Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Институт кибернетики
Направление подготовки 09.04.02 «Информационные системы и технологии»
Кафедра информационных систем и технологии

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема работы
Разработка программного обеспечения системы управления требованиями

УДК 004.8

Студент

Группа	ФИО	Подпись	Дата
8ИМ5А	Ногербек Н.Д.		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент каф. ИСТ	Мальчуков А.Н.	к.т.н		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Попова С. Н.	к.э.н		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Акулов П.А	–		

ДОПУСТИТЬ К ЗАЩИТЕ:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
Доцент каф. ИСТ	Мальчуков А.Н.	к.т.н		

Томск – 2017 г.

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Код результатов	Результат обучения (выпускник должен быть готов)
Общепрофессиональные компетенции	
Р1	Воспринимать и самостоятельно приобретать, развивать и применять математические, естественнонаучные, социально-экономические и профессиональные знания для решения нестандартных задач, в том числе в новой или незнакомой среде и в междисциплинарном контексте.
Р2	Владеть и применять методы и средства получения, хранения, переработки и трансляции информации посредством современных компьютерных технологий, в том числе в глобальных компьютерных сетях.
Р3	Демонстрировать культуру мышления, способность выстраивать логику рассуждений и высказываний, основанных на интерпретации данных, интегрированных из разных областей науки и техники, выносить суждения на основании неполных данных, анализировать профессиональную информацию, выделять в ней главное, структурировать, оформлять и представлять в виде аналитических обзоров с обоснованными выводами и рекомендациями.
Р4	Анализировать и оценивать уровни своих компетенций в сочетании со способностью и готовностью к саморегулированию дальнейшего образования и профессиональной мобильности. Владеть, по крайней мере, одним из иностранных языков на уровне социального и профессионального общения, применять специальную лексику и профессиональную терминологию языка.
Профессиональные компетенции	
Р5	Разрабатывать стратегии и цели проектирования, критерии эффективности и ограничения применимости, новые методы, средства и технологии проектирования геоинформационных систем (ГИС) или промышленного программного обеспечения.
Р6	Планировать и проводить теоретические и экспериментальные исследования в области создания интеллектуальных ГИС и ГИС технологии или промышленного программного обеспечения с использованием методов системной инженерии.
Р7	Осуществлять авторское сопровождение процессов проектирования, внедрения и сопровождения ГИС и ГИС технологий или промышленного программного обеспечения с использованием методов и средств системной инженерии, осуществлять подготовку и обучение персонала.

Р8	Формировать новые конкурентоспособные идеи в области теории и практики ГИС и ГИС технологий или системной инженерии программного обеспечения. Разрабатывать методы решения нестандартных задач и новые методы решения традиционных задач. Организовывать взаимодействие коллективов, принимать управленческие решения, находить компромисс между различными требованиями как при долгосрочном, так и при краткосрочным планировании.
Общекультурные компетенции	
Р9	Использовать на практике умения и навыки в организации исследовательских, проектных работ и профессиональной эксплуатации современного оборудования и приборов, в управлении коллективом.
Р10	Свободно пользоваться русским и иностранным языками как средством делового общения.
Р11	Совершенствовать и развивать свой интеллектуальный и общекультурный уровень. Проявлять инициативу, в том числе в ситуациях риска, брать на себя всю полноту ответственности.
Р12	Демонстрировать способность к самостоятельному обучению новым методам исследования, к изменению научного и научно-производственного профиля своей профессиональной деятельности, способность самостоятельно приобретать с помощью информационных технологий и использовать в практической деятельности новые знания и умения, в том числе в новых областях знаний, непосредственно не связанных со сферой деятельности, способность к педагогической деятельности.

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Институт кибернетики
Направление подготовки 09.04.02 «Информационные системы и технологии»
Кафедра информационных систем и технологии

УТВЕРЖДАЮ:
Зав. кафедрой
_____ Мальчуков А.Н.
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

магистерской диссертации

Студенту:

Группа	**ФИО**
8ИМ5А	Ногербек Нуржан Даулетбекулы

Тема работы:

Разработка и программная реализация алгоритма детектирования номерных знаков на изображениях	
Утверждена приказом директора (дата, номер)	№ 897/с от 20.02.2017

Срок сдачи студентом выполненной работы:	

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе	Разработка программного обеспечения системы управления требованиями. В результате данной магистроской диссертации был предложен инструментарий, позволяющий студентам обучающихся по дисциплины «Инженерия требований к системам», изучить процесс формирования технических требований и их согласования с заинтересованными лицами (анг. stackholder) и изучить жизненый цикл ИС.

Перечень подлежащих исследованию, проектированию и разработке вопросов	Обзор и анализ основных методов детектирования автомобильных номерных знаков, проектирование структуры и содержания основных классов инфраструктуры разрабатываемого программного обеспечения, разработка основных компонентов программного обеспечения, расчет ресурсоэффективности и ресурсосбережения, анализ вредных производственных факторов.
Перечень графического материала	UML диаграммы, изображение с концептуальным уровнем, структура основных классов, изображения представлений и таблицы с требованиями.

Консультанты по разделам выпускной квалификационной работы

Раздел	**Консультант**
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Попова С.Н.
Социальная ответственность	Акулов П.А.
Раздел на иностранном языке	Горбатова Т.Н.

Названия разделов, которые должны быть написаны на русском и иностранном языках:

Разделы введение, обзор систем управления требованиями, проектирование систем управления требованиями, реализовация системы управления требованиями, финансовый менеджмент, ресурсоэффективность и ресурсосбережение, социальная ответственность и заключение должны быть написаны на русском языке.

Раздел введение должен быть написан на английском языке.

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	

Задание выдал руководитель:

Должность	**ФИО**	**Ученая степень, звание**	**Подпись**	**Дата**
Доцент каф. ИСТ	Мальчуков А.Н.	к.т.н		

Задание принял к исполнению студент:

Группа	**ФИО**	**Подпись**	**Дата**
Доцент каф. ИСТ	Ногербек Н. Д.		

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Институт Кибернетики
Направление подготовки 09.04.02 «Информационные системы и технологии»
Уровень образования: магистр
Кафедра Информационных систем и технологии
Период выполнения: осенний семестр 2016 г. – весенний семестр 2017 учебного года

Форма представления работы:

Магистерская диссертация

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
27.06.2016	Постановка задачи и анализ предметной области	10
28.08.2016	Составление и утверждение технического задания	5
12.09.2016	Подбор и изучение материалов по тематике	15
12.09.2016	Анализ существующих на рынке аналогов	15
12.09.2016	Составление календарного плана	10
10.11.2016	Проектирование архитектуры веб-системы	10
14.02.2017	Разработка системы управления требованиями в веб-фреймворке Django	10
14.02.2017	Улучшение производительности веб-системы за счет оптимизации кода	15
15.05.2017	Тестирование конечного программного обеспечения системы управления требованиями	5
01.06.2017	Оформление пояснительной записки	5

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент каф. ИСТ	Мальчуков А.Н.	к.т.н		

СОГЛАСОВАНО:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
ИСТ	Мальчуков А.Н.	к.т.н		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8ИМ5А	Ногербек Нуржан Даулетбекулы

Институт	Институт Кибернетики	**Кафедра**	Информационных систем и технологий
Уровень образования	Магистр	**Направление специальность**	09.04.02 Информационные системы и технологии

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих	На основании информации, представленной в научных статьях и публикациях, аналитических материалах, статистических бюллетенях и изданиях, нормативно-правовых документах, определить методику расчета экономической эффективности.
2. Нормы и нормативы расходования ресурсов	
3. Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования	
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Оценка коммерческого потенциала инженерных решений (ИР)	Оценка ресурсной, социальной эффективности НИ и потенциальных рисков.
2. Формирование плана и графика разработки и внедрения ИР	Планирование этапов работы, определение календарного графика и трудоемкости разработки.
3. Составление бюджета инженерного проекта (ИП)	Затраты на материальные ресурсы, электроэнергию, заработную плату, страховые взносы, накладные расходы.
4. Оценка ресурсной, финансовой, социальной, бюджетной эффективности исследования	Оценка сравнительной эффективности проекта
Перечень графического материала (с точным указанием обязательных чертежей)	
1. Перечень работ и продолжительность их выполнения 2. Трудозатраты на выполнение проекта 3. Линейный график работ 4. Расчет затрат на материалы 5. Затраты на заработную плату 6. Смета затрат на разработку проекта 7. Оценки научно-технического уровня НИР	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Попова С.Н.	к.э.н		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ИМ5А	Ногербек Нуржан Даулетбекулы		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	**ФИО**
8ИМ5А	Ногербек Нуржан Даулетбекулы

Институт	ИК	**Кафедра**	Информационных систем и технологий
Уровень образования	Магистр	**Направление/специальность**	09.04.02 Информационные системы и технологии

Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика, рабочая зона) и области его применения	Объектом исследования являются среда разработки PyCharm, языки программирования Python и JavaSctript; язык гипертекстовой разметки HTML; каскадные таблицы стилей CSS; технологии JSON, AJAX и JQuery. Целью магистерской диссертации является разработка системы управления требованиями, которая обеспечивала бы системный подход к сбору, документированию и отслеживанию требований системы, что позволила бы обеспечить единое виденье разрабатываемой системы у всех заинтересованных лиц.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1 Производственная безопасность: 1.1 Анализ выявленных вредных факторов при разработке и эксплуатации проектируемого решения в следующей последовательности; 1.2 Анализ выявленных опасных факторов при разработке и эксплуатации проектируемого решения в следующей последовательности.	Производственная безопасность на стадии разработки веб-системы. 1.1 В качестве вредных факторов выделены: - Недостаточная освещенность рабочей зоны; - Монотонный режим работы; - Умственное перенапряжение. 1.2 В качестве опасных факторов выделены: - Опасность возникновения пожара; - Опасность поражения электрическим током.
2 Экологическая безопасность: 2.1 Анализ воздействия объекта на окружающую среду; 2.2 Разработать решения по обеспечению экологической безопасности со ссылками на НТД по охране окружающей среды.	2.1 Влияние объекта исследования на окружающую среду: – Утилизация компьютерной техники. – Утилизация бумаги. 2.3 Мероприятия по защите окружающей среды. - Отказ использования бумаги для документа оборота и переписке между участниками за счет ведения всех действий в электронном формате непосредственно в самой системе.
3 Безопасность в чрезвычайных ситуациях:	3.1 Возможные чрезвычайные ситуации: – Пожар;

3.1 Перечень возможных ЧС при разработке и эксплуатации проектируемого решения; 3.2 Выбор наиболее типичной ЧС; 3.3 Разработка действий в результате возникшей ЧС и мер по ликвидации её последствий.	– Социальные чрезвычайные ситуации (кибертерроризм). 3.2 Типичная чрезвычайная ситуация: – Возгорание (пожар); – Социальная чрезвычайная ситуация (терроризм). 3.3 Мероприятия по предотвращению наиболее типичной ЧС – пожара, согласно нормативным документам: НПБ 105-03 и ППБ 01–03.
4 Правовые и организационные вопросы обеспечения безопасности: 4.1 Специальные (характерные при эксплуатации объекта исследования, проектируемой рабочей зоны) правовые нормы трудового законодательства; 4.2 Организационные мероприятия при компоновке рабочей зоны.	4.1 Описание правовых норм для работ, связанных с работой за ПЭВМ согласно следующим документам: – Трудовой кодекс Российской Федерации" от 30.12.2001 N 197-ФЗ (ред. от 30.12.2015); 4.2 Влияние реализации веб-системы на работу компаний разрабатывающие информационные системы: – Обеспечение единого виденье разрабатываемой информационной системы у всех заинтересованных лиц (stakeholders); – Поэтапный сбор и документирование всех данных; – Создания технического задания к разрабатываемой информационной системе в электронном формате pdf; – Проведение анализа изменений. – Экономия времени и денежных средств. – Низкий порог вхождения в программу, не требует от потенциального пользователя дополнительных знаний о программе и навыков, интуитивно понятный интерфейс и высокое юзабилити.

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Акулов П.А	–		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ИМ5А	Ногербек Н. Д.		

РЕФЕРАТ

Выпускная квалификационная работа состоит из 127 страниц, 11 таблиц, 27 рисунков, 23 источников, 3 приложении.

Ключевые слова: система управления требованиями, программное обеспечение, веб-фреймворк, концептульный уровень, системный уровень, технологичный уровень, технические требования, технические задания, комплекс задач, функции, группа требований, контроль версий, комплекс задач, спецификации.

Объектом исследования является задача по разработке программного обеспечения системы управления требованиями.

Целью магистерской диссертации является – разработать программное обеспечение, которое позволяло бы пользователям сформировать простой и понятных документ, обеспечивающий единое виденье разрабатываемой ИС у всех заинтересованных лиц разного уровня (от руководителя до разработчика).

В результате данной магистроской диссертации был предложен инструментарий, позволяющий студентам обучающихся по дисциплины «Инженерия требований к системам», изучить процесс формирования технических требований и их согласования с заинтересованными лицами (анг. stackholder) и изучить жизненый цикл ИС.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном информационном мире несмотря на неприрывный и внушительный прогресс в индрустрии разработки информационных систем у многих IT компаний по прежнему возникают трудности при выявлении, сборе и документировании технических требований. Технологические требования настолько сложны, что зачастую топ-менеджмент IT компаний не понимают их в полной мере и подписывают «не глядя». Также из-за неправильно систематизированных и сформулированных требований IT компаниям не всегда удается вовремя закончить проекты и предоставить заказчикам ожидаемую функциональность. В результате – нарушение сроков и изменение требований, что приводит к убытком в IT компании.

По данным отчетов аналитического агентства «The Standard Group International» в 2016 году порядка 52.7% IT-проектов в странах СНГ и Европы во время разработки столкнулись с трудностями, которые оказывали огромное влияние на время разработки, качество и бюджет IT-проекта и впоследсвии привели к изменению ранее запланированных целей и ожидаемых результатов. Согласно статистике порядка 32 процентов подобных IT-проектов были окончены досрочно и не завершены [1]. Одними из основных причин подобной плачевной статистики в отчете указывались такие вещи как неправильно систематизированные технические требования и ошибки при управлении рисками в проектах.

Следует понимать, что управление требованиями является одним из важных этапов проектной деятельности. Проектирование информационных систем (далее ИС) – сложная комплексная задача, требующая не только учета огромного количества предъявляемых требований со стороны заинтересованных сторон, но и их систематизация. Систематизация же подразумевает под собой, то что будут объединяться различные требования по различным факторам и критериям в единую, структурированную иерархию

фундаментальных целей согласно взаимосвязям, между внешними и внутренними техническими требованиями.

В литературе [2] систематизация технических требований характеризуется с помощью:

- определения свяхей между составом технических требований к ИС;
- иерархии групп требований, а также их компонентов и детализированных спецификаций, определяемых единую концепцию классификации.

Требования являются основой любой информационной системы. Они определяют потребности заинтересованных лиц (заинтересованными лицами являются личности, на которых оказывает влияние разрабатываемая система) и функционал, которым информационная система должна обладать, чтобы удовлетворить эти потребности.

Разработка требований к информационным системам это – процесс выявления, формулирования, анализа, документирования и верификации требований, подлежащих к реализации в разрабатываемом программном продукте. Разработку технических требований принято считать самой сложной частью проектирования. Данная работа подразделяется на извлечение, анализ, документирование и проверку требований. Анализ технических требований цикличный, на каждой итерации происходит детализация высокоуровневых требований и подтверждение правильности требований будущими пользователями системы. Управление требованиями нужно рассматривать как неприрывный процесс на протяжении всего цикла создания программного обеспечения (далее ПО).

Целью магистерской диссертации является – разработать программное обеспечение, которое позволяло бы пользователям сформировать простой и понятных документ, обеспечивающий единое виденье разрабатываемой ИС у всех заинтересованных лиц разного уровня (от руководителя до разработчика). Сформированный с помощью програмного обеспечения

единый документ, отражающий иерархию и взаимосвязи всей системы требований, давал бы заинтересованным пользователям ответы на вопросы: «Что будет разрабатываться?», «Каким образом?», «Каков конечный результат?», а также обеспечивал бы единое виденье жизненного цикла ИС. Эффективносность функционирования ИС зависит от четырех стадий. Это предпроектная стадия, затем проектировочная стадия, третьим идет стадия внедрение и завершает данный список стадия функционирования. Качества проектной деятельности влияет на работоспособность будущей ИС, из-за чего все стадии делаться на несколько этапов и предусматривает составление технической документации, отражающей результаты работ.

Разработанная в рамках данной магистерской диссертации система управления требованиями дает инструментарий, позволяющий студентам обучающихся по дисциплины «Инженерия требований к системам», изучить процесс формирования технических требований и их согласования с заинтересованными лицами (анг. stackholder) и изучить жизненый цикл ИС.

1 ОБЗОР СИСТЕМ УПРАВЛЕНИЯ ТРЕБОВАНИЯМИ

1.1 Сравнение систем управления требованиями

Для структурированного хранения требований и их детализации используют соответствующие инструментальные средства – системы управления требованиями (далее СУТ). СУТ имеют слудеющие важные ключевые особенности:

- возможность указания связей между требованиями;
- возможность построения выборок в различных представлениях;
- функции отслеживания изменений;

Данный базовый и неисчерпывающий список функций дает возможность отслеживать и контролировать ошибки проектирования, сокращать время и повышать точность формирования оценки изменений технических требований.

В настоящее время на рынке широкое распространение получили такие СУТ как: IBM Rational RequisitePro, Telelogic DOORS и Borland Caliber RM. Также для контроля техническими требованиями в процессе создания ПО можно использовать легковесные веб-системы управления проектами, например, Redmine, Trello и т.д. Далее приведены некоторые особенности и сравнительный анализ подобных СУТ.

IBM Rational RequisitePro – средство управления требованиями к ПО при разработке программного продукта. Оно позволяет разработчикам определять и управлять требованиями, систематизировать и отслеживать изменения, которые могут возникнуть на протяжении всего жизненного цикла проекта, создавать с помощью RationalRequisitePro качественные сценарии использования, организовывать и повышать эффективность совместной работы [3].

IBM Rational/Telelogic DOORS – это семейство решений для управления требованиями, которое позволяет оптимизировать обмен информацией о

требованиях, контролировать большой объем взаимосвязанной информации, обеспечивает проверку выполнения требований и управление ими. IBM Rational/Telelogic DOORS успешно используется при создании сложных наукоемких изделий [3].

Borland Caliber RM – корпоративная система управления требованиями, которая разработана для повышения качества создаваемых продуктов, путем улучшения взаимодействия между участниками команды, упрощения анализа влияний требований и процесса передачи информации и возможности непрерывного сбора пожеланий заинтересованных в проекте лиц на всех этапах жизненного цикла проекта [4].

Redmine – это одна из наиболее популярных и современных систем управления задачами. Это открытое серверное веб-приложение для управления проектами и задачами, написанное на Ruby. Redmine представляет собой приложение на основе веб-фреймворка RubyonRails и распространяется согласно GeneralPublicLicense. Система позволяет проводить мониторинг и контроль выполнения задач [5].

На основании выполненного аналитического обзора были выделены критерии оптимального выбора СУТ. Далее в таблице 1.1 приведено сравнение наиболее популярных СУТ по этим критериям.

На основании изучения теоретических основ разработки и управления требованиями к ПО был выполнен сравнительный анализ различных систем управления требованиями. Каждая система обладает определенными достоинствами и недостатками. Для использования в крупных и сложных проектах становится рациональным использование коммерческих СУТ. При грамотном использовании они позволяют легче обнаруживать ошибки на ранних этапах проектирования и оценивать влияние изменений на систему, возможность их реализации, сроки и стоимость. Однако в небольших или некоммерческих проектах будет неуместным использовать данные тяжеловесные и коммерческие СУТ.

Таблица 1.1 – Сравнение систем управления требованиями.

Критерии	IBM Rational RequisitePro	IBM Rational Telelogic DOORS	Borland Caliber RM	Redmine
Веб-интерфейс	+	+	+	+
Трассировка требований	+	+	+	–
Авторизация доступа	+	+	+	+
Права доступа	+	+	+	–
Интеграция со средствами визуального моделирования UML	+	+	+	+
Интеграция с системами управления версиями	+	+	+	+
Работа с нетекстовыми объектами	–	+	+	+
Оповещения об изменении требований	+	+	+	–
Бесплатность	–	–	–	–
Легковесность	–	–	–	–

1.2 Модель систематизации требований

1.2.1 Концепция модели

Формирование эталонной модели систематизации требований позволяет определить рамочную модель или фреймворк (англ. framework) для

использования при формализации и систематизации технических требований к информационной системе. Фреймворк, например, может представлять собой структуру (шаблон) документа, в котором будет отражаться результат формализации требований, а также набор принципов (подходов) к их систематизации. Рассмотрим подробнее данную модель систематизации (см. рис. 1.1).

На концептуальном уровне модель представляет собой иерархию уровней абстракции (англ. *views)* с точки зрения профессиональной позиции каждой заинтересованной стороны. Дж. Захман в своей архитектурной онтологии называет эти уровни Reification Transformations [6]. В случае рассмотрения архитектурной составляющей информационной системы (особенно на низких уровнях абстракции) можно определить соответствия с онтологией представлений архитектуры информационной системы (viewpoints) [7].

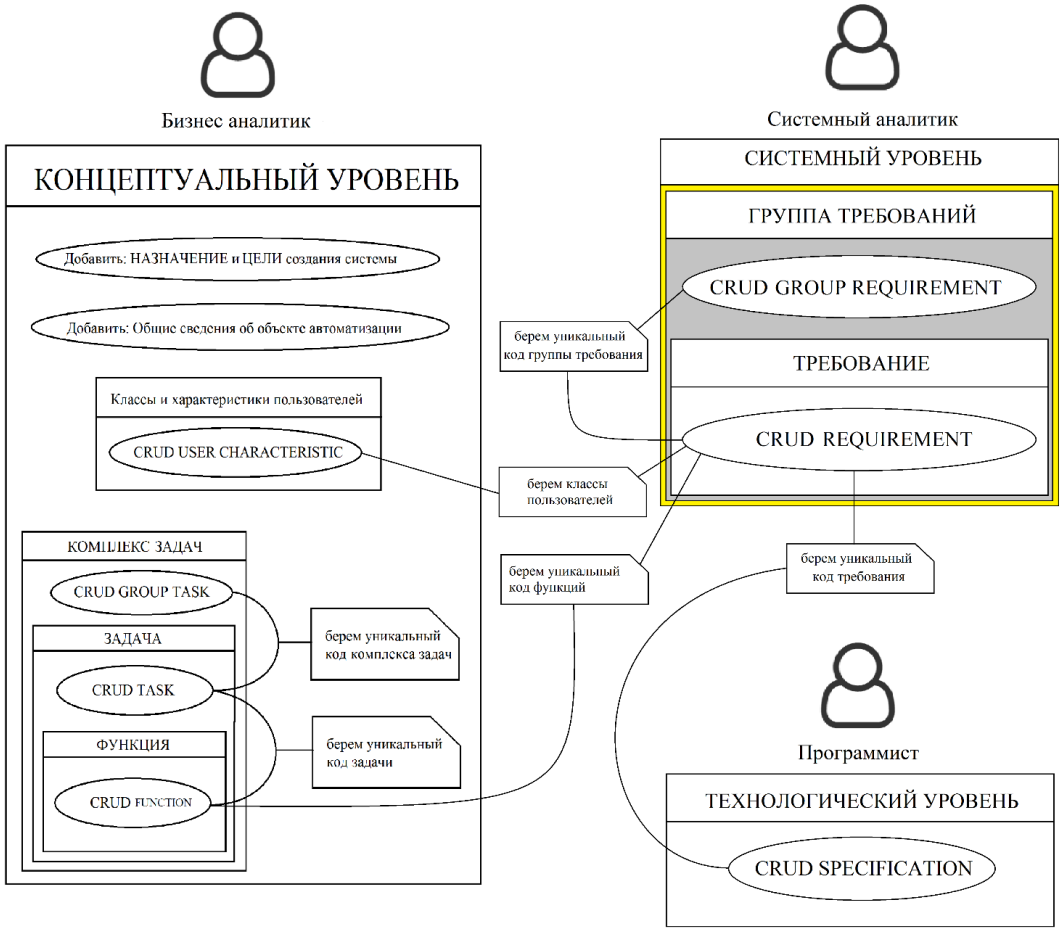

Рисунок 1.1 – Модель систематизации требований.

1.2.2 Описание уровней модели

Главная задача описанных далее уровней – обеспечение сквозного и однозначного понимания целей и задач, поставленных на уровне владельца бизнеса (руководителя), вплоть до построения конечной технологической модели, отражающейся в логическом программном коде исполняемых файлов, обеспечивающих непрерывность бизнеса. Рассмотрим уровни подробнее.

Концептуальный уровень: взгляд внешнего заказчика в лице владельца бизнеса (business scope planners), и бизнес-аналитика (англ. business analyst).

Описывается назначение и цели создания системы, а также бизнес-логика. Бинес-логика – это реализация предметной области в информационной системе, которая может иметь различные виды представлений. В общем случае это целостное описание характеристик объекта автоматизации, включающее в себя:

- общие сведения об объекте автоматизации;
- описание комплексов задач;
- функциональные компоненты системы (выполняемые функции);
- характеристики пользователей в будущей системе.

Данные описания могут представляться в различных формах, в том числе в виде модели бизнес-процессов.

В данной работе представлена архитектура программного обеспечения информационной системы, представляющая собой обеспечение потребностей автоматизируемых бизнес-процессов набором функциональных компонентов (функций) системы. Практическая реализация этого описания может иметь различные представления. Самый простой вариант – перечень групп требований (англ. types of requirements). Группа требований – это логическое объединение однотипных требований, т. е. результат объединения их в единую систему [9]. Обобщением в рамках данного описания, а также приведением его к ГОСТу 34 «Стандарты на разработку автоматизированных систем» является обозначение уровня классификации, определяющего, к какому виду

обеспечения и к какой составляющей системы относится та или иная группа требований (декомпозиция комплексов задач на функции).

Системный уровень: взгляд системного аналитика (англ. system analyst).

Уровень отражает логическое описание работы информационной системы, а именно перечень требований и их взаимосвязь. При некотором приближении можно назвать этот уровень системным проектом. Системный проект – это общее техническое представление информационной системы, он включает в себя полноценную функциональную модель, информационную модель и технические требования. Кроме того, в данном разделе может определяться объем и организация работ, требуемые ресурсы и проектные риски.

Технологический уровень: взгляд разработчика (англ. developer).

Описывается технологическая модель системы в виде спецификаций для программиста. Спецификация определяет, что и каким образом должна делать информационная система, а также детальный алгоритм ее функционирования.

Рассмотренная выше модель регламентирует иерархию и взаимосвязь всей системы требований (уровней). Каждый ее элемент определяется на различных уровнях интерпретации для заинтересованных сторон. Таким образом, взаимосвязи между требованиями, обозначенные на верхнем уровне, однозначно должны определяться уровнем ниже. Это главное правило модели.

Далее рассмотрим примеры использования описанной модели.

1.3 Применение модели

1.3.1 Название, цели и задачи

Первый этап применения модели — это формализация бизнес-логики будущей информационной системы. Главной задачей на данном этапе является определение назначения и целей создания ИС (см рис. 1.2). Рисунок 1.2 был взят и адаптирован из статьи [10].

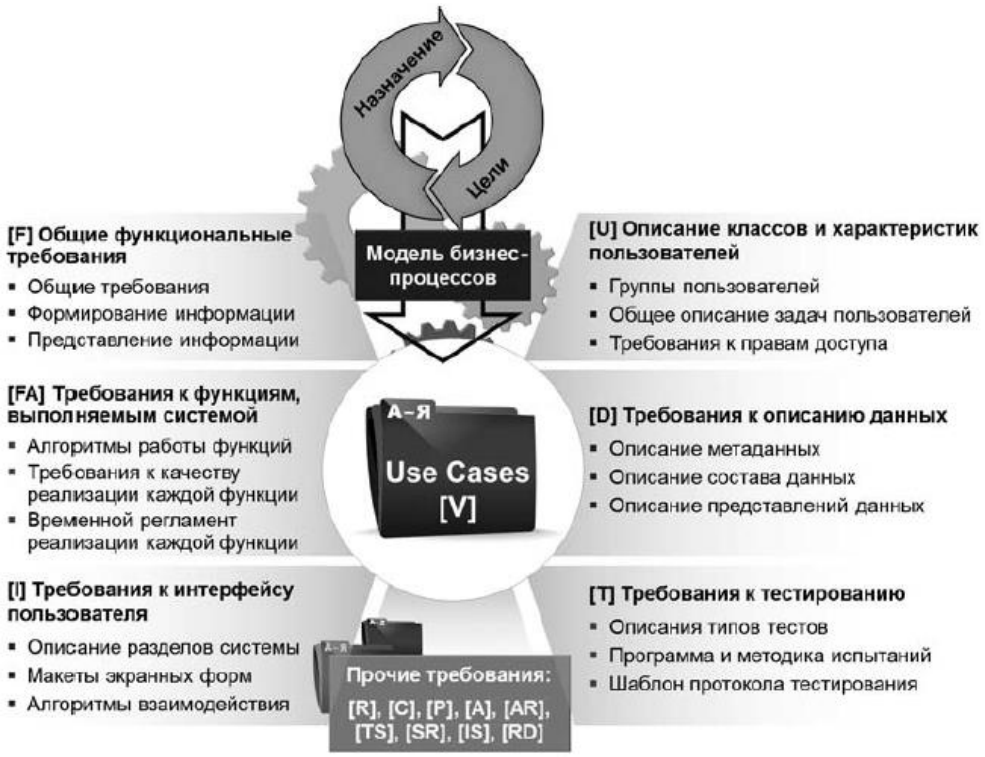

Рисунок 1.2 – Модель описания концепции и архитектуры ИС.

Для определения целей и задач, особенно в том случае, когда система проектируется для российского заказчика или для заказчика из стран СНГ, рекомендуется следовать требованиям ГОСТа 34. В рассматриваемом случае актуальными являются следующие требования [11]:

- назначение информационной системы должно отражать вид автоматизируемой деятельности и список объектов автоматизации, на которых планируется её применять;
- цели создания системы должны описывать названия и необходимые значения производственных, технических, финансовых или прочих показателей объекта автоматизации. Данные показатели должны быть достигнуты в результате реализации автоматизированной системы.

При описании целей следует использовать такие языковые конструкции, в которых будут присутствовать категории изменения: «улучшить», «увеличить», «уменьшить», «повысить», «снизить», «упорядочить», «предоставить» и т. д. Из контекста читателю должно быть понятно, что разрабатываемая система действительно улучшит жизнь пользователя,

автоматизировав его деятельность; должно присутствовать сравнение типа «есть – должно быть». Кроме того, рекомендуется указывать критерии оценки достижения целей системы, т. е. то, каким образом мы оценим выполнение системой поставленных целей.

Еще одним подходом к описанию целей может быть способ формирования цели по технологии SMART (мнемоническая аббревиатура, используемая в менеджменте и проектном управлении для определения целей и постановки задач – от англ. Specific, Measurable, Achievable, Realistic, Timely). Метод отличается от описанного выше в первую очередь тем, что позволяет в одном предложении понять не только «что, зачем, кто», но и «где? когда? как?». При определении цели данным методом она по умолчанию содержит критерий для оценки достижения.

Кроме того, есть еще один полезный прием. Он позволяет в ходе интервью заинтересованных сторон правильно сформулировать цели, отталкиваясь от множества сформулированных задач по SMART. На первом этапе интервью необходимо заполнить таблицу со всеми атрибутами. Из полученного множества задач необходимо убрать дубликаты, найти пересечения и взаимоисключения, максимально сократив список. Имея данный список, проведя «проектную сессию», можно сформулировать то назначение и цели системы, которые будут удовлетворять всему объему поставленных задач [12].

1.3.2 Модель бизнес процессов

Далее для получения единого видения проектируемой информационной системы у всех заинтересованных лиц необходимо формализовать алгоритмы действий, выполняемые разрабатываемой информационной системой. Как известно, алгоритм представляет собой последовательный набор инструкций, определяющий порядок действий для решения задачи за установленное время. Для описания подобного рода алгоритмов целесообразно исползовать модели бизнес-процессов.

Существует множество методологий описания бизнес-процессов, применяемых в различных ситуациях, однако выбор конкретной методологии в любом случае зависит от контекста решаемой задачи. Основными требованиями к модели являются наглядность последовательности выполнения операций, а также явное описание данных и документов, используемых в под процессах. Модель должна полно и однозначно описывать алгоритм действий, происходящих в ИС. Точность построенной модели должна быть такого уровня, который позволит понять логику работы системы и перейти к детальному формированию требований.

С ориентацией на стандарты серии ГОСТ 34 модель бизнес-процессов может быть представлена в виде комплексов задач с соответствующей декомпозицией на функции, выполняемые системой.

На рисунке 1.3 представлен пример модели бизнес-процессов (см. уровень А.0), в которой выделены комплексы задач (см. уровень А.1 – А.6). Каждый подпроцесс описывается в формате нотации «flowchart» с описанием входных и выходных данных для каждой функции, выполняемой системой.

Когда модель бизнес-процессов построена, можно проследить, как назначение и цели системы на верхнем уровне закрепляют рамки автоматизируемого бизнес-процесса (взаимосвязь цели функционирования и характеристики объекта автоматизации). На данном этапе очевидно, что формализация модели бизнес-процессов является ключевым фактором для принятия множества управленческих решений в ходе согласования и определения рамок проекта по автоматизации объекта, проектирования его архитектуры.

1.3.3 Варианты использования системы

Следующий шаг систематизации требований должен обеспечить общее понимание концепции разрабатываемой информационной системы между бизнес-аналитиком, системным-аналитиком и разработчиком. Этот процесс

определяет переход от концептуального уровня на архитектурный. Для обеспечения данного понимания наилучшим образом подходит описание так называемых вариантов использования (англ. use cases), опирающихся на разработанную модель бизнес-процессов.

Рисунок 1.3 – Детальное описание группы задач и бизнес модель происходящих процессов.

Практика создания вариантов использования как средств уточнения требований к поведению информационных систем и бизнес-процессов широко используется в мировой практике. Варианты использования обеспечивают комплексное понимание функциональных требований, сопровождение которых необходимо на всех этапах жизненного цикла (англ. life-cycle) системы, показывая, как именно будет применяться система в различных ситуациях.

На первый взгляд идея создания вариантов использования кажется довольно простой. Тем не менее, определяя (формализуя) набор вариантов использования, сначала необходимо закрепить уровень их детализации. Для этого поставим вопрос: с какой целью будем описывать данный use case, и что

получим в результате? Постановка целей и задач каждого компонента требований позволяет определить требуемый уровень детализации, а также не писать лишнего, что экономит время формализации требований.

Формализация вариантов использования ни в коем случае не является конечным результатом проектирования ИС. Скорее даже наоборот, это один из первых этапов, обеспечивающий понимание функциональных требований к системе. Обозначить роли актёров, сущности и варианты использования – вот то, с чего необходимо начинать определение функционала и пользователей будущей информационной системы.

Варианты использования определяют набор реализуемых и ошибочных последовательностей, которые могут совершаться при взаимодействии актёра и системы (подсистемы/класса) для достижения некоторой цели. Совокупность вариантов использования дает разработчику комплексное понимание для проектирования архитектуры приложения, осуществляемого на фазе разработки (англ. elaboration). На этом принципе построена, в частности, методология разработки ПО Rational-Unified-Process, созданная компанией Rational Software [13].

Таким образом, по результатам анализа назначения и целей описания вариантов использования можно определить ключевую предпосылку систематизации требований: варианты использования являются связующим звеном между пониманием достижения системой целей с позиций всех заинтересованных сторон, реализуя их бизнес-процессы посредством архитектурных и функциональных требований к информационной системе.

Формирование документа, содержащего вышеупомянутые разделы уже на этом этапе может быть значимым содержанием концепции создания информационной системы (англ. vision). Данный документ – отправная точка для закрепления единого видения у всех заинтересованных лиц к автоматизируемым процессам. Следовательно, на описываемом этапе были представлены общие подходы к определению данных разделов, далее рассмотрим принципы формализации и систематизации требований.

1.4 Взаимосвязь, классификация и кодирование требований

1.4.1 Классификация требований

Ранее описывалось, что модель систематизации определяет комплексную взаимосвязь бизнес-процессов, вариантов использования и групп технических требований. Однако набор групп требований должен находиться в каждом случае отдельно, отталкиваясь от эталонной модели (например, по документу software requirements specification [14], или составу types requirements [15]). Для конкретной информационной системы в ходе ее проектирования необходимо обозначить свой, уникальный набор групп, целостно описывающий требования заинтересованных лиц.

Для того чтобы возможно было построить взаимосвязи (ссылки) между требованиями, а также для повышения удобства их использования и восприятия, необходимо определить коды для всех групп описаний информационной системы (англ. description). Для удобства результат классификации рекомендуется помещать в начале документа, что позволяет сразу установить состав технологического содержания документа. В таблице 1.2 приведен пример классификации требований.

Фактически представленная структура является основной для содержания структуры документа, так как включает все разделы описания информационной системы.

1.4.2 Взаимосвязь требований

На концептуальном уровне взаимосвязь между видами требований описывает отношения между их элементами согласно модели «сущность-связь». Отношения между элементами могут иметь как прямую, так и опосредованную связь. Прямая связь представляет собой отношения между элементами групп требований: «один к одному», «один ко многим».

Таблица 1.2 – Классификатор элементов описания.

Код	Элемент описания	Раздел
BP	Модель бизнес-процессов (описание комплексов задач)	Описание бизнес-логики
U	Описание классов и характеристик пользователей	
V	Описание вариантов использования	
FA	Требование к функциям (задачам), выполняемым системой	Основные требования к системе
I	Требования к интерфейсу пользователя	
D	Требования к описанию (составу) данных	
R	Требования к отчетам	
AR	Требования к правам доступа	
A	Требование к администрированию, управлению доступом и безопасностью системы	
P	Требования к средствам интеграции	
F	Общие функциональные требования	Дополнительные требования к системы
C	Требования к справочникам и классификаторам системы	
IS	Требования к безопасности	
RD	Требования к надежности	
T	Требования к тестированию	
M	Требования к математическому обеспечению	Требования к видам обеспечения
TS	Требования к техническому обеспечению	
SR	Требования к программному обеспечению	

Для лучшего понимания рассмотрим несколько примеров для прямых связей:

- вариант использования «Расчета затрат на общехозяйственные расходы» по принципу «один к одному» относится к алгоритму работы функции «Расчет затрат на общехозяйственные нужды»;
- вариант использования «Расчета затрат на общехозяйственные расходы» по принципу «один ко многим» относится к отчетам «Административно-управленческие расходы» и «Амортизационные отчисления».

К опосредованной связи относится вид связи «многие ко многим» (ее невозможно отобразить, однако проблему решает преобразование в две связи «один ко многим»). Приведем пример такой связи: требования к отчетам по принципу «многие ко многим» относятся к требованиям к описанию данных. Но эта опосредованная связь прослеживается прямыми связями «один ко многим» через различные варианты использования.

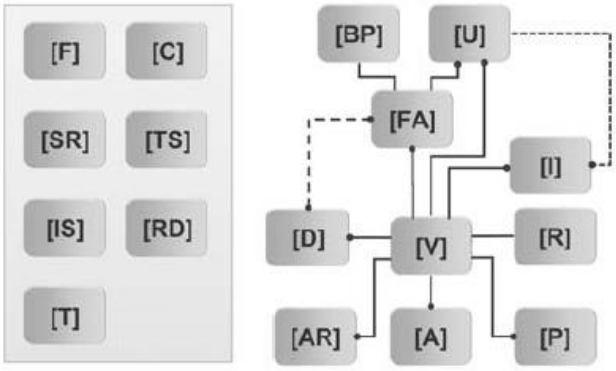

Рисунок 1.4 – Взаимосвязь требований.

Рассмотрим пример взаимосвязи основных требований (см. рис. 1.4). Центральным элементом требований являются варианты использования [V]. Описание каждого из элементов (определенный вариант использования) ссылается [16]:

- на описание классов и характеристик пользователей [U] (связь типа «один к одному», так как каждый вариант использования определяется для конкретного пользователя);
- требования к функциям (задачам), выполняемым системой [FA];

- требования к отчетам с результатами текущей работы [R];
- требования к пользовательскому интерфейсу (UI – англ. user interface) [I];
- требования к администрированию, управлению доступом и безопасностью системы [A];
- требования к основным пользовательским правам доступа [AR];
- требования к описанию (составу) данных [D].

Функции (задачи), выполняемые системой [FA] в свою очередь по принципу «один к одному» ссылаются на описания бизнес-процессов [BP] и на требования к средствам интеграции [P]. Остальные (дополнительные) требования в данном случае не имеют непосредственной прямой связи между требованиями и описываются в следующем составе [17]:

- общие функциональные требования [F];
- требования к справочникам и классификаторам [C];
- требования к безопасности [IS];
- требования к надежности [RD];
- требования к тестированию [T];
- требования к математическому обеспечению [M];
- требования к аппатарному обеспечению [TS];
- требования к программному обеспечению [SR].

Некоторые из требований рекомендуется определять прямой связью. Например, требования к тестированию системы должны быть однозначно «завязаны» на варианты использования или требования к алгоритмам работы функций (в зависимости от подходов к тестированию).

Прямые взаимосвязи между различными элементами-требованиями в документе рекомендуется определять перекрестными ссылками. Чтобы понимать, на какой конкретный элемент мы ссылаемся, необходимо обозначить подход к кодированию элементов.

1.4.3 Кодирование требований

Ранее в разделе был определен подход к кодировке требований (если быть точным, в данном случае это виды требований). Каждому типу (группе) требований соответствует некий первичный код, используемый в качестве первичного элемента в общей кодировке.

Далее подход к кодированию требований определяется иерархией элементов, изображенной на рис. 1.5:

Рисунок 1.5 – Кодирование требований.

Каждая группа требований может содержать множество элементов-требований, которые по смыслу рекомендуется объединить в группу. Таким образом, группировка представляет собой смысловое объединение различных требований. Рассмотрим пример: в группу требований «Расчет затрат» могут входить элементы «Расчет затрат на общехозяйственные расходы», «Расчет затрат на общепроизводственные расходы» и др.

Следующий элемент кода – это номер элемента из множества требований в рамках группы, т. е. собственно сами требования. Далее к каждому требованию может прилагаться его спецификация (последний элемент), в которой содержится техническая информация для аналитиков и разработчиков. Состав спецификации зависит от вида требований.

1.5 Описание групп требований

1.5.1 Основные группы

Ранее в разделах были определено отличие описания бизнес-логики от требований к информационной системе. Однако при декомпозиции от комплексов задач до конкретных функций, выполняемых системой, нельзя обойтись без их привязки к вариантам использования и классам пользователей. Поэтому на уровне описания системного аналитика необходимо формализовать уже классифицированные ранее элементы:

- [V] Описание вариантов использования.
- [U] Описание классов и характеристик пользователей.

Кроме того, в рассматриваемом примере к основным требованиям к системе относятся следующие группы:

- [F] Общие функциональные требования.
- [FA] Требования к функциям (задачам), выполняемым системой.
- [I] Требования к интерфейсу пользователя.
- [D] Требования к описанию данных.
- [T] Требования к тестированию.

Описание групп и элементов требований (в рамках типа) рекомендуется отображать в форме таблицы, чтобы удобнее ориентироваться по всей структуре этого вида требований. Эталонная шапка таблицы – три столбца «Код», «Наименование требования» и «Примечание». Однако в зависимости от вида требований таблица может дополняться другими необходимыми (в зависимости от контента) столбцами. Далее подробнее рассмотрим данные виды требований.

1.5.2 Описание вариантов использования

В настоящем разделе описывается состав вариантов использования информационной системы. Как уже рассматривалось ранее, эта группа является ключевым связующим звеном для всех требований к системе. В

таблице 1.3 представлен пример оформления вариантов использования, в случае если один вариант использования определяет одну роль пользователя. Далее в разделе приведен пример детализации описания варианта использования.

1.5.3 Описание классов и характеристик пользователей

Описание классов и характеристик пользователей может содержать следующие разделы:

- Классы пользователей;
- Общее описание задач пользователей;
- Требования к правам доступа.

Таблица 1.3 – Требование к вариантом использования.

Код	Вариант использования	Пользователь
V.01.00	Требования к основным вариантам использования	
V.01.01	AS – Варианты использования для роли Администратор	AS
V.01.02	AN – Варианты использования для роли Руководитель	AN
V.01.03	RU – Варианты использования для роли Руководитель	RU
V.01.04	OP – Варианты использования для роли Оператор	OP
V.01.05	PO – Варианты использования для роли Учреждение ПО	PO
V.01.06	NP – Варианты использования для роли Незарегистрированный пользователь	NP

В этой группе требований следует описывать характеристики пользователей, выполняемые ими задачи и их права доступа. Для удобства стоит обозначить соответствующим кодом каждый класс пользователя (т. е. его роль). В таблице 1.4 представлен пример описания классов (ролей) пользователей. На следующем этапе (описание задач и прав доступа) можно детализировать требования на основе общего подхода к систематизации

требований, только с привязкой к коду конкретного пользователя.

Далее перейдем к разделу основных требований к информационной системе.

Таблица 1.4 – Классы и характеристики пользователей.

Роль	Код	Описание
Администратор	AD	Лицо, отвечающее за обеспечение целостного функционирования системы.
Аналитик	AN	Лицо, отвечающее за содержательное функционирование системы. Строит рейтинги учреждений, создает проект премирования на основе рейтинга.
Руководитель	RU	Получает агрегированную информацию по формированию рейтингов учреждений и распределению премий.
Оператор	OP	Лицо, выполняющие работы по информационному наполнению системы и контролю корректности данных и т. п .
Учреждение	FI	Филиал организации. Имеет доступ к своей персональной информации.

1.5.4 Общие функциональные требования

Общие функциональные требования могут содержать следующие элементы описания [17]:

- Общие требования.
- Формирование информации.
- Представление информации.

В настоящем разделе преимущественно содержатся требования, применимые к другим элементам требований или к системе в целом. Например, система должна хранить историю изменения значений или иметь возможность изменения настроек профиля пользователя. В таблице 1.5 представлен пример общих требований.

Таблица 1.5 – Общие функциональные требования.

F.01.00	Общие требования	Примечания
F.01.01	Работа пользователей с информационной системой должна быть в режиме online через тонкий клиент пользователя (браузер)	С применением таких браузеров как: Internet Explorer 8 и выше, Firefox 4 и выше, Google Chrome 11 и выше, Safari 5 и выше.
F.01.02	В ИС должен быть предусмотрен UI (пользовательский интерфейс) для редактирования логической структуры портала и публикации различных видов информационных материалов	Виды информационных материалов: Новостные сообщения, статьи, документы формата MS Word.
F.01.03	Система должна обеспечивать доступ к информационным материалам посредствам интернет-портала, поддерживающего навигацию пользователей в соответствии с многоуровневым (иерархическим) классификатором	

1.5.5 Требования к функциям, выполняемым системой.

Требования к функциям, выполняемым системой, могут содержать следующие разделы:

- Описание алгоритмов работы функций.
- Требования с описанием качества реализации всех функции.
- Временные скоки реализации всех функции.

Данная группа содержит требования к функциям (задачам), выполняемым системой. Функции (задачи) есть продолжение (декомпозиция) комплексов задач, определяемых бизнес-процессами. Функции (задачи) также могут вырабатываться на основе вариантов использования, при применении объектного подхода к проектированию информационной системы. Однако первый вариант наиболее удобен для построения процессоориентированных

систем. В данном случае примером может являться та же группа требований «Расчет затрат», в которую могут входить элементы «Расчет затрат на общехозяйственные расходы», «Расчет затрат на общепроизводственные расходы» и др. В этом случае в полном соответствии (прямая связь «один к одному») с бизнес-процессами определяются функциональные алгоритмы с аналогичными названиями.

Таблица 1.6 – Требования к работе алгоритмам функций.

CODE	Код функции	Функция	Пользователи
FA.01.00	A.1	Анализ и верификация исходных данных.	
FA.01.01	A.1.1	Загрузка массива данных.	AS, OP
FA.01.02	A.1.2	Верификация данных.	AS, OP, RU
FA.01.03	A.1.3	Утверждение данных.	AS, OP
FA.06.00	A.6	Формирование государственных заданий.	
FA.06.01	A.6.1	Расчет затрат на оказание образовательной услуги.	AS, AN
FA.06.02	A.6.2	Расчет затрат на общехозяйственные нужды.	AS, AN
FA.06.03	A.6.3	Расчет затрат на содержание имущества.	AS, AN
FA.06.04	A.6.4	Формирование проекта.	RU, AS, AN

В примере (табл. 1.6) показана ситуация, когда в табличной форме указывается код функции бизнес-процесса и пользователи, использующие данную функцию и соответствующий бизнес-процесс.

1.5.6 Требования к интерфейсу пользователя

Требования к интерфейсам пользователя могут содержать следующие разделы:

- Описание разделов системы.
- Макеты экранных форм.

Требования к интерфейсам зачастую представляют собой три смысловых блока: общие требования к интерфейсам, требования к элементам управления и детализируемые требования к конкретным интерфейсам пользователя (экранным формам). Первые два зачастую описываются текстовой информацией, последнюю группу требований для лучшего понимания рекомендуется визуализировать в виде макета интерфейса. В примере (табл. 1.7) представлены требования к интерфейсу пользователя.

Таблица 1.7 – Требования к интерфейсу пользователя.

Код	**Требования**
I.01.00	Общие требования
I.01.01	Потенциальный юзер системы должен иметь возможность перехода по гиперссылкам информационной системы, чтобы получить доступ к данным.
I.01.02	Юзер должен иметь возможность получения информации.
I.01.03	ИС не должна предоставлять кнопки без названия или без помеченных специальной характерной иконкой.
***	***
I.02.00	Требования к элементам управления
I.02.01	Предупреждение об обязательности заполнения поля.
I.02.02	Подчеркнутый текст — признак того, что элемент становится активным при открытии формы.
I.02.03	Строка заголовка, данные, итоговая строка – недоступны для редактирования, строку данных можно выбирать.

1.5.7 Требования к описанию данных

Требования к описанию данных могут содержать следующие элементы [18]:

- Описание метаданных.
- Описание состава данных.
- Описание представлений данных.

В настоящем разделе описываются требования, содержащие информацию о данных, хранящихся и обрабатываемых в информационной системе. Описание метаданных представляет собой информацию об атрибутах и их характеристиках. Описание состава данных включает необходимые табличные формы и перечень соответствующих атрибутов. Описание представлений данных содержит информацию о виде табличных форм и отчетов. Таким образом, определяются требования к описанию и составу данных (от их атрибутов до визуального представления отчетных форм). В таблице 1.8 представлен пример описания требований к данным.

Таблица 1.8 – Требования к описанию данных.

Код	**Требования**	**Примечания**
D.01.00	Требования к списку метаданных объекта данных	
D.02.00	Требования к составу данных	
D.02.01	Нормативные затраты	FA.06.01
D.02.02	Налоги	FA.06.02
D.02.03	Тарифы	FA.06.03
D.02.04	Затраты учреждения на тепловую энергию в отчетном периоде	FA.06.03
D.02.05	Затраты учреждения на электроэнергию в отчетном периоде	FA.06.03
D.03.00	Требования к представлению данных	
D.03.01	А.1.2. Верификация данных	FA.01.02

1.5.8 Требования к тестированию

Требования к тестированию могут содержать следующие элементы [19]:

- Описания типов тестов.
- Программа и методика испытаний.
- Шаблон протокола тестирования.

В этом разделе следует определить и описать подходы к тестированию системы и приемке системы заказчиком (заинтересованными лицами). Проведение тестов может быть привязано как к конкретным алгоритмам

работы функций (компонентное тестирование), так и к вариантам использования (функциональное тестирование), или, например, к интерфейсам пользователя (юзабилити-тестирование и тестирование интерфейса пользователя), а также к другим видам технических требований. Для этого устанавливается прямая связь (вида «один к одному» или «один ко многим») и описываются требования к тестированию каждой отдельной функции (в случае с функциональным тестированием).

Для формальных процедур тестирования рекомендуется определять формат протокола тестирования в целях сокращения в будущем трудозатрат на оформление результатов тестирования. Подобные протоколы должны подписываться участками приемочной комиссии при демонстрации системы или ее сдачи (показа) заказчику.

1.6 Спецификация требований

Спецификации требований предназначены для описания конкретных реализаций функций, вариантов использования, интерфейсов и т. п. Именно в них описывается сам элемент группы технических требований и его особенности. Например, если это алгоритм функции, то описываются все шаги его реализации, особенности, исключения и т. п., т. е. дается вся информация, которая понадобится разработчику при реализации этой функции в виде программного кода.

2 ПРОЕКТИРОВАНИЕ СИСТЕМЫ УПРАВЛЕНИЯ ТРЕБОВАНИЯМИ

Одним из распространённых способов описания архитектуры является представленный Филиппом Крухтеном метод под названием «4+1 представление» [20]. В данном методе важнейшие аспекты архитектуры ИС описываются с помощью различных представлений системы: логическим представлением, представлением процесса, представлением реализации и представлением развертывания. Далее вышеупомянутые представления объяденяются в единое представление – представление прецедентов. Каждое из данных представлений описывают разные аспекты архитектуры ПО, как показано на рис. 2.1.

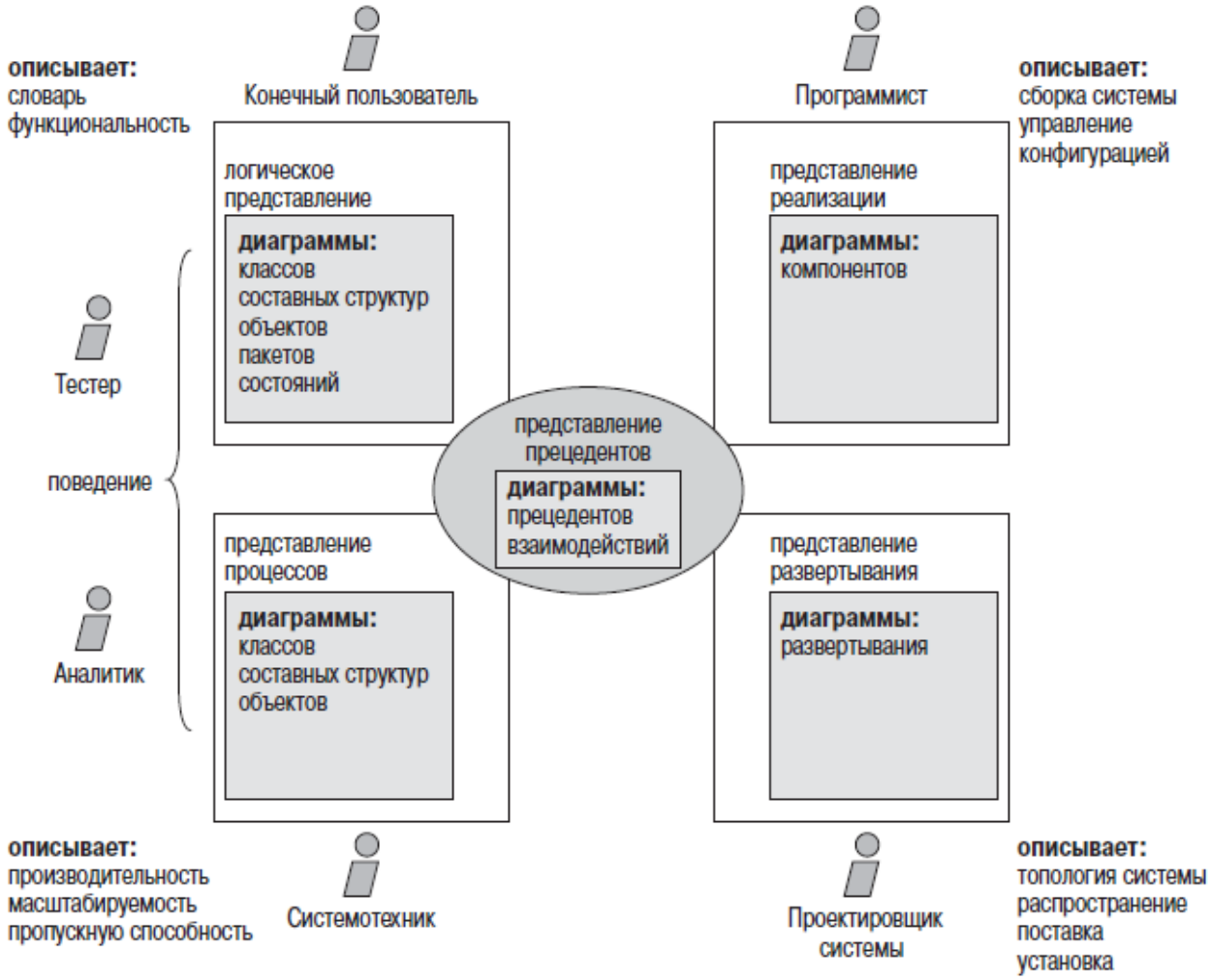

Рисунок 2.1 – Архитектура системы в методе «4+1 представление».

Представления [21]:

- Логическое представление описывает предметную область как неких набор основных классов и объектов. Основной акцент в данном представлении уделяется отображению того, как данные объекты и классы реализуют требуемое поведение информационной системы.
- Представление процессов имитирует выполняемые процессы информационной системы как активные классы.
- Представление реализации имитирует файлы и компоненты, реализующую физическую базу из кода для системы. Данное представление показывает взаимосвязь между элементами и управляет настройками списка компонентов для определения версии информационной системы.
- Представление развертывания имитирует физическое развертывание артефактов на физические вычислительные узлы. Данное обеспечение дает возможность имитировать распределения артефактов между узлами распределенной системы.
- Представление прецедентов описывает технические требования, предъявляемые к системе, как список прецедентов. Эти прецеденты обеспечивают базу для создания остальных представлений.

После создания 4+1 представлений существует возможность с помощью UML (Unified Modeling Language) моделей составить анализ всех основных аспектов архитектуры ИС. В случаи использования итеративного жизненного цикла UP (Unified Process), архитектура «4+1» создается не за один проход, так как модернизируется со временем.

2.1 Определение пользователей системы

При предпроектой деятельности немаловажно знать типы пользователей, которые будут пользоваться будущей информационной системой. Описывая и группируя пользователей будущей ИС и изучая их требования, можно узнать множества информации. К примеру, можно узнать

о том, как целесообразно будет спроектировать релиационную базу данных, какие ограничения заложить в информационную систему, определить права доступа, как группировать веб-страницы, как обеспечить юзабилити, какие данные необходимо отобразить в представлениях веб-сайте.

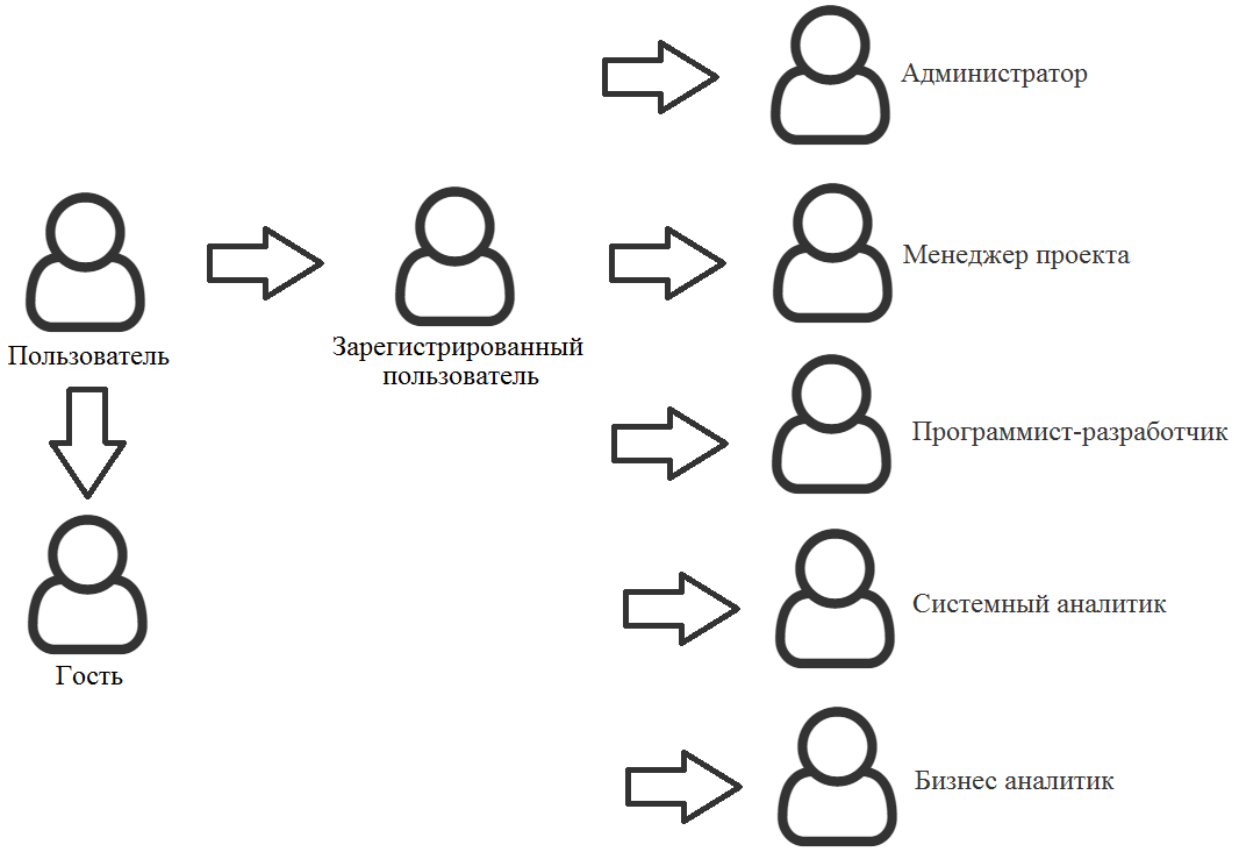

Рисунок 2.2 – Иерархия пользователей системы.

Таблица 2.1 – Классы пользователей.

Класс пользователей	Символ	Описание
Администратор	AD	Лицо, которое отвечает за целостность функционирования информационной системы. Также администратор осуществляет работы по информационному наполнению ИС и контролю корректности информации и т.п. Администратор обладает максимальными правами в ИС.
Бизнес аналитик	BA	Лицо, указывающее назначение и цели создания системы, описывает комплекс

		задач, и предоставляет перечень используемых функций системы.
Системный аналитик	CA	Лицо, составляющие группу требований в системе управления требованиями.
Менеджер проекта	MP	Лицо, указывающее классы и характеристики пользователей конкретного проекта в системе управления требованиями.
Программист-разработчик	RA	Лицо, описывающие технологическую модель системы в виде спецификаций для программиста.
Гость	NP	Пользователь портала, использующий доступ к общей информации. Имеет доступ к внешнему контуру.

На рисунке 2.2 отображены несколько групп пользователей (актеров), которые в будущем будут соприкосаться с ИС. Основным и общим типом пользователя является «Пользователь». Стрелки указывают отношение, то есть принадлежность категории пользователей: гости и зарегистрированные пользователи. Характеристики, используемые у обеих групп пользователей, приписаны исполнителю «Пользователь». У пользователя с разными ролями в проектах системе есть различные функции. Далее в разделе 2.2 будут детально рассмотрены диаграмма вариантов использования (анг. use cases).

2.2 Информация о диаграммах вариантов использования

Диаграммы вариантов использования или диаграмма прецедентов дает информацию о функциональном назначении информационной системы или дает понимание того, как работает ИС. Создание диаграммы use cases имеет следующие основные цели [22]:

- определить рамки и содержимое предметной области;
- создать единые требования к функционалу проектируемой ИС;
- разработать концептуальную модель ИС для её дальнейшей детализации в виде логических и физических моделей;

- подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

В диаграммы вариантов использования разрабатываемая ИС описывается в виде многочисленных сущностей и пользователей, которые взаимодействуют с ИС. Актеры с определенной ролью в проекте можно назвать любую сущность, которая взаимодействуют с информационной системой извне. Подобной сущностью быть как человек, так и техническое устройство. Use cases в первую очередь служит для описания веб-сервисов, которые ИС предоставляет пользователю. Диаграмма вариантов использования может описываться пояснительным текстом, который раскрывает особенность и семантику составляющих её компонентов.

Отдельный вариант использования обозначается на диаграмме эллипсом, внутри которого содержится его краткое название или имя в форме глагола с пояснительными словами.

Цель варианта использования заключается в том, чтобы определить законченный аспект или фрагмент поведения некоторой сущности без раскрытия её внутренней структуры. В качестве такой сущности может выступать система или любой элемент модели, который обладает собственным поведением.

Каждый вариант использования соответствует отдельному сервису, который предоставляет моделируемая сущность по запросу актера, то есть определяет способ применения этой сущности. Сервис, который инициализируется по запросу актера, представляет собой законченную неделимую последовательность действий.

Рисунок 2.4 – Диаграмма вариантов использования в рамках проекта.

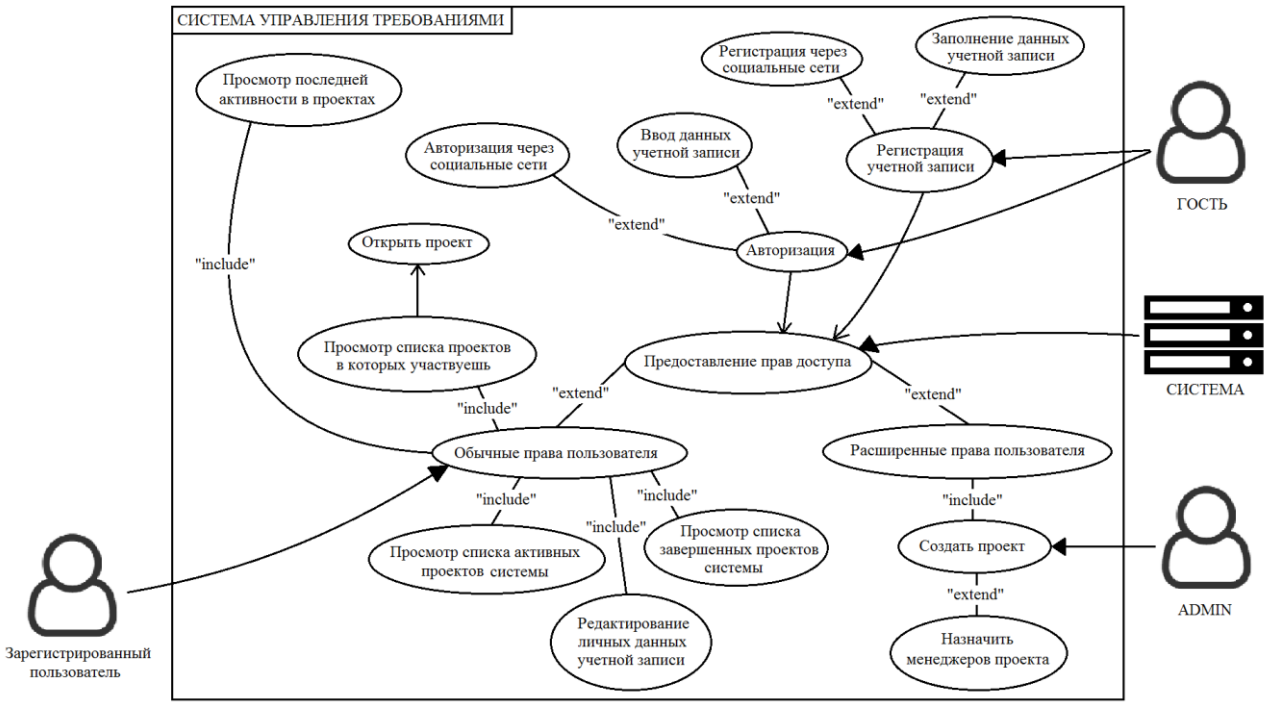

Рисунок 2.3 – Диаграмма вариантов использования.

Варианты использования могут применяться как для спецификации внешних требований к проектируемой системе, так и для спецификации функционального поведения уже существующей системы. Множество вариантов использования в целом должно определять все возможные стороны ожидаемого поведения системы. Кроме этого, варианты использования неявно устанавливают требования, определяющие, как актеры должны взаимодействовать с системой, чтобы иметь возможность корректно работать с предоставляемыми сервисами. Для удобства множество вариантов использования может рассматриваться как отдельный пакет.

Примерами вариантов использования могут являться следующие действия: проверка состояния текущего счета клиента, оформление заказа на покупку товара, получение дополнительной информации о кредитоспособности клиента, отображение графической формы на экране монитора и другие действия.

2.3 Диаграмма последовательности

Удобное средство для обозначения очередности следования друг за другом различных стимулов (сообщений), с помощью которых объекты взаимодействуют между собой. Например, когда нужно проработать буквально по шагам какой-то очень важный участок выполнения программы. Главный акцент - порядок и динамика поведения, т.е. как и в каком порядке происходят события. Отличие от диаграммы классов: Диаграмма классов дает статическую картинку, т.е. описание которое не меняется во время выполнения программы. Отличие от диаграммы коммуникаций (или как она раньше называлась colaboration): Диаграмма последовательности фокусирует наше внимание на очередности выполнения по времени, а диаграмма коммуникаций - на составляющих элементах [23]. Обычно нормальные люди стараются описывать одной диаграммой только один определенный кейс (UseCase, вариант использования), например: "оставить коммент к сообщению к записе", "стать автором редактирования записи" и

т.д. Диаграммы последовательности, которые описывают всю систему сразу, представляют из себя монстра, пожирающего внимание, сознание, силы, время и мозг разработчика. Итак, предлагаю рассмотреть простенькую диаграмму последовательности.

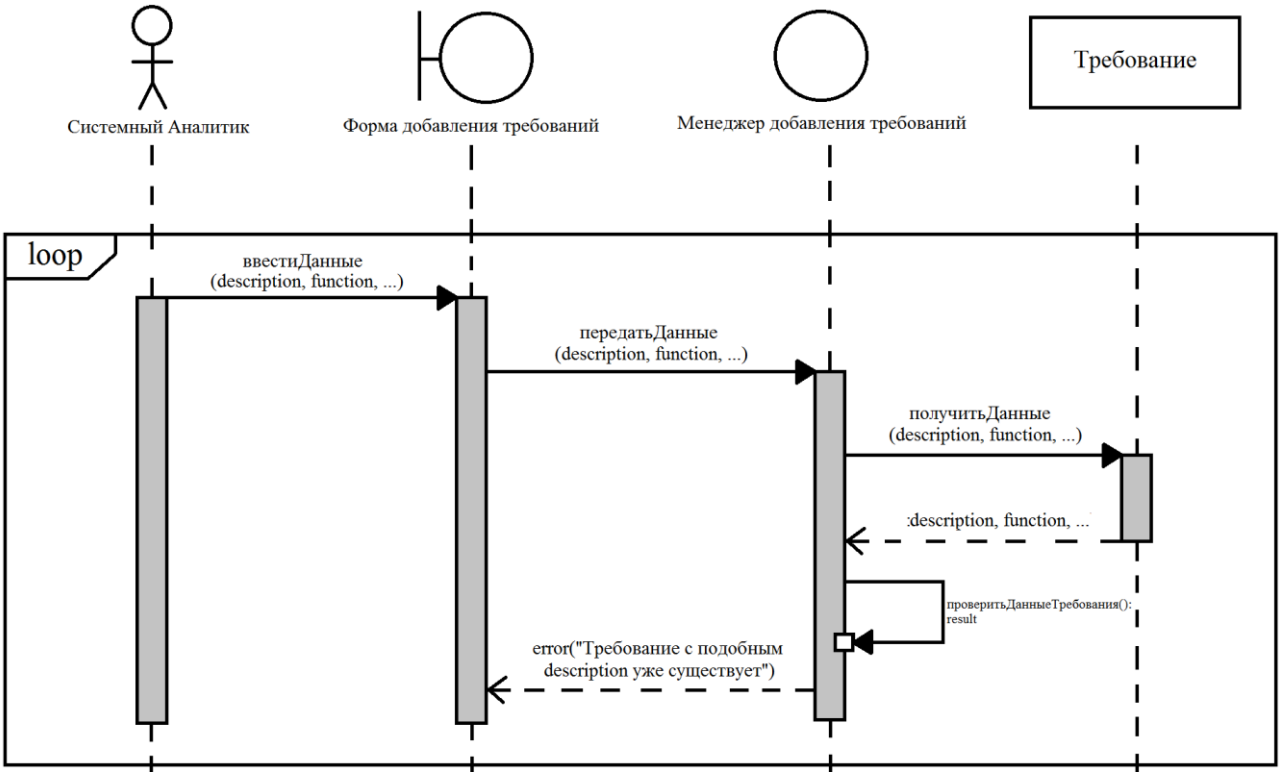

Рисунок 2.4 – Диаграмма последовательности для прецендента «Создать Требование».

2.4 Диаграмма классов

Диаграмма классов (англ. Static Structure Diagram) – диаграмма, демонстрирующая классы системы, их атрибуты, методы и взаимосвязи между ними.

Существует два вида:

- Статический вид диаграммы рассматривает логические взаимосвязи классов между собой;
- Аналитический вид диаграммы рассматривает общий вид и взаимосвязи классов, входящих в систему.

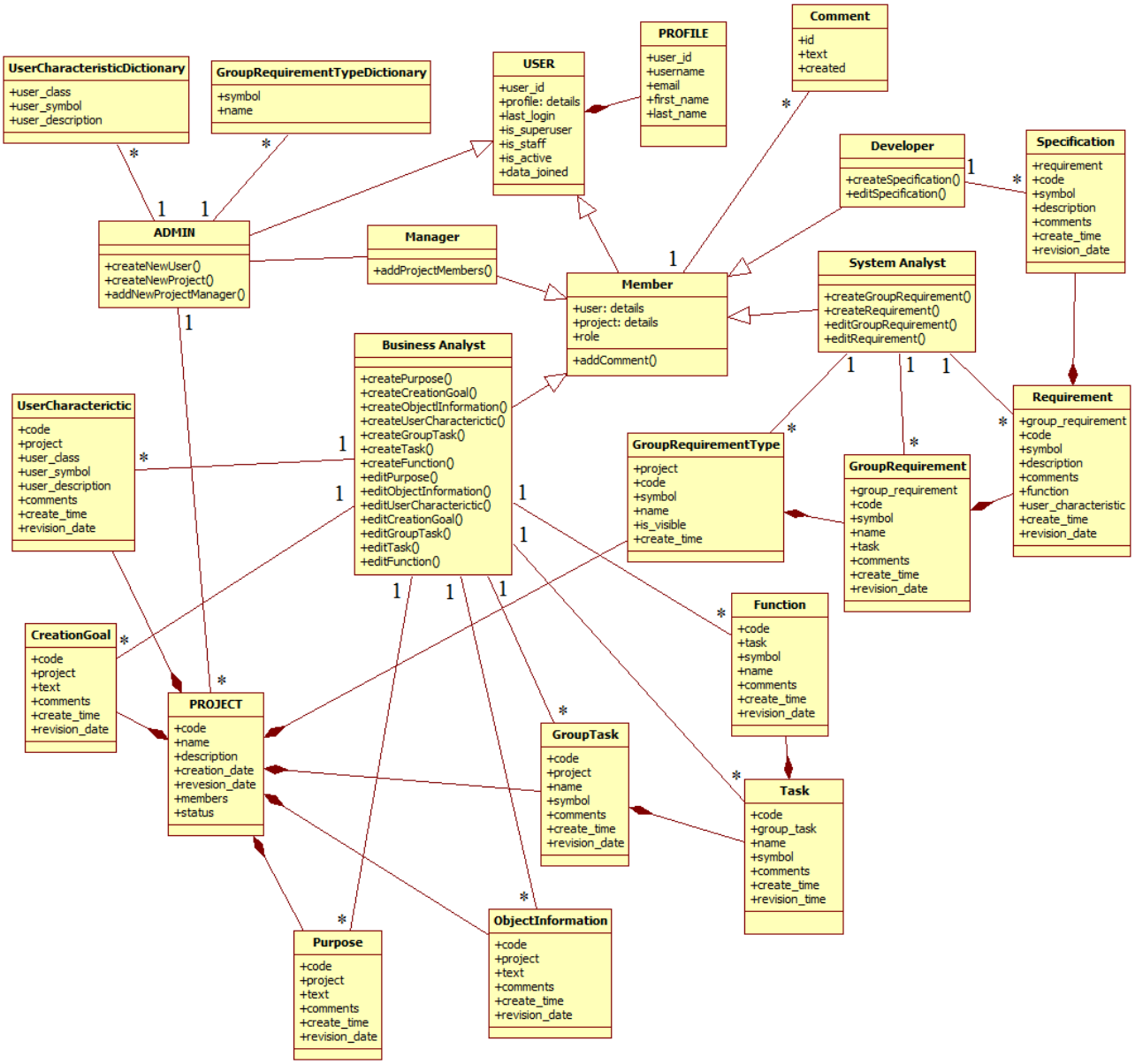

Рисунок 2.5 – Диаграмма классов.

На сегоднящний день присутствуют различные мнения на счет формирования диаграмм классов в зависимости от поставленных целей их дальнейшего использования:

- Первой точкой зрения является концептуальная идея. В данной точки зрения диаграмма классов описывает модель предметной области;
- Второй точкой зрения является спецификации. В данной точки зрения диаграмма классов используется при проектировании ИС;

- Третей точкой зрения является реализации. В данной точки зрения диаграмма классов состоит из классов, используемые в программном коде (при использовании объектно-ориентированных языков программирования).

2.4 Диаграмма развертывания

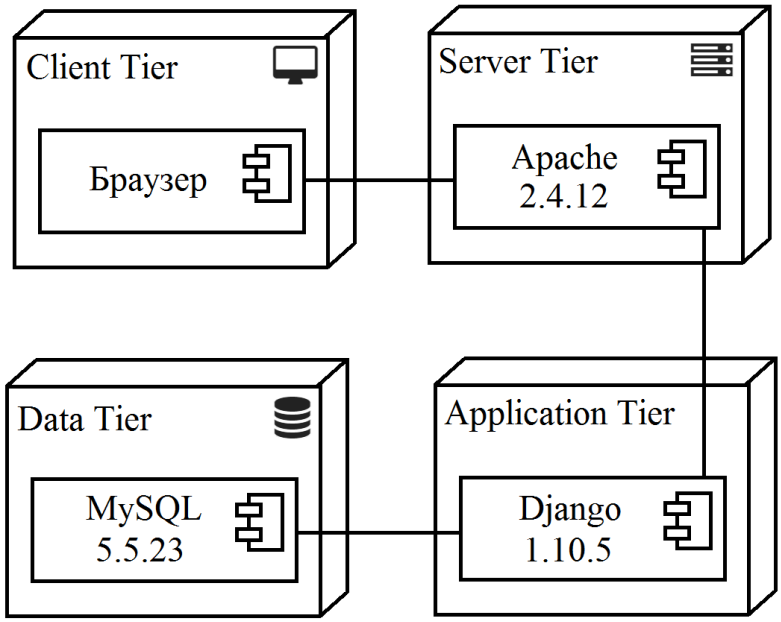

Рисунок 2.6 – Диаграмма развертывания.

Диаграмма развёртывания (анг. Deployment Diagram) в UML описывает физическое развертывание компанентов на узлах. Deployment Diagram указывает, какие аппаратные компоненты (если быть точнее узлы) существуют (например, веб-сервер, сервер базы данных, сервер приложения), какие программные компоненты (артефакты) работают на каждом узле.

При создании диаграммы развертывания преследовались следующие цели:

- определить и описать взаимосвязь компонентов ИС по её физическим узлам;
- указать физическую связь между узлами ИС на стадии её применения;

- на ранних стадиях проектирования определить узкие места ИС и настроить её топологию для достижения ожидаемой производительности.

СУТ использует клиент-серверную технологию. В качестве клиента выступает браузер пользователя, в качестве сервера – совокупность программно-аппаратных средств, необходимых для работы разработанной системы. На стороне клиента используется браузер, а на стороне сервера веб-сервер Apache, СУБД MySQL и веб-фреймворк Django на языке программирования Python. Диаграмма развертывания представлена на рисунке 2.6.

3 РЕАЛИЦИЯ СИСТЕМЫ УПРАВЛЕНИЯ ТРЕБОВАНИЯМИ

3.1 Серверная часть

Серверная логика веб-системы управления требованиями написана на языке программирования Python версии 3.5. За основу был взят веб-фреймворк Django версии 1.10.5.

Проект построенный на веб-фреймворке Django состоит из одного или нескольких так называемых приложений (анг. application). Каждое подобное приложение создается отчуждаемыми и подключаемыми, что позволяет использовать данные application в нескольких проектах. Это основная особенность архитектуры веб-фреймворка Django от его конкурентов. Одним из основных негласных правил веб-фреймворка является концепция DRY (анг. Don`t repeat yourself). Данная концепция разработки программного обеспечения, в первую очередь позволяет минимизировать повторения программного кода, особенно в информационных системах с многочисленными слоями абстрагирования.

Следует отметить, что в сравнение от другими веб-фреймворками, обработчики URL в Django настраиваются явно при помощи регулярных выражений, а не выводятся автоматически из структуры моделей и контроллеров.

Во время работы с реалиционной базой данных веб-фреймворк Django использует собственную ORM (Object Relational Mapping) из коробки, в которой модель данных описывается классами языка программирования Python. Далее на основе данных классов генерируется схема базы данных, что безусловно упращает процесс разработки.

Проект Django представляет собой настраиваемую пользователем среду разработки. Главные компоненты среды разработки Django следующие:

− Объектно-реляционное отображение (анг. object-relational mapping) для создания моделей данных;

− Детально прописанный и визуально оформленный интерфейс

администратора, нацеленный для конечных пользователей системы;

- Спроектированный механизм адресования (URL);
- Язык шаблонов, для дизайнеров;
- Система кэширования;

Для удобной работы с веб-фреймворком Django была использована интегрированная среда разработки для языка программирования Python – PyCharm. PyCharm дает возможность производить анализ кода, а также дает удобный, современный графический отладчик, инструмент для запуска тестовых сценариев и что немаловажно поддерживает веб-разработку на веб-фреймворке Django.

В качестве системы управления базой данных была использована реализионная база данных MySQL. При работе с данной СУБД было замечено, что данные и индексы хранятся отдельно, в разных файлах. После работы с данной СУБД были следующие наблюдения, что конструкция MySQL лучше подходит для очень широкого диапазона современных систем.

3.2 Клиентская часть

Клиентом разрабатываемой системы управления требованиями является браузер. Клиентская часть веб-системы написана с использованием HTML (от англ. HyperText Markup Language – «язык гипертекстовой разметки»), CSS (Cascading Style Sheets – каскадные таблицы стилей) и мультипарадигменным языком программирования JavaScript.

В качестве основного фреймворка для реализации интерфейса представлений был использован Bootstrap 4, который предоставляет разработчикам коллекцию инструментов для создания веб-страниц.

Также использованы JavaScript библиотеки JQuery и Ajax для реализации пользовательского интерфейса системы управления требованиями.

Библиотека JQuery позволяет реализовать взаимодействие JavaScript и HTML. Данная библиотека при разработке облегчала доступ к получению элементов DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими. Также библиотека jQuery предоставляет разработчикам API для работы с AJAX. Вместо прямого указания обработчика событий нажатия кнопки, управление передаётся JQuery, которая идентифицирует кнопки и затем преобразует его в обработчик события клика. Подобное разделение структуры и поведения именуется концепцией ненавязчивого JavaScript.

AJAX был использован при разработке системы управления требованиями для создания интерактивных пользовательских интерфейсов веб-страниц, позволяющих в «фоновом» обмене данными браузера с сервером Apache. В итоге, при обновлении информации страница в браузере не перезагружается полностью, и в целом система становятся быстрее и удобнее для конечного пользователя.

3.3 Архитектура MVC

Разработка системы управления требованиями велась в веб-фреймворке Django на языке программирования Python. Поддерживаемый организацией «Django Software Foundation» проект «Django» является свободным веб-фреймворком, использующий шаблон проектирования MVC (Model-View-Controller, Модель-Представление-Контроллер). Данный шаблон проектирования предоставляет три отдельных компонента как контроллер, представление и модель. Данная концепция позволяет настраивать каждый компонент независимо друг от друга.

Модель предоставляет данные и методы работы с ними: запросы в базу данных, проверка на корректность. Модель не зависит от представления – не знает, как данные визуализировать – и контроллера – не имеет точек взаимодействия с пользователем –, просто предоставляя доступ к данным и управлению ими.

Модель строится таким образом, чтобы отвечать на запросы, изменяя своё состояние, при этом может быть встроено уведомление «наблюдателей».

Представление отвечает за отображение данных модели данных пользователю, реагируя на изменения модели.

Контроллер обеспечивает «связи» между пользователем и системой. Контролирует и направляет данные от пользователя к системе и наоборот. Использует модель и представление для реализации необходимого действия.

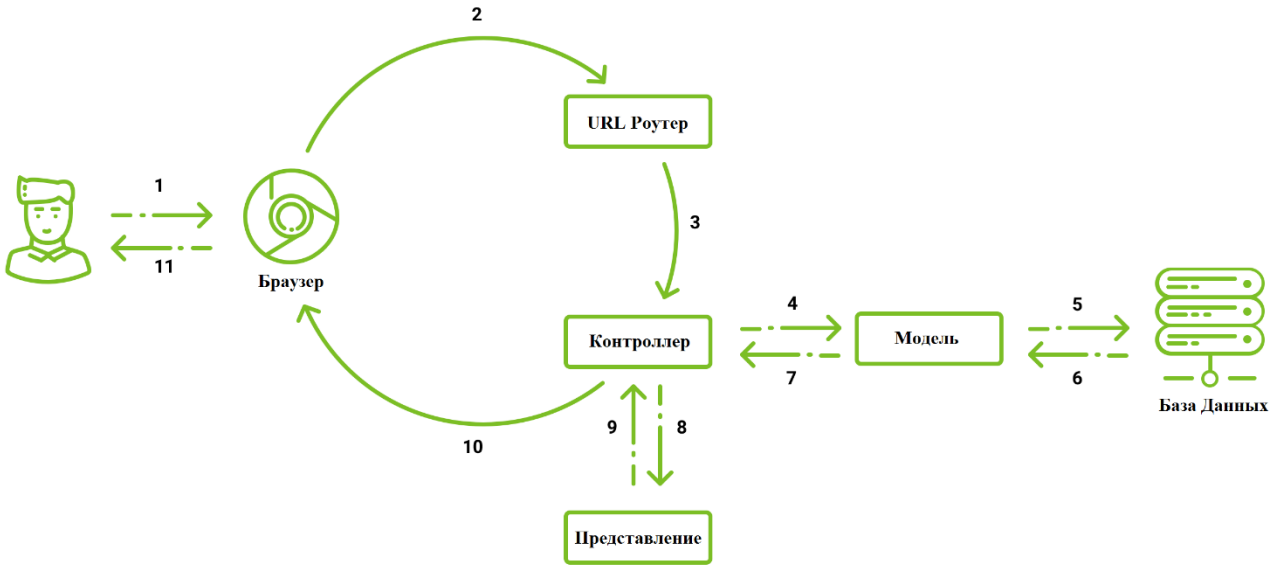

Рисунок 3.1 – Пример работы шаблона проектирования MVC.

Основной целью применения данной концепции в программном обеспечении системы управления требованиями преследовалось желание отделения бизнес-логики от её внешнего вида. Подобное четкое разделения повышает возможность повторного использования кода в написании программного кода. В проекте данная концепция нашло применения также в случаи, когда пользователь должен быть информирован одинаковыми данные одновременно в различных контекстах.

Весь код моделей данных приложен в разделе «Приложения А».

Далее детально расписаны несколько из моделей данных.

Модель данных «User» хранит данные о пользователе системы со следующими полями:

- username – Обязательное поле. Поле может содержит до 30 символов. Только буквы, цифры и подчеркивание.
- first_name – Необязательное поле. Поле может содержать до 30 символов.
- last_name – Необязательное поле, которое может содержать до 30 символов.
- email – Необязательное поле. Содержит Email адрес пользователя системы.
- password – Обязательное поле. Данное поле хранит хэш и метаданные о пароле. (Django не сохраняет открытый пароль.) Пароль может быть любой длинны и содержать любое количество символов.
- is_staff – Булево значение. Указывает имеет ли пользователь доступ к интерфейсу администратора.
- is_active – Булево значение. Указывает активный аккаунт или нет. Советуем устанавливать это поле в False вместо удаления; если в вашем приложение существуют связи с моделью пользователя, они не будет сломаны. Это поле не контролирует может пользователь войти или нет. Бэкэнд идентификации не проверяет значение is_active, поэтому если вы хотите ограничить вход для пользователей с is_active равным False, делайте это в собственном представлении входа. Однако, класс формы AuthenticationForm используемый в представлении login() выполняет проверку is_active, аналогично и метод проверки прав has_perm() и вход в интерфейс администратора Django. Все эти методы/функции вернут False для неактивных пользователей.
- is_superuser – Булево значение. Определяет, что пользователь имеет все права без явного их присвоения пользователю.
- last_login – Время последнего входа пользователя. По умолчанию устанавливается в текущее время.
- date_joined – Время создания аккаунта. По умолчанию устанавливается в текущее время при создании аккаунта.

Модель данных «Project» хранит информацию о проекте и содержит следующие поля данных:

- code – Поле является первичным кличем и принимает значения формата uuid4 (анг. Universally Unique Identifier). Главным назначение uuid4 является, то что оно позволяет различным ИС уникально идентифицировать данные без централизированной координации. Из чего следует вывод, что любой может реализовать uuid4 и применять его для идентификации записей с полной уверенности, что данный идентификатор непреднамеренно никогда не будет использован для чего-то ещё. В реализованном в рамках данной магистерской диссертации системе, значение для поля code генерировалась автоматически при создании объекта Project. За это отвечает пакет uuid4 языка программирования Python.
- name – Поле хранит текстовое значение названия проекта. Принимает максимальное количество символом равное 250.
- description – Большое текстовое поле для описание проекта.
- publication_date – Дата публикации проекта. Поле хранит дату и время, представленные объектом datetime.datetime Python.
- creation_date – Поле хранит информацию о времени и дате создания объекта Project. В системе значение поле генерируется автоматически за счет атрибута auto_now_add.
- reversion_date – Поле хранит информацию о дате и времени последнего обновления объекта Project.
- members – Поле используется для определения связки многое-ко многому с моделью Member, где хранится информация о участниках проекта. В проекте могут быть разные участники, также данные пользователи могут быть участниками разных проектов.
- status – Поле хранит информацию о статусе проекта. Может быть (public) открытым или закрытым (privite).

Модель данных «GroupRequirement» описывает данные о группе требований. Объекты, создаваемые на основе данной модели привязаны к типу

группы требования. Тогда как модель данных «Requirement» описывает данные о требовании и привязано к объекту группе требования. У каждого объекта «GroupRequirement» и «Requirement» есть поля данных:

- code – Первичный ключ созданной записи.
- symbol – Тестовое поле описывающее символ записи (группы требования или требования).
- description – Тестовое поле описывающее описание записи (группы требования или требования).
- comments – Поле связывает модели с моделью «Comment», связью многое ко многому. В поле хранятся информации о комментариях к определенной записи. Данное поле является важным, так как на их основе можно будет следить за причинами изменения в записях объектов.
- creation_time – Поле хранит информацию о времени и дате создания объекта. В системе значение поле генерируется автоматически за счет атрибута auto_now_add.

Поля данных, которые используются лишь в «GroupRequirement»:

task – Поле хранит ссылку на задачу, которую покрывает конкретная группа требования. Поле используется лишь для описания группы требования «[FA] Требования к алгоритмам работы функций».

Поля данных, которые используются лишь в «Requirement»:

function – Поле хранит ссылки на функции, которые покрывает конкретная задача. Также следует упомянуть, что данное поле не используется во всех требованиях.

user_characterictic – Поле хранит информацию о пользователе использующий функцию, которая покрывает данное требование. Поле используется лишь для описания требований: [V] Требование к вариантам использования, Требования к алгоритмам работы функций.

Представления в веб-фреймворке Django принято называть шаблонами (template), которые предоставляют удобный для дизайнера синтаксис для управления отображением информации пользователю. Django позволяет

динамически генерировать HTML. Самый распространенный подход – использование шаблонов. Шаблоны содержат статический HTML и динамические данные, рендеринг которых описан специальным синтаксисом.

Проект Django может использовать один или несколько механизмов создания шаблонов (или ни одного, если вы не используете шаблоны). Django предоставляет бэкенд для собственной системы шаблонов, которая называется - язык шаблонов Django (Django template language, DTL), и популярного альтернативного шаблонизатора Jinja2.

Django предоставляет стандартный API для загрузки и рендеринга шаблонов, независимисо от используемого бэкенда. Загрузка включает в себя поиск шаблона по названию и предварительную обработку, обычно выполняется загрузка шаблона в память. Рендеринг означает передачу данных контекста в шаблон и возвращение строки с результатом.

Основные представления системы управления требованиями указаны указаны в приложении «Б».

4 ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ

4.1. Потенциальные потребители программного обеспечения

Разработанное в рамках данной магистерской диссертации программное обеспечение системы управления требованиями, в первую очередь, предназначено для IT компаний разрабатывающие информационные системы. Веб-система позволит автоматизировать рутинный процесс составления технических требований и заданий к разрабатываемым информационным системам. Веб-системой предусмотрена возможность изменения данных, обсуждения с помощью комментариев любых изменений между участниками проекта и просмотр истории изменений, что в совокупности обеспечит единое виденье разрабатываемой ИС у заинтересованных лиц.

Также веб-система может быть полезна студентам высших учебных заведений для учебных нужд при проектной деятельности программных продуктов, так как веб-система позволяет структурировать данные и наглядно увидеть трассировку (взаимосвязь) данных.

4.2 Организация и планирование работ

Одной из составляющих успешной реализации проекта служит рациональное планирование занятости каждого из его участников, а также определение сроков выполнения определенных этапов работы над проектом.

В данном подразделе составляется список проводимых работ, определяются их исполнители и продолжительность. Так как число исполнителей не превышает двух, линейный график работ является наиболее удобным и компактным способом представления данных планирования.

Таблица 4.1 – Перечень работ и продолжительность их выполнения

Этапы работы	**Исполнители**	**Загрузка исполнителей**
Постановка целей и задач	НР	НР – 100%
Составление и утверждение технического задания	НР, С	НР – 90% С – 10%
Подбор и изучение материалов по тематике	НР, С	НР – 10% С – 90%
Анализ существующих на рынке аналогов	С	С – 100%
Составление календарного плана	НР, С	НР – 70% С – 30%
Проектирование архитектуры веб-системы	НР, С	НР – 10% С – 90%
Разработка системы управления требованиями в веб-фреймворке Django	С	С – 100%
Улучшение производительности веб-системы за счет оптимизации кода	НР, С	НР – 10% С – 90%
Тестирование конечного программного обеспечения системы управления требованиями	НР, С	НР – 30% С – 70%
Оформление пояснительной записки	С	С – 100%
Подведение итогов	НР, С	НР – 60% С – 40%

Примечание к таблице 4.1: НР — научный руководитель; С — студент.

4.2.1 Продолжительность этапов работ

Расчет продолжительности этапов работ осуществляется двумя методами:

- технико-экономическим;
- опытно-статистическим.

Для расчета ожидаемого значения продолжительности работ $t_{ож}$ применяются две оценки: $t_{\min}$ и $t_{\max}$ (метод двух оценок).

$$t_{ож} = \frac{3 \cdot t_{\min} + 2 \cdot t_{\max}}{5}, \qquad (4.1)$$

где $t_{\min}$ – минимальная трудоемкость работ, чел/дн;

$t_{\max}$ – максимальная трудоемкость работ, чел/дн.

Для выполнения перечисленных в таблице 4.1 работ требуются специалисты: научный руководитель (НР) и студент (С).

Для построения линейного графика рассчитывается длительность этапов в рабочих днях, а затем осуществляется её перевод в календарные дни. Расчёт продолжительности выполнения каждого этапа в рабочих днях ($T_{РД}$) выполняется по формуле:

$$T_{РД} = \frac{t_{ожс}}{K_{ВН}} \cdot K_{Д}, \quad (4.2)$$

где $t_{ожс}$ – продолжительность работы, дн.;

$K_{ВН}$ – коэффициент выполнения работ ($K_{ВН} = 1$);

$K_{Д}$ – коэффициент, учитывающий дополнительное время на компенсацию непредвиденных задержек и согласование работ ($K_{Д} = 1,2$).

Расчёт продолжительности этапа в календарных днях осуществляется по формуле:

$$T_{КД} = T_{РД} \cdot T_{К}, \quad (4.3)$$

где $T_{КД}$ – продолжительность выполнения этапа в календарных днях;

$T_{РД}$ – продолжительность выполнения этапа в рабочих днях;

$T_{К}$ – коэффициент календарности.

Коэффициент календарности рассчитывается по формуле:

$$T_{К} = \frac{T_{КАЛ}}{T_{КАЛ} - T_{ВД} - T_{ПД}}, \quad (4.4)$$

где $T_{КАЛ}$ – календарные дни, $T_{КАЛ} = 365$;

$T_{ВД}$ – выходные дни, $T_{ВД} = 52$;

$T_{ПД}$ – праздничные дни, $T_{ПД} = 10$.

Подставив значения в формулу 4.4, получим следующий результат:

$T_{К} = \frac{365}{365-52-10} = 1,205.$

В таблице 4.2 приведена длительность этапов работ и число исполнителей, занятых на каждом этапе

Таблица 4.2 – Временные показатели проведения научного исследования

Этап	**Исполнители**	**Продолжительность работ, дни**			**Длительность работ, чел/дн.**			
					$T_{РД}$		$T_{КД}$	
		t_{min}	t_{max}	$t_{ож}$	**НР**	**С**	**НР**	**С**
Постановка целей и задач	НР	2	3	2.4	2.88	0	3.47	0
Составление и утверждение технического задания	НР, С	2	3	2.4	2.59	0.29	3.12	0.34
Подбор и изучение материалов по тематике	НР, С	4	6	4.8	0.576	5.18	0.69	6.24
Анализ существующих на рынке аналогов	С	4	6	4.8	0	5.76	0	6.9
Составление календарного плана	НР, С	2	3	2.4	2.016	0.864	2.42	1.04
Проектирование архитектуры веб-системы	НР, С	10	12	10.8	1.296	11.664	1.56	14.05

Этап	**Исполнители**	**Продолжительность работ, дни**			**Длительность работ, чел/дн.**			
					$T_{РД}$		$T_{КД}$	
		t_{min}	t_{max}	$t_{ож}$	**НР**	**С**	**НР**	**С**
Разработка системы управления требованиями в веб-фреймворке Django	С	10	12	10.8	0	12.96	0	15.61
Улучшение производительности веб-системы за счет оптимизации кода	НР, С	10	12	10.8	1.2	11.66	1.446	14
Тестирование конечного программного обеспечения системы управления требованиями	НР, С	7	9	7.8	2.808	6.552	3.38	7.89
Оформление пояснительной записки	С	3	4	3.4	0	4.08	0	4.9
Проверка работы	НР, С	3	4	3.4	2.448	1.632	2.9	1.96
Итого:		**57**	**74**	**63.8**	**15.8**	**60.48**	**18,96**	**72,93**

Таблица 4.3 – Календарный план-график проведения работ

Этап	$T_{КД}$ **НР**	$T_{КД}$ **С**	**Февраль**			**Март**			**Апрель**			**Май**			**Июнь**		
			1	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**3**
Постановка целей и задач	3.47	0															
Составление и утверждение технического задания	3.12	0.34															
Подбор и изучение материалов по тематике	0.69	6.24															
Анализ существующих на рынке аналогов	0	6.9															
Составление календарного плана	2.42	1.04															
Проектирование архитектуры веб-системы	1.56	14.05															
Разработка системы управления требованиями в веб-фреймворке Django	0	15.61															
Улучшение производительности веб-системы за счет оптимизации кода	1.446	14															
Тестирование конечного программного обеспечения системы управления требованиями	3.38	7.89															
Оформление пояснительной записки	0	4.9															
Проверка работы	2.9	1.96															

Примечание к таблице 7.3: НР − ; С −

4.2.2 Расчет накопления технической готовности

В данном разделе производится оценка текущих результатов работы над проектом. Величина накопления готовности работы показывает, на сколько процентов по окончании текущего этапа выполнен общий объем работ по проекту в целом.

Степень готовности определяется формулой:

$$СГ_i = \frac{TP_i^H}{TP_{общ}} = \frac{\sum_{k=1}^{i} TP_k}{TP_{общ}} = \frac{\sum_{k=1}^{i}\sum_{j=1}^{m} TP_{km}}{\sum_{k=1}^{I}\sum_{j=1}^{m} TP_{km}} \quad (4.5)$$

где $TP_{общ}$ – общая трудоемкость проекта;

TP_i (TP_k) – трудоемкость *i*-го (*k*-го) этапа проекта, $i = \overline{1,I}$;

TP_i^H – накопленная трудоемкость *i*-го этапа проекта по его завершении;

TP_{ij} (TP_{kj}) – трудоемкость работ, выполняемых *j*-м участником на *i*-м этапе.

Таблица 4.4 – Нарастание технической готовности работы

Этап	**ТР$_i$,%**	**СГ$_i$,%**
Постановка целей и задач	2,3	2,3
Составление и утверждение технического задания	2,3	4,6
Подбор и изучение материалов по тематике	7,87	12,47
Анализ существующих на рынке аналогов	2,3	14,78
Составление календарного плана	13,44	28,21
Проектирование архитектуры веб-системы	11,52	39,73
Разработка системы управления требованиями в веб-фреймворке Django	13,44	53,16
Улучшение производительности веб-системы за счет оптимизации кода	11,52	64,68

Тестирование конечного программного обеспечения системы управления требованиями	11,52	76,19
Оформление пояснительной записки	18,24	94,43
Проверка работы	5,57	100

4.3 Расчёт сметы затрат на выполнение проекта

Состав затрат на научно-исследовательскую работу в рамках магистерской диссертации состоит из всех расходов, необходимых для реализации комплекса работ, составляющих содержание данного исследования. Так как научно-исследовательская работа проводилась на домашнем компьютере, без аренды помещения и в программном обеспечении с бесплатной студенческой лицензией расчет сметной стоимости производится по следующим статьям затрат:

- материалы и покупные изделия;
- заработная плата;
- социальный налог;
- расходы на электроэнергию (без освещения);
- амортизационные отчисления;
- оплата услуг связи;
- прочие (накладные расходы) расходы.

4.3.1 Расчёт затрат на материалы

К данной статье расходов относится стоимость всех материалов, расходуемых непосредственно в процессе выполнения работ.

Таблица 4.5 – Расчёт затрат на материалы

Наименование	**Единица измерения**	**Количество**	**Цена за ед., руб**	**Затраты на материалы, ($З_м$), руб.**
Бумага формата А4 для принтера	Упаковка	1	250	250
Картридж для принтера	Штука	1	650	650
Шариковая ручка	Штука	1	9	9
Итого				909

Транспортно-заготовительные расходы (ТРЗ) составляют 5% от отпускной цены материалов. Расходы на материалы с учётом ТРЗ:

$C_{MAT} = 889 \cdot 1{,}05 = 954{,}45$ руб.

7.3.2 Расчёт заработной платы

Данная статья расходов включает заработную плату научного руководителя и студента, а также премии, входящие в фонд заработной платы. Расчет основной заработной платы выполняется на основе трудоёмкости выполнения каждого этапа и величины месячного оклада исполнителя.

Величина месячного оклада научного руководителя (МОНР) получена из открытых данных, размещенных на официальном сайте Национального исследовательского Томского политехнического университета. Величина месячного оклада инженеров (МОИ) берется как месячный оклад инженера кафедры.

Основной расчет фонда заработной платы выполняется по формуле:

$$ЗП_{дн-т} = МО/N, \quad (4.7)$$

где МО – месячный оклад, руб.;

N – количество рабочих дней в месяц, при шестидневной рабочей неделе – $N = 24{,}91$, а при пятидневной рабочей неделе – $N = 20{,}58$.

Среднедневная заработная плата научного руководителя равна:

$$\text{ЗП}_{\text{дн}-\text{т}} = \frac{26\,300}{24{,}91} = 1\,055{,}8\ \frac{\text{руб.}}{\text{раб. день}}.$$

А среднедневная тарифная заработная плата инженеров равна:

$$\text{ЗП}_{\text{дн}-\text{т}} = \frac{14874{,}45}{20{,}58} = 722{,}76\ \frac{\text{руб.}}{\text{раб. день}}.$$

Затраты времени по каждому исполнителю в рабочих днях взяты из таблицы 4.2. Для перехода от тарифной суммы заработка исполнителя, связанной с участием в проекте, к соответствующему полному заработку необходимо будет тарифную сумму заработка исполнителя, связанной с участием в проекте умножить на интегральный коэффициент. Интегральный коэффициент находится по формуле:

$$\text{К}_{\text{и}} = \text{К}_{\text{пр}} \cdot \text{К}_{\text{доп.ЗП}} \cdot \text{К}_{\text{р}}, \qquad (4.8)$$

где $\text{К}_{\text{пр}}$ – коэффициент премий, $\text{К}_{\text{пр}} = 1{,}1$;

$\text{К}_{\text{доп.ЗП}}$ – коэффициент дополнительной зарплаты, при шестидневной рабочей неделе $\text{К}_{\text{доп.ЗП}} = 1{,}188$, а при пятидневной рабочей неделе $\text{К}_{\text{доп.ЗП}} = 1{,}113$;

$\text{К}_{\text{р}}$ – коэффициент районной надбавки, $\text{К}_{\text{р}} = 1{,}3$.

Таблица 4.6 – Затраты на заработную плату

Исполнитель	Оклад, руб./мес	$\text{ЗП}_{\text{дн}-\text{т}}$, руб./раб.день	Затраты времени, раб.дни	Коэффициент	Фонд з/платы, руб.
НР	26300	1055,8	27	1,699	48432,71
С	14874,45	722,76	99	1,59	113769,65
Итого:					162202,36

4.3.3 Расчет отчисления на социальные нужды

Взнос в социальные фонды установлен в размере 30,2% от заработной платы. Размер взноса рассчитываются по формуле:

$$С_{соц} = С_{ЗП} \cdot 0{,}302, \quad (4.9)$$

где $С_{ЗП}$ – размер заработной платы.

Подставив необходимые значения в формулу 5.10 получим:

$$С_{соц} = 162202{,}36 \cdot 0{,}302 = 48985{,}11 \text{ руб.}$$

4.3.4 Расчет затрат на электроэнергию

Затраты на электроэнергию рассчитываются по формуле:

$$С_{эл.об.} = Р_{об} \cdot t_{об} \cdot Ц_{э}, \quad (4.10)$$

где $Р_{об}$ – мощность, потребляемая оборудованием, кВт;

$t_{об}$ –время работы оборудования, час;

$Ц_{э}$ – тариф на 1 кВт·час. Для ТПУ, $Ц_{э} = 5{,}8$ руб./кВт · час.

Время работы оборудования вычисляется на основе итоговых данных таблицы 5.3 для инженера ($Т_{рд}$) из расчета, что продолжительность рабочего дня равна 8 часов.

$$t_{об} = Т_{РД} \cdot K_t, \quad (4.11)$$

где K_t – коэффициент использования оборудования по времени, $K_t = 0{,}9$.

Мощность, потребляемая оборудованием, определяется по формуле:

$$Р_{об} = Р_{ном} \cdot K_C, \quad (4.12)$$

где K_C – коэффициент загрузки;

$Р_{ном}$ – номинальная мощность оборудования, кВт. Для технологического оборудования малой мощности $K_С = 1$.

Таблица 4.7 – Затраты на электроэнергию технологическую

Наименование оборудования	Время работы оборудования $t_{ОБ}$, час	Потребляемая мощность $Р_{ОБ}$, кВт	атраты $Э_{ОБ}$, руб.
Персональный компьютер инженера	791,6	0,09	372,89
Итого:			**372,89**

4.3.5 Расчет амортизационных расходов

Для расчета амортизационных расходов используется формула:

$$С_{АМ} = \frac{Н_А \cdot Ц_{ОБ} \cdot t_{рф} \cdot n}{F_Д}, \quad (4.13)$$

где $Н_А$ – годовая норма амортизации единицы оборудования;

$Ц_{ОБ}$ – балансовая стоимость единицы оборудования с учетов ТЗР, стоимость ПК инженера – 20 500 руб.;

$t_{рф}$ – фактическое время работы оборудования в ходе выполнения проекта, $t_{рф} = 98{,}95 \cdot 8 = 791{,}6$ часов;

n – число задействованных однотипных единиц оборудования;

$F_Д$ – действительный годовой фонд времени работы соответствующего оборудования, $F_Д = 298 * 8 = 2384$ часа.

$Н_А$ определяется по формуле:

$$Н_А = \frac{1}{СА}, \quad (4.12)$$

где СА – срок амортизации, который можно получить из постановления правительства РФ «О классификации основных средств, включенных в амортизационные группы» Для электронно-вычислительной техники СА свыше 2 лет до 3 лет включительно. В данной работе примем СА=2,5 года. Тогда

$$Н_А = \frac{1}{2,5} = 0,4.$$

$$С_{АМ}(ПК) = \frac{0,4 \cdot 20\,500 \cdot 791,6 \cdot 1}{2384} = 2722,78 \text{ руб}$$

В результате начислено амортизации 2722,78 рублей.

4.3.6 Расчет расходов на услуги связи

Расходы на услуги связи определены наличием подключения к сети Интернет на компьютере, использованном в данной работе.

Ежемесячная оплата, согласно тарифу TPUnet, составляет 350 рублей. В соответствии с таблицей 7.3, трудоемкость выполняемой задачи составляет четыре календарных месяца. Таким образом, сумма расходов на услуги связи составляет 4*350 = 1750 руб. Общая сумма расходов $С_{св} = 1400$

4.3.7 Расчет прочих расходов

Прочие расходы следует принять равными 10% от суммы всех предыдущих расходов. Они находятся по формуле:

$$С_{проч} = (С_{мат} + С_{ЗП} + С_{соц} + С_{эл.об.} + С_{АМ} + С_{св}) \cdot 0,1, \quad (4.13)$$

Где $С_{мат}$ – расходы на материалы, руб.;

$С_{ЗП}$ – основная заработная плата, руб.;

$С_{соц}$ – расходы на единый социальный налог, руб.;

$С_{эл.об.}$ – расходы на электроэнергию, руб.;

$С_{АМ}$ – амортизационные расходы, руб.;

$С_{св}$ – расходы на услуги связи, руб.

Подставив полученные выше результаты, получим:

$C_{проч}$ = (954,45 + 162202,36 + 48985,11+ 372,89 + 2722,98 + 1400) ·0,1 = 21663,779 рублей.

4.3.8 Расчет общей себестоимости разработки

Проведя расчет по всем статьям сметы затрат на разработку, можно определить общую себестоимость проекта.

Таблица 4.8 – Смета затрат на разработку проекта

Статья затрат	**Условное обозначение**	**Сумма, руб.**
Материалы и покупные изделия	$C_{мат}$	954,45
Основная заработная плата	$C_{ЗП}$	162202,36
Отчисления в социальные фонды	$C_{соц}$	48985,11
Расходы на электроэнергию	$C_{эл.об.}$	372,89
Амортизационные отчисления	$C_{АМ}$	2722,98
Расходы на услуги связи	$C_{св}$	1400
Прочие расходы	$C_{проч}$	21661,68
Итого:	**238299,48**	

Таким образом, затраты на разработку составили С = 238299,48 рублей.

4.3.9 Расчёт прибыли

Прибыль следует принять в размере 20% от полной себестоимости разработки. Прибыль составляет:

$238299{,}48 * 0{,}2 = 47659{,}898$, руб.

4.3.10 Расчёт НДС

НДС составляет 18% от суммы затрат на разработку и прибыли:

$(238299{,}48 + 47659{,}898) * 0{,}18 = 51472{,}688$, руб.

4.4 Оценка экономической эффективности

Выполнение научно-исследовательских работ оценивается уровнями достижения экономического, научного, научно-технического и социального эффектов.

Для итоговой оценки результатов проекта в зависимости от поставленных целей в качестве критерия эффективности принимается один из видов эффекта, а остальные используются в качестве дополнительных характеристик.

На данном этапе внедрение нет возможности оценить экономический эффект в количественных показателях. Так как данная разработка является моделью для дальнейшей модификации при решении конкретно поставленной модели. Следовательно, в дальнейшем необходимо рассчитывать данный показатель исходя из заявленных параметров и условий. Поэтому в качестве критерия эффективности проекта оценим научно-технический уровень НИР.

4.4.1 Оценка научно-технического уровня НИР

Научно-технический уровень характеризует влияние проекта на уровень и динамику обеспечения научно-технического прогресса в данной области. Для оценки научной ценности, технической значимости и эффективности, планируемых и выполняемых НИР, используется метод балльных оценок. Каждому фактору по принятой шкале присваивается определенное количество баллов. Обобщенная оценка проводится по сумме баллов по всем показателям. На её основе делается вывод о целесообразности НИР.

Интегральный показатель научно технического уровня НИР определяется по формуле:

$$I_{\text{НТУ}} = \sum_{i=1}^{3} R_i \cdot n_i, \qquad (4.13)$$

где $I_{\text{НТУ}}$ – интегральный индекс научно-технического уровня;

R_i – весовой коэффициент i-го признака научно-технического эффекта;

n_i – количественная оценка i-го признака научно-технического эффекта, в баллах.

Таблица 4.9 – Весовые коэффициенты признаков НТУ

Признаки научно-технического эффекта НИР	**Характеристика признака НИР**	R_i
Уровень новизны	Систематизируются и обобщаются сведения, определяются пути дальнейших исследований	0,40
Теоретический уровень	Разработка способа	0,10
Возможность реализации	Время реализации в течение первых лет	0,50

Баллы для оценок уровня новизны, теоретического уровня и возможности реализации приведены в таблицах 4.10 – 4.12.

Таблица 4.10 – Баллы для оценки уровня новизны

Уровень новизны	**Характеристика уровня новизны**	**Баллы**
Принципиально новая	Новое направление в науке и технике, новые факты и закономерности, новая теория, вещество, способ	8–10
Новая	По-новому объясняются те же факты, закономерности, новые понятия дополняют ранее полученные результаты	5–7
Относительно новая	Систематизируются, обобщаются имеющиеся сведения, новые связи между известными	2–4
Не обладает новизной	Результат, который ранее был известен	0

Таблица 4.11 – Баллы значимости теоретических уровней

Теоретический уровень полученных результатов	**Баллы**
Установка закона, разработка новой теории	10
Глубокая разработка проблемы, многоспектральный анализ взаимодействия между факторами с наличием объяснений	8
Разработка способа (программного обеспечения)	6
Элементарный анализ связей между фактами (наличие гипотезы, объяснения версии, практических рекомендаций)	2
Описание отдельных элементарных факторов, изложение наблюдений, опыта, результатов измерений	0,5

Таблица 4.12 – Возможность реализации результатов по времени

Время реализации	**Баллы**
В течение первых лет	10
От 5 до 10 лет	4
Свыше 10 лет	2

В таблице 4.13 указано соответствие качественных уровней НИР значениям показателя, рассчитываемого по формуле 4.11).

Таблица 4.13 – Оценка научно-технического уровня НИР

Фактор НТУ	**Значимость**	**Уровень фактора**	**Выбранный балл**	**Обоснование выбранного балла**
Уровень новизны	0,4	Новая	5	Облегчит разработку других структур
Теоретический уровень	0,1	Разработка способа	6	Описание принципа работы системы с заданными параметрами
Возможность реализации	0,5	В течение первых лет	10	Быстрая разработка с помощью различных инструментальных средств

Интегральный показатель научно-технического уровня составляет:

$I_{НТУ} = 0{,}4 \cdot 5 + 0{,}1 \cdot 6 + 0{,}5 \cdot 10 = 7.6$.

Таблица 4.14 – Оценка уровня научно-технического эффекта

Уровень НТЭ	**Показатель НТЭ**
Низкий	1 – 4
Средний	4 – 7
Высокий	8 – 10

Таким образом, научно-исследовательская работа имеет средний уровень научно-технического эффекта.

5 СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ

Научно-исследовательская деятельность выполнялась в помещении кафедры «Информационные системы и технологии» десятого корпуса ТПУ в кабинете 402А. Помещение оснащено видео-дисплейными терминалами (ВДТ), персональными электронно-вычислительными машинами (ПЭВМ), компьютерными столами, стульями, столом для коллективной работы, огнетушителями, кондиционером, противопожарной сигнализацией и датчиками дыма.

Для обеспечения производственной безопасности необходимо проанализировать воздействия на человека вредных и опасных производственных факторов, которые могут возникать при разработке проекта.

Производственный фактор считается вредным, если воздействие этого фактора на человека может привести к его заболеванию. Производственный фактор считается опасным, если его воздействие может привести к травме.

Все производственные факторы классифицируются по группам элементов: физические, химические, биологические и психофизические. Для данной работы целесообразно рассмотреть физические и психофизические вредные и опасные факторы производства, характерные для рабочей зоны программиста, разработчика приложения, пользователя. Выявленные факторы представлены в таблице 5.1.

Таблица 5.1 – Вредные и опасные производственные факторы при выполнении работ за ПЭВМ

Источник фактора, наименование видов работ	Факторы (по ГОСТ 12.0.003-74)		Нормативные документы
	Вредные	Опасные	
1) Работа за ПК	1) Недостаточная освещенность рабочей зоны; 2) Умственное перенапряжение; 3) Монотонный режим работы.	1) Опасность поражения электрическим током; 2) Опасность возникновения пожара.	1) СН 2.2.4/2.1.8.562-96; 2) СанПиН 2.2.4.548-96; 3) СанПиН 2.2.2/2.4.1340-03; 4) СП 52.13330.2011; 5) ГОСТ Р 12.1.019-2009 ССБТ; 6) СНиП 21-01-97.

5.1.1 Вредные производственные факторы

5.1.1.1 Недостаточная освещенность рабочей зоны

Недостаточная освещенность рабочей зоны является вредным производственным фактором, возникающим при работе с ПЭВМ, уровни которого регламентируются СП 52.13330.2011.

Причиной недостаточной освещенности являются недостаточность естественного освещения, недостаточность искусственного освещения, пониженная контрастность.

Работа с компьютером подразумевает постоянный зрительный контакт с дисплеем ПЭВМ и занимает от 80 % рабочего времени. Недостаточность освещения снижает производительность труда, увеличивает утомляемость и количество допускаемых ошибок, а также может привести к появлению профессиональных болезней зрения.

Разряд зрительных работ программиста и оператора ПЭВМ относится к разряду III и подразряду Г (работы высокой точности). В таблице 5.2 представлены нормативные показатели искусственного освещения при работах заданной точности.

Таблица 5.2 – Требования к освещению помещений промышленных предприятий для операторов ПЭВМ

Характеристика зрительной работы	Наименьший или эквивалентный размер объекта различения, мм	Разряд зрительной работы	Подразряд зрительной работы	Контраст объекта с фоном	Характеристика фона	Искусственное освещение		
						Освещённость, лк		
						При системе комбинированного освещения		При системе общего освещения
						всего	В том числе от общего	
Высокой точности	0,264	III	г	Средний, большой	Светлый, средний	400	200	200

Для создания и поддержания благоприятных условий освещения для операторов ПЭВМ, их рабочие места должны соответствовать санитарно-эпидемиологическим правилам СанПиН 2.2.2/2.4.1340-03. Рабочее помещение должно иметь естественное и искусственное освещение, соответствующее показателям, представленным в таблице 5.6. Для рассеивания естественного

освещения следует использовать жалюзи на окнах рабочих помещений. В качестве источников искусственного освещения должны быть использованы люминесцентные лампы, лампы накаливания – для местного освещения.

5.1.1.2 Умственное перенапряжение

Умственное перенапряжение вызывается большим объемом информации, которую надо анализировать, и чтобы избежать умственного перенапряжения необходимо устраивать небольшие перерывы в течение рабочего дня продолжительностью не более 5 минут.

При умственной работе, по сравнению с физической работой потребление кислорода мозгом увеличивается в 15-20 раз. Если для умственной работы требуется значительное нервно-эмоциональное напряжение, то возможны значительные изменения кровяного давления, пульса. Длительная работа этого характера может привести к заболеванию, в частности сердечно-сосудистым и некоторым другим заболеваниям.

5.1.1.3 Монотонный режим работы

При работе с ПЭВМ основным фактором, влияющим на нервную систему программиста или пользователя, является огромное количество информации, которое он должен воспринимать. Это является сложной задачей, которая очень сильно влияет на сознание и психофизическое состояние из-за монотонности работы. Поэтому меры, позволяющие снизить воздействие этого вредного производственного фактора, которые регулируются СанПиН 2.2.2/2.4.1340-03, являются важными в работе оператора ПЭВМ. Они позволяют увеличить производительность труда и предотвратить появление профессиональных болезней.

Организация работы с ПЭВМ осуществляется в зависимости от вида и категории трудовой деятельности. Виды трудовой деятельности разделяются на

3 группы: группа А – работа по считыванию информации с экрана с предварительным запросом; группа Б – работа по вводу информации; группа В – творческая работа в режиме диалога с ПЭВМ. Работа программиста-разработчика рассматриваемой в данной работе относится к группам А и Б, в то время, как деятельность пользователя приложения относится к группе В. Категории трудовой деятельности, различаются по степени тяжести выполняемых работ. Для снижения воздействия рассматриваемого вредного фактора предусмотрены регламентированные перерывы для каждой группы работ – таблица 5.3.

Таблица 5.3 – Суммарное время регламентированных перерывов в зависимости от продолжительности работы, вида категории трудовой деятельности с ПЭВМ

Категория работы с ПЭВМ	**Уровень нагрузки за рабочую смену при видах работ с ПЭВМ**			**Суммарное время регламентированных перерывов, мин.**	
	группа А, количество знаков	**группа Б, количество знаков**	**группа В, ч**	**при 8-часовой смене**	**при 12-часовой смене**
I	до 20 000	до 15 000	до 2	50	80
II	до 40 000	до 30 000	до 4	70	110
III	до 60 000	до 40 000	до 6	90	140

Для предупреждения преждевременной утомляемости пользователей ПЭВМ рекомендуется организовывать рабочую смену путем чередования работ с использованием ПЭВМ и без него. В случаях, когда характер работы требует постоянного взаимодействия с компьютером (работа программиста-разработчика) с напряжением внимания и сосредоточенности, при исключении возможности периодического переключения на другие виды трудовой деятельности, не связанные с ПЭВМ, рекомендуется организация перерывов на

10–15 мин. через каждые 45–60 мин. работы. При высоком уровне напряженности работы рекомендуется психологическая разгрузка в специально оборудованных помещениях.

5.1.2 Опасные производственные факторы

5.1.2.1 Опасность поражения электрическим током

Поражение электрическим током является опасным производственным фактором и, поскольку программист имеет дело с электрооборудованием, то вопросам электробезопасности на его рабочем месте должно уделяться особое внимание. Нормы электробезопасности на рабочем месте регламентируются СанПиН 2.2.2/2.4.1340-03, вопросы требований к защите от поражения электрическим током освещены в ГОСТ Р 12.1.019-2009 ССБТ.

Электробезопасность – система организационных и технических мероприятий и средств, обеспечивающих защиту людей от вредного и опасного воздействия электрического тока, электрической дуги, электромагнитного поля и статического электричества.

Опасность поражения электрическим током усугубляется тем, что человек не в состоянии без специальных приборов обнаружить напряжение дистанционно.

Помещение, где расположено рабочее место оператора ПЭВМ, относится к помещениям без повышенной опасности ввиду отсутствия следующих факторов: сырость, токопроводящая пыль, токопроводящие полы, высокая температура, возможность одновременного прикосновения человека к имеющим соединение с землей металлоконструкциям зданий, технологическим аппаратам, механизмам и металлическим корпусам электрооборудования.

Основным организационным мероприятием по обеспечению безопасности является инструктаж и обучение безопасным методам труда, а

также проверка знаний правил безопасности и инструкций в соответствии с занимаемой должностью применительно к выполняемой работе.

К мероприятиям по предотвращению возможности поражения электрическим током относятся:

- С целью защиты от поражения электрическим током, возникающим между корпусом приборов и инструментом при пробое сетевого напряжения на корпус, корпуса приборов и инструментов должны быть заземлены;
- При включенном сетевом напряжении работы на задней панели корпуса приборов должны быть запрещены;
- Все работы по устранению неисправностей должен производить квалифицированный персонал;
- Необходимо постоянно следить за исправностью электропроводки.

5.1.2.2 Опасность возникновения пожара

Возникновение пожара является опасным производственным фактором, т.к. пожар на предприятии наносит большой материальный ущерб, а также часто сопровождается травмами и несчастными случаями. Регулирование пожаробезопасности производится СНиП 21-01-97.

В помещениях с ПЭВМ повышен риск возникновения пожара из-за присутствия множества факторов: наличие большого количества электронных схем, устройств электропитания, устройств кондиционирования воздуха; возможные неисправности электрооборудования, освещения, или неправильная их эксплуатация может послужить причиной пожара.

Возможные виды источников воспламенения:

- Искра при разряде статического электричества;
- Искры от электрооборудования;
- Искры от удара и трения;
- Открытое пламя.

Для профилактики организации действий при пожаре должен проводиться следующий комплекс организационных мер: должны обеспечиваться регулярные проверки пожарной сигнализации, первичных средств пожаротушения; должен проводиться инструктаж и тренировки по действиям в случае пожара; не должны загромождаться или блокироваться пожарные выходы; должны выполняться правила техники безопасности и технической эксплуатации электроустановок; во всех служебных помещениях должны быть установлены «Планы эвакуации людей при пожаре и других ЧС», регламентирующие действия персонала при возникновении пожара.

Для предотвращения пожара помещение с ПЭВМ должно быть оборудовано первичными средствами пожаротушения: углекислотными огнетушителями типа ОУ-2 или ОУ-5; пожарной сигнализацией, а также, в некоторых случаях, автоматической установкой объемного газового пожаротушения.

5.2 Экологическая безопасность

5.2.1 Влияние объекта исследования на окружающую среду

В данном разделе рассматривается воздействие на окружающую среду деятельности по разработке проекта, а также самого продукта в результате его реализации на производстве.

В ходе выполнения ВКР и дальнейшем использовании алгоритмов отсутствуют выбросы каких-либо вредных веществ в атмосферу и гидросферу, следовательно, загрязнение воздуха и воды не происходит.

Люминесцентные лампы, применяющиеся для искусственного освещения рабочих мест, также требуют особой утилизации, т.к. в них присутствует от 10 до 70 мг ртути, которая относится к чрезвычайно-опасным химическим веществам и может стать причиной отравления живых существ, а также загрязнения атмосферы, гидросферы и литосферы. Сроки службы таких

ламп составляют около 5-ти лет, после чего их необходимо сдавать на переработку в специальных пунктах приема.

Во время разработки и написания ВКР образовывался мусор, такой как: канцелярские принадлежности, бумажные отходы, неисправные комплектующие персонального компьютера, люминесцентные лампы.

5.2.2 Мероприятия по защите окружающей среды

Для уменьшения вредного влияния на литосферу необходимо производить сортировку отходов и обращаться в службы по утилизации для дальнейшей переработки или захоронения.

В основном, организации, занимающиеся приёмом и утилизацией ртуть содержащих отходов, принимают люминесцентные лампы в массовых количествах. Лампа состоит из электронного блока — выгодный компонент для реставрации и утилизации; колба и цоколь также ценное сырье. По стране утилизацией «ртутных» ламп занимаются более 50 фирм, но единственное их условие — деньги, которые вы должны заплатить за вывоз.

Такие лампы нельзя выкидывать в мусоропровод или уличные контейнеры, а нужно отнести в свой районный ДЕЗ (Дирекция единичного заказчика) или РЭУ (Ремонтно-эксплуатационное управление), где есть специальные контейнеры. Там они принимаются бесплатно, основанием должна служить утилизация в соответствии с Управлением Федеральной службы по надзору в сфере защиты прав потребителей и благополучия человека по Томской области. Пункты приёма отработавших свой срок люминесцентных ламп по городам можно найти в интернете.

Переработка макулатуры представляет собой многоэтапный процесс, цель которого заключается в восстановлении бумажного волокна и, зачастую, других компонентов бумаги (таких как минеральные наполнители) и использование их в качестве сырья для производства новой бумаги.

Организации, занимающиеся покупкой сломанных компьютеров на запчасти, готовы платить за запчасти деньги, которые они сэкономят на покупке новых деталей, необходимых для ремонта. Такие организации принимают даже битую и залитую чем-то технику. Компьютерная техника (или ее компоненты) может также заинтересовать тех, кто скупает старые платы и радиодетали для получения из них после переработки драгоценных и редких металлов. Многие сетевые гипермаркеты электронной техники периодически устраивают программу утилизации. Условия такие: за старую бытовую технику вам предложат неплохую скидку на последующую покупку в этом магазине. Также можно самостоятельно отвезти сломанный компьютер в пункт приема металлолома не составит труда. Такие точки приема есть в каждом городе.

Также следует отменить, что разрабатываемая в рамках ВКР система позволяет создавать сложные по структуре документы технических задании для компаний разрабатывающие информационные системы, вести документооборот и переписку между участниками непосредственно в самой системе в электронном формате, что экономит бумагу.

5.3 Безопасность в чрезвычайных ситуациях

5.3.1 Основные чрезвычайные ситуации в офисном помещении

Чрезвычайные ситуации бывают техногенного, природного, биологического, социального или экологического характера.

При работе в кабинете могут возникнуть следующие классификации чрезвычайных ситуаций:

- Преднамеренные/непреднамеренные;
- Техногенные: взрывы, пожары, обрушение помещений, аварии на системах жизнеобеспечения/природные – связанные с проявлением стихийных сил природы.

- Экологические – это аномальные изменения состояния природной среды, такие как загрязнения биосферы, разрушение озонового слоя, кислотные дожди/ антропогенные – являются следствием ошибочных действий людей.
- Биологические – различные эпидемии, эпизоотии, эпифитотии;
- Социальные – это обстановка на определенной территории, сложившаяся в результате опасного социального явления, которое повлекло в результате человеческие жертвы, ущерб здоровью, имуществу или окружающей среды;
- Комбинированные.

5.3.2 Типичные чрезвычайные ситуации

5.3.2.1 Пожар (возгорание)

Наиболее вероятная чрезвычайная ситуация, которая может возникнуть при работе с ПЭВМ – пожар, так как в современных ЭВМ очень высокая плотность размещения элементов электронных схем. В непосредственной близости друг от друга располагаются соединительные провода и кабели, при протекании по ним электрического тока выделяется значительное количество теплоты, при этом возможно оплавление изоляции и возникновение возгорания.

Биологические, так как программист работает в кабинете и контактирует с большим количеством людей, в том числе с другими сотрудниками, то велик риск заражения одного сотрудника от другого (чем больше людей, тем выше риск). В связи с большим скоплением народа в одном помещении появляется необходимость в непрерывном проветривании, что приводит к образованию сквозняков, что так же может сказаться на здоровье.

Возникновение других видов ЧС – маловероятно.

5.3.2.2 Социальная чрезвычайная ситуация (кибертерроризм)

Терроризм – это метод, посредством которого организованная группа или партия стремятся достичь провозглашенные ими цели через систематическое использование насилия.

Компьютерный терроризм (кибертерроризм) – использование компьютерных и телекоммуникационных технологий (прежде всего, интернета) в террористических целях.

В киберпространстве могут быть использованы различные способы для совершения кибертеракта:

– Получение несанкционированного доступа к государственным и военным секретам, банковской и личной информации;

– Нанесение ущерба отдельным физическим элементам информационного пространства, например, разрушение сетей электропитания, создание помех;

– Использование специальных программ для разрушения аппаратных средств;

– Кража или уничтожение информации, программ и технических ресурсов путем преодоления систем защиты, внедрения вирусов, программных закладок;

– Воздействие на программное обеспечение и информацию;

– Раскрытие и угроза публикации закрытой информации;

– Захват каналов средств массовой информации с целью распространения дезинформации, слухов, демонстрации мощи террористической организации и объявления своих требований;

– Уничтожение или активное подавление линий связи, неправильная адресация, перегрузка узлов коммуникации;

– Проведение информационно-психологических операций.

Использование документов и данных хранящихся в системы, включая личные переписки участников, дают киберпреступником возможность использовать их в злых умыслах.

В результате чего, физические или юридические лица, чьи данные были использованы в злых умыслах могут быть подвержены шантажу и вымогательству со стороны киберпреступников, что может негативно сказаться на репутации и финансовому состоянию компаний разрабатывающие информационные системы. Также в создаваемых документах технических заданий могут содержаться информации о государственной или коммерческой тайне.

5.3.3 Действия в результате возникновения чрезвычайной ситуации и мер по ликвидации ее последствий

При работе компьютерной техники выделяется много тепла, что может привести к пожароопасной ситуации. Источниками зажигания так же могут служить приборы, применяемые для технического обслуживания, устройства электропитания, кондиционеры воздуха. Серьёзную опасность представляют различные электроизоляционные материалы, используемые для защиты от механических воздействий отдельных радиодеталей.

В связи с этим, участки, на которых используется компьютерная техника, по пожарной опасности относятся к категории пожароопасных «В».

Меры, соблюдение которых поможет исключить с большой вероятностью возможность возникновения пожара:

– Для понижения воспламеняемости и способности распространять пламя кабели покрывают огнезащитным покрытием;

– При ремонтно-профилактических работах строго соблюдаются правила пожарной безопасности;

– Помещения, в которых должны располагаться ПЭВМ проектируют I или II степени огнестойкости;

– Каждое из помещений, где производится эксплуатация устройств ПЭВМ, должно быть оборудовано первичными средствами пожаротушения и обеспечено инструкциями по их применению. В качестве средств пожаротушения разрешается использование углекислотного огнетушителя типа ОУ-2, ОУ-5(описание ниже), а также порошковый тип. Применение пенных огнетушителей не допускается, так как жидкость пропускает ток;

– Устройства ПЭВМ необходимо устанавливать вдали отопительных и нагревательных приборов (расстояние не менее 1 м и в местах, где не затруднена их вентиляция и нет прямых солнечных лучей);

– Разрабатываются организационные меры по обучению персонала навыкам ликвидации пожара имеющимися в наличии средствами тушения пожара до прибытия пожарного подразделения.

При пожаре люди должны покинуть помещение в течение минимального времени.

В помещениях с компьютерной техникой, недопустимо применение воды и пены ввиду опасности повреждения или полного выхода из строя дорогостоящего электронного оборудования.

Для тушения пожаров необходимо применять углекислотные и порошковые огнетушители, которые обладают высокой скоростью тушения, большим временем действия, возможностью тушения электроустановок, высокой эффективностью борьбы с огнем. Воду разрешено применять только во вспомогательных помещениях.

5.4 Правовые и организационные вопросы обеспечения безопасности

5.4.1 Описание правовых норм для работ, связанных с работой за ПЭВМ

Регулирование отношений между работником и работодателем, касающихся оплаты труда, трудового распорядка, особенности регулирования труда женщин, детей, людей с ограниченными способностями и проч., осуществляется законодательством РФ, а именно трудовым кодексом РФ.

Нормальная продолжительность рабочего времени не может превышать 40 часов в неделю.

Порядок исчисления нормы рабочего времени на определенные календарные периоды (месяц, квартал, год) в зависимости от установленной продолжительности рабочего времени в неделю определяется федеральным органом исполнительной власти, осуществляющим функции по выработке государственной политики и нормативно-правовому регулированию в сфере труда.

Продолжительность ежедневной работы (смены) не может превышать:

- Для работников в возрасте от 15 до 16 лет – 5 часов, в возрасте от 16 до 18 лет – 7 часов;
- Для учащихся общеобразовательных учреждений, образовательных учреждений начального и среднего профессионального образования, совмещающих в течение учебного года учебу с работой, в возрасте от 14 до 16 лет – 2,5 часа, в возрасте от 16 до 18 лет – 4 часов;
- Для инвалидов – в соответствии с медицинским заключением, выданным в порядке, установленном федеральными законами и иными нормативными правовыми актами российской федерации.

Для работников, занятых на работах с вредными и (или) опасными условиями труда, где установлена сокращенная продолжительность рабочего времени, максимально допустимая продолжительность ежедневной работы (смены) не может превышать:

- При 36-часовой рабочей неделе - 8 часов;
- При 30-часовой рабочей неделе и менее - 6 часов.

Продолжительность работы (смены) в ночное время сокращается на один час без последующей отработки. К работе в ночное время не допускаются: беременные женщины; работники, не достигшие возраста 18 лет, за исключением лиц, участвующих в создании и (или) исполнении художественных произведений, и других категорий работников в соответствии с настоящим Кодексом и иными федеральными законами.

В течение рабочего дня (смены) работнику должен быть предоставлен перерыв для отдыха и питания. Время предоставления перерыва и его конкретная продолжительность устанавливаются правилами внутреннего трудового распорядка или по соглашению между работником и работодателем.

Всем работникам предоставляются выходные дни (еженедельный непрерывный отдых).

Организация-работодатель выплачивает заработную плату работникам. Возможно удержание заработной платы только в случаях, установленных ТК РФ ст. 137. В случае задержки заработной платы более чем на 15 дней, работник имеет право приостановить работу, письменно уведомив работодателя.

Законодательством РФ запрещена дискриминация по любым признакам и принудительный труд.

Если пользователь постоянно загружен работой с ЭВМ, приемлемой является поза сидя. В положении сидя основная нагрузка падает на мышцы, поддерживающие позвоночный столб и голову. В связи с этим при длительном сидении время от времени необходимо сменять фиксированные рабочие позы.

Исходя из общих принципов организации рабочего места, в нормативно-методических документах сформулированы требования к конструкции рабочего места.

Основными элементами рабочего места программиста являются: рабочий стол, рабочий стул (кресло), дисплей, клавиатура, мышь; вспомогательными - пюпитр, подставка для ног.

Взаимное расположение элементов рабочего места должно обеспечивать возможность осуществления всех необходимых движений и перемещений для эксплуатации и технического обслуживания оборудования.

Рабочие места с ЭВМ должны располагаться па расстоянии не менее 1,5 м от стены с оконными проемами, от других стен – на расстоянии 1 м, между собой – на расстоянии не менее 1,5 м. При размещении рабочих мест необходимо исключить возможность прямой засветки экрана источником естественного освещения.

При размещении ЭВМ на рабочем месте должно обеспечиваться пространство для пользователя величиной не менее 850 м. Для стоп должно быть предусмотрено пространство по глубине и высоте не менее 150 мм, по ширине – не менее 530 мм. Располагать ЭВМ на рабочем месте необходимо так, чтобы поверхность экрана находилась на расстоянии 400 – 700 мм от глаз пользователя. Конструкция рабочего места и взаимное расположение всех его элементов (сиденье, органы управления, средства отображения информации и т.д.) должны соответствовать антропометрическим, физиологическим и психологическим требованиям, а также характеру работы.

Рабочее кресло обеспечивает поддержание рабочей позы в положении сидя, и чем длительнее это положение в течение рабочего дня, тем жестче должны быть требования к созданию удобных и правильных рабочих сидений.

Высота поверхности сиденья должна регулироваться в пределах 400 – 550 мм. Ширина и глубина его поверхности должна быть не менее 400 мм. Поверхность сиденья должна быть плоской, передний край – закругленным. Сиденье и спинка кресла должны быть полумягкими, с нескользящим, не электризующимся и воздухопроницаемым покрытием, материал которого обеспечивает возможность легкой очистки от загрязнения.

Опорная поверхность спинки стула должна иметь высоту 280 – 320 мм, ширину – не менее 380 мм и радиус кривизны горизонтальной плоскости – 400 мм. Расстояние сцинки от переднего края сиденья должно регулироваться в пределах 260 – 400 мм.

Рабочее место должно быть оборудовано устойчивой и просто регулируемой подставкой для ног, располагающейся, по возможности, по всей ширине отводимого участка для ног. Подставка должна иметь ширину не менее 300 мм, глубину не менее 400 мм, регулировку по высоте до 150 мм и по углу наклона опорной поверхности подставки до 20. Поверхность подставки должна быть рифленой, по переднему краю иметь бортик высотой 10 мм.

При организации рабочего пространства необходимо учитывать индивидуальные антропометрические параметры пользователя с соответствующими допусками на возможные изменения рабочих поз и потребность в перемещениях.

Рациональной рабочей позой может считаться такое расположение тела, при котором ступни работника расположены на плоскости пола или на подставке для ног, бедра сориентированы в горизонтальной плоскости, верхние части рук – вертикальный угол локтевого сустава колеблется в пределах 70 – 90, запястья согнуты под углом не более чем 20, наклон головы – в пределах 15 – 20, а также исключены частые ее повороты.

5.4.2 Влияние реализации веб-системы на работу компаний разрабатывающие информационные системы

Основным направлением реализации разработанного продукта является применение его в качестве веб-системы, которая позволит создавать технические задания в формате pdf для компаний занимающие разработкой информационных систем.

Технологические требования настолько сложны, что порой топ-менеджеры не понимают их и подписывают «не глядя». В результате – нарушение сроков и изменение требований, что приводит к денежным затратам и неустойкам со стороны компаний. Разрабатываемая в рамках ВКР система позволяет сформировать простой и понятный документ в электронном формате pdf с техническими требованиями, который позволит специалистам разного уровня (от руководителей до разработчика) ответить на вопросы «Что будет разрабатываться?», «Каким образом?», «Каков конечный результат?». Другими словами, система обеспечит единое виденье разрабатываемой информационной системы у всех заинтересованных лиц.

Другой положительный момент системы, переписка между участниками определенного проекта ведется непосредственно в самой программе, что экономит время пользователей системы.

Документооборот предусмотренный системой также дает возможность делать все действия в самой системе, не принуждая пользователя использовать другие сервисы по обмену документов.

Разработанная веб-система решает проблемы нехватки времени, денежных средств и бытовые проблемы компаний разрабатывающие информационные системы, так как все участники системы могут вести диалог в самой системе. В результате система позволяет автоматизировать рутинные дела компаний, занимающихся разработкой информационных систем на этапе создания технических требований.

ЗАКЛЮЧЕНИЕ

Далее в магистерской диссертации были отражены следующие важные выводы:

- Концепция систематизации технических требований не должна быть ограничена набором конкретных методов и инструментов, а лишь отражает возможные правила их использования с точки зрения профессиональной позиции для каждой заинтересованной стороны.
- Классификация и кодирования требований, имеющих однозначные взаимосвязи на всех уровнях иерархии своих описаний, разрешает проблемы информационного характера, связанные со сложностью восприятия и работы с множеством требований к ИС.
- Набор групп требований определяется в зависимости от концепции разрабатываемой информационной системы и её будущего содержания.

В рамках магистркой диссертации было разработано программное обеспечение системы управления требованиями, которое позволяет пользователям сформировать простой и понятных технический документ, обеспечивающий единое виденье разрабатываемой ИС и автомитизируемых процессов у всех заинтересованных лиц разного уровня (от руководителя до разработчика). Сформированный с помощью програмного обеспечения единый документ, отражающий иерархию и взаимосвязи всей системы требований, дает заинтересованным пользователям ответы на вопросы: «Что будет разрабатываться?», «Каким образом?», «Каков конечный результат?», а также обеспечивал бы единое виденье жизненного цикла ИС. Эффективносность функционирования ИС зависит от четырех стадий (предпроектная, проектировочная, внедрение, функционирование). От качества проектировочных работ зависит эффективность функционирования будущей

ИС, поэтому каждая стадия разделяется на несколько этапов и предусматривает составлениет технической документации, отражающей результаты работ.

Разработанная в рамках данной магистерской диссертации система управления требованиями дает инструментарий, позволяющий студентам обучающихся по дисциплины «Инженерия требований к системам», изучить процесс формирования технических требований и их согласования с заинтересованными лицами (анг. stackholder) и изучить жизненый цикл ИС.

ОПУБЛИКОВАННЫЕ РАБОТЫ

1. Ногербек Н.Д. Обзор MVC- веб-фреймворков. [Электронный ресурс] / науч. рук. Мальчуков А.Н. // Молодежь и современные информационные технологии: сборник трудов XIII Международной научно-практической конференции студентов, аспирантов и молодых ученых, г. Томск, 9-13 ноября 2015 г. в 2 т. / Национальный исследовательский Томский политехнический университет (ТПУ), Институт кибернетики (ИК) ; ред. кол. Т. Е. Мамонова [и др.]. — 2015. — Т. 2. — [С. 107-108]. — Заглавие с титульного экрана. — Свободный доступ из сети Интернет. — Adobe Reader.

Режим доступа:
http://portal.tpu.ru/f_ic/files/science/activities/msit/msit2015_tom2.pdf

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. George A. Miller. The Magical Number Seven, Plus or Minus Two // The Psychological Review. 1956. Vol. 63. P. 81-97.
2. Берталанфи Л. фон Общая теория систем — Критический обзор // Исследования по общей теории систем. М.: Прогресс, 1969. С. 23-82.
3. Системный анализ // Википедия. Открытая энциклопедия. URL: httpУ/ru.wikipedia.org/Систем- ный анализ (дата обращения: 01.06.2017).
4. Качало А. В. Теория систем и системного анализа: учебное пособие для высших учебных заведений. М.: Горячая линия – Телеком, 2009. – 216 с.
5. Данилин А, Слюсаренко А. Архитектура и стратегия. Инь и Янь информационных технологий предприятия. М.: Интернет-Ун-т Информ. Технологий. 2005. — 504 с.
6. John A. Zachman. The Zachman Framework: The Official Concise Definition // Zachman International Web Site. URL: http://zachmaninternational.com/index.php/home-article/13 (дата обращения: 01.06.2017).
7. ISO/IEC/IEEE 42010 Systems and software engineering — Architecture description // JTC 1/ SC 7. ISO publications, 2011.
8. Systems Engineering Fundamentals // Department Of DefenseSystemManagementCollege. Fort Belvoir, Virginia, 2001.
9. Ковалев С. М, Ковалев С. В. Современные методологии описания бизнес-процессов — просто о сложном // Консультант директора. 2004. № 12.
10. Коберн А. Современные методы описания функциональных требований к системам. М.: Лори, 2002. — 266 с.
11. RationalUnifiedProcess // Википедия. Открытая энциклопедия. URL: http://ru.wikipedia.org/wi-ki/Rational_Unified_Process (дата обращения: 01.06.2017)
12. RationalSoftware// Википедия. Открытая энциклопедия. URL: http://ru.wikipedia.org/ wiki/Rational_Software (дата обращения: 01.06.2017)

13. Джим Арлок, Айла Нейштадт, UML 2 и Унифицированный процесс. Практический объектно-ориентированный анализ и проектирование, 2-ое издание, изд. «Символ», 2010г.

14. Крег Ларман, Применение UML 2.0 и шаблонов проектирования. Введение в объектно-ориентированный анализ, проектирование и итеративную разработку, изд. «», 2011 г.

15. P. Grassle, H. Baumann, P. Baumann, Design and architecture of systems, изд. «Packt», 2014 г.

16. Russ Miles, Kim Hamilton, Learning UML 2.0, изд. «Oreill"y», 2013 г.

17. Montreal University Team, Software Architecture Document in fulfillment of Seon 344 Winter, 2009 г.

18. Головчинер М. Н., Проектирование информационных систем – методическое указания, Томск, 2010г.

19. Stellman A., Greene J. Applied software project management // O'Reilly Media, 2005. — 308 p.

20. А.В. Симкин Подход к комплексному применению методологий систематизации требований // Научно-практический журнал «Прикладная информатика». – 2013. – №1(43).

21. Управление требованиями к IT-проектам // Habrahabr // URL: http://www.habrahabr.ru/post/75131/ (дата обращения: 02.06.2017)

22. ISO/IEC/IEEE 42010 Systems and software engineering — Architecture description // JTC 1/ SC 7. ISO publications, 2011.

23. Systems Engineering Fundamentals // Department of Defense System Management College. Fort Belvoir, Virginia, 2001.

```
from itertools import product
from string import ascii_uppercase

from django.template import RequestContext
from django.template.context_processors import csrf
from django.utils.translation import ugettext_lazy as _
from django.shortcuts import render, get_object_or_404, redirect
from django.http import JsonResponse, HttpResponse
from django.template.loader import render_to_string, get_template

from project.activities import CommentPosted
from .widgets import PaginatorMixin
import reversion
import json
import re
from weasyprint import HTML, CSS
from reversion.models import Version
from .models import (Project, GroupRequirementType, GroupTask, UserCharacteristic, UserCharacteristicDictionary,
                     Function, Member, Task, Comment, Purpose, CreationGoal, ObjectInformation,
                     GroupRequirementTypeDictionary, GroupRequirement, Requirement, Specification, Image)
from .forms import (MemberAddForm, GroupRequirementTypeFormSet, GroupRequirementForm, RequirementForm,
GroupTaskForm, TaskForm, UserCharacteristicForm, FunctionForm, CommentForm, PurposeForm,
CreationGoalForm, ObjectInformationForm, V_RequirementForm, FA_GroupRequirementForm,
FA_RequirementForm, DF_GroupRequirementForm, DF_RequirementForm, SpecificationForm,
RequirementFormTree, ImageFormSet)


def project_list(request):
    objects = Project.objects.filter(status='open', member__user=request.user)
    page = request.GET.get('page')
    max_number = 3
    page_object = PaginatorMixin(objects, max_number, page).queryset_paginated()
    page_range = PaginatorMixin(objects, max_number, page).page_numbering()
    projects = page_object.object_list
    context = {
        'projects': projects,
        'page_object': page_object,
        'page_range': page_range
    }
    return render(request, 'project/project_list.html', context)


def project_detail(request, project_code):
    project = get_object_or_404(Project, pk=project_code, status='open')
    group_requirement_types = GroupRequirementType.objects.filter(project=project_code, is_visible=True)
    group_tasks = GroupTask.objects.filter(project=project_code).order_by('create_time', 'symbol')
    user_characteristics = UserCharacteristic.objects.filter(project=project_code).order_by('-create_time')
    request_user_is_system_analyst = project.member_set.filter(user=request.user, role='system_analyst').exists()
    request_user_is_manager = project.member_set.filter(user=request.user, role='manager').exists()
    request_user_is_business_analyst = project.member_set.filter(user=request.user, role='business_analyst').exists()
    request_user_is_developer = project.member_set.filter(user=request.user, role='developer').exists()
    group_task_comment_form = CommentForm(request.POST)
    user_characteristic_comment_form = CommentForm(request.POST)
    creation_goal_comment_form = CommentForm(request.POST)
```

```
    creation_goals = CreationGoal.objects.filter(project=project_code)
    creation_goal_is_exist = CreationGoal.objects.filter(project=project_code).exists()
    purpose_comment_form = CommentForm(request.POST)
    purposes = Purpose.objects.filter(project=project_code)
    purpose_is_exist = Purpose.objects.filter(project=project_code).exists()
    object_information_comment_form = CommentForm(request.POST)
    object_information_all = ObjectInformation.objects.filter(project=project_code)
    object_information_is_exist = ObjectInformation.objects.filter(project=project_code).exists()
    member = project.member_set.get(user=request.user)

    context = {
        'project': project,
        'group_requirement_types': group_requirement_types,
        'group_tasks': group_tasks,
        'user_characteristics': user_characteristics,
        'request_user_is_manager': request_user_is_manager,
        'request_user_is_system_analyst': request_user_is_system_analyst,
        'request_user_is_business_analyst': request_user_is_business_analyst,
        'request_user_is_developer': request_user_is_developer,
        'group_task_comment_form': group_task_comment_form,
        'user_characteristic_comment_form': user_characteristic_comment_form,
        'purpose_comment_form': purpose_comment_form,
        'purposes': purposes,
        'purpose_is_exist': purpose_is_exist,
        'creation_goal_is_exist': creation_goal_is_exist,
        'creation_goals': creation_goals,
        'creation_goal_comment_form': creation_goal_comment_form,
        'object_information_all': object_information_all,
        'object_information_comment_form': object_information_comment_form,
        'object_information_is_exist': object_information_is_exist,
        'role': member.get_role_display(),
    }
    return render(request, 'project/project_detail.html', context)


def project_member_add(request, project_code):
    data = dict()
    project = get_object_or_404(Project, pk=project_code)
    if request.method == 'POST':
        form = MemberAddForm(request.POST)
        if form.is_valid():
            user = form.cleaned_data['user']
            if Member.objects.filter(user=user, project=project_code).exists():
                form.add_error('user', _('This user is already a member of the project!'))
            else:
                new_member = form.save(commit=False)
                new_member.project = project
                new_member.save()
                data['form_is_valid'] = True
                data['html_project_member'] = render_to_string('project/project_member_list.html', {'project': project})
        else:
            data['form_is_valid'] = False
    else:
        form = MemberAddForm()
    context = {'project': project, 'form': form}
    data['html_project_member_form'] = render_to_string('project/project_member_add.html', context, request=request)
```

```
    return JsonResponse(data)


@reversion.create_revision()
def group_task_add(request, project_code):
    data = dict()
    project = get_object_or_404(Project, pk=project_code)
    request_user_is_business_analyst = project.member_set.filter(user=request.user, role='business_analyst').exists()
    group_task_comment_form = CommentForm(request.POST)
    if request.method == 'POST':
        group_task_form = GroupTaskForm(request.POST)
        if group_task_form.is_valid():
            name = group_task_form.cleaned_data['name']
            if GroupTask.objects.filter(name=name, project=project_code).exists():
                group_task_form.add_error('name', _('Group task with the same name is already exist in this project!'))
            else:
                group_task = group_task_form.save(commit=False)
                symbols = [''.join(i) for i in product(ascii_uppercase, repeat=1)] + [a + b for a, b in product(ascii_uppercase,
repeat=2)]
                for symbol in symbols:
                    if not GroupTask.objects.filter(project=project_code, symbol=symbol).exists():
                        group_task.symbol = symbol
                        break
                group_task.project = project
                group_task.save()
                data['form_is_valid'] = True
                group_tasks = GroupTask.objects.filter(project=project_code).order_by('create_time')
                context = {'project': project, 'group_task': group_task, 'group_tasks': group_tasks,
'group_task_comment_form': group_task_comment_form, 'request_user_is_business_analyst':
request_user_is_business_analyst}
                context.update(csrf(request))
                data['html_group_task'] = render_to_string('project/group_task_list.html', context)
                reversion.set_user(request.user)
                reversion.set_comment('CREATE')
        else:
            data['form_is_valid'] = False
    else:
        group_task_form = GroupTaskForm()
    context = {'project': project, 'group_task_form': group_task_form}
    data['html_group_task_form'] = render_to_string('project/group_task_add.html', context, request=request)
    return JsonResponse(data)


@reversion.create_revision()
def group_task_edit(request, project_code, group_task_code):
    data = dict()
    group_task = get_object_or_404(GroupTask, pk=group_task_code)
    project = get_object_or_404(Project, pk=project_code)
    group_task_comment_form = CommentForm(request.POST)
    request_user_is_business_analyst = project.member_set.filter(user=request.user, role='business_analyst').exists()
    if request.method == 'POST':
        group_task_form = GroupTaskForm(request.POST, instance=group_task)
        if group_task_form.is_valid():
            group_task_form.save()
```

ПРИЛОЖЕНИЕ Б

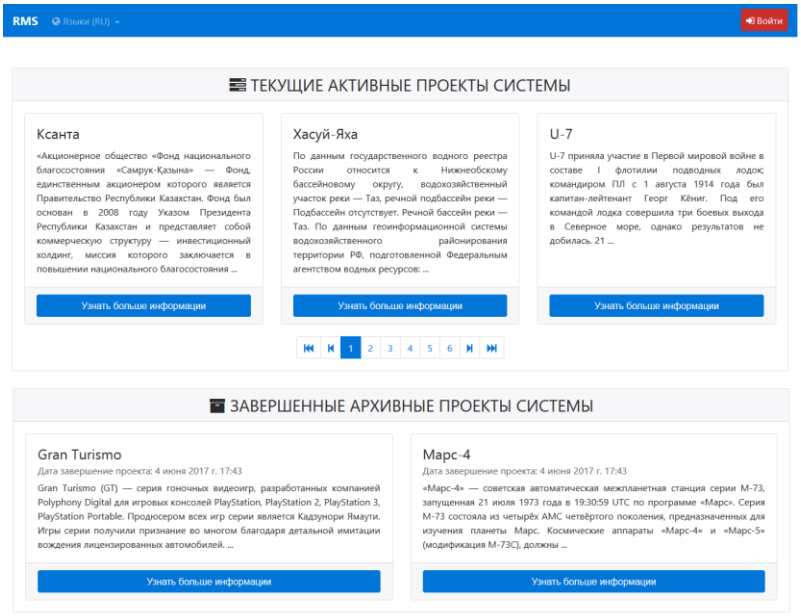

Рисунок Б.1 – Главная страница для неавторизированных пользователей.

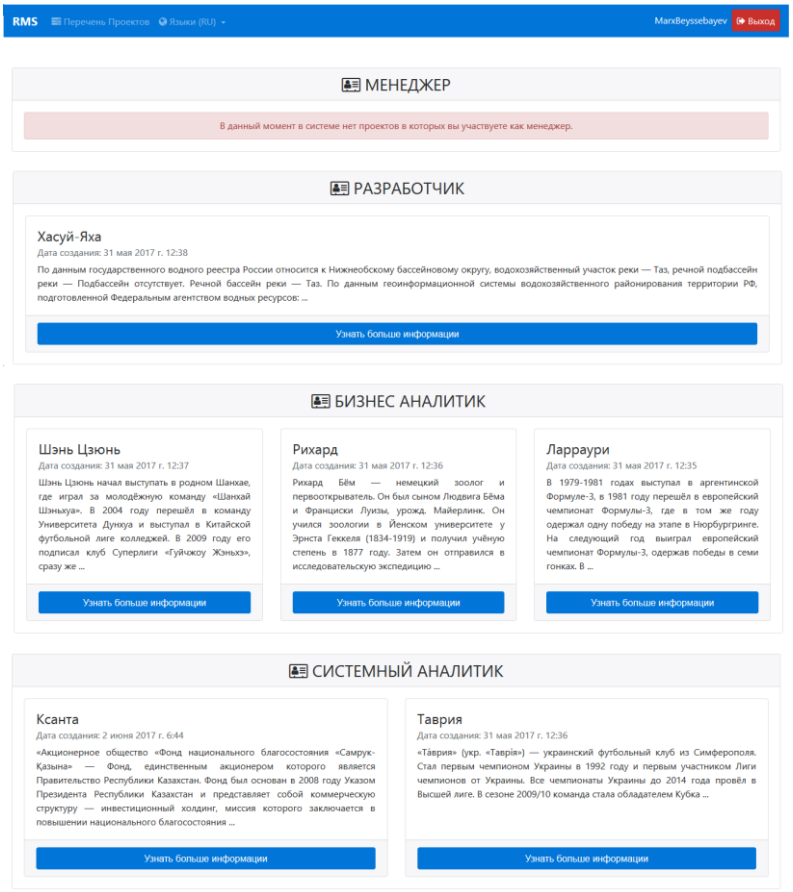

Рисунок Б.2 – Главная страница для авторизированных пользователей.

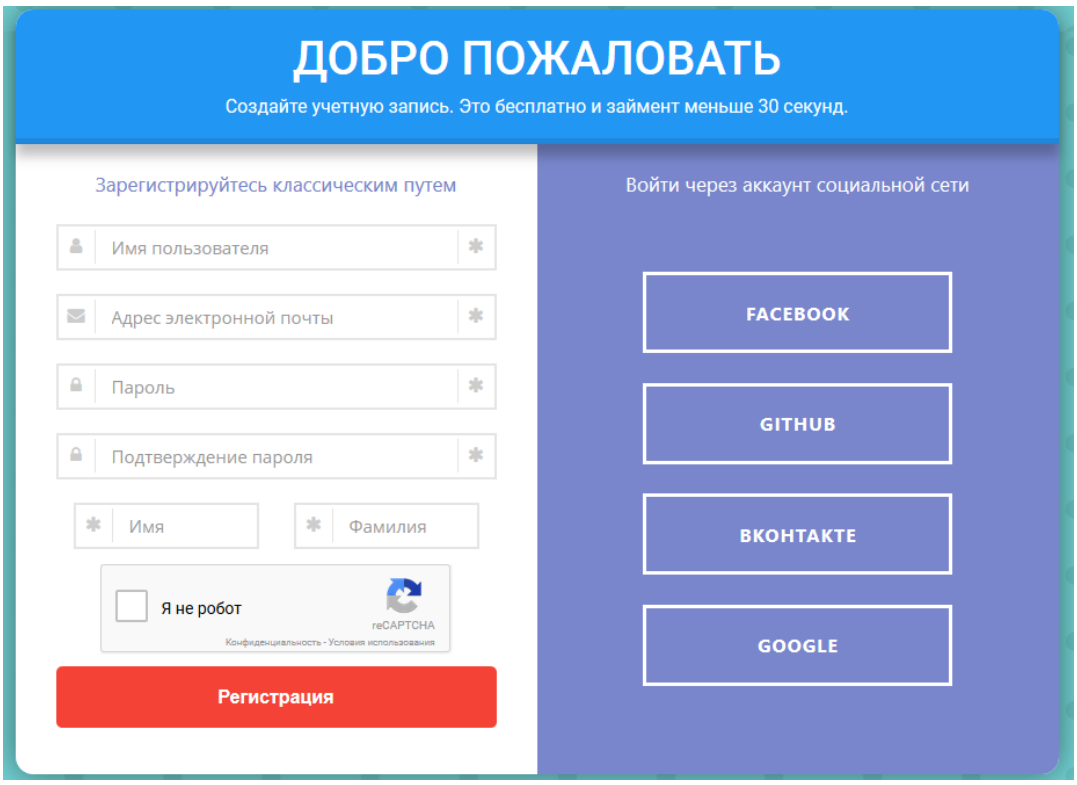

Рисунок Б.3 – Форма регистрации.

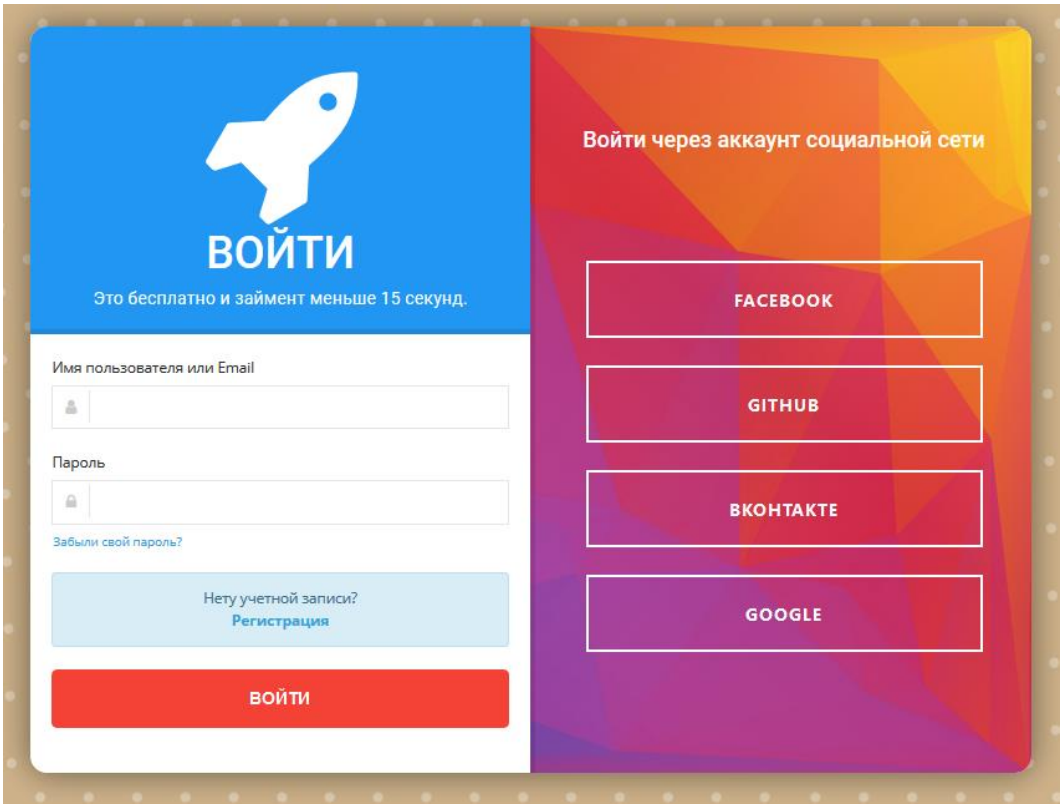

Рисунок Б.4 – Форма авторизации.

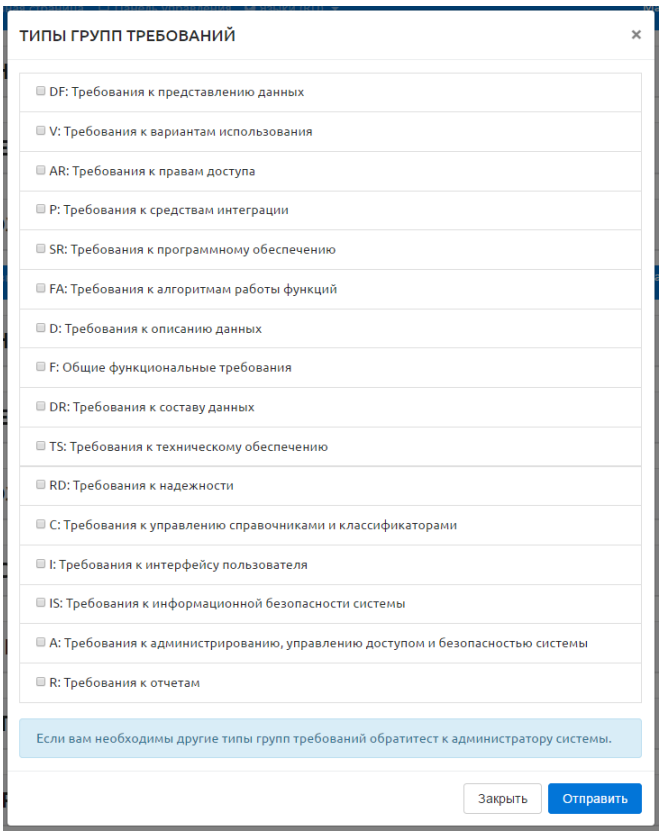

Рисунок Б.5 – Форма добавления типа группы требования.

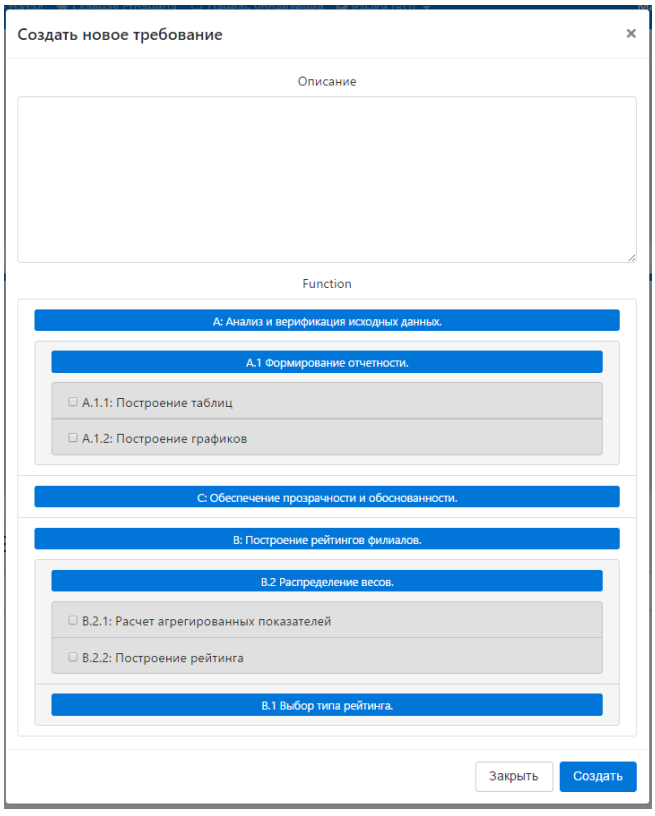

Рисунок Б.6 – Форма создания требования.

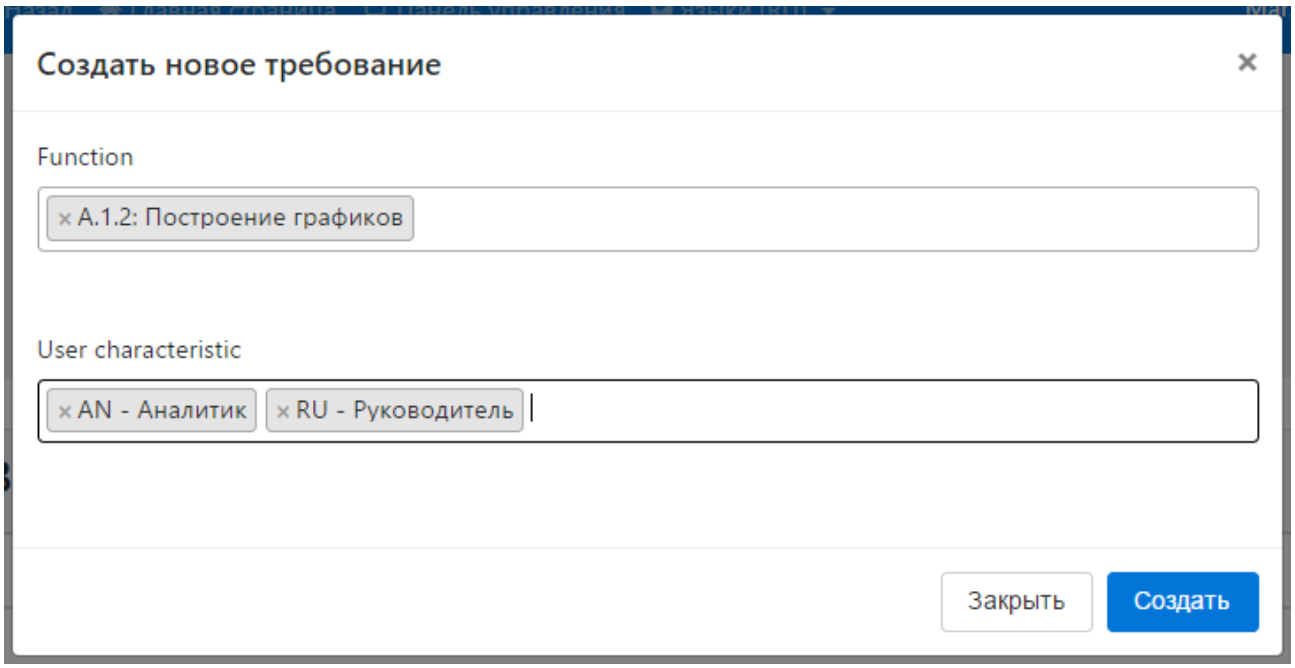

Рисунок Б.7 – Форма создания требования для типа FA.

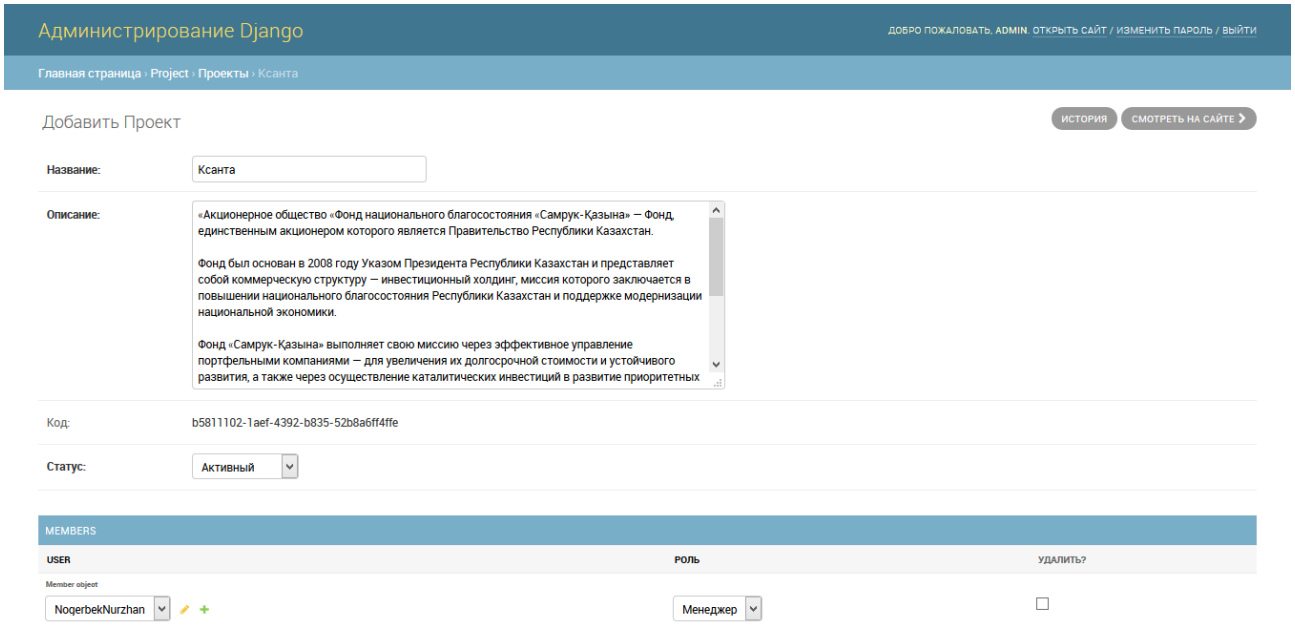

Рисунок Б.8 – Форма создания проекта на странице администратора.

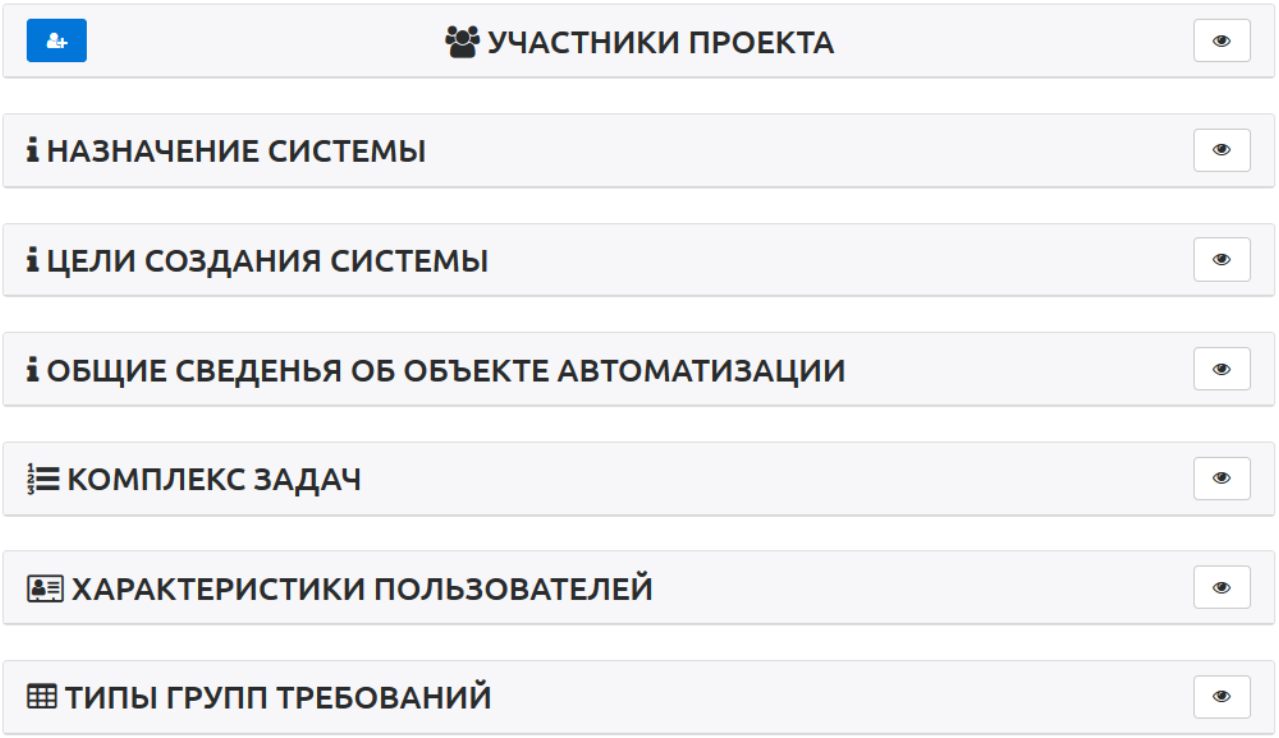

Рисунок Б.9 – Раскрываемые блоки на странице «Детали Проекта».

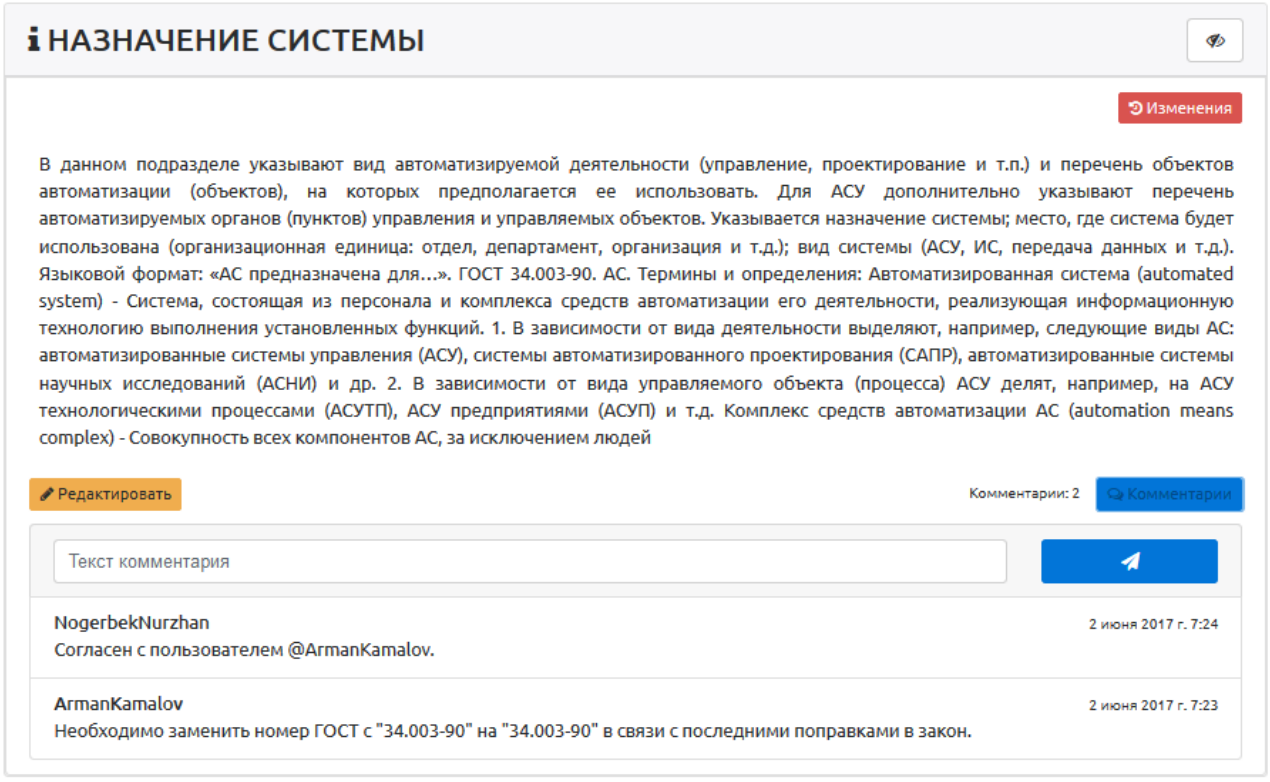

Рисунок Б.10 – Блок раздела «Назначения системы».

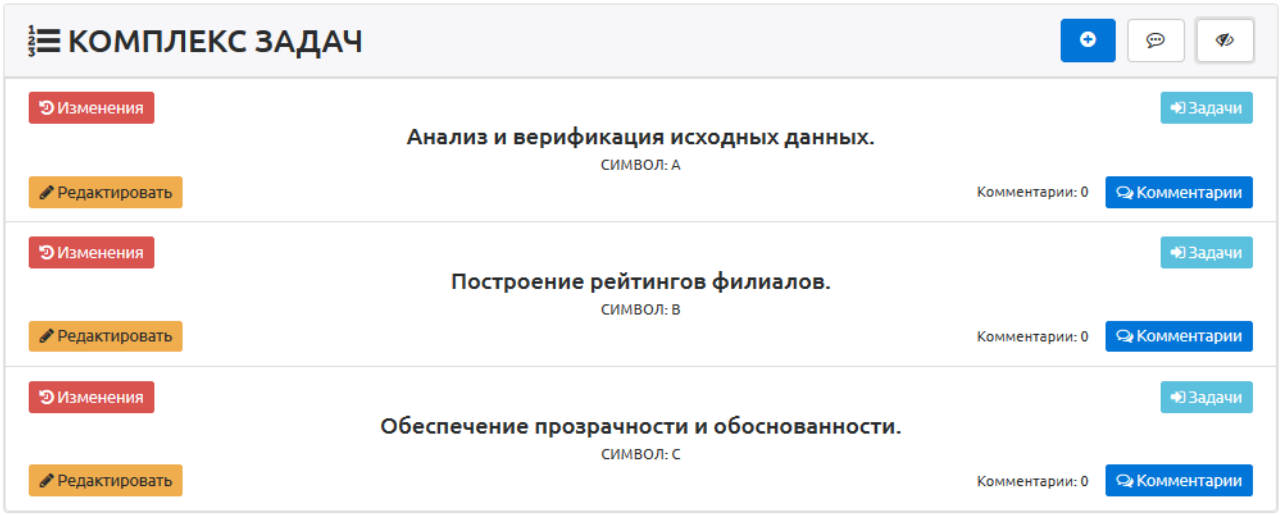

Рисунок Б.11 – Блок раздела «Комплекс задач».

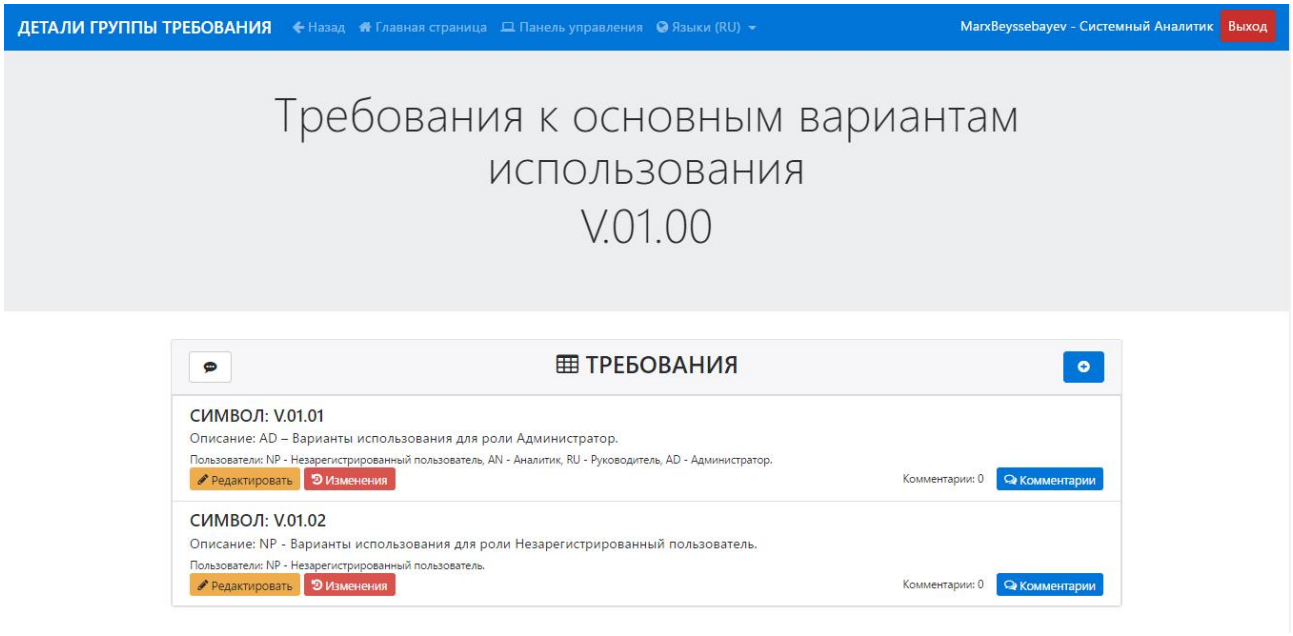

Рисунок Б.12 – Страница «Детали группы требований».

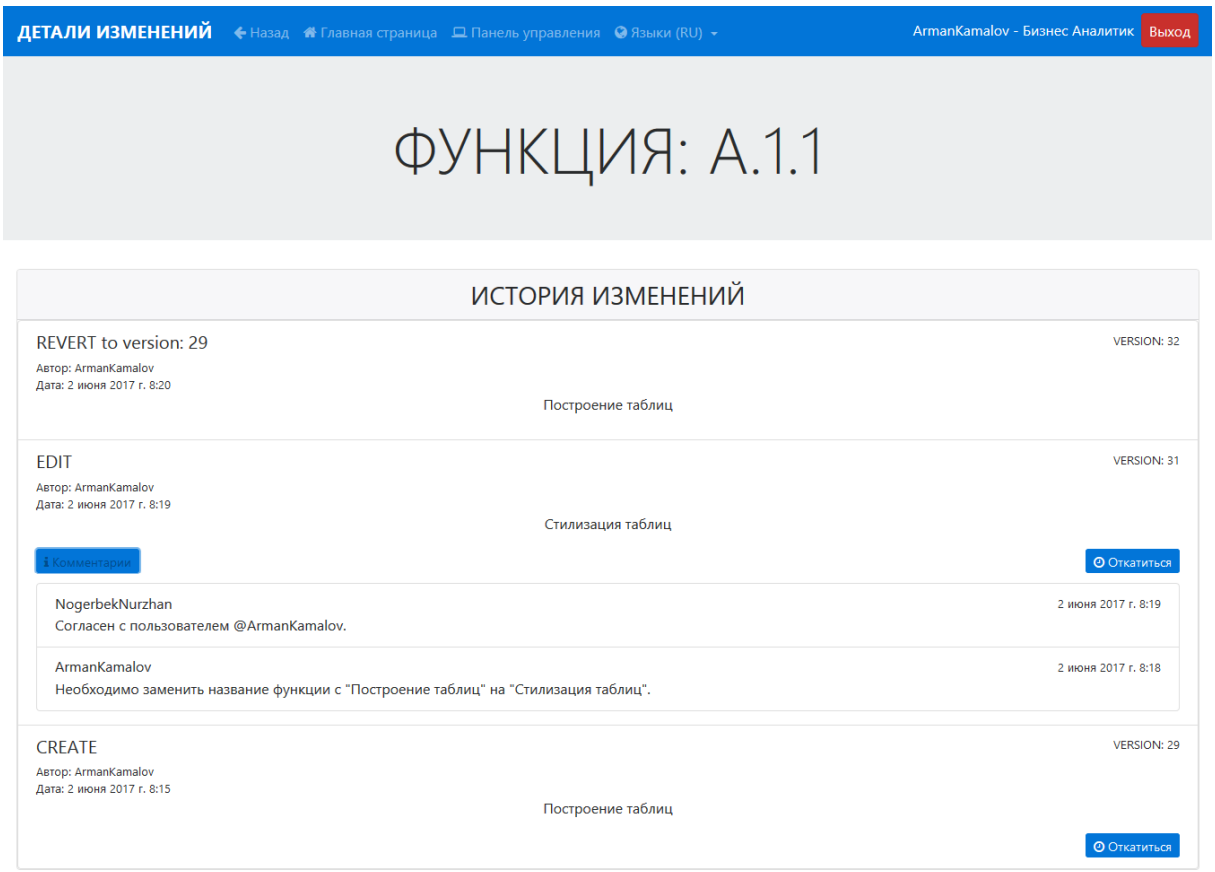

Рисунок Б.13 – Страница «Детали группы требований».

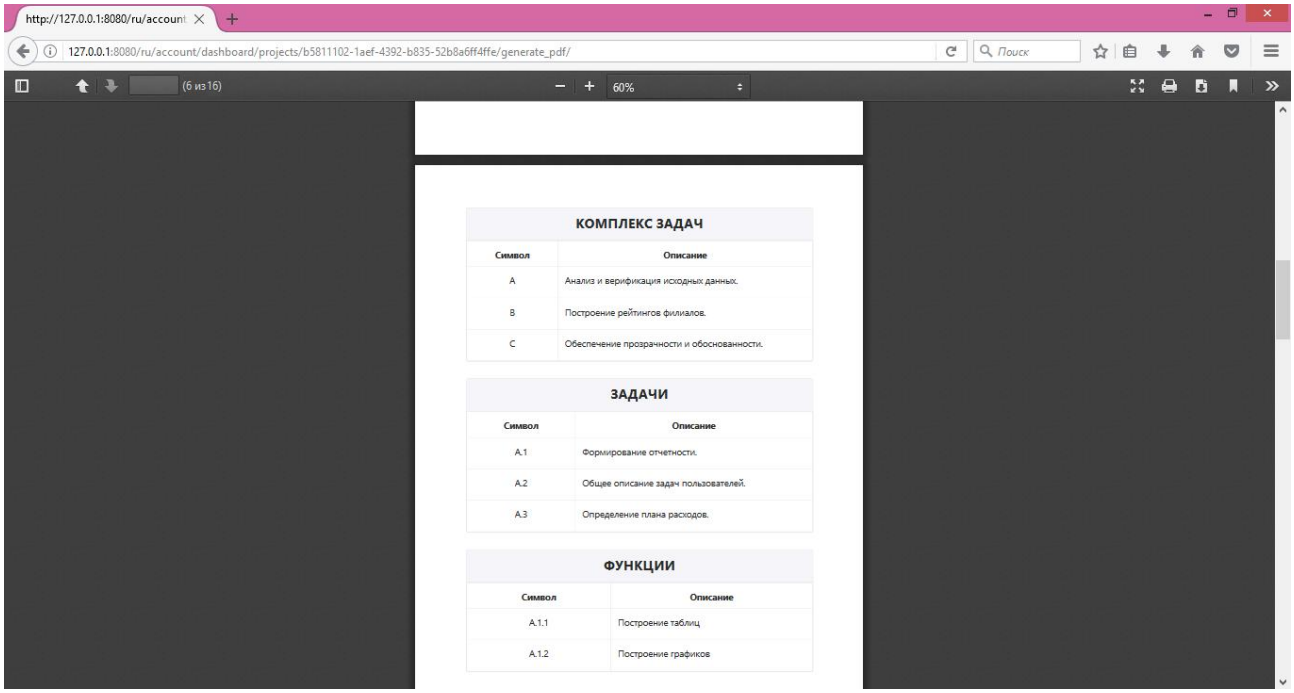

Рисунок Б.14 – Окно браузера с сгенерированном документом «Техническое задание».

ПРИЛОЖЕНИЕ В

Раздел на иностранном языке

Раздел «Requirements management system»

Студент:

Группа	ФИО	Подпись	Дата
8ИМ5А	Ногербек Нуржан Даулетбекулы		

Консультант проф. кафедры:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент каф. ИСТ	Мирошниченко Е.А.	к.т.н.		

Консультант – лингвист кафедры ИЯИК:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель	Горбатова Т.Н.			

REQUIREMENTS MANAGEMANT SYSTEM

In the modern information world in spite of impressive progress in the industry of information systems, many IT companies still have difficulties in identifying, collecting, documenting, and managing technical requirements to under development information systems. The technological requirements are so complex that sometimes top management of IT companies don't understand them fully and sign an agreement without deep analysis of subject field. After incorrectly formulated technical requirements, IT companies sometimes are not able to finish projects and to provide customers with the expected functionality. As a result – companies finish there projects out of time and always change requirements, which leads to monetary damages in IT companies.

Requirements management is one of the important stages in designing information systems. Requirements management is the process of documenting, analyzing, tracing, prioritizing and agreeing on requirements and then controlling change and communicating to relevant stakeholders. It is a continuous process throughout a project. A requirement is a capability to which a project outcome (product or service) should conform. Design of information systems is a complex task, which requires not only taking into account the huge number of requirements from the stakeholders, but also their systematization.

Systematization of requirements is described with the help of:

- relationships between the composition of information system requirements.
- some group requirements, their components and detailed specifications.

Requirements define the needs of stakeholders and the functionality, which must be following by the information system. Stakeholder is a personality. System influenced to them.

The purpose of requirements management is to ensure that an organization documents, verifies, and meets the needs and expectations of its customers and internal or external stakeholders. Requirements management begins with the analysis and elicitation of the objectives and constraints of the organization. Requirements

management includes requirements support, integrating requirements and work organization (attributes for requirements).

The purpose of the master's thesis is to develop software for requirements management system. This information system allows users to create a simple and understandable technical document that provides common vision of future system or automated processes to all stakeholders of different levels (from company executive to the developer). This technical document, generates requirements management system, describes the hierarchy and relationships of the system components. By using this system, stakeholders could get answers to the questions like «What will be developed?», «How does the system works?» and «What is the final result?».

The software of the requirements management system, which was created in the master's thesis, could be useful for TPU students in the study of the discipline «Engineering requirements for the systems».

The established traceability used in managing requirements to report if a company’s and a stakeholder’s interests are satisfied in terms of compliance, completeness, coverage, and consistency. Traceability also supports change management a part of requirements management in understanding the impacts of changes through requirements or other related elements (functional impacts through relations to functional architecture), and facilitation introducing these changes.

Requirement management system, which was created in this master's thesis, has the next important features:

- the ability to specify relationships between requirements;
- the possibility of constructing samples in different views;
- tracking changes.

This basic function package provides the ability to track and monitor design errors and improve the accuracy of formation evaluation changes. Such requirementы management systems as IBM Rational RequisitePro, Telelogic DOORS and Borland Caliber RM are widespread today. Also for requirements management in software

development process, we can use lightweight web project management systems as Redmine or Trello. Some features and comparative analysis of these systems are given below.

IBM Rational RequisitePro is a software which we can use to manage the requirements when developing software. It allows developers to define and manage requirements, organize and track changes that may occur throughout the project life cycle. Also RequisitePro allows to create qualitative scenarios, to organize and increase the efficiency of collaboration.

IBM Rational/Telelogic DOORS is a requirements management system, which allows to optimize the information exchange requirements to control a large amount of interconnected information provides verification of compliance and management. DOORS used successfully in the creation of sophisticated high-technology products.

Borland Caliber RM is an enterprise requirements management system designed to improve quality of products by improving interaction between team members. This program also ease of analysis, influences requirements and process of information transfer and the possibility of continuous collection of the wishes of the interested in the project, individuals at all stages of the project life cycle.

Requirements management involves communication between the project team members and stakeholders, and adjustment to requirements changes throughout the course of the project. To prevent one class of requirements from overriding another, constant communication among members of the development team is critical. For example, in software development for internal applications, the business has such strong needs that it may ignore user requirements, or believe that in creating use cases, the user requirements are being taken care of.

Redmine is one of the most popular and modern systems of task management. It is open source server-side web application for managing projects and tasks, written in Ruby. Redmine is an application based on a web framework RubyonRails and distributed according to General Public License. The system allows monitoring and control of tasks. Redmine is a lightweight and free system that does not require a

dedicated or virtual hosting. On the basis of performed analytical review criteria have been formulated for the optimal selection of the system. The following table provides a comparison of the most popular systems on these criteria.

Table B.1 – A comparative analysis of requirements management systems.

Criteria	IBM Rational RequisitePro	IBM Rational Telelogic DOORS	Borland Caliber RM	Redmine
The web interface	+	+	+	+
Tracing requirements	+	+	+	–
Access authorization	+	+	+	+
Access rights	+	+	+	–
Integration with visual UML modeling	+	+	+	+
Integration with version control systems	+	+	+	+
Work with non-text objects	–	+	+	+
The change notification of requirements	+	+	+	–
Free license	–	–	–	–
Lightness	–	–	–	–

Requirements traceability is concerned with documenting the life of a requirement. It should be possible to trace back to the origin of each requirement and every change made to the requirement should therefore be documented in order to achieve traceability. Even the use of the requirement after the implemented features have been deployed and used should be traceable.

Requirements come from different sources, like the business person ordering the product, the marketing manager and the actual user. These people all have different requirements for the product. Using requirements traceability, an implemented feature can be traced back to the person or group that wanted it during the requirements elicitation. This can, for example, be used during the development process to prioritize the requirement, determining how valuable the requirement is to a specific user. It can also be used after the deployment when user studies show that a feature is not used, to see why it was required in the first place.

Requirements activities

At each stage in a development process, there are key requirements management activities and methods. To illustrate, consider a standard five-phase development process with Investigation, Feasibility, Design, Construction and Test, and Release stages.

Investigation

In Investigation, the first three classes of requirements are gathered from the users, from the business and from the development team. In each area, similar questions are asked; what are the goals, what are the constraints, what are the current tools or processes in place, and so on. Only when these requirements are well understood can functional requirements be developed.

In the common case, requirements cannot be fully defined at the beginning of the project. Some requirements will change, either because they simply weren't extracted, or because internal or external forces at work affect the project in mid-cycle.

The deliverable from the Investigation stage is a requirements document that has been approved by all members of the team. Later, in the thick of development, this document will be critical in preventing scope creep or unnecessary changes. As the system develops, each new feature opens a world of new possibilities, so the requirements specification anchors the team to the original vision and permits a controlled discussion of scope change.

While many organizations still use only documents to manage requirements, others manage their requirements baselines using software tools. These tools allow requirements to be managed in a database, and usually have functions to automate traceability (e.g., by allowing electronic links to be created between parent and child requirements, or between test cases and requirements), electronic baseline creation, version control, and change management. Usually such tools contain an export function that allows a specification document to be created by exporting the requirements data into a standard document application.

In the Feasibility stage, costs of the requirements are determined. For user requirements, the current cost of work is compared to the future projected costs once the new system is in place. Questions such as these are asked: "What are data entry errors costing us now?" Or "What is the cost of scrap due to operator error with the current interface?" Actually, the need for the new tool is often recognized as these questions come to the attention of financial people in the organization.

Business costs would include, "What department has the budget for this?" "What is the expected rate of return on the new product in the marketplace?" "What's the internal rate of return in reducing costs of training and support if we make a new, easier-to-use system?"

Technical costs are related to software development costs and hardware costs. "Do we have the right people to create the tool?" "Do we need new equipment to support expanded software roles?" This last question is an important type. The team must inquire into whether the newest automated tools will add sufficient processing power to shift some of the burden from the user to the system in order to save people time.

The question also points out a fundamental point about requirements management. A human and a tool form a system, and this realization is especially important if the tool is a computer or a new application on a computer. The human mind excels in parallel processing and interpretation of trends with insufficient data. The CPU excels in serial processing and accurate mathematical computation. The

overarching goal of the requirements management effort for a software project would thus be to make sure the work being automated gets assigned to the proper processor. For instance, “Don’t make the human remember where she is in the interface. Make the interface report the human’s location in the system at all times.” Or “Don’t make the human enter the same data in two screens. Make the system store the data and fill in the second screen as needed.”

The deliverable from the Feasibility stage is the budget and schedule for the project.

Design

Assuming that costs are accurately determined and benefits to be gained are sufficiently large, the project can proceed to the Design stage. In Design, the main requirements management activity is comparing the results of the design against the requirements document to make sure that work is staying in scope.

Again, flexibility is paramount to success. Here’s a classic story of scope change in mid-stream that actually worked well. Ford auto designers in the early ‘80s were expecting gasoline prices to hit $3.18 per gallon by the end of the decade. Midway through the design of the Ford Taurus, prices had centered to around $1.50 a gallon. The design team decided they could build a larger, more comfortable, and more powerful car if the gas prices stayed low, so they redesigned the car. The Taurus launch set nationwide sales records when the new car came out, primarily because it was so roomy and comfortable to drive.

In most cases, however, departing from the original requirements to that degree does not work. So the requirements document becomes a critical tool that helps the team make decisions about design changes.

In the construction and testing stage, the main activity of requirements management is to make sure that work and cost stay within schedule and budget, and that the emerging tool does in fact meet requirements. A main tool used in this stage is prototype construction and iterative testing. For a software application, the user interface can be created on paper and tested with potential users while the framework

of the software is being built. Results of these tests are recorded in a user interface design guide and handed off to the design team when they are ready to develop the interface. This saves their time and makes their jobs much easier.

Hardly would any software development project be completed without some changes being asked of the project. The changes can stem from changes in the environment in which the finished product is envisaged to be used, business changes, regulation changes, errors in the original definition of requirements, limitations in technology, changes in the security environment and so on. The activities of requirements change management include receiving the change requests from the stakeholders, recording the received change requests, analyzing and determining the desirability and process of implementation, implementation of the change request, quality assurance for the implementation and closing the change request. Then the data of change requests be compiled, analyzed and appropriate metrics are derived and dovetailed into the organizational knowledge repository.

Requirements management does not end with product release. From that point on, the data coming in about the application's acceptability is gathered and fed into the Investigation phase of the next generation or release. Thus the process begins again.

Acquiring a tool to support requirements management is no trivial matter and it needs to be undertaken as part of a broader process improvement initiative. It has long been a perception that a tool, once acquired and installed on a project, can address all of its requirements management-related needs. However, the purchase or development of a tool to support requirements management can be a costly decision. Organizations may get burdened with expensive support contracts, disproportionate effort can get misdirected towards learning to use the tool and configuring it to address particular needs, and inappropriate use can lead to erroneous decisions. Organizations should follow an incremental process to make decisions about tools to support their particular needs from within the wider context of their development process and tooling. The tools are presented in Requirements traceability.

The model of requirement systematization:

Organizing of the reference model of requirement systematization allows defining a model or framework for use in the formalization and systematization of information system`s technical requirements. The framework, for example, may consist of a structure (template) document, which will show the result of the requirement formalization and set of principles (approaches) to their classification.

At a conceptual level, the model consists of a hierarchy of levels of abstraction from the point of view of the professional position of each stakeholder. John. Zachman, in his architectural ontology calls these levels Reification Transformations. In cases where architectural components of information systems (especially at lower levels of abstraction) determine compliance with the ontology representations of the architecture information systems (viewpoints).

Description of model levels

The main task of the following levels provide a through and clear understanding of the objectives and tasks set at the level of the business owner (head), until the construction of the technological model, reflected in the logical program code executable files, ensuring business continuity.

Conceptual level: opinion of the external customer in the face of the business owner, Manager, process and business analyst.

Describes the purpose and objectives of the system and business logic. Business logic is the implementation of the subject area in the information system, which can have different kinds of views. In the general case, it is a holistic description of the characteristics of an automation object, which includes:

- overview of the automation object;
- description of tasks;
- functional components of the system.

These descriptions can be submitted in various forms, including in the form of models of business processes. Architectural level: the view of the system architect. Application architecture of an information system represent the needs of automated

business processes with functional components of the system. The practical implementation of this description may have different views. The easiest option – a list of group requirements. Group requirement is a logical grouping of similar requirements. In other words, the result of combining requirements into a single system.

System level shows a logical description of information system operation, namely the list of requirements and their relationships. With some approximation we can say that this level of system project level. System design is the overall technical performance of information system; it includes full functional model, information model and technical requirements. In addition, this section may define the scope and organization of work, required resources and project risks.

Technology level: the opinion of the developer.

Describes the technological model of the system in the form of specifications for the programmer. Specification defines, how information system works, as well as a detailed algorithm of its functioning.

The above model regulate hierarchy and interrelation of all system requirements (levels). Each element defined on different levels of interpretation for stakeholders. Thus, the relationship between requirements defined at the top level, uniquely shaped by the level below. This is the main rule of the model.

The first stage of the model application is the formalization of business logic of the future information system. The main task at this stage is the definition of the purpose and creation goal of information system. When system is designed for Russian customer, it is recommended to follow the requirements of State Standard 34. In this case, relevant are the following requirements:

- the purpose of the system will determine the type of automated activities and a list of automation objects on which we want to use it;
- the creation goal of the system should regulate the names and the required values of technical, technological or commercial object to be achieved by the establishment of the automated system.

From the context of the purpose description, it should be clear to the reader that the developed system will really improve the user's life by automating its activities. It is recommended to specify the criteria for assessing the achievement of system objectives. We rate the implementation of system goals.

Another approach to the description of goals can be a way of forming goals with SMART technology. When you define a target by this method, by default it contains the criteria to assess the achievement. In addition, there is another useful technique. He allows in interview of stakeholders properly formulate the objectives, building on many of the formulated SMART tasks. At the first stage interview, you must fill the table with all attributes. At the first stage interview, you must fill the table with all attributes. From the obtained set of tasks it is necessary to remove duplicates, find intersections and mutually exclusive, minimizing the list. With the help of such list, it is possible to formulate the purpose and creation goals of the system that will satisfy the entire volume of tasks.

Business processes model

If we want a single vision of the designed information system in the mind of all stakeholders, we need to formalize algorithms of actions of the developed information system. As we know, the algorithm is a sequential set of instructions that defines the procedure for solving the problem within a set time. To describe such algorithms, it is advisable to use models of business processes.

There are many methodologies for describing business processes used in different situations, however, the choice of a specific methodology in any case depends on the context of the problem being solved. The main requirements for the model are the visibility of order execution, as well as a clear description of the data and documents used in the process. The model fully and clearly describe the sequence of actions inside information system. The accuracy of the constructed model needs to be a level that will allow us to understand the logic of the system and go to the detailed generation of requirements.

State standard 34 standards say that business processes model can be represented in the form of complexes of tasks with appropriate breakdown on the functions performed by the system.

When the business processes model was built, you can see how the purpose and creation goals of the system at the top level fixed frame automated business process. At this stage it is obvious that the formalization of business processes is a key factor for making many management decisions during the negotiation and definition of project automation object design of its architecture.

The system usage

The next step of requirement systematization should ensure a common understanding of the concept of information system between the business analyst, architect and developer. This process determines the transition from the conceptual level to the architectural. To ensure this understanding best fits the description of use cases based on the developed model business processes.

The practice of creating use cases as a means to clarify the requirements to the behavior of information systems and business processes is widely used in the world practice. Use cases provide a comprehensive understanding of functional requirements, which support is required at all stages of the life cycle of the system, showing how it will apply the system in different situations.

At first glance, the idea of use seems simple. However, specifying the set of use cases, you first need to consolidate their level of detail. What purpose will describe this use case, and what will be the result? Setting goals and tasks for each requirement component allows us to determine the required level of detail and not to write too much, which saves time of formalizing the requirements.

The formalization of use cases in any case is not the result of information system design. Quite the contrary, it is one of the first steps to understand the functional requirements for the system. With the identification of the role of actors, entities and use cases it is necessary to start defining the functionality and users of future information systems.

Use cases define a set of ongoing and incorrect sequences that may occur in the interaction of the actor and the system to achieve a certain goal. A set of use cases allows the developer of a comprehensive understanding for the design of the application architecture implemented in the development phase.

Thus, according to the analysis purpose and goals description of the use cases you can define the key prerequisite of systematization of requirements: use cases are the link between understanding the achievement of system goals from the standpoint of all stakeholders, while implementing their business processes via architectural and functional requirements for information system.

The technical requirements document, containing the above-mentioned topics at this stage may be the meaningful content of the concept of creation of the information system. This document is the starting point to consolidate a common vision from all stakeholders to automated processes. Therefore, the general approaches to the definition of these sections was presented at this stage. We consider further the principles of formalization and systematization of requirements.

Requirement classification

Earlier it was described that the model of systematization determines the complex relationship of business processes, use cases and technical requirements. However, the set of group requirements would be in each case separately, starting from the reference model. For a particular information system during its design it is necessary to identify the unique set of groups, holistically describing the requirements of stakeholders.

In order to build relationships between requirements, as well as to improve the ease of use and perception, it is necessary to define codes for all descriptions of information systems. For convenience, the classification result it recommended to place at the beginning of the document that allows you to immediately install the technical content of the document. Table shows 2 an example of requirement classification.

In fact, the structure is the main content of the document structure, because it includes all the sections of the description an information system.

Table B.2 – Classification of description elements.

Code	Element description	Section
BP	Business processes model (description of group tasks)	Description of the business logic
U	The description of the classes and characteristics of users	
V	Description of use cases	
FA	The requirement for the functions (tasks) performed by the system	Basic system requirements
I	Requirements for the user interface	
D	Requirements for the description of the data	
R	The reporting requirements	
AR	Requirements for access rights	
A	The requirement for administration, access control and security system	
P	Requirements for integration	
F	General functional requirements	Additional system requirements
C	Requirements for the reference and classification system	
IS	Security requirements	
RD	Reliability requirements	
T	Testing requirements	
M	Requirements for software development	Requirements for types of support
TS	Requirements for technical support	
SR	The software requirements	

Requirement relationships

At a conceptual level, the relationship between types of requirements describes the relationship between elements in the model "entity-relationship". The relationships between elements can have both direct and indirect communication. A direct link represents the relationship between the elements of the groups of requirements: «one-to-one», «one-to-many».

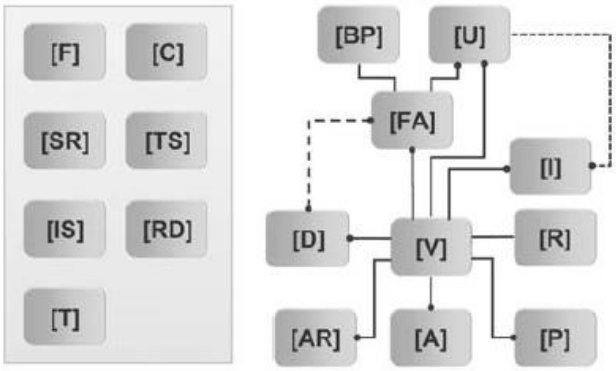

Figure 1 – Requirement relationships.

The central element of the requirements are requirements for use cases [V]. A description of each of the elements (specific use) refers to:

- the description of the classes and characteristics of users [U] (relationship type one-to-one, as each use case is defined for a specific user);
- requirements to functions (tasks) performed by the system [FA];
- the reporting requirements [R];
- the requirements for user interfaces [I];
- the requirements for administration, access control and system security [A];
- requirements of access rights [AR];
- requirements to description (composition) of data [D].

Some of the requirements are recommended to be determining with direct link. For example, the requirements for testing system needs to be related to the use cases or requirements to the algorithms of functions (depending on the approach to testing). It is recommended to define direct relationship between various elements of the

requirements document by cress-references. To understand which specific element we refer to, you should define the approach to coding elements.

The purpose of the master's thesis is to develop software for requirements management system, which can create pdf file with all technical documentation of future information system.