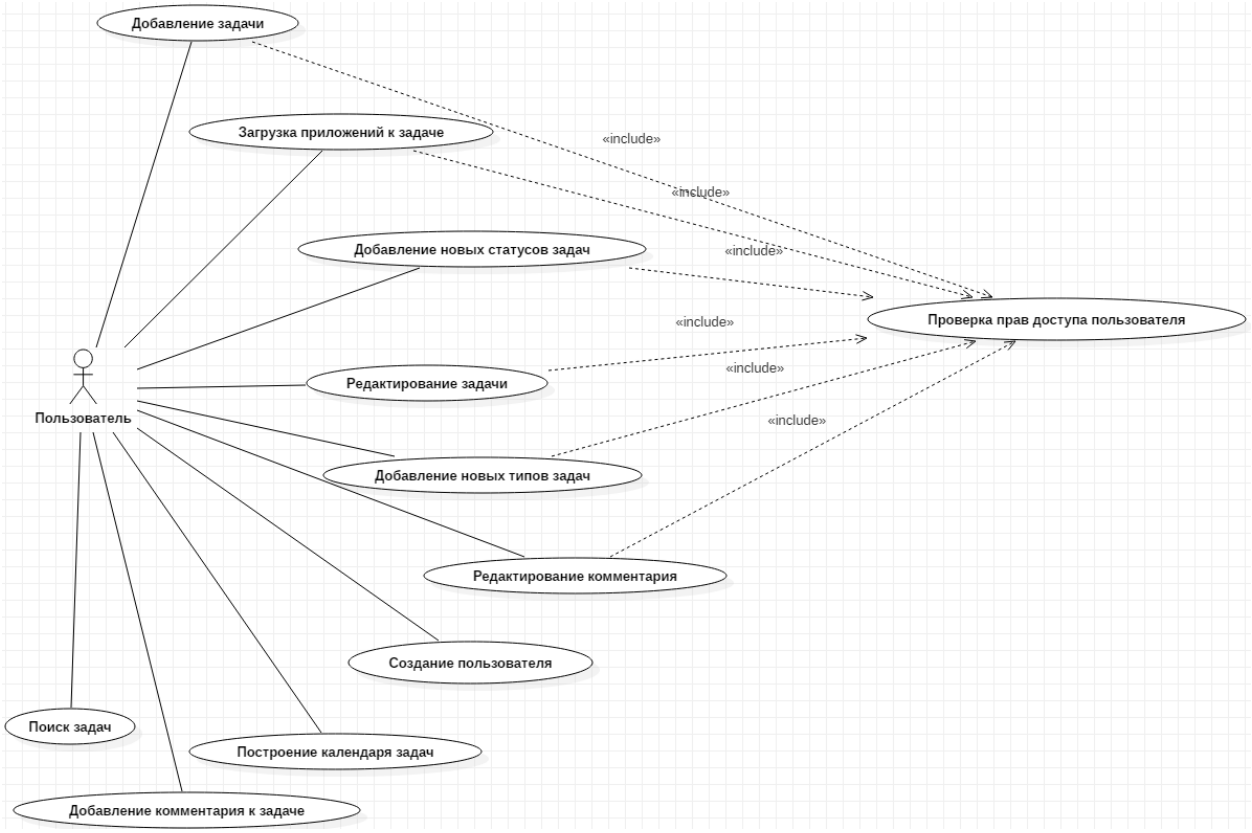

Рис 2.2 – Диаграмма прецедентов

2.5.2 Список приоритетов прецедентов

Приоритеты прецедентов приведены в таблице 2.1

Таблица 2.1 – Приоритеты прецедентов

Прецедент	**Приоритет**
Создание пользователя	Средний
Проверка прав доступа пользователя	Высокий
Добавление задачи	Высокий
Редактирование задачи	Средний
Загрузка приложений к задаче	Низкий
Добавление комментария к задаче	Средний
Редактирование комментария	Средний
Добавление новых типов задач	Низкий
Добавление новых статусов задач	Низкий
Построение календаря задач	Низкий
Поиск задач	Высокий

2.6 Перечень рисков

На этапе Начало была проведена инициация рисков с разделением на проектные, технические и коммерческие риски

2.6.1 Проектные риски

1. **Риски управления требованиями.** На этапах Конструирование и Переход проектная команда может понять, что изначальные требования не соответствовали реальности. Для минимизации риска следует детально провести интервьюирование заказчика, наиболее полно составить ТЗ и согласовать его с заказчиком.
2. **Риски, связанные с персоналом**. Участники проектной команды могут уйти на больничный, быть уволены или показать низкую производительность, что увеличит сроки разработки проекта. Для минимизации риска следует предусмотреть возможность растягивания графика разработки.

2.6.2 Технические риски

1. **Риски недостаточной оценки сложности**. На этапах Начало и Конструирование могут возникнуть проблемы проектирования и реализации каких-либо модулей или компонентов, из-за чего следует заложить в бюджет проекта средства на найм дополнительного персонала или экспертов–консультантов.
2. **Риски неверной реализации.** На этапах Конструирование и Переход может выясниться, что какие-либо части системы реализованы неправильно, с ошибками в логике работы или с полной неработоспособностью. Для минимизации рисков следует проводить постоянное тестирование на соответствие требованиям, интеграционное тестирование и нагрузочное тестирование системы
3. **Риски несовместимости технологий.** На этапах Конструирование и Переход может выясниться, что какие-либо технологии, используемые системой могут оказаться несовместимы, как друг с другом, так и с используемой платформой. Для минимизации рисков следует провести анализ технологий, которые будут использоваться в проекте

2.6.3 Коммерческие риски

1. **Риски потери финансирования**. В течение разработки проекта может возникнуть недостаток денежных средства для продолжения работ. Для минимизации риска следует выделить отдельную статью расходов в бюджете одного или нескольких ключевых партнёров на срок реализации проекта, которая позволит покрыть превышение стоимости разработки в случае его возникновения

2.7 Описание возможной архитектуры

При составлении архитектуры системы был проведён анализ основных проектных требований, модели предметной области и выявленных прецедентов. Результат приведён на рисунке 2.3.

Принято решение использовать шаблон проектирования MVC, поскольку в нём работа системы организована таким образом, чтобы модификация одного из компонентов оказывала минимальное воздействие на остальные

ля реализации системы выбран язык C# (.net core 2.1), поскольку он имеет технологию asp .net core mvc, предназначенную для реализации веб-приложений по шаблону mvc.

В связи с требованиями языка C# (.net core 2.1) серверная ОС может быть принадлежать как семействам Windows ОС, так и ОС основанных на ядре Linux.

Принято решение использовать ОС Debian 9.1, поскольку она является свободно распространяемой и довольно проста в первоначальной настройке и администрировании.

Для развертывания сервера приложения будет использоваться Docker для минимизации риска несовместимости технологий и ОС сервера, а также позволит упростить повторную развертку на другом сервере.

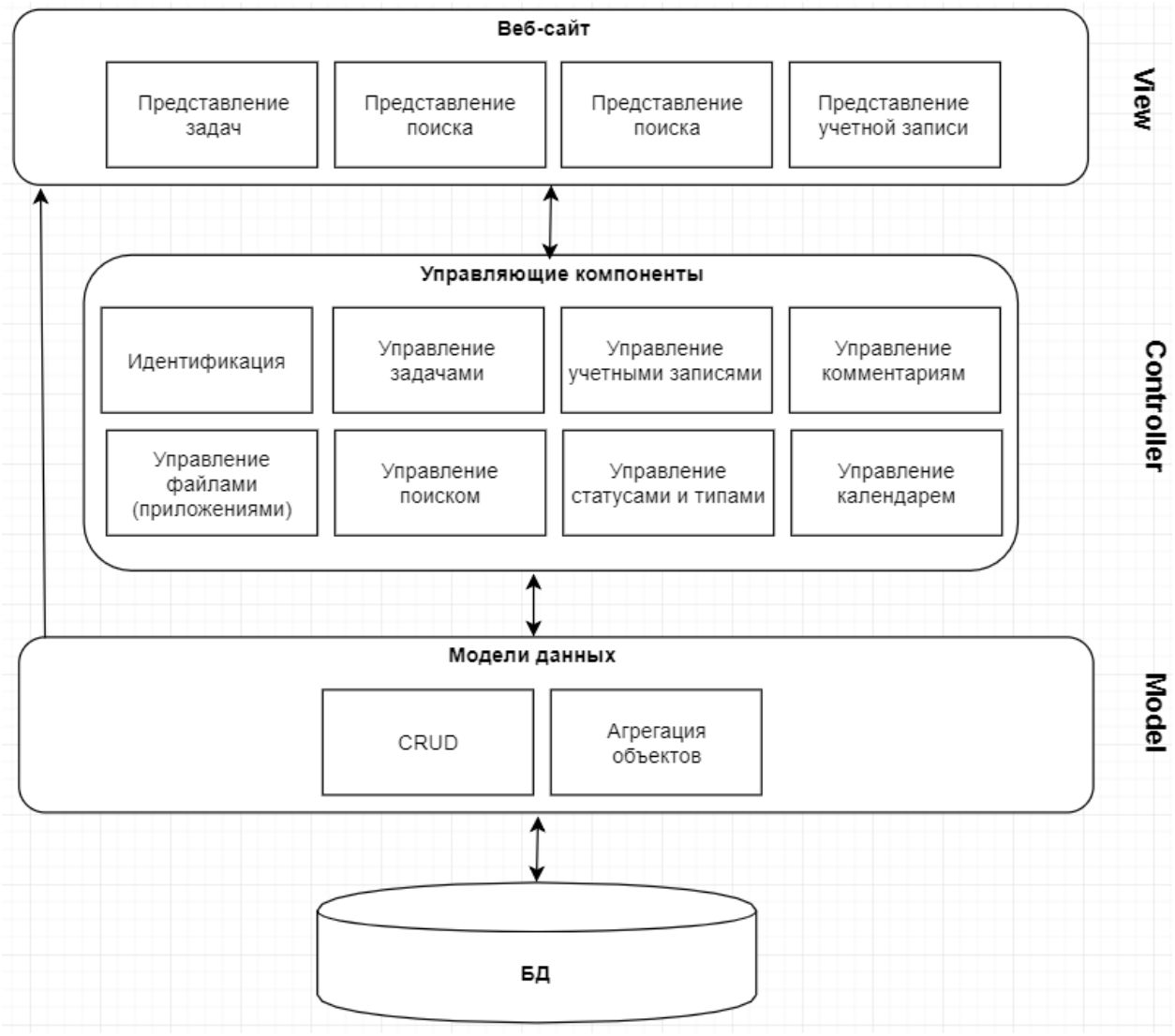

Рис 2.3 – Базовая архитектура системы

Для обеспечения работы сервера с HTTP– и HTTPS–запросами используются веб-серверы Nginx и Apache.

Благодаря использованию ORM выбранный язык разработки нечувствителен к используемой СУБД. Выбрана СУБД PostgreSQL, поскольку её основными особенностями являются поддержка БД практически неограниченного размера и лёгкая расширяемость.

2.8. Оценка проекта по COCOMO II

Для оценка стоимости, затрат и длительности проекта используется конструктивная модель стоимости этапа композиции приложения на основе объектных указателей. Оценка сложности экранов приведена в таблице 2.2. Оценка сложности отчётов приведена в таблице 2.3.

Таблица 2.2 – Оценка сложности экранов

Экран	**Количество представлений**	**Сложность**
Экран входа	1	Простая
Главный экран	2	Средняя
Экран задачи	4	Средняя
Экран календаря	3	Средняя
Экран поиска	2	Простая
Экран статусов	1	Простая
Экран типов	1	Простая
Экран профиля	2	Простая

Таблица 2.3 – Оценка сложности отчетов

Экран	**Количество представлений**	**Сложность**
Отчет об изменении задачи	3	Средняя
Отчет о статусах задач за временной промежуток	3	Средняя

Язык C# является языком программирования четвёртого поколения, поэтому в расчёте количества объектных указателей появится объект 4GL. Но будет использоваться вес 10, такой же, как и для 3GL компонентов. Оценка количества объектных указателей приведена в таблице 2.4.

Таблица 2.4 – Оценка количества объектных указателей

Тип объекта	**Количество**	**Вес**	**Итого**
Экран простой	5	1	5
Экран средний	3	2	6
Отчет средний	2	5	10
4GL компонент	1	10	10
Итого			31

Для проектной команды разработка подобного рода систем является абсолютно новой задачей, поэтому процент повторного использования кода $\%REUSE = 0$.

Новые объектные указатели рассчитаны в формуле 2.1.

$$NOP = OP \times \frac{100-\%REUSE}{100} = 31 \times \frac{100-0}{100} = 31 \quad (2.1)$$

Примем, что опытность разработчиков и зрелость среды разработки номинальные, тогда $PROD = 13$. Затраты рассчитаны в формуле 2.2.

$$\text{ЗАТРАТЫ} = \frac{NOP}{PROD} = \frac{31}{13} = 2.38 \text{ чел.} - \text{мес.} \quad (2.2)$$

Примем, что среднее значение рабочего коэффициента является номинальным и равно 15000$. Стоимость рассчитана в формуле 2.3

$$\text{СТОИМОСТЬ} = \text{ЗАТРАТЫ} \times \text{РАБ}_{\text{КОЭФ}} = 2.38 \times 15000\$ = 35700\$ \quad (2.3)$$

Параметры вычисления длительности разработки приведены в таблице 2.5.

Таблица 2.5 – Параметры вычисления длительности разработки

Масштабный фактор	**Значение**	**Описание**
Предсказуемость PREC	3	Отчасти непредсказуемый
Гибкость разработки FLEX	3	Редкое расслабление в работе
Разрешение архитектуры- /риска RESL	4	Разрешение риска 40%
Связность группы TEAM	1	Высокая кооперативность
Зрелость процесса PMAT	3	Номинальная зрелость

Показатель степени B рассчитан в формуле 2.4

$$B = 1.01 + 0.01 \times \sum_{i=1}^{5} W_i = 1.01 + 0.14 = 1.15 \quad (2.4)$$

Процент увеличения номинального графика примем за 50
Тогда $SCEDPercentage$ = 150.
Длительность разработки оценена в формуле 2.5.

$$TDEV = \left[3 \times \text{ЗАТРАТЫ}^{(0{,}33+0{,}2 \times [B-1.01])}\right] \times \frac{\text{SCEDPercentage}}{100} = 6.138 \text{ мес} \quad (2.5)$$