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

Школа информационных технологий и робототехники
Направление подготовки 09.04.02 Информационные системы и технологии
Отделение школы (НОЦ) информационных систем и технологий

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема работы
Разработка геоинформационной подсистемы для программного комплекса SCADA Infinity

УДК 004.422.83.55.002.6

Студент

Группа	ФИО	Подпись	Дата
8ИМ6А	Айдаров Шукурдин Бахтиярович		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор ОИТ	Ким Валерий Львович	д.т.н., профессор		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН	Старикова Екатерина Васильевна	к.филос.н., доцент		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОКД	Король Ирина Степановна	к.х.н., доцент		

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Профессор ОИТ	Марков Николай Григорьевич	д.т.н., профессор		

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Код результатов	Результат обучения (выпускник должен быть готов)
	Обще профессиональные 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	Выполнять инновационные инженерные проекты по разработке аппаратных и программных средств автоматизированных систем различного назначения с использованием современных методов проектирования, систем автоматизированного проектирования, передового опыта разработки конкурентно способных изделий.
Р6	Планировать и проводить теоретические и экспериментальные исследования в области проектирования аппаратных и программных средств автоматизированных систем с использованием новейших достижений науки и техники, передового отечественного и зарубежного опыта. Критически оценивать полученные данные и делать выводы.
Р7	Осуществлять авторское сопровождение процессов проектирования, внедрения и эксплуатации аппаратных и программных средств автоматизированных систем различного назначения.
	Общекультурные компетенции

Р8	Использовать на практике умения и навыки в организации исследовательских, проектных работ и профессиональной эксплуатации современного оборудования и приборов, в управлении коллективом.
Р9	Осуществлять коммуникации в профессиональной среде и в обществе в целом, активно владеть иностранным языком, разрабатывать документацию, презентовать и защищать результаты инновационной инженерной деятельности, в том числе на иностранном языке.
Р10	Совершенствовать и развивать свой интеллектуальный и общекультурный уровень. Проявлять инициативу, в том числе в ситуациях риска, брать на себя всю полноту ответственности.
Р11	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»**

Школа информационных технологий и робототехники
Направление подготовки (специальность) 09.04.02 Информационные системы и технологии
Отделение школы (НОЦ) информационных систем и технологий

УТВЕРЖДАЮ:
Руководитель ООП

_____ _____ _____
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

Магистерской диссертации

(бакалаврской работы, дипломного проекта/работы, магистерской диссертации)

Студенту:

Группа	**ФИО**
8ИМ6А	Айдаров Шукурдин Бахтиярович

Тема работы:

Разработка геоинформационной подсистемы для программного комплекса SCADA Infinity	
Утверждена приказом директора (дата, номер)	

Срок сдачи студентом выполненной работы:	

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе	Список потребностей

Перечень подлежащих исследованию, проектированию и разработке вопросов	Обзор существующих решений, картографических сервисов и инструментов разработки геоинформационных систем. Анализ вариантов реализации системы. Проектирование и реализация системы.
Перечень графического материала	
Консультанты по разделам выпускной квалификационной работы	
Раздел	**Консультант**
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Старикова Е.В.
Социальная ответственность	Король И. С.
Раздел на иностранном языке	Комиссарова О.В., Мирошниченко Е.А.
Названия разделов, которые должны быть написаны на русском и иностранном языках:	
Обзор существующих решений и картографических сервисов, анализ вариантов реализации	

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	

Задание выдал руководитель:

Должность	**ФИО**	**Ученая степень, звание**	**Подпись**	**Дата**
Профессор ОИТ	Ким Валерий Львович	д.т.н., профессор		

Задание принял к исполнению студент:

Группа	**ФИО**	**Подпись**	**Дата**
8ИМ6А	Айдаров Шукурдин Бахтиярович		

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

Школа информационных технологий и робототехники
Направление подготовки (специальность) 09.04.02 Информационные системы и технологии
Уровень образования Магистр
Отделение школы (НОЦ) информационных систем и технологий
Период выполнения осенний / весенний семестр 2017/2018 учебного года

Форма представления работы:

Магистерская диссертация

(бакалаврская работа, дипломный проект/работа, 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)
	Аналитический обзор	
	Проектирование системы	
	Реализация системы	
	Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	
	Социальная ответственность	
	Обязательное приложение на иностранном языке	
	Оформление пояснительной записки	

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор ОИТ	Ким Валерий Львович	д.т.н., профессор		

СОГЛАСОВАНО:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Профессор ОИТ	Марков Николай Григорьевич	д.т.н., профессор		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА «ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8ИМ6А	Айдаров Шукурдин Бахтиярович

Школа	Информационных технологий и робототехники	**Отделение**	Информационных систем и технологий
Уровень образования	магистратура	**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	Человеческие ресурсы – 3 чел
2. Нормы и нормативы расходования ресурсов	Нормы рабочего времени в 2018 году: Количество календарных дней – 365; Количество рабочих дней – 247; Количество выходных и праздничных дней – 119.
3. Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования	Тарифы страховых взносов в 2018 году: Пенсионный фонд РФ – 22%; Фонд социального страхования РФ – 2,9%; Федеральный фонд обязательного медицинского страхования – 5,1%.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Оценка коммерческого и инновационного потенциала НТИ	1.1 Комплексный анализ слабых и сильных сторон, возможностей и угроз для реализации проекта. 1.2 Описание качества разработки и ее перспективности на рынке.
2. Планирование проектных работ	2.1 Определение организационной структуры проекта. 2.2 Определение трудоемкости выполнения работ. 1.3 Разработка календарного плана выполнения проекта.
3. Определение ресурсной, финансовой, экономической эффективности	3.1 Расчет бюджета проекта. 3.2 Определение эффективного варианта исполнения.
Перечень графического материала *(с точным указанием обязательных чертежей):*	
1. Карта сегментирования отечественного рынка 2. Диаграмма Исикавы 3. Диаграмма Гантта	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН	Старикова Екатерина Васильевна	к.филос.н., доцент		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ИМ6А	Айдаров Шукурдин Бахтиярович		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА «СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8ИМ6А	Айдаров Шукурдин Бахтиярович

Школа	Информационных технологий и робототехники	**Отделение**	Информационных систем и технологий
Уровень образования	магистратура	**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 является геоинформационная система, предназначенная для отображения технологических объектов, информации о параметрах и ходе технологического объекта на географической карте.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Производственная безопасность 1.1. Анализ выявленных вредных факторов при разработке и эксплуатации проектируемого решения. 1.2. Анализ выявленных опасных факторов при разработке и эксплуатации проектируемого.	Производственная безопасность при выполнении работ по разработке и эксплуатации геоинформационной системы для отображения технологических объектов, информации о параметрах и ходе технологического объекта на географической карте. 1.1 К группе вредных факторов отнесено: • превышение допустимых показателей электромагнитного излучения. 1.2 К группе опасных факторов отнесено: • опасность поражения электрическим током.
2. Экологическая безопасность	2.1 Влияние объекта исследования на окружающую среду: • образование отходов.
3. Безопасность в чрезвычайных ситуациях	3.1 Возможные чрезвычайные ситуации при разработке и эксплуатации проектируемого решения: • пожар.
4. Правовые и организационные вопросы обеспечения безопасности	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); • ГОСТ 12.2.032-78 Система стандартов безопасности труда (ССБТ). Рабочее место при выполнении работ сидя. Общие эргономические требования; • ГОСТ Р 50923-96 Дисплеи. Рабочее место оператора. Общие эргономические требования и требования к производственной среде. Методы измерения.

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

ДДолжность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОКД	Король Ирина Степановна	д.х.н., доцент		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ИМ6А	Айдаров Шукурдин Бахтиярович		

РЕФЕРАТ

Выпускная квалификационная работы содержит 154 страницы, 7 таблиц, 17 рисунков, 8 приложений.

Ключевые слова: геоинформационная система, SCADA-система, технологический объект, технологическая информация, маркеры, мониторинг.

Объектом исследования является геоинформационная система для отображения технологических объектов, информации о параметрах и ходе технологического процесса на географической карте.

Цель работы – расширение функциональности программного комплекса SCADA Infinity функциями отображения технологических объектов, информации о параметрах и ходе технологического процесса на географической карте.

В процессе работы проводились обзор существующих продуктов, источников картографических материалов, инструментов разработки геоинформационных систем, а также проектирование и разработка системы.

ТЕРМИНЫ И СОКРАЩЕНИЯ

В данной работе применены описанные ниже термины и сокращения с соответствующими определениями.

ГИС: геоинформационная система.

SCADA: программный пакет, предназначенный для разработки или обеспечения работы в реальном времени систем сбора, обработки, отображения и архивирования информации об объекте мониторинга или управления.

OPC UA: спецификация, определяющая передачу данных в промышленных сетях и взаимодействие устройств в них.

InfinityServer: сервер обработки данных.

Infinity HMI: программа разработки и просмотра графических мнемосхем.

CRUD: сокращение от английского create, read, update, delete – «создать, прочесть, обновить, удалить».

API: сокращение от английского Application Programming Interface – программный интерфейс приложения.

REST: сокращение от английского Representational State Transfer – архитектурный стиль взаимодействия компонентов распределенного приложения в сети.

ActiveX: технология разработки программных компонентов, пригодных к использованию из программ, написанных на разных языках программирования.

ПК: программный комплекс.

UML: сокращение от английского Unified Modeling Language – унифицированный язык моделирования.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В нынешнем мире информационных технологий сложно представить современное производство без использования *SCADA*-системы. Основными функциями данных систем являются сбор, первичная обработка, накопление, архивирование и отображение информации о параметрах и ходе технологического процесса, обнаружение аварийных ситуаций и оперативное управление технологическим процессом. Внедрение и эксплуатация подобных систем позволяет контролировать и управлять сложными технологическими процессами, экономить средства, повышать эффективность и безопасность производства.

Одной из таких *SCADA*-систем является отечественный программно-инструментальный комплекс для реализации систем управления технологическими процессами *SCADA Infinity*, относящаяся к разработкам компании ЭлеСи.

Технологический процесс ряда отраслей характерен наличием распределенных и мобильных технологических объектов. Особенно это характерно для предприятий, работающих в таких отраслях, как нефтегазовая, транспортная, энергетическая и т.д. Для подобных технологических процессов существует потребность в ведении мониторинга за территориально распределенными и мобильными объектами и их состояниями, для обеспечения оперативного реагирования при возникновении на них событий.

До недавнего времени не было потребности в данной функциональности, но с появлением новых рынков и инновационных проектов появилась потребность в расширении ПК *SCADA Infinity* функциональностью отображения технологических объектов и информации о состоянии технологического процесса на географической карте.

Целью данной системы является расширение возможностей ПК *SCADA Infinity* функциями отображения технологических объектов и информации о состоянии технологического процесса на географической карте в ГИС-ориентированном интерфейсе. Также обеспечение конкурентного преимущества перед отечественными *SCADA*-система и поддержание общего тренда развития зарубежных *SCADA*-систем.

Для достижения поставленной цели требуется решить ряд задач: изучение и анализ предметной области, выявление и документирование требований к системе, проектирование архитектуры, пользовательского интерфейса и базы данных, а также реализация намеченных функциональных возможностей.

В настоящей работе рассматриваются вопросы, возникающие при проектировании и реализации проектов по созданию геоинформационных систем, предназначенные для визуализации информации о состоянии технологического процесса и обеспечению мониторинга технологических объектов. Особое внимание уделено определению архитектурного стиля, проектированию и реализации архитектурного решения системы.

В первом разделе проведены обзоры существующих решений, источников картографических материалов и инструментов разработки геоинформационных системы. А также выполнен анализ вариантов решения, с учетом всех ограничений и требований, задачи отображения технологических объектов и информации о состоянии технологического процесса на географической карте.

Во втором разделе описан этап проектирования системы. Определены типы приложений, архитектурный стиль, основные варианты использования и развертывания. Приведены структурные схемы компонентов системы, эскизы пользовательского интерфейса и диаграммы *UML*.

В третьем разделе рассмотрен процесс реализации системы. Приведены диаграммы зависимостей сборок компонентов, фрагменты исходного текста и структура модульных тестов.

В четвертом разделе приведены результаты оценки финансового менеджмента и ресурсоэффективности и ресурсосбережения проекта.

В пятом разделе представлены результаты анализа социальной ответственности.

1 АНАЛИТИЧЕСКИЙ ОБЗОР

1.1 Обзор существующих решений

Информация о существующих аналогах, и качественный анализ их достоинств и недостатков во многих случаях может стать основой успеха проекта. Этот этап позволяет определить удавшиеся концепции продуктов-конкурентов, которые стоит реализовать в разрабатываемой системе. А также избежать ряда принятых неблагоприятных решений, ставшие причиной неудовлетворительных отзывов пользователей о продукте, или даже их провала.

Ниже приводится краткое описание наиболее популярных *SCADA*-систем, в которых объедены функции сбора данных и диспетчерского управления с функциями геоинформационных систем (ГИС).

1.1.1 GISize Wonderware

Зарубежная компания Wonderware, занимающая разработкой и поставкой программных решений для управления технологическими операциями, предоставляет современный инструмент *GISize* для разработки решений в области автоматизации. *GISize* объединяет богатый набор функциональных возможностей с технологиями виртуальной реальности, 3D-визуализации, мобильными и геоинформационными технологиями.

Широкий набор функций *GISize* для работы с геоданными позволяют создавать полноценные специализированные слои для технологических объектов и определять на них объекты с пространственным и атрибутивными данными. Атрибутивные данные могут представлять собой информацию состоянии технологического процесса, интеграция с технологическим процессом

осуществляется средствами платформы диспетчерского управления *InTouch OMI* [10], относящая к разработкам *Wonderware*.

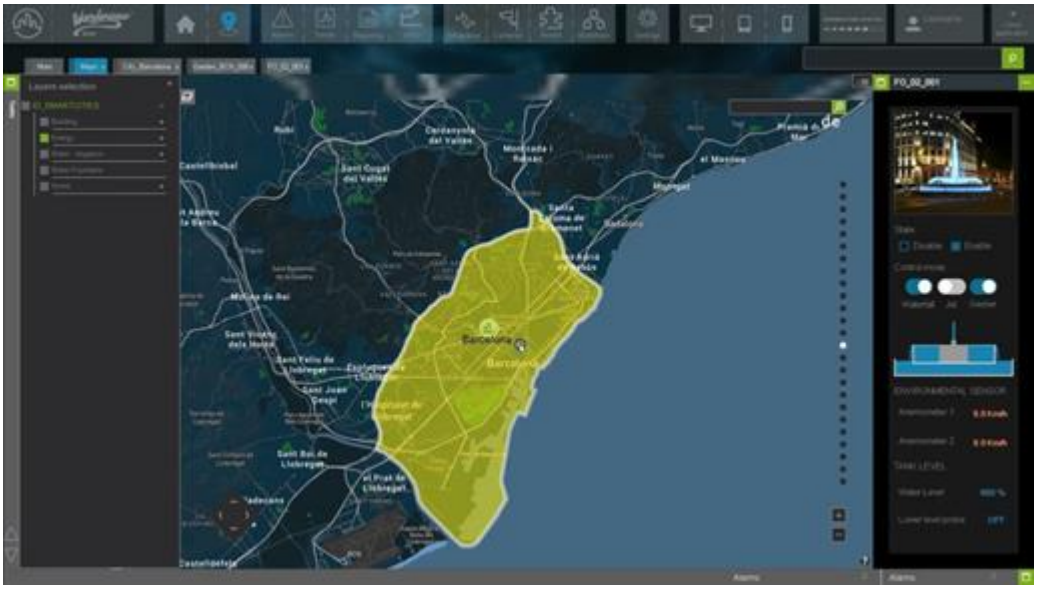


Рисунок 1.1 – Отображение карты

Функциональные возможности *GISize* в областях виртуальной реальности и 3D-визуализации позволяют создавать виртуальные модели производственного оборудования или помещений. Эти среды могут использоваться для различных целей, таких как симуляция систем и расширенное использование информации в промышленных контекстах, интерактивное управление оборудованием при помощи устройств виртуальной реальности, позволяющие отобразить информацию, связанную с местоположением оборудования или человека, тем самым повысить эффективность управления ими (рисунок 1.2).

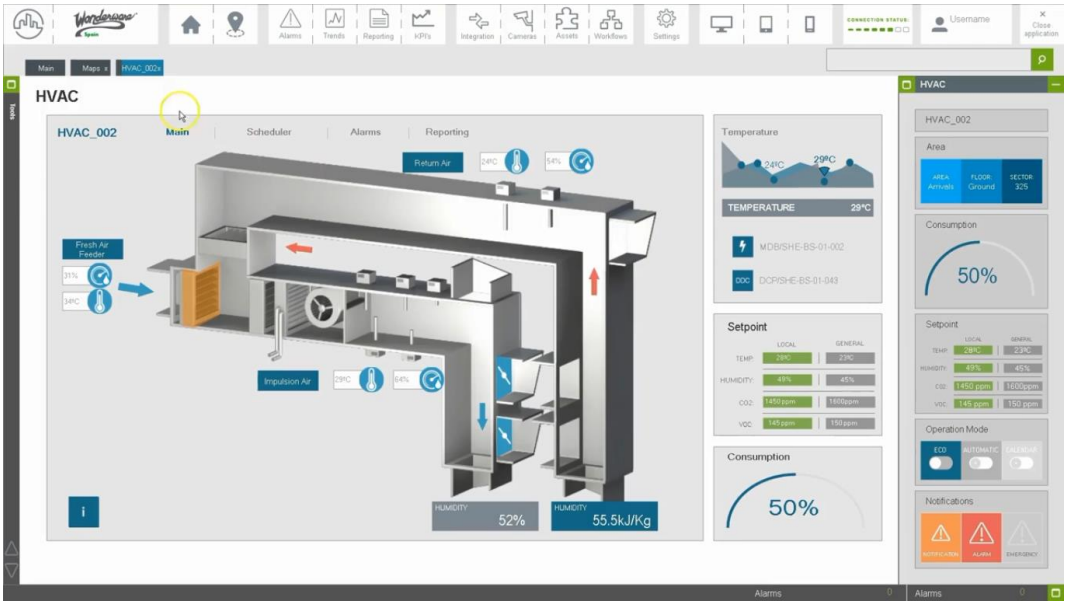


Рисунок 1.2 – Интерактивное управление оборудованием

Для создания слоев карты и интерактивных проектов в GISize предусмотрена мощная интегрированная среда разработки с простым и удобным пользовательским интерфейсом, который представлен на рисунке 1.3. В среде предусмотрено создание различных шаблонов объекта, позволяющие ускорить процесс разработки. При помощи данной среды можно создавать решения любого масштаба для любых отраслей промышленности.

Следует также отметить возможности *GISize* интеграции с различными поставщиками геоданных с помощью стандартных *WMS* и *REST* сервисов, и создания карт в векторном формате.

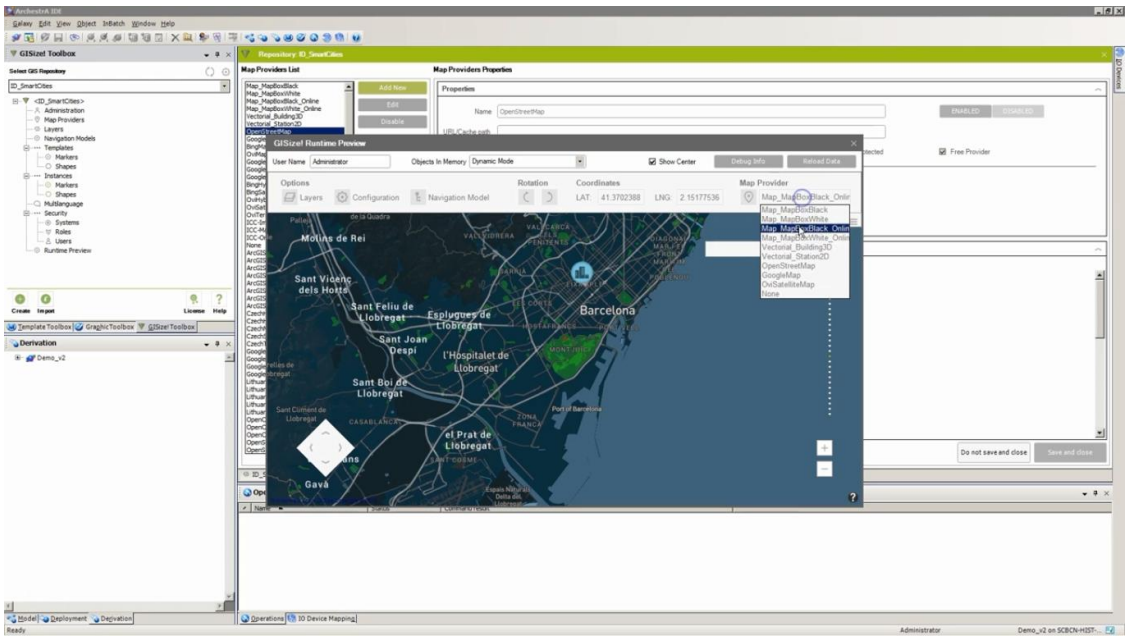


Рисунок 1.3 – Встроенная IDE GISize

Интуитивно-понятный и удобный пользовательский интерфейс, 3D-визуализация и виртуальная реальность, и широкий набор функциональных возможней *GISize* непременно можно отнести к достоинствам этого продукта, но функциональная мощность и качество данного продукта стали причиной его высокой стоимости.

В качестве другого недостатка стоит выделить тесную зависимость компонентов *GISize* с продуктами *Microsoft*. Платформа *InTouch* рассчитана на ОС *Microsoft* версии *Windows 7* и более поздние, совместима с СУБД *Microsoft* начиная с версии *SQL Server 2008 SP3*.

1.1.2 Геоинформационный модуль ПК PcVue

Программный комплекс сбора данных и диспетчерского управления технологическими процессами PcVue, относится к разработкам зарубежной компании *ARC Informatique* [11]. Данный ПК объединяет технологию картографирования с классическими функциями *SCADA*-систем и имеет в своем

составе геоинформационный модуль, позволяющий отобразить технологические объекты на географической карте в виде маркеров различных типов. ГИС-ориентированный интерфейс модуля, представлен на рисунке 1.4.

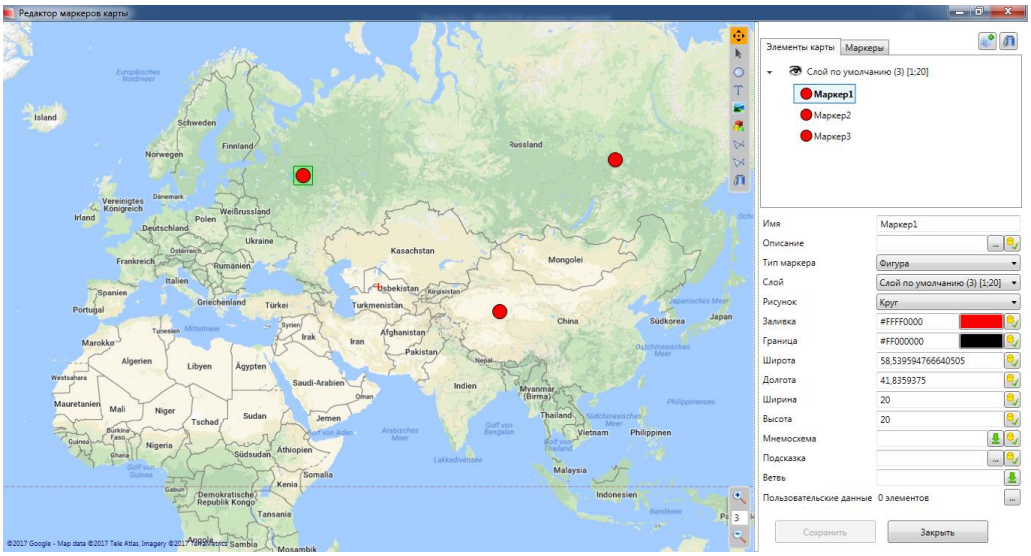


Рисунок 1.4 – ГИС-модуль PcVue

В *PcVue* определены маркеры следующих типов: текстовый, символьный, фигурный, маркер типа «изображение», линейный и полигонный маркер. Маркеры первых четырех типов являются точечными географическими объектами, линейный маркер представляет собой одномерный объект, описываемый двумя координатами, а полигонный маркер – двумерный объект, описывающий некоторую область.

Каждый тип маркера обладает характерным ему набором свойств. Значение определенного свойства может быть установлено двумя способами. При первом способе значение свойства определяет пользователь, в поле ввода.

При втором способе в качестве источника значения, пользователь указывает узел в дереве переменных сервера сбора данных, а система сама выполняет все операции, связанные с чтением и установкой актуального

значения свойству маркера. Сервер сбора данных представляет собой стандартный *OPC* сервер. Окно выбора узла из дерева переменных, приведено на рисунке 1.5.

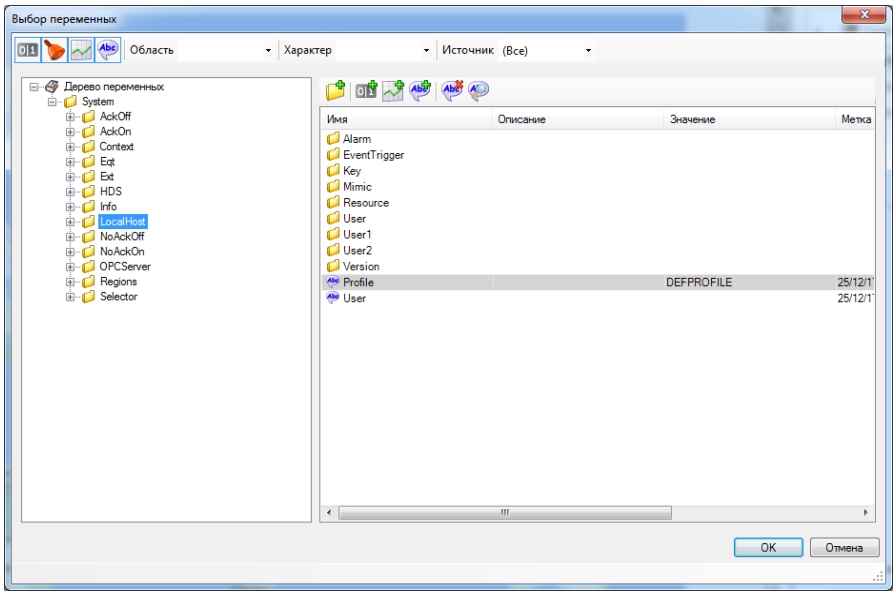


Рисунок 1.5 – Выбор переменных

Из недостатков стоит отметить неудобство пользовательского интерфейса для работы с маркерами, процесс размещения маркеров на карте требует максимальной сосредоточенности. Причиной этому стали плохо проработанные курсоры и элементы управления маркерами. Также еще одним недостатком является отсутствие возможности доступа к геоданным во внутренней сети, расположенный за прокси-сервером. Но данный недостаток компенсируется возможностью экспорта геоданных в файл, который может быть перемещен и использован на компьютере, находящийся за межсетевым экраном.

К достоинствам данного продукта, следует отнести большое количество поддерживаемых картографических сервисов и простоту развертывания. В текущей версии *PcVue 11* поддерживает *Google Maps, Bing Maps, OpenStreetMap* и ряд других сервисов.

1.1.3 Астра: ГИС-SCADA

Программный комплекс для построения систем диспетчеризации и автоматизации инженерных сетей *ГИС-SCADA,* относится к разработкам отечественной научно-производственной компании Астра [12]. Данный ПК представляет собой сконфигурированный и оптимизированный инструмент для теплоснабжающих, сервисных и управляющий компаний, водоканалов и муниципальных органов управления. В составе *ГИС-SCADA* имеются интерактивные карты с административными и инженерными слоями, мнемосхемы с возможностью управления, расчетные и аналитические модули, а также интуитивно-понятный и простой редактор проектов.

Геоинформационный модуль *ГИС-SCADA* содержит стандартные слои административной карты в растровом и векторных форматах с детализацией для разных масштабов, а также специализированные слои для отраслей тепло- и водоснабжения.

Для представления технологических объектов в *ГИС-SCADA* существует предопределенный набор специализированных шаблонов, ориентированные на одну из перечисленных выше отраслей. Интеграция с технологическим процессом может быть осуществлена при помощи компонентов, входящих в состав *ГИС-SCADA* или средствами стандартных OPC серверов.

НПК Астра предлагает готовые продукты на основе *ГИС-SCADA* для решения задач учета тепловой энергии, газа, электричества и воды, а также ряд других продуктов.

Из числа недостатков *ГИС-SCADA* следует выделить сложность адаптации данного продукта для решения задач в других отраслях, сложность развертывания, а также отсутствие кроссплатформенности. Компоненты *ГИС-SCADA* ориентированы на платформу Microsoft.

Основным преимуществом данного продукта является низкая стоимость относительно других подобных продуктов.

1.2 Варианты решения задачи и их анализ

Существуют различные варианты для решения задач отображения технологической информации на географической карте. Одним из вариантов является адаптация полнофункциональной геоинформационной системы для решения поставленных задач, другим вариантом является создание информационных систем, способных извлекать технологическую информацию и представлять их на карте.

1.2.1 Применение существующей ГИС

Одним из вариантов является адаптация универсальной, полнофункциональной ГИС для решения задач отображения технологических объектов и информации о состоянии технологического процесса на географической карте. В процессе адаптации потребуется разработать соответствующие модули для выбранной ГИС и найти пути интеграции с программой разработки и просмотра графических мнемосхем *Infinity HMI*. Интеграция с *Infinity HMI* является одним из требований к разрабатываемой системе, полный список технических требований приведен в Приложении 1.

Более того современные универсальные ГИС обладают огромной функциональной мощностью, что не требуется в контексте поставленной задачи. Также стоит отметить, что подобные ГИС требуют дополнительных затрат по их специфической поддержке, включающую установку, настройку, сопровождение и т.д. С учетом вышесказанного данный вариант создания системы был отвергнут.

1.2.2 Разработка настольного приложения

Другим вариантом создания системы является разработка полнофункционального настольного приложения средствами программных ГИС-компонентов, предоставляющие инструменты для работы с картографическими данными. Этот подход позволяет создать систему с необходимым функционалом и специализированным пользовательским интерфейсом для решения задач настоящей системы. Также в этом подходе, предоставляется возможным реализовать интеграцию с *Infinity HMI* посредством технологии *ActiveX*.

Интеграция с *Infinity HMI* средствами *ActiveX*, предполагает создание настольного приложения в виде элемента управления *ActiveX*. Средой выполнения для этого элемента управления является контейнер *ActiveX*, реализованный в *Infinity HMI*. Так как в данном элементе управления будет реализовано вся функциональность системы, то ему большое количество ресурсов памяти и процессора. Однако наличие в *Infinity HMI* подобных компонентов снижают его производительность. Это создаст пользователю огромные неудобства и может нарушит работу других компонент в *Infinity HMI*. Поэтому данный вариант создания системы также был отвергнут.

1.2.3 Разработка настольного клиента элемента управления ActiveX и веб-сервера

С учетом требований к интеграции, приведенные в приложении 1, и ограничений *Infinity HMI*, появился третий вариант создания системы, который и был избран. Этот вариант предполагает разбиение системы на три компонента. Первым компонентом является облегченный элемент управления *ActiveX*, в

котором сосредоточена функциональность, связанная с отображением информации на географической карте.

Второй компонент представляет собой насыщенное настольное приложение, в котором сосредоточена функциональность редактора маркеров и ряд других дополнительных функций. Данное приложение получило название среды разработки проектов, так как с его помощью создаются маркеры и определяется информация, которая будет представлена на карте.

Третий компонент – веб-сервер, он является основным звеном системы, обеспечивающий централизованное хранение и доступ к данным остальным компонентам системы.

Преимущества этого варианта заключаются в следующем:

- тонкий элемент управления *ActiveX* не замедляет и не нарушает работу *Infinity HMI*;
- функциональность, необходимая только для создания маркеров сосредоточена в отдельном приложение, которое без необходимости не запускается и не тратит ресурсы компьютера;
- наличие веб-сервера позволяет гибко добавлять в систему новых клиентов, за счет стандартизированного интерфейса взаимодействия.

1.3 Обзор источников картографических данных

В данном разделе проводится краткий обзор популярных картографических сервисов с целью определения основного источника геоданных, способ интеграции с ними и возможных ограничений.

1.3.1 Информационно-поисковый сервис Google Maps

Информационно-поисковый веб-сервис *Google Maps* компании *Google*, предоставляет доступ к огромному объему картографического материала в векторном, растровом или гибридном режимах [13].

Google предлагает огромное количество инструментов и программных интерфейсов для использования данных и услуг сервиса *Google Maps*, ориентированные на различные языки программирования и платформы. Наиболее популярными являются *JavaScript API*, *Static Maps API*, *Android API* и *SDK for IOS*, позволяющие интегрировать карты в приложения для различных платформ. Стоит отметить, программный интерфейс *Static Maps API* является более универсам, так как он предполагает взаимодействие по стандартному протоколу HTTP(s) и не ориентирован на конкретную платформу [13].

Материалы и услуги сервисов Google Maps предоставляются бесплатно для некоммерческого применения, но с рядом ограничений на использования. Для корпоративного развертывания существует определенная лицензионная политика [13].

1.3.2 Поисково-информационный картографический сервис Яндекс.Карты

Одним из геоинформационных сервисов российской разработки является поисково-информационный картографический сервис *Яндекс.Карты*, относящийся к разработкам компании *Яндекс*. Данный сервис предоставляет геоданные и позволяет выполнять поиск по карте, получать информацию о пробках, прокладывать маршруты и панорамы улиц крупных городов и т.д [14].

Подобно Google, Яндекс предоставляет программный интерфейс и инструменты для интеграции с сервисом. Программный интерфейс образуют

набор сервисов, позволяющие использовать картографические материалы и технологии Яндекса в различных приложениях. Данный интерфейс состоит из *JavaScript API* и *HTTP API, API* поиска по организациям и *Static API* [14].

JavaScript API представляет собой набор программных компонентов на языке JavaScript, позволяющие размещать на страницах сайта интерактивные карты. Эти компоненты позволяют встроить в приложение карту с поиском по названию географического объекта и организациям, и использовать ряд других функций, доступные на *Яндекс.Картах*.

HTTP API предоставляет доступ к отдельным возможностям сервиса *Яндекс.Карт*, таким как *геокодер*, средствами протокола *HTTP(s)*.

Static API позволяет получить растровое изображение нужного фрагмента карты, при помощи запросов *HTTP(s)*. Добавляя в запрос разные параметры и задавая их значения, можно определить центр карты, её размер и область показа, отметить нужные объекты и отобразить пробки.

Яндекс предоставляет свободный доступ к картографическим материалам в некоммерческих целях, но с некоторыми ограничениями. В бесплатной версии *API Яндекс.Карт* можно получать изображения карт размерами не более 600x450 пикселей. Такое изображение обязательно должно быть размещено на общедоступном сайте или в приложении. Также бесплатный *API* нельзя использовать для мониторинга транспорта и в закрытых системах. Для коммерческого использования существует платная версия *API* с различными тарифами [14].

1.3.3 Картографический сервис Bing Maps

Bing Maps является картографическим сервисом компании *Microsoft* [15]. Интеграция с *Bing Maps* может быть осуществлена средствами *REST* сервисов или с помощью программных компонентов.

REST сервисы разделены на две группы. К первой группе относятся сервисы *Spatial Data Services* с помощью которых можно геокодировать, сохранить и запросить геоданные, используя стандартные *HTTP*(s) запросы.

Вторая группа сервисов – *REST Services*, предоставляющие такие операции, как создание статических карт с маркерами, геокодированными адресами и маршрутами.

Кроме сервисов, *Microsoft* предлагает набор готовых программных компонентов, позволяющие создавать настольные, мобильные и веб-приложения. Для разработки настольных приложений предназначены элементы управления *Bing Maps WPF Control*, обеспечивающие взаимодействие с сервисом *Bing Maps*, визуализацию и управление картографическими данными, добавление маркеров, анимации и различных фигур на карту, а также ряд дополнительных функций.

Для создания веб-приложений предназначен высокопроизводительный, удобный и многофункциональный набор элементов управления *Bing Maps Version 8 Control*. Этот набор позволяет выполнять кластеризацию маркеров, строить тепловые карты, создавать анимации, а также он поддерживает форматы *GeoJSON* и *GeoXML* [15].

Использование картографических данных Bing Maps предполагает заключение одной из лицензионных соглашений Microsoft [15].

1.3.4 Картографический сервис OpenStreetMap

OpenStreetMap – популярный некоммерческий картографический веб-сервис. Картографические данные *OpenStreetMap* созданы на основе данных с *GPS* устройств, аэрофотографии, видеозаписи, спутниковые снимки и панорамы улиц, предоставленные некоторыми компаниями. В основе сбора данных

положен принцип «Википедии», позволяющий каждому зарегистрированному пользователю вносить изменения в картографические данные.

Преимуществом сервиса является свободный доступ к геоданным, ставший основной причиной его популярности. Картографические материалы сервиса распространяются на условиях свободной лицензии *Open Database License* [16], предоставляющий право свободно распространять, изменять и использовать эти данные в приложениях, в том числе коммерческих.

Подобно сервисам *Google Maps*, *Bing Maps* и *Яндекс.Карты*, *OpenStreetMap* предоставляет *REST API*, позволяющий другим системам интегрироваться с ним с помощью стандартных *HTTP*-запросов.

Из числа недостатков *OpenStreetMap* следует выделить наличие неточностей, из-за принципа «Википедии», в картографических данных.

1.4 Обзор инструментов разработки геоинформационных систем

В данном разделе приводится краткое описание инструментов, позволяющие разрабатывать ГИС-приложения. Рассматриваются полноценные универсальные ГИС и программные библиотеки с открытым исходным кодом.

Основной целью данного раздела является определение подходящего инструменте для работы с картографическими сервисами и их материалами.

1.4.1 Инструменты ArgGis

Американская компания *ESRI Inc* [19] предлагает программные инструменты с огромной функциональной мощностью для разработки геоинформационных систем. К наиболее популярным относятся инструменты для создания настольных приложений *ArgGis Engine* и для разработки ГИС-серверов *ArgGis for Server*.

ArgGis Engine позволяет разрабатывать как собственные ГИС-приложения, так и добавлять новые функции в продукты *ESRI*. Основу данного инструмента образуют две составляющие: инструменты для разработки *ArcGIS Engine Developer Kit* и компоненты выполнения *ArgGis Engine*.

ArgGis for Server предоставляет все необходимое для разработки ГИС-серверов, позволяющие формировать, поддерживать и предоставлять различные картографические материалы через сеть. Данный инструмент позволяет разрабатывать различные геоинформационные веб-приложения и сервисы для доступа из веб-браузеров, настольных и мобильных клиентов.

ESRI предлагает качественную документацию и техническую поддержку своих продуктов.

Качество и обилие функциональных возможностей инструментов *ESRI* определили их стоимость. Немногие организации способны позволить себе инструменты подобного уровня.

1.4.2 MapXtreme .Net

Комплект разработчика программного обеспечения *MapXtreme .Net* фирмы *Pitney Bowes Software Inc* [18], позволяет разрабатывать настольные и веб-приложения. Этот инструмент тесно интегрирован с платформой *Microsoft .Net*, и полностью состоят из объектной модели этой платформы. В составе *MapXtreme .Net* содержатся программные компоненты для технологий *Windows Forms* и *WPF*. Эти компоненты позволяют визуализировать картографические материалы, манипулировать, анализировать, производить вычисления над ними, а также выполнять ряд дополнительных операций.

Компоненты, предназначенные для создания клиентских приложений, позволяют взаимодействовать со стандартными *WMS* и *WFS* сервисами для обмена данными.

Основным преимуществом *MapXtreme .Net* является высокая скорость разработки за счет готовых к использованию и удобных программных компонентов.

В качестве недостатков следует отметить высокую стоимость, отсутствие качественной документации и технической поддержки, а также как следствие тесной связи с платформой *Microsoft*, отсутствие кроссплатформенности [18].

1.4.3 Quantum GIS

Свободная кроссплатформенная универсальная настольная ГИС *Quantum GIS* (*QGIS*) предоставляет широкий набор функции для работы геоданными. *QGIS* – это одна из самых популярных универсальных ГИС, которая распространяется свободно и предоставляет широкий набор инструментов для работы с геоданными [21].

QGIS поддерживает огромное количество форматов и источников геоданных, и предоставляет инструменты анализа, визуализации и редактирования геоданных. В настоящий момент поддерживаются пространственные таблицы *PostgreSQL* и *SpatiaLite*, векторные и растровые форматы, *GeoJSON*, я также ряд других.

Из числа преимуществ стоит выделить набор инструментов, позволяющие решать задачи анализа, исследования и управления данными. *QGIS* предоставляет возможность использовать инструменты анализа, выборки, геопроцессинга, управления геометрией и базами данных. Также в нем присутствует возможность создавать карты и исследовать пространственные данные, используя такие инструменты и функции, как перепроецирование «на лету», подписывание объектов, определение и выборка объектов, редактирование, просмотр и поиск атрибутов, экспортировать данные, а также ряд дополнительных функций.

Для адаптации *QGIS* к особым требованиям существует набор программных библиотек, позволяющие изменять и расширять возможности данной универсальной ГИС, а также создавать полноценные приложения на базе *QGIS*. На данный момент библиотеки существуют для языков программирования *C++* и *Python*.

В качестве недостатка следует выделить отсутствие локализованных элементов пользовательского интерфейса и сложность конфигурирования.

1.4.4 XtraMap

Компания *DevExpress* предлагает инструменты для разработки геоинформационных настольных и веб-приложений, в виде готовых к использованию программных компонент для таких технологий, как *Windows Forms, WPF, ASP.Net и ASP.Net Core* [23]. Данные инструменты обладают всеми необходимыми функциями визуализации, анализа и манипулирования геоданными.

Преимуществом инструментов *DevExpress* является качественные, локализованные и современные программные компоненты, позволяющие за короткие сроки разработать приложение.

Стоимость годовой подписки на линейку продуктов *DevExpress* составляет 2200 долларов США. Она включается в себя техническую поддержку, доступ к исходному коду и набор дополнительных инструментов тестирования и анализа [23].

1.4.5 GMap.Net

Кроссплатформенная программная библиотека с открытым исходным кодом *GMap.Net* [22], предназначена для разработки настольных и мобильных

приложений. В составе *GMap.Net* имеются программные компоненты для технологий *Windows Forms* и *WPF*, предоставляющие возможности создания маршрутов, маркеров, визуализацию и экспорт картографических материалов из сервисов *Google Maps*, *Yahoo Maps*, *Bing Maps*, *OpenStreetMap*, *Яндекс.Карты* и др. Альтернативными источниками геоданных для *GMap.Net* служат пространственные базы данных *PostgreSQL* и *SpatiaLite*.

Из числа преимуществ *GMap.Net* можно выделить качественный набор обучающих уроков с демонстрационными примерами использования этой библиотеки.

Выводы по разделу:

Обзор существующих решений позволил определить способы представления технологических объектов и информации о состоянии технологического процесса в ГИС-ориентированном интерфейсе. В качестве основного способа, избран вариант отображения этой информации при помощи маркеров, так как для реализация данного способа требует меньших ресурсов.

При использовании картографических материалов сервисов Google Maps, Яндекс.Карты и Bing Maps в коммерческих целях, требуется заключить лицензионное соглашение, позволяющие использовать данных этих сервисов. В качестве основного источника картографических данных, в виду его доступности, выбран *OpenStreetMap*.

В соответствие с выбранным подходом отображения информации на карте, был избран инструмент *GMap.Net* для работы с картами и маркерами. Функциональные возможности данного инструмента позволяют решить поставленные задачи.

2 ПРОЕКТИРОВАНИЕ

Одним из факторов, влияющих на успех любой системы является качество ее проектного решения. Этап проектирования является одним из самых важных и ответственных. Для ИС от качества выполнения данного этапа зависит эффективность, гибкость, расширяемость, сроки разработки, тестируемость системы и многое другое.

Процесс проектирования выполняется в соответствии с итеративной методикой, которая описанная в источнике [1]. Данная методика предполагает пять основных шагов, каждый из которых разбит на отдельные аспекты. Она определяет итеративный процесс создания архитектуры, позволяющий выработать различные варианты архитектуры, которые дорабатываются в ходе итераций. Итогом всех итераций является архитектурное решение, наиболее соответствующее разрабатываемой ИС [1].

Исходными данными проектирования являются технические требования, приведенные в Приложение Б.

2.1 Определение целей, потребителей и ограничений архитектуры

Под целями архитектуры понимается задачи и ограничения, очерчивающие архитектуру и процесс проектирования, определяющие объем работ и помогающие понять, когда следует завершить этап проектирования. Наличие четких целей позволяет сосредоточиться на архитектуре и правильном выборе проблем для решения [1].

Основными целями разработки архитектуры являются создание общих моделей для облегчения понимания системы и предоставление абстрактных конструкций, позволяющие приступить к этапу реализации.

Основными потребителями архитектурного решения являются разработчики и тестировщики.

Одним из источников ограничений проектируемой системы является программа разработки и просмотра графических мнемосхем *Infinity HMI*, а точнее ее производительность. Согласно техническим требованиям, представленным в приложении А, необходимо обеспечить интеграцию проектируемой системы с *Infinity HMI*. Единственный известный на данный момент способ интеграции с *Infinity HMI* основан на технологии *ActiveX*. Интеграция предполагает создание элемента управления *ActiveX*, который будет выполняться в среде *Infinity HMI*, но наличие тяжелых элементов управления *ActiveX* сильно ухудшают производительность *Infinity HMI* и может нарушить работу других ее компонентов.

2.1.1 Определение варианта развертывания

Варианты развертывания системы в физической среде должны рассматриваться как часть процесса проектирования, так как ограничения этой среды могут вносить коррективы в архитектурные решения.

В качестве основного варианта выбрано 2-уровневое развертывание, диаграмма представлена на рисунке 2.1. На сервере развернуты веб-приложение, СУБД *PostgreSQL* и *InfinityServer*. В качестве исполняемой среды может использоваться операционная система *Window* 7 или более поздней версии.

Клиентские приложения развернуты на компьютере, в котором организуется автоматизированное рабочее место средствами *Infinity HMI*. Среда разработки проектов запускается и выполняется на операционной системе *Windows* 7 или более поздней версии, а исполняемой средой для компонента *ActiveX* является программа *Infinity HMI*.

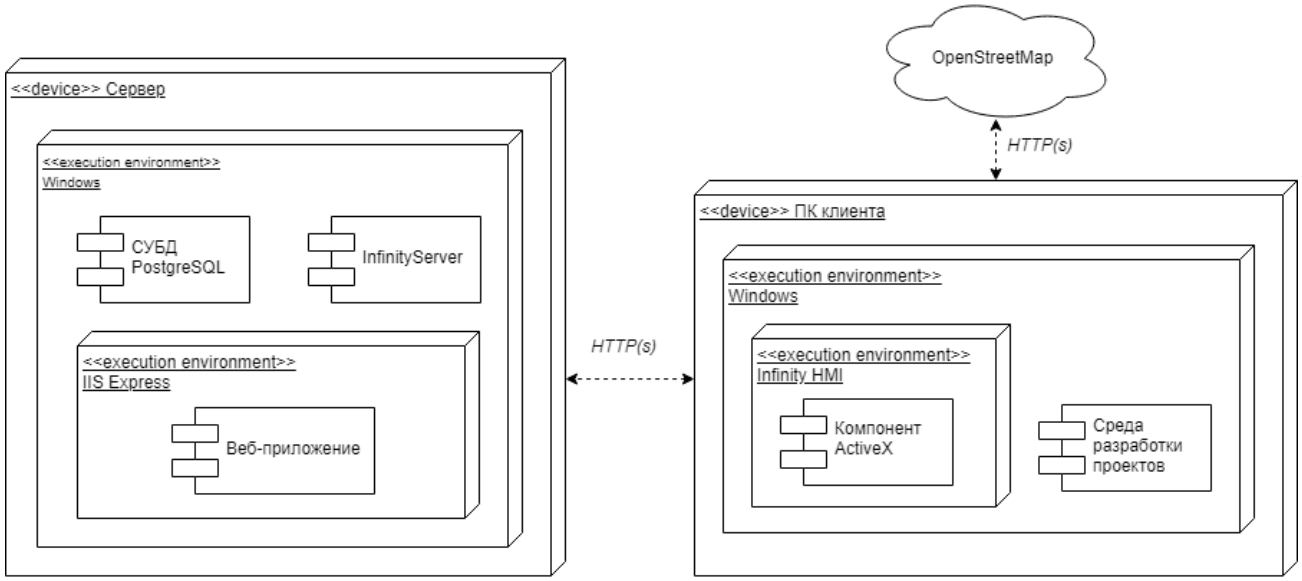


Рисунок 2.1 – Диаграмма развертывания

Стоит также отметить, что данный вариант развертывания не является единственно возможным. В некоторый случаях допустим вариант 3-уровневого развертывания, а также вариант нераспределенного развертывания [1].

3-уровневое развертывание позволяет эффективно сконфигурировать аппаратную часть для обеспечения максимального соответствия требованиям каждого компонента системы, но для этого необходимо разместить каждый компонент на отдельном устройстве.

2.2 Основные варианты использования системы

Варианты использования оказывают влияние на многие аспекты разрабатываемой системы. Они играют особо важную роль в обеспечении будущего успеха создаваемой системы [1]. Выявленные важные с точки зрения архитектуры варианты использования, можно применять как средство оценки применимости или неприменимости возможных вариантов архитектуры.

Диаграммы основных вариантов использования системы, представлены в приложении В.

2.3 Определение архитектурного стиля и типа приложений

В качестве архитектурного стиля проектирования основных функциональных частей системы, избрана N-уровневая архитектура, описанная в источнике [1]. Данный стиль определяет способ разделения функциональности системы на сегменты, которые могут физически размещаться на разных устройствах. Характеристиками данной архитектуры являются функциональная декомпозиция приложения, сервисные компоненты и их распределённое развёртывание, следствием которого является повышенная масштабируемость, доступность, управляемость и эффективность использования ресурсов. Каждый уровень абсолютно независим от всех остальных, кроме тех, с которыми он непосредственно взаимодействует.

Согласно рекомендациям, из источника [1], в случае, если создаваемое приложение предназначено для использования во внутренней сети организации, таковой является проектируемое решение, где все компоненты системы будут располагаться в закрытой сети, следует использовать только три уровня.

2.4 Архитектура системы

Высокоуровневая схема системы, состоящая из трех уровней представлена на рисунке 2.2.

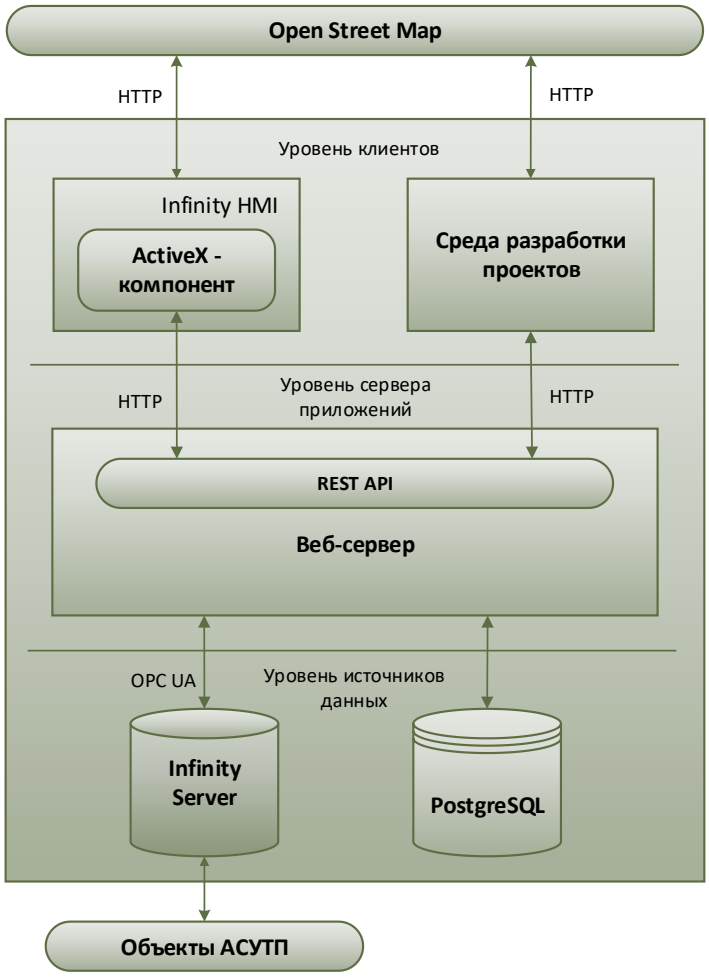


Рисунок 2.2 – Трехуровневая архитектура системы

На уровне клиентов расположены настольный клиент и элемент управления *ActiveX*, обеспечивающий интеграцию с *Infinity HMI*. Эти клиенты относятся к типу насыщенных клиентов, представляющие собой самодостаточные приложения с пользовательским интерфейсом и логикой представления, обеспечивающие отображение данных с помощью элементов управления пользовательского. Данные клиенты могут поддерживать сценарии работы без постоянного подключения к сети [1].

Уровень клиентов взаимодействует с уровнем сервера приложений, на котором расположен веб-сервер. Веб-сервер является сервисным приложением,

обеспечивающий функциональность централизованного хранения и доступа к данным для совместного использования клиентами. Взаимодействие веб-сервера и клиентов осуществляется с помощью программного интерфейса, предоставляемый веб-сервером. Этот интерфейс основан на стандартном протоколе *HTTP(s)*.

Третий уровень представляет источников данных. На нем находятся СУБД *PostgreSQL* и сервер ввода/вывода *Infinity Server*. *Infinity Server* является компонентом программного комплекса *SCADA Infinity*, предназначен для непрерывного мониторинга технологических процессов и управления технологическим оборудованием.

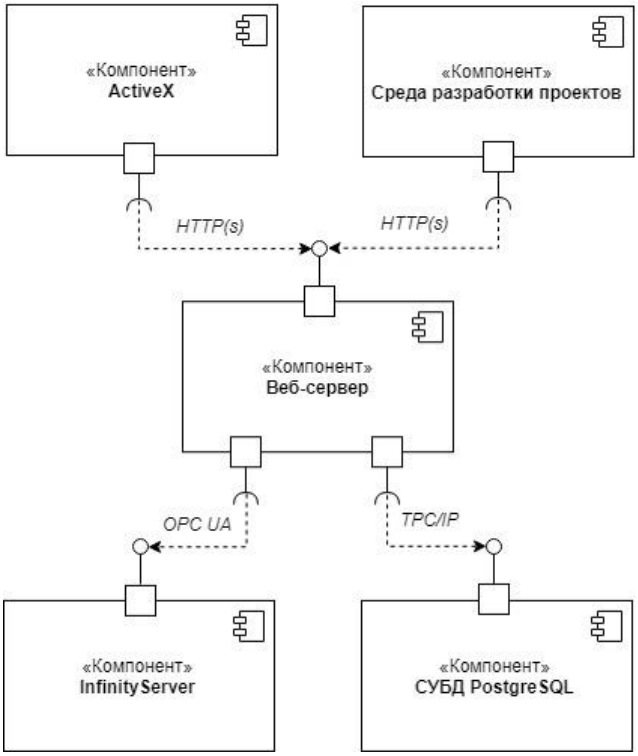


Рисунок 2.3 – Диаграмма компонентов системы

2.4.1 Уровень источников данных

2.4.1.1 База данных

Основными критериями выбора в качестве СУБД *PostgreSQL*, стали ее доступность, функциональная мощность, богатая система типов, поддержка популярных форматов данных, надежность и безопасность.

Используя реляционную модель, спроектирована схема базы данных, которая представлена в Приложении Г.

Основной сущностью предназначенная для представления технологического объекта на географической карте, является *Marker*. Атрибуты этой сущности приведены в таблице 2.1.

Таблица 2.1 – Атрибуты сущности *Marker*

Атрибут	Описание
Name	Название маркера.
Geometry	Описывает геометрию маркера для представления его на карте. Геометрия точечного маркера описана двумя координатами.
MarkerKind	Описывает тип маркера.
MarkerGroupId	Идентификатор группы маркеров, в которой состоит этот маркер. Внешний ключ.

Каждый технологический объект обладает определенным набором свойств, характеризующее его состояние. Для моделирования данных свойств спроектирована сущность *Property*, ее атрибуты представлены в таблице 2.2.

Таблица 2.2 – Атрибуты сущности *Property*

Атрибут	Описание
Name	Название свойства технологического объекта.
Value	Значение свойства.
StatesTable	Таблица состояний свойства.
UaNodeId	Идентификатор переменной в адресном пространстве OPC UA сервера.

Атрибут	Описание
IsDynamic	Если значение этого атрибута равно *true*, то сущность привязана к переменной OPC UA сервера.
MarkerId	Идентификатор маркера, которому принадлежит это свойство. Внешний ключ.

Таблица состояний сущности *Property* позволяет реализовать отображение значений переменной OPC UA сервера на значение, определенное пользователем. Например, пусть некоторая сущность *Property* представляет состояние светофора, регулирующее движение автомобильного транспорта. Пусть источником текущего состояния светофора является значение переменной из адресного пространства OPC UA сервера, при этом информация о состояние светофора определено целыми числами, где 0 – красный, 1– желтый и 2 – зеленый. Чтобы сопоставить число определенному цвету сигнала светофора, применяется таблица состояний. Каждая запись этой таблицы представляет собой пару ключ-значение, где ключом является некоторое значение переменной, а в качестве значения используется информация, определенная пользователем. В данном выше примере ключом является числа 0, 1 и 2, а значением является цвет сигнала светофора.

Для поддержания типизации значения атрибута *Value* и записей *StatesTable* сущности *Property*, спроектирована сущность *PropertyMetadata*, описывающая типы значение атрибута *Value* и ключа таблицы состояний свойства.

Таблица 2.3 – Атрибуты сущности *PropertyMetadata*

Атрибут	Описание
ValueType	Тип данных значения свойства.
StatesTableKeyType	Тип данных ключа таблицы состояний.
PropertyId	Идентификатор свойства, с которым связаны метаданные. Внешний ключ.

Сущность *MarkerGroup* моделирует группы маркеров, над которыми можно выполнять групповые операции и задавать параметры, применяемые для всех маркеров, состоящих в группе. Описание атрибутов сущности *MarkerGroup* приведены в таблице 2.4.

Таблица 2.4 – Атрибуты сущности *MarkerGroup*

Атрибут	Описание
Name	Название группы.
Visible	Определяет видимость маркеров на карте, принадлежащие этой группе.
ZIndex	Задает очередность отображения маркеров текущей группы при наложении с маркерами другой группы.
Scale	Ступени масштаба, в пределах которого маркеры этой группы видимы.
ProjectId	Идентификатор проекта, в которой
ImageId	Идентификатор изображения, которая используется в качестве иконки для отображения пользователю. Внешний ключ.

Маркеры и группы маркеров определены в проекте, которая описана сущностью *Project*. Атрибуты этой сущности представлены в таблице 2.5.

Таблица 2.5 – Атрибуты сущности *Project*

Атрибут	Описание
Name	Название группы.
CreatorName	Определяет видимость маркеров этой группы.
CreationDate	Дата создания проекта.
LastEditorName	Имя пользователя последним редактировавший проект.
LastModifiedDate	Дата последнего редактирования проекта.
Description	Пользовательское описание к проекту.

С каждым проектом связана конфигурация карты, в которой определены различные ограничения. Конфигурация карты представлена сущностью *MapOptions*, ее атрибуты описаны в таблице 2.6.

Таблица 2.6 – Атрибуты сущности *MapOptions*

Атрибут	Описание
AllowMove	Атрибут определяет доступность перемещения по карте.
DefaultScale	Ступень масштаба по умолчанию.
MapSources	Список допустимых источников карт.
Center	Координата центрирования карты.
LowerRightCorner	Координата нижнего правого угла прямоугольника.
UpperLeftCorner	Координата верхнего левого угла прямоугольника.
AllowableScale	Допустимый диапазон ступеней масштаба.
ProjectId	Идентификатор проекта, которому принадлежит конфигурация.

Значение атрибутов *LowerRightCorner* и *UpperLeftCorner* образуют прямоугольную область на карте, в пределах которого допускается перемещение. Стоит также отметить, что атрибут *AllowMove* задает более приоритетное ограничение, чем атрибуты *LowerRightCorner* и *UpperLeftCorner*. Если значение атрибута *AllowMove* установлено в *false*, то атрибуты *LowerRightCorner* и *UpperLeftCorner* игнорируются.

Изображения являются такими же ресурсами, как и выше перечисленные сущности. Они используются клиентскими приложения в пользовательском интерфейсе, а также один из типов маркеров предназначен для отображения различных изображений на карте. Для того чтобы обеспечить централизованный доступ к этим изображениям, было принято решение хранить их на веб-сервере. Более того эти изображения организуются в библиотеки изображения, что упрощает работу с ними.

Библиотека изображений описана сущностью *ImageLibrary*, которая за исключением идентификатора имеет только атрибут *Name*, представляющее ее название.

Сущность *Image* описывает информацию о изображении, ее атрибуты приведены в таблице 2.7. Стоит отметить, что *Image* всего лишь содержит информацию о расположении изображения в хранилище, хэш-код и название

этого изображения. Физически изображение расположено в файловой системе веб-сервера.

Таблица 2.7 – Атрибуты сущности *Image*

Атрибут	Описание
Name	Название изображения.
HashCode	Хэш-код изображения.
Path	Путь до изображения в хранилище изображений.
ImageLibraryId	Идентификатор библиотеки изображений. Внешний ключ.

Информация о хэш-коде позволяет исключить загрузку на веб-сервер дубликатов изображений.

Для моделирования пользователей и их ролей используются сущности, предоставляемые системой авторизации *ASP.NET Core Identity* [7]. На схеме, приведенной в приложении Б, указана лишь часть атрибутов этих сущностей.

2.4.1.2 Infinity server

Основными функциями сервера вводы/вывода *Infinity Server* являются непрерывный контроль технологического процесса, опрос системы автоматики и телемеханики, логическая и математическая обработка поступающих данных, предоставление доступа к оперативным значениям технологических параметров средствами протоколов *OPC DA, OPC AE* и *OPC UA*.

2.4.2 Уровень сервера приложений

Данный уровень содержит веб-сервер, структура которого представлена на рисунке 2.4.

Архитектурное решение веб-сервера основано на популярной концепции слоев, используемая разработчиками программного обеспечения для разделения

сложных систем на более простые части. Эта концепция позволяет сгруппировать связанную функциональность в отдельных слоях, выстраиваемых вертикально, поверх друг друга [2]. Разделение системы на слои позволяет достичь строгого разделения функциональности и обеспечить необходимый уровень абстракции, следствием которых является гибкость, удобство, простота обслуживания, слабое связывание между слоями системы, а также увеличение повторного использования компонентов системы [1].

Рисунок 2.4 – Структура веб-сервера

Архитектуру веб-сервера составляют три слоя: слой доступа к данным, слой предметной области и слой сервисов.

В слое доступа к данным сосредоточена функциональность взаимодействия с внешними источниками данных. Для проектируемого решения источниками данных являются база данных, расположенная в СУБД *PostgreSQL* и сервер ввода/вывода Infinity Server, обмен данными с которым выполняется средствами протокола *OPC UA*.

Слой предметной области содержит компоненты, обеспечивающие логическую валидацию данных, реализацию правил манипулирования и доступа к этим данным, а также компоненты, предоставляющие стандартные *CRUD*-операции с данными.

Компоненты слоя сервисов организуют открытый программный интерфейс, обеспечивающий доступ к функциональности слоя предметной области.

Подробнее компоненты отдельного слоя описаны ниже.

2.4.2.1 Слой доступа к данным

Основной задачами компонентов этого слоя, как было описано выше, являются обеспечение обмена данными с внешними источниками и предоставление необходимого уровня абстракции вышележащему слою. Данный слой предоставляет программные интерфейсы и их реализации слою предметной области, скрывая при этом детали взаимодействия с источниками данных.

Для описания данных внешних источников определена система классов, позволяющие оперировать этими данными в удобном виде. Диаграмма основных классов, описывающие данные представлены в Приложении В.

Совокупность классов-сущностей, интерфейсов и их реализаций позволяют обеспечить работу с данным в унифицированном виде.

Диаграмма классов, обеспечивающие взаимодействие с источниками данных представлена в приложении В. Класс *UaClient* реализует логику и скрывает детали взаимодействия со стандартным сервером *OPC UA*, в том числе и с *SCADA InfinityServer*. Этот класс реализует интерфейс *IUaClient*, определяющий методы установления соединения с сервером *Connect*, чтения значения переменных *ReadValues*, записи значений в переменные *WriteValues*, а

также метод *GetNodeChildren*, предназначенный для обхода дерева переменных сервера *OPC UA*.

Для работы с базой данных определены классы, описывающие контексты доступа к данным. Класс *MarkerContext* описывает контекст доступа к данным, связанные с маркерами, включая проекты и их конфигурации.

Контекст, необходимый для работы с изображениями представлен классом *ImageContext*. *ImageContext* и *MarkerContext* являются наследниками класса *DbContext*, относящийся к платформе *EntityFrameworkCore* [1].

Для управления пользователями и их ролями предназначен контекст, описанный классом *UserContext*. Данный класс, в отличие вышеперечисленных, является наследником класса *IdentityDbContext*, относящийся к системе авторизации *ASP.NET Core Identity* [7].

Разделение контекстов позволяет создать изолированные области работы с данными, обеспечивая при этом простоту и удобство.

Также для того чтобы обеспечить абстрагирование от конкретной платформы объектно-реляционного отображения, были спроектированы классы и интерфейсы: *MarkerStorage, ImageStorage, IMarkerStorage, IImageStorage* и обобщённый интерфейс *IRepository*. Класс *EfRepository* реализует интерфейс *IRepository*, скрывая от компонентов, использующих его, детали работы с платформой *EntityFrameworkCore*.

2.4.2.2 Слой предметной области

В данном слое сосредоточена основная функциональность системы и логика предметной области, связанная с извлечением, управлением, применением правил, обеспечением непротиворечивости и действительности данных. Слой состоит из компонентов, предоставляющие программные интерфейсы, доступные для использования в слое сервисов. Также в этом слое

определены классы, реализующие паттерн *Data Transfer Object (DTO)* [2]. Диаграмма классов *DTO* представлена в Приложении Г. Основной задачей этих классов в данной системе является передача данных между веб-сервером и клиентскими приложениями. Свойства классов *DTO*, связанные с маркерами, проектами и изображениями, аналогичны классам-сущностям, описанные в разделе 2.4.2.1, за исключением дополнительного свойства *Owner* в классе *ProjectDto*. Данное свойство определяет пользователя, который владеет проектом в настоящий момент времени.

Классы *NodeDto, ValueNodeDto, NodeIdDto, DataValueDto, WriteValueDto* относятся к контексту работы с протоколом *OPC UA*. *NodeDto* и *ValueNodeDto* предназначены для передачи информации о переменных (узел) адресного пространства сервера *OPC UA*. Каждая переменная имеет уникальный идентификатор в пределах одного сервера *OPC UA*, для передачи информации об идентификаторе переменной предусмотрен класс *NodeIdDto*.

Класс *DataValueDto* обеспечивает передачу информации о значении переменной. Также следует отметить, что запись некоторого значения в переменную *OPC UA*, кроме самого значения, требует ряд дополнительной информации, для передачи этой информации предусмотрен класс *WriteValueDto*. Подробнее с протоколом *OPC UA* можно ознакомиться в источнике [8].

Диаграмма классов, обеспечивающую основную функциональность данного слоя приведена в приложении Д. Названия этих классов и интерфейсов оканчивается суффиксом «*Service*», так как они предоставляют некоторую услугу другим компонентам системы. Стоит различать компоненты данного слоя от компонентов слоя сервисов. Задачей компонентов этого слоя является предоставление услуг извлечения, управления, применения правил, обеспечения непротиворечивости и действительности данных, в свою очередь задачей компонентов слоя сервисов является прием и обработка запросов клиента,

преобразование данных запроса и передача их компонентам слоя предметной, а также формирование и отправка ответов клиентов на основе результатов, полученных от компонентов слоя предметной области.

Некоторые объекты (ресурсы) в приложении, подобные проекту, группе маркеров, маркерам и библиотеке изображений, агрегируют другие объекты. Проекты агрегируют группы маркеров, группы агрегирует маркеры, а маркеры – свойства и т.д. Для обеспечения более эффективной работы с этими ресурсами, спроектированы обобщенные интерфейсы *IRootCrudService* и *ICrudService*, определяющие стандартные *CRUD*-операции для работы с агрегирующими и агрегируемыми ресурсами соответственно.

Применением правил и обеспечением непротиворечивости данных маркеров занимается система классов-валидаторов, также представленная в приложении Д. Для каждого типа маркера определен соответствующий класс валидатора, экземпляры этих валидаторов создаются средствами статического класса *MarkerValidatorFactory*. Этот класс реализует паттерн проектирования *Абстрактная фабрика*.

Класс *UaService*, реализуя интерфейс *IUaService* предоставляют услуги взаимодействия с серверами *OPC UA*. Этот класс инкапсулирует логику работы с несколькими серверами *OPC UA*, при этом делегируя работу с конкретным сервером классу *UaClient* из слоя доступа к данным. То есть *UaService* предоставляет услуги чтения или записи значений в переменные, определенные на разных серверах *OPC UA*. Требование коммуникации с несколькими серверами приводится в приложении Б.

Для управления параллельным доступом к ресурсам веб-сервера, спроектированы компоненты, совместно реализующие пессимистическую блокировку. К ним относятся интерфейс *ILocker* и классы *MemoryLocker* и

LockOwner. Следует отметить, что подобное решение управления параллельным доступ описан в источнике [2].

2.4.2.3 Слой сервисов

Данный слой предоставляет фасад, обеспечивающий коммуникацию клиентов с веб-сервером. Приемом и обработкой запросов клиентов, а также формированием ответов занимаются классы-контроллеры. Диаграмма классов данного слоя представлена в приложении Е.

На рисунке 2.5 приведены классы, обеспечивающие фильтрацию запросов. Эти классы реализуют логику валидации запросов до выполнения метода контроллера, в случае если запрос является не валидным, клиенту формирует ответ с кодом ошибки.

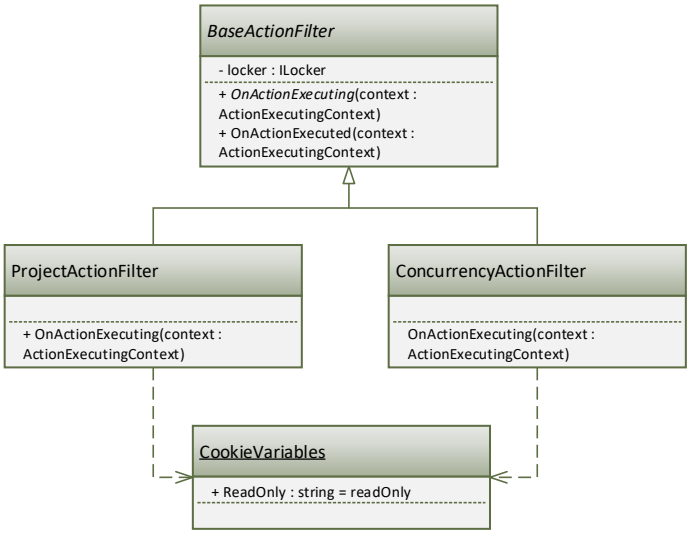


Рисунок 2.5 – Диаграмма классов-фильтров

Функциональность авторизации размещена в слое сервисов. Она основана на стандарте *JSON Web Token* (*JWT*). *JWT* представляет собой строку в которой закодирована информация, необходимая для авторизации пользователя. Для генерации этой строки в слое сервисов присутствуют соответствующие компоненты. Диаграмма классов этих компонент представлена на рисунке 2.6

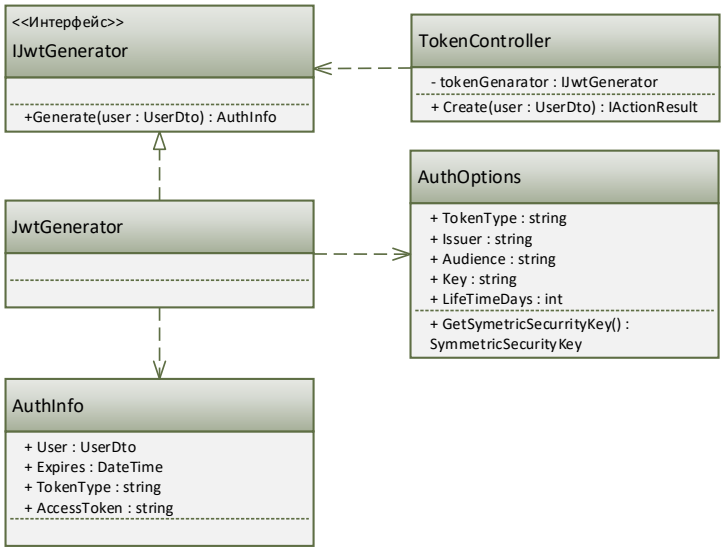


Рисунок 2.6 – Генератор *JWT*

Интерфейс *IJwtGenerator* определяет компонент, предназначенный для генерации строки *JWT*, а класс *JwtGenerator* является реализацией этого интерфейса. Класс *AuthOptions* описывает конфигурационные данные, необходимые для генерации строки *JWT*.

Класс *AuthInfo* описывает данные авторизованного пользователя с дополнительной информацией о типе и дате окончания действия *JWT*.

Для обработки запросов авторизации существует контроллер *TokenController*.

2.4.2.4 Сквозная функциональность

В архитектурном решении большинства приложений почти всегда присутствует функциональность, охватывающая все слои системы. Такую функциональность называют сквозной функциональностью. Обычно к ней относят журналирование, валидацию, аутентификацию и управление

исключениями [2]. К сквозной функциональности веб-сервера относятся журналирование и управление исключениями, функциональность валидации и аутентификации делегированы слою сервисов. Ниже подробнее раскрываются вопросы журналирования и управления исключениями.

Тщательно продуманное решение управления исключениями позволяет упростить архитектурное решение приложения, повысить безопасность и управляемость. При проектировании стратегии управления исключениями системы, применялись следующие рекомендации:

- перехватываются только внутренние исключения, которые могут обработаны или к которым необходимо добавить дополнительную информацию;
- исключения не используются как средство управления логикой выполнения приложения;
- исключения распространяются вверх к слоям, где они протоколируются и преобразовываются для передачи на следующий слой или клиенту.

Для слоя доступа к данным применяется стратегия распространения исключений, в которой разрешается распространение исключений на следующий слой. Применение этой стратегии позволяет не засорять компоненты этого слоя логикой обработки исключений, что делает их более простыми и удобными.

В слое предметной области используется стратегия перехвата и повторного формирования исключений, генерируемые слоем доступа к данным. Перехваченные исключения протоколируются, затем вместо них формируются более общие и уместные исключения для слоя сервисов.

Исключения слоя предметной области перехватываются в методах классов-контроллеров, затем на их основе формируются коды ошибок.

Последние передаются клиенту в ответе. Коды ошибок описаны константами в классе *ErrorCodes*, представленный на рисунке 2.7.

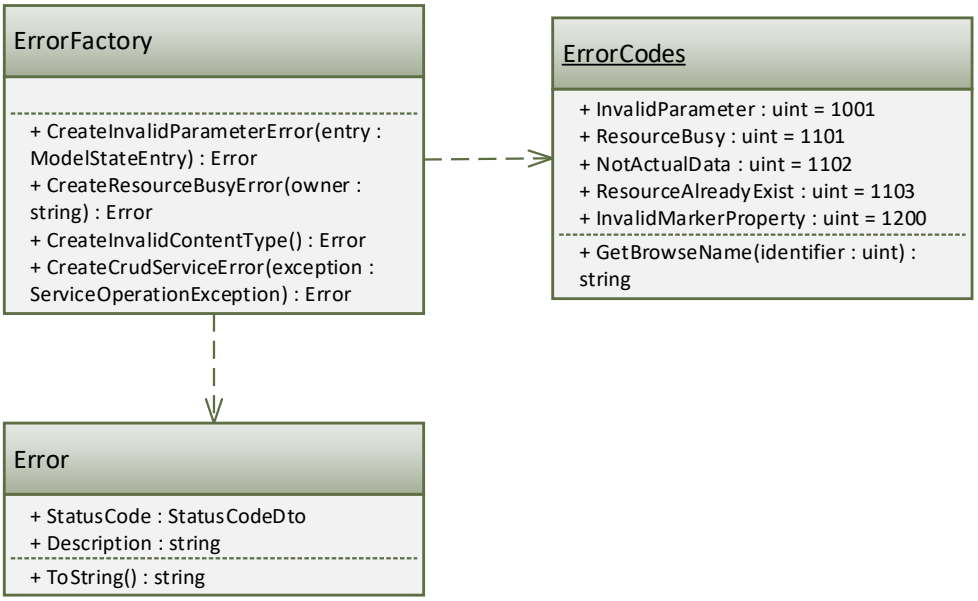


Рисунок 2.7 – Диаграмма классов, описывающие сообщения об ошибках

Журнал – один из наиболее надежных источников информации о состоянии приложения. Ведение качественного журнала приложения позволяет определить причины сбоев в работе приложения, расходуя при этом минимум ресурсов. Стратегия журналирования системы использует следующие рекомендации:

- в журнал заносятся сообщения с информацией о пользователе, идентификаторе запроса (для слоя сервисов), выполняемая операция, меры, принятые для выхода из ситуации сбоя;
- в журнал не записывается информация, которая не актуальна для текущей ситуации или уже известна из текущего контекста;
- конфиденциальная и важная информация о работе приложения выводится только в режиме разработки [1].

Процесс журналирования разделен на следующие уровни:

- *Trace*. Этот уровень используется для отладки проблем с конфиденциальными данными;
- *Debug*. Этот уровень используется для записи в журнал информации о состояниях компонентов системы, идентификаторах процессов и т.д.;
- *Information*. Используется журналирования информации о входных и выходных данных компонентов системы. Эта информация не содержит конфиденциальных и критически важных для приложения данных;
- *Error*. Используется для записи в журнал информации о возникших исключениях и сбоях.

Следует отметить, что уровни *Trace* и *Debug* используются исключительно во время разработки приложения.

2.4.3 Уровень клиентов

При проектировании клиента применяется шаблон проектирования *Model View Presenter* [3]. Основная цель этого шаблона отделить элементы управления пользовательского интерфейса от компонентов логики представления. Такое разделение функциональности повышает удобство обслуживания, тестируемость и возможность повторного использования.

2.4.3.1 Компоненты пользовательского интерфейса

Компоненты пользовательского интерфейса представляют собой визуальные элементы, отображающие данные пользователю и принимающие пользовательский ввод. Эти компоненты обычно называют *Представлениями* (*Views*). При проектировании компонентов пользовательского интерфейса клиента применяются следующие рекомендации:

- окна приложения необходимо разбить на отдельные пользовательские элементы управления, что позволит сделать их простыми и обеспечит возможность повторного использования;
- следует избегать иерархии наследования пользовательских элементов управления. При возникновении необходимости обеспечения возможности повторного использования компонентов, следует применять композицию, а не наследование;
- при создании специализированных элементов управления следует использовать существующие элементы управления платформы.

Основной элемент управления, предназначенный для отображения карты и предоставления ряда функций для работы с ней представлен классом *MapControl*. Этот содержит панель инструментов, в котором расположены инструменты фильтрации и ряд дополнительных инструментов. Выполнение низкоуровневых операций с картой делегированы компоненту *GMapControl*, относящийся к инструменту *GMap.NET*. Диаграмма классов представлена на рисунке 2.8.

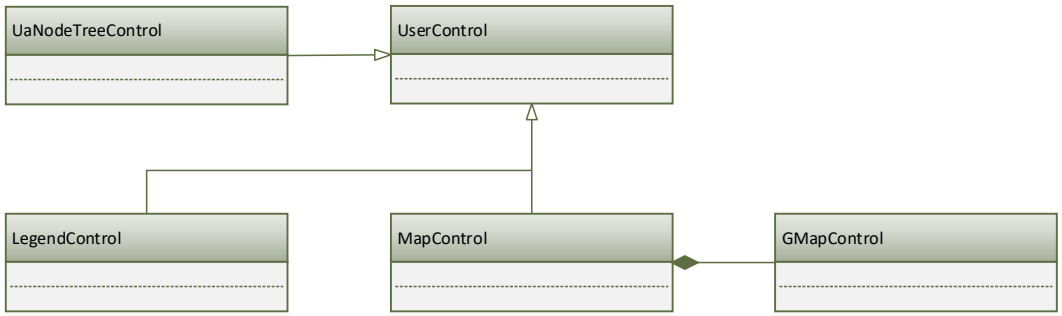


Рисунок 2.8 – Диаграмма основных классов пользовательского интерфейса

Класс *LegendControl* обеспечивает отображение информации о группах и маркерах проекта.

Для отображения дерева переменных существует класс *UaNodeTreeControl*.

На рисунке 2.9 представлен эскиз основного окна компонента *ActiveX*. Окно состоит из панели управления группами и маркерами, расположенный слева, панели фильтров, расположенный на верхней части окна, а также панели карты, занимающая основную часть окна.

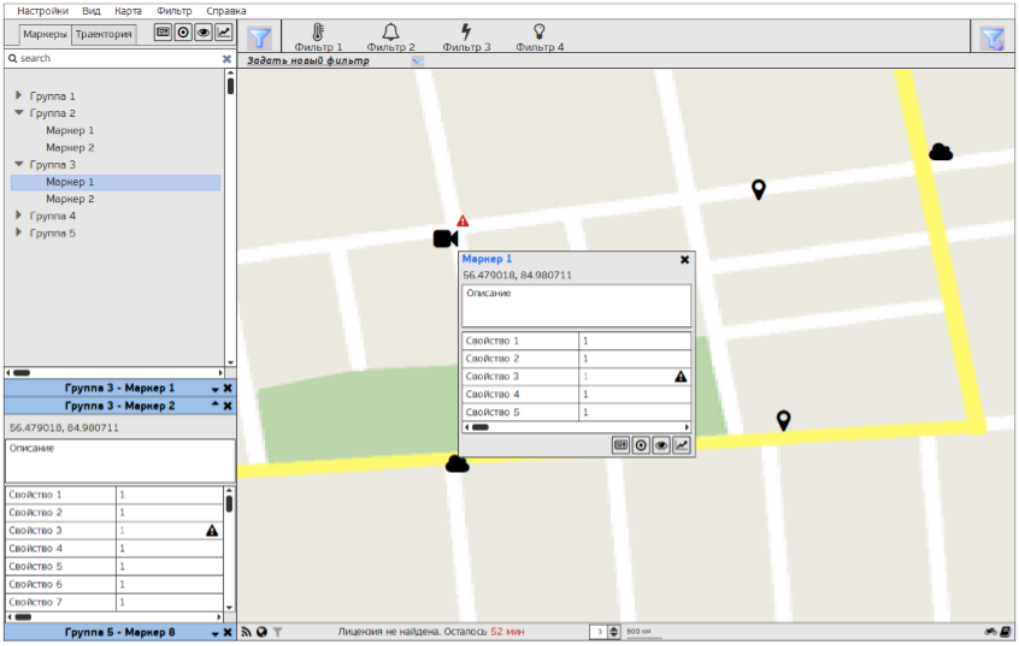


Рисунок 2.9 – Эскиз основного окна компонента ActiveX

На рисунке 2.10 приведена панель фильтров. Эта панель представлена в развернутом и свернутых видах. В развернутом виде показан процесс создания фильтра.

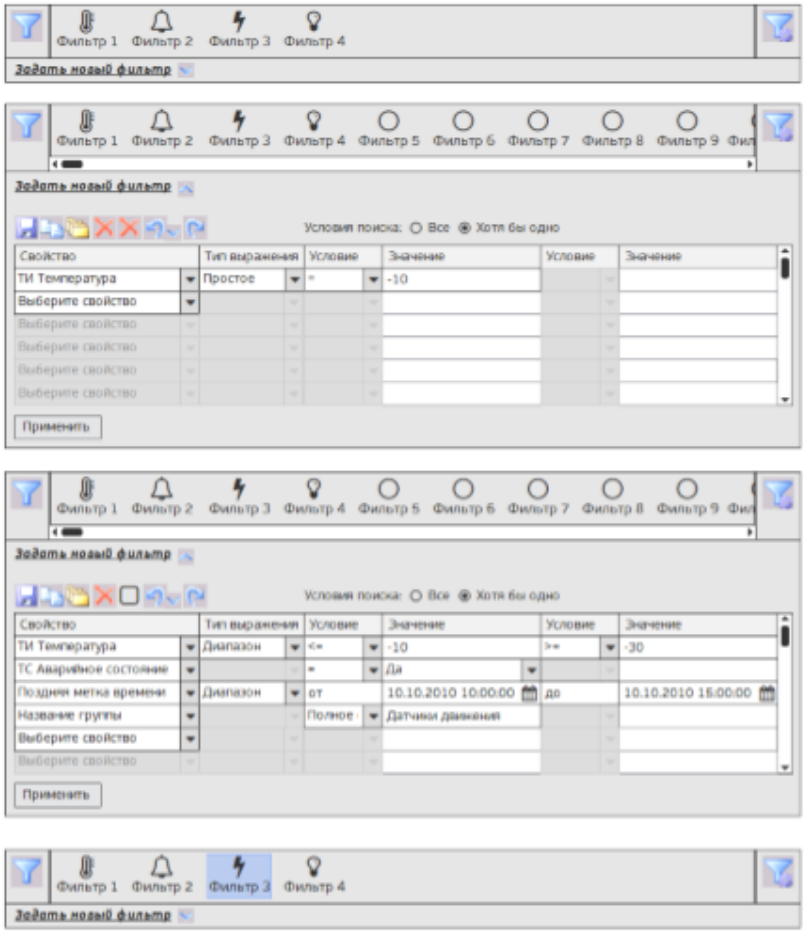


Рисунок 2.10 – Эскизы панели фильтров

На рисунке 2.11 представлено основное окно редактора проектов.

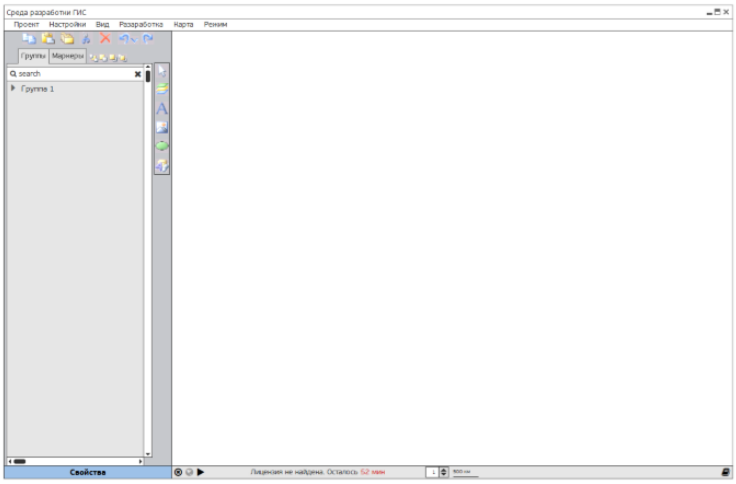


Рисунок 2.11 – Эскиз основного окна среды разработки проектов

Основное окно среды разработки проектов состоит из панели управления группами и маркерами, панели инструментов и маркеров. Основная часть окна предназначена для отображения карты.

2.4.3.2 Компоненты логики представления

Компоненты логики представления предназначены для реализации не визуальных аспектов пользовательского интерфейса. В данной системе к этим аспектам относят взаимодействие с веб-сервером, выполнение различных операций, управление стеком выполненных операции, поддержка и выполнение пользовательских команд. При проектировании компонентов логики представления используются следующие рекомендации:

- компоненты, предназначенные для хранения состояний следует отнести к компонентам логики представления;
- агенты, использующиеся для коммуникации с внешними системами, также следует реализовать в виде компонентов логики представления;
- не следует размещать в компонентах логики представления реализации сценариев формирования визуального представления пользовательского интерфейса.

Операцию, которую можно выполнить над маркером или связанными с ними данными (объектами) описывает абстрактный класс *Command*. Этот класс и его наследники реализуют шаблон проектирования *Команда*, позволяющий инкапсулировать некоторую операцию в объекте. Класс *Command* определяет два метода *Do* и *Undo*, позволяющий выполнить команду и отменить ее соответственно. Диаграмма классов, с основными командами представлена на рисунке 2.12.

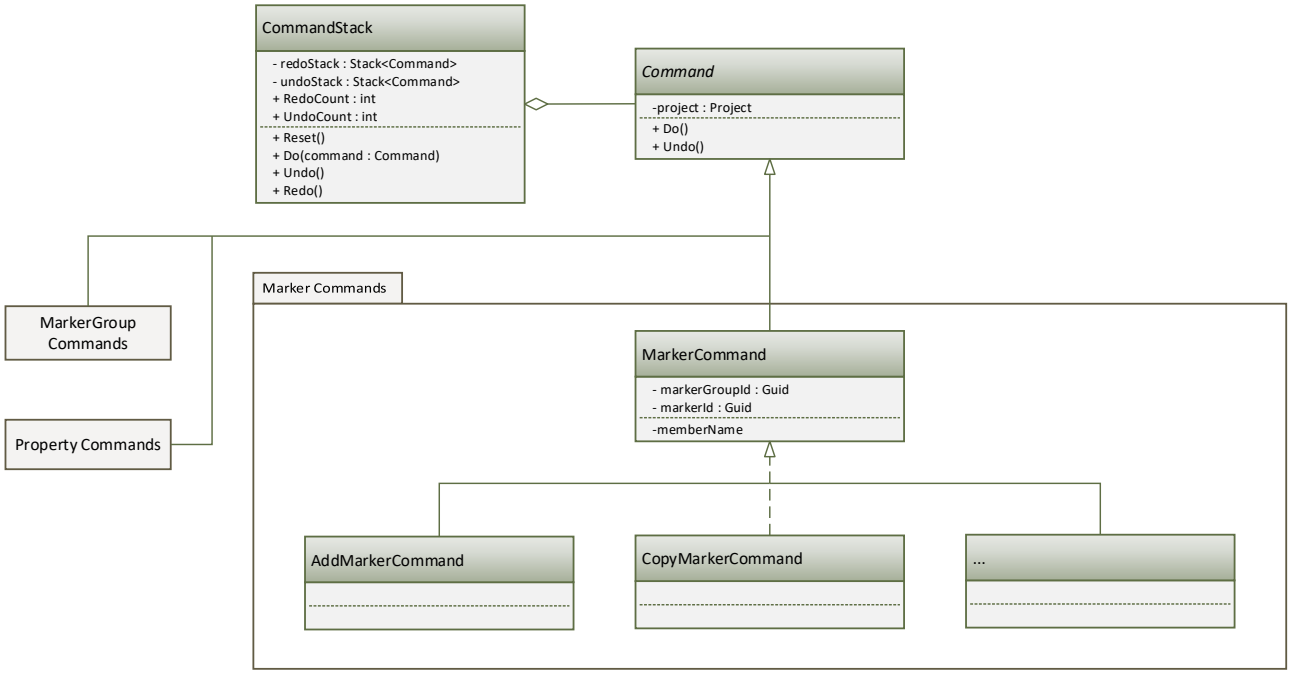


Рисунок 2.12 – Команды

Каждую конкретную операцию реализуют наследники *Command*. В этих классах определена вся необходимая информация для выполнения операции и ее отмены.

Выполненные команды агрегируются стеком команд *CommandStack* для обеспечения возможности отмены *Undo* или повторного его выполнения *Redo*.

Интерфейс *IApiClient* и *ApiClient* описывают агента, обеспечивающий коммуникацию с веб-сервером.

2.4.3.3 Модели представления

Модели представления описывают данные поступающие с веб-сервера, в формате пригодном для использования в пользовательском интерфейсе и компонентах логики приложения. Эти модели реализуют паттерн *View Model* [2], содержащий агрегированные данные из различных источников и преобразованные в формат, обеспечивающий удобство их отображения в

пользовательском интерфейсе. Также эти модели могут содержать логику валидации данных.

При проектировании моделей представления применяются следующие рекомендации:

- следует использовать модели представления для обеспечения требований отображения данных в пользовательском интерфейсе;
- следует обеспечить привязку моделей представления к элементам управления пользовательского интерфейса, если возможно средствами платформы, или предоставить соответствующие интерфейсы и события для поддержки привязки данных;
- логику валидацию данных следует разместить в соответствующей модели представления;
- следует обеспечить сериализацию моделей представления, требуемые для сохранения на жестком диске клиента.

Классы, описывающие модели представлений приведены на рисунке 2.13.

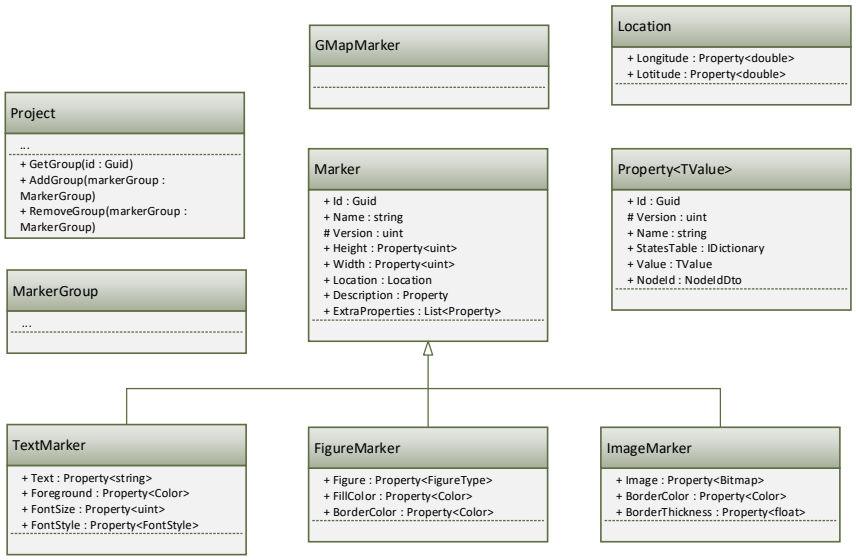


Рисунок 2.13 – Диаграмма классов моделей представления

Свойства классов, представляющих группы маркеров, проекты и конфигурацию проекта аналогичны классам *DTO* слоя предметной области, за исключением дополнительных методов, обеспечивающие удобство в использовании этих классов. Но классы, предназначенные для описания маркеров модифицированы под нужны пользовательского интерфейса. Абстрактный класс *Marker* определяет общие свойства для всех маркеров, а конкретный вид маркера, определен наследниками этого класса. Также класс *Marker* является наследником класса *GMapMarker*, относящийся к инструменту *GMap.NET*. Наследование в этом контексте позволяет обеспечить полиморфизм, то есть использовать классы маркеров в компонентах *GMap.NET*.

Также следует обратить внимание на класс *Property*, в отличие от класса *PropertyDto*, расположенного в слое предметной области, этот класс является обобщенным. Обобщение позволяет обеспечить строгую типизацию свойства *Value* этого класса, следствием которого является выявлении ошибок на этапе компиляции, связанных с типами данных.

Выводы по разделу:

Результатом этапа проектирования стало архитектурное решение, содержащие основные абстрактные модели, позволяющие перейти к этапу реализации.

РЕАЛИЗАЦИЯ

3.1 Реализация веб-сервера

Каждый слой веб-сервера определен в собственной сборке, диаграмма зависимостей сборок проекта представлена на рисунке 3.1.

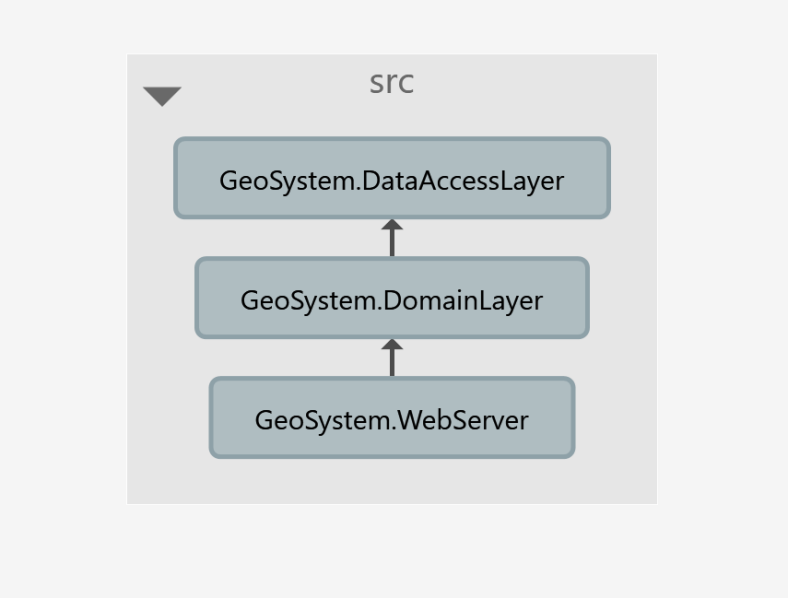


Рисунок 3.1 – Диаграмма зависимостей сборок веб-сервера

Для реализации слоев доступа к данным и предметной области используется сборки, реализующие спецификацию *.NET Standard 2.0* [6]. Необходимые инструменты для разработки сборок, соответствующие этой спецификации предоставляются интегрированной средой разработки *Microsoft Visual Studio 2017* [6].

Функциональность слоя доступа к данным реализована в сборке *GeoSystem.DataAccessLayer*. В данной сборке определены все классы, относящиеся к этому слою. В качестве инструмента объектно-реляционного отображения используется кроссплатформенная технология доступа к данным *Entity Framework Core*. Эта технология предоставляет абстракцию для работы с

базой данных, при этом для работы с конкретной базой данных необходимо использовать соответствующую реализацию этой абстракции. Одной из реализаций этой абстракции является драйвер *Npgsql* [6], предназначенный для работы с СУБД *PostgreSQL*.

Для коммуникации с серверами *OPC UA* в сборке доступа к данным реализован агент, интерфейс которого представлен ниже:

```
public interface IUaClient
{
    bool IsConnected { get; }
    void Connect(string endpoint, string connectionName);
    DataValueCollection ReadValues(NodeId[] nodeIdentifiers);
    StatusCodeCollection WriteValues(WriteValueCollection values);
    IEnumerable<Node> GetNodeChildren(NodeId nodeId, NodeClass[] classes);
    void Reconnect();
}
```

Описание методов приводилось в разделе 2. Ниже представлен фрагмент исходного текста класса, реализующий данный интерфейс:

```
public class UaClient : IUaClient, IDisposable
    {
        private readonly string configSectionName;
        private ApplicationConfiguration configuration;
        private Session session;

        public UaClient(string configSectionName = @"Ua\UaClient")
        {
            this.configSectionName = configSectionName;
        }

        public void Dispose()
        {
            session?.Dispose();
        }

        public bool IsConnected => session?.Connected ?? false;

        public void Connect(string endpoint,
            string connectionName = "GeoSystem connection")
        {
            if (string.IsNullOrWhiteSpace(endpoint))
                throw new ArgumentException("Endpoint is null or white space.");

            if (configuration == null)
                configuration = LoadConiguration();

            configuration.Validate(ApplicationType.Client).Wait();

            var securityConfiguration = configuration.SecurityConfiguration;
            var certificate = securityConfiguration.ApplicationCertificate
                                  .Certificate
                              ?? CreateCertificate2(configuration);
```

```
            securityConfiguration.ApplicationCertificate
                .Certificate = certificate;

            configuration.ApplicationUri = Utils
                .GetApplicationUriFromCertificate(certificate);

            configuration.CertificateValidator.CertificateValidation += (s, e) =>
            {
                if (configuration.SecurityConfiguration
                    .AutoAcceptUntrustedCertificates)
                    e.Accept = true;
            };

            var selectedEndpoint = CoreClientUtils.SelectEndpoint(
                endpoint,
                true,
                configuration.TransportQuotas.OperationTimeout);

            var endpointConfiguration = EndpointConfiguration
                .Create(configuration);

            var configuredEndpoint = new ConfiguredEndpoint(description: selectedEndpoint,
                configuration: endpointConfiguration, collection: null);

            session = Session.Create(
                configuration,
                configuredEndpoint,
                false,
                connectionName,
                (uint) configuration.ClientConfiguration.DefaultSessionTimeout,
                new UserIdentity(new AnonymousIdentityToken()),
                null).Result;
        }

        public DataValueCollection ReadValues(NodeId[] nodeIdentifiers)
        {
            var itemsToRead = new ReadValueIdCollection();
            itemsToRead.AddRange(
                nodeIdentifiers.Select(nodeId => new ReadValueId
                {
                    NodeId = nodeId,
                    AttributeId = Attributes.Value
                }));

            session.Read(null, 0, TimestampsToReturn.Both,
                itemsToRead, out var values, out var diagnosticInfo);

            ClientBase.ValidateResponse(values, itemsToRead);
            ClientBase.ValidateDiagnosticInfos(diagnosticInfo, itemsToRead);

            return values;
        }

        //...
    }
```

Реализация данного класса основан на программной библиотеке, предоставляемый организацией *OPC Foundation* [9].

Реализация слоя предметной области выполнена в сборке *GeoSystem.DomainLayer*. Следует отметить, что перемещение информации из классов-сущностей в классы *DTO* является трудоемкой, поэтому для этих целей применяется инструмент *Automapper* [6], позволяющий копировать данные из одних объектов в другие.

Для реализации слоя сервисов используется технология *ASP.NET Core Web API*, предоставляющий необходимые инструменты для разработки веб-сервисов. Реализация этого слоя расположена в сборке *GeoSystem.WebServer*, которая является исполняемой сборкой.

3.2 Реализация клиентов

В качестве основной технологии для реализации пользовательского интерфейса используется *Windows Forms*.

Для реализации функциональности, связанной с картой, используется инструмент *GMap.Net*.

3.2.1 Реализация компонента ActiveX

Создание элемента *ActiveX* предполагает реализации программных интерфейсов, определенные спецификаций технологией COM [6]. Следует отметить, что список интерфейсов весьма широк, поэтому *Microsoft* предоставляет инструменты, позволяющие автоматически реализовать эти интерфейсы.

Фрагмент исходного текста класса, представляющий элемент управления *ActiveX* представлен ниже:

```
[Guid("7110F53C-E7B2-4B7C-8774-318C0B669E61")]
```

```
[ProgId("GeoSystem.ActiveXClient")]
[ClassInterface(ClassInterfaceType.None)]
[ComSourceInterfaces(typeof(IActiveXEvents))]
[ComVisible(true)]
public partial class ActiveXControl : UserControl , IActiveXMethods
{
    private static string ComponentName = "GeoSystem.ActiveXClent";

    public ActiveXControl()
    {
        InitializeComponent();
    }

    //...
}
```

С помощью программного атрибута *GUID* данному классу задается глобальный уникальный идентификатор. Этот идентификатор позволяет избежать конфликтов между элементами управления *ActiveX*.

Следующий программный атрибут *ProgId* устанавливает программный идентификатор. Данный идентификатор является записью в реестре ОС, которая связанна с глобальным идентификатором элемента управления *ActiveX*. Программный идентификатор также идентифицирует класс, описывающий элемент управления *ActiveX*, но в отличие от глобального идентификатора имеет понятный человеку вид.

Атрибут *ClassInterface* определяет тип стандартного *COM*-интерфейса, который должен быть сгенерирован для класса, обеспечивающее взаимодействие с *COM*-объектами. Тип генерируемого интерфейса определяется перечислением *ClassIntefaceType*, значение которого передается в конструктор атрибута *ClassInterface*. Возможные значения перечисления *ClassIntefaceType* приводятся в источнике [6]. В данном случае используется значение *ClassIntefaceType.None*, указывающий, что для класса не требуется стандартный *COM*-интерфейс, так как для этого класса явно определен интерфейс, предоставляющий возможность взаимодействия с *COM*-объектами.

Список доступных методов для *COM*-объектов определяет интерфейс *IActiveXMethods*:

```
[ComVisible(true)]
```

```
[Guid("D0843C2E-3E1E-4B98-836A-45CDFBCC2978")]
public interface IActiveXMethods
{
    [ComVisible(true)]
    void Find(double longitude, double latitude);

    //...
}
```

Интерфейс *IActiveXMethods* реализуется классом, описывающим элемент управления *ActiveX*. Внешние *COM*-объекты используют методы интерфейса *IActiveXMethods* для изменения состояния этого элемента управления.

Атрибут *ComSourceInterfaces*, приведенный в фрагменте исходного текста класса элемента управления *ActiveX*, задает набор интерфейсов, определяющие события этого элемента управления, доступные *COM*-объектам. В данном случае используется один интерфейс, фрагмент исходного текста этого интерфейса приведен ниже:

```
[Guid("901EE2A0-C47C-43ec-B433-985C020051D5")]
[InterfaceType(ComInterfaceType.InterfaceIsIDispatch)]
public interface IActiveXEvents
{
    [DispId(1)]
    void OnGoToMnemonicSchema(string identifier);

    //...
}
```

Атрибут *InterfaceType* определяет тип интерфейса для модели *COM*. Типы интерфейсов определены в перечислении *ComInterfaceType*, значение которого передается конструктору атрибута *InterfaceType*. В данном случае используется тип *ComInterfaceType.InterfaceIsIDispatch*, позволяющий только позднее связывание.

Каждый метод в интерфейсе *IActiveXEvents* помечен атрибутом *DispId*, позволяющий платформе *.NET* автоматически реализовать один из базовых *COM*-интерфейсов *IDispatch*.

Чтобы элемент управления *ActiveX* стал доступным для использования, его необходимо зарегистрировать в реестре ОС. Регистрация осуществляется в процессе развертывания системы, отмена регистрации – в процессе

деинсталляции. Ниже приведены методы, осуществляющие регистрацию компонента и ее отмену:

```
[ComRegisterFunction]
public static void Register(string key)
{
    StringBuilder sb = new StringBuilder(key);
    sb.Replace(@"HKEY_CLASSES_ROOT\", "");

    RegistryKey regKey = Registry.ClassesRoot.OpenSubKey(sb.ToString(), true);
    regKey.SetValue("", ComponentName);
    RegistryKey ctrl = regKey.CreateSubKey("Control");
    ctrl.Close();
    RegistryKey insertb = regKey.CreateSubKey("Insertable");
    insertb.Close();

    RegistryKey inprocServer32 = regKey.OpenSubKey("InprocServer32", true);
    inprocServer32.SetValue("CodeBase", Assembly.GetExecutingAssembly().CodeBase);
    inprocServer32.Close();
    regKey.Close();
}

[ComUnregisterFunction]
public static void Unregister(string key)
{
    StringBuilder sb = new StringBuilder(key);
    sb.Replace(@"HKEY_CLASSES_ROOT\", "");

    RegistryKey regKey = Registry.ClassesRoot.OpenSubKey(sb.ToString(), true);

    regKey.DeleteSubKey("Control", false);
    regKey.DeleteSubKey("Insertable", false);

    regKey.OpenSubKey("InprocServer32", true);

    regKey.DeleteSubKey("CodeBase", false);
    regKey.Close();
}
```

На рисунке 3.2 представлена программа *Infinity HMI* с интегрированной в нее картой.

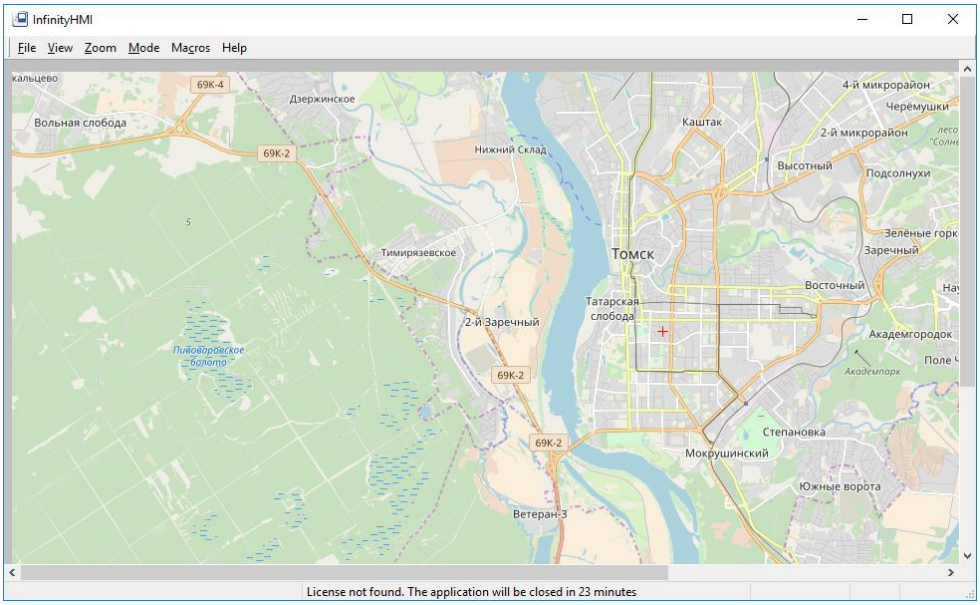


Рисунок 3.2 – Интегрированная карта в *Infinity HMI*

3.3 Unit-тестирования

Модульное тестирование осуществляется с помощью популярной платформы *XUnit*, распространяемый вместе со средой *Visual Studio 2017*.

Ниже на рисунке 3.2 представлена диаграмма зависимостей сборок, включая сборки модульных тестов. На этой диаграмме можно увидеть, что для каждой сборки системы созданы сборки модульных тестов. Причина размещения модульных тестов в отдельных сборках заключается в том, что код модульных не должен распространятся с приложением, так как он нужен исключительно для тестирования.

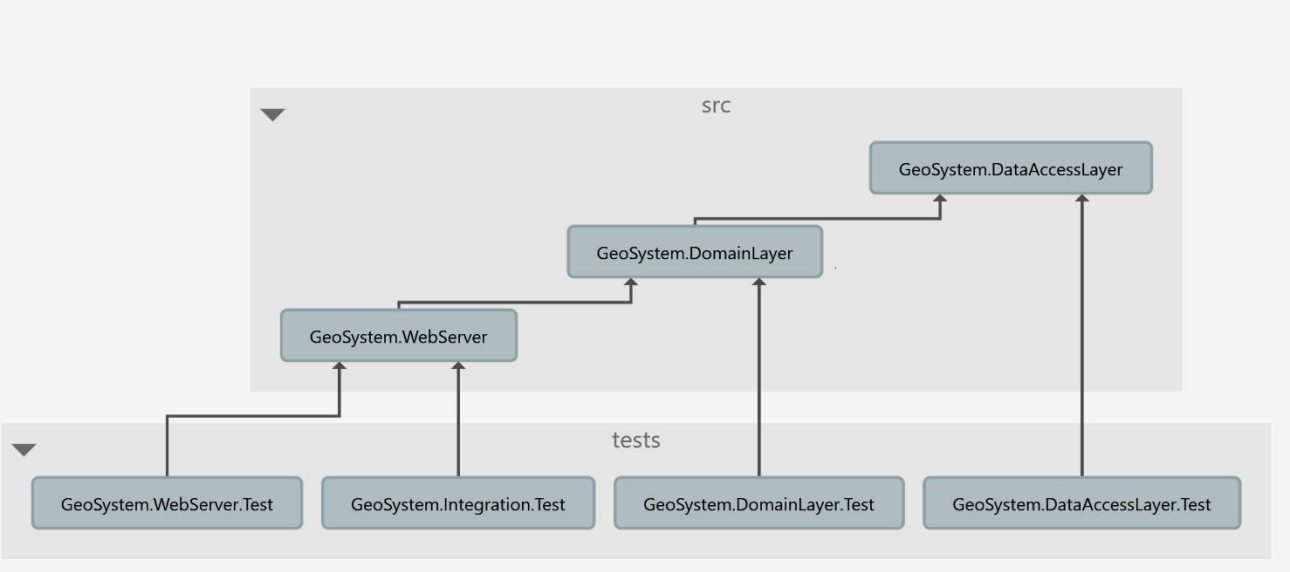


Рисунок 3.2 – Диаграмма зависимостей сборок с исходным кодом и тестами

При организации модульных тестов соблюдается некоторые правила. Во-первых, структура папок тестового проекта соответствует структуре папок тестируемой сборки. Во-вторых, все тесты, относящиеся к одному классу, собраны в одном файле с суффиксом «Tests». Внутри тестового класса тесты упорядочены при помощи вложенных классов. Каждый внутренний класс специализируется на тестировании определенного сценария или функции. Ниже представлен фрагмент теста, обеспечивающий тестирование класса, в котором реализована пессимистическая блокировка:

```
public class MemoryLockerTests
    {
        public class ValidateAcquireLock
        {
            [Theory(DisplayName = "Acquire lock with valid parameters.")]
            [InlineData("6C5FD9E7-99F7-4F54-A267-2EE1CD83D478", "developer")]
            [InlineData("C1D20598-B932-41CF-BE4E-C476F628FE75", "developer")]
            [InlineData("29E2102C-48CC-4A7C-849F-6E1D1CF778BD", "operator")]
            public void AcquireLock(string lockable, string user)
            {
                // arrange
                var guid = Guid.Parse(lockable);
                var locker = new MemoryLocker();

                // act
                locker.AcquireLock(guid, user);
```

```
            // assert
            Assert.True(locker.HasLock(guid));
            Assert.True(locker.IsOwner(guid, user));
            Assert.True(locker.IsUserHasLock(user));
            Assert.Equal(locker.Owner(guid), user);
        }

        private Guid lockedResource = new Guid("6C5FD9E7-99F7-4F54-A267-2EE1CD83D478");
        private string owner = "user";

        [Theory(DisplayName = "Acquire when resource is locked.")]
        [InlineData("6C5FD9E7-99F7-4F54-A267-2EE1CD83D478", "developer")]
        [InlineData("6C5FD9E7-99F7-4F54-A267-2EE1CD83D478", "operator")]
        [InlineData("6C5FD9E7-99F7-4F54-A267-2EE1CD83D478", "admin")]
        public void ThrowsAcquireLock(string lockable, string user)
        {
            var guid = Guid.Parse(lockable);
            var locker = new MemoryLocker();

            locker.AcquireLock(lockedResource, owner);
            locker.UpdateActivityTime(user);

            Assert.Throws<InvalidOperationException>(() =>
                locker.AcquireLock(guid, user));
        }

        //...
    }

    public class ValidateReleaseLock
    {
        [Theory(DisplayName = "Release lock with valid parameters.")]
        [InlineData("6C5FD9E7-99F7-4F54-A267-2EE1CD83D478", "developer")]
        [InlineData("C1D20598-B932-41CF-BE4E-C476F628FE75", "developer")]
        [InlineData("29E2102C-48CC-4A7C-849F-6E1D1CF778BD", "operator")]
        public void ReleaseLock(string lockable, string user)
        {
            var guid = Guid.Parse(lockable);
            var locker = new MemoryLocker();
            locker.AcquireLock(guid, user);

            locker.RealeseLock(guid);

            Assert.False(locker.HasLock(guid));
            Assert.False(locker.IsUserHasLock(user));
            Assert.Null(locker.Owner(guid));
        }

        //...
    }

    public class ValidateTimeout
    {
        [Theory(DisplayName = "Validate lock timeout.")]
        [InlineData("6C5FD9E7-99F7-4F54-A267-2EE1CD83D478", "developer")]
        [InlineData("C1D20598-B932-41CF-BE4E-C476F628FE75", "developer")]
        [InlineData("29E2102C-48CC-4A7C-849F-6E1D1CF778BD", "operator")]
        public void Timeout(string locable, string user)
        {
```

```
            var locker = new MemoryLocker(TimeSpan.FromMilliseconds(100));

            var guid = Guid.Parse(locable);
            locker.AcquireLock(guid, user);

            Thread.Sleep(110);

            Assert.False(locker.HasLock(guid));
            Assert.False(locker.IsUserHasLock(user));
        }
    }

    //...
}
```

Приведенный выше класс содержит внутренние классы, обеспечивающие тестирование определенного метода класса *MemoryLocker*. Тесты, определенные в классе *ValidateAcquireLock*, обеспечивают тестирование получения блокировки. При этом рассматриваются все возможные сценарии, которые могут возникнуть при получении блокировок.

Также на базе *Unit*-тестов выполнятся интеграционное тестирования веб-сервера. Ниже представлен фрагмент теста:

```
public class ProjectTests
{
    public class ResponseArray
    {
        [JsonProperty("Data")]
        public ProjectDto[] projects { get; set; }
        public Error[] Errors { get; set; }
    }

    public class ValidRequests
    {
        private readonly TestServer server;
        private readonly HttpClient client;
        private readonly string token;

        public ValidRequests()
        {
            var testConfigBuilder = new ConfigurationBuilder();
            testConfigBuilder.AddJsonFile("appsettings.json");

            server = new TestServer(new WebHostBuilder()
                .UseConfiguration(testConfigBuilder.Build())
                .UseStartup<Startup>());

            client = server.CreateClient();

            client.Authorize().Wait();
        }
```

```
        [[Fact(DisplayName = "Request to get all project.")]
        public async Task GetAll()
        {
            var response = await client.GetAsync(ApiRequests.Projects());

            response.EnsureSuccessStatusCode();

            var responseString = await response.Content.ReadAsStringAsync();
            var responseData = JsonConvert
                .DeserializeObject<ResponseArray>(responseString);

            Assert.NotEmpty(responseData.projects);
        }

        //...
    }
}
```

В конструкторе класса *ValidRequests* загружается конфигурация и запускается веб-приложение на тестовом сервере. Затем создается экземпляр клиента, последний выполняет авторизацию с помощью метода расширения *Authorize*. Созданный клиент используется в методах, обеспечивающие тестирование. Подобным методом является *GetAll*, тестирующий сценарий получения проектов.

Выводы по разделу:

В результате реализации создан веб-сервер и компонент *ActiveX*, обеспечивающие отображение технологических объектов и информацию о состоянии технологического процесса на географической карте.

4 ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ

Введение

Современный процесс разработки программного обеспечения, на стадии предпроектного анализа должен включать этап экономического исследования, который позволит оценить коммерческую ценность и перспективность разработки. Необходимость в проведении данного этапа обусловлена тем, что рынок информационных технологий очень быстро развивается и пополняется новым программным обеспечением, изучение этих программ позволяет выявить их преимущества и недостатки, которые можно учесть при разработке собственных программ.

Объектом исследования данной работы является геоинформационная система для отображения технологических объектов, информации о параметрах и ходе технологического процесса на географической карте. С целью определения коммерческой ценности, оценки сроков выполнения работ и ресурсов проекта проводится экономическое исследование.

В данном разделе приведены результаты оценки коммерческого потенциала, перспективности разработки с позиции ресурсоэффективности и ресурсосбережения.

4.1 Задачи экономического исследования

1. Выявления слабых и сильных сторон продуктов конкурентов.
2. Определение перспективности разработки.
3. Выявление слабых и сильных сторон проекта, а также возможностей и угроз для реализации проекта.
4. Определение организационной структуры проекта.

5. Определение трудоемкости выполнения работ.
6. Разработка календарного плана.
7. Расчет бюджета проекта.

4.2 Оценка коммерческого потенциала и перспективности разработки с позиции ресурсоэффективности и ресурсосбережения

4.2.1 Потенциальные потребители результатов проекта

Основными потребителями продукта являются предприятия и организации, технологический процесс которых характерен наличием распределенных и мобильных технологических объектов. Например, к их числу относятся предприятия и организации нефтегазовой, транспортной, энергетической, горнодобывающей отраслей и т.д. На рисунке 4.1 представлена карта сегментирования отечественного рынка программных продуктов, объединяющие классические функции SCADA-систем с функциями геоинформационных приложений.

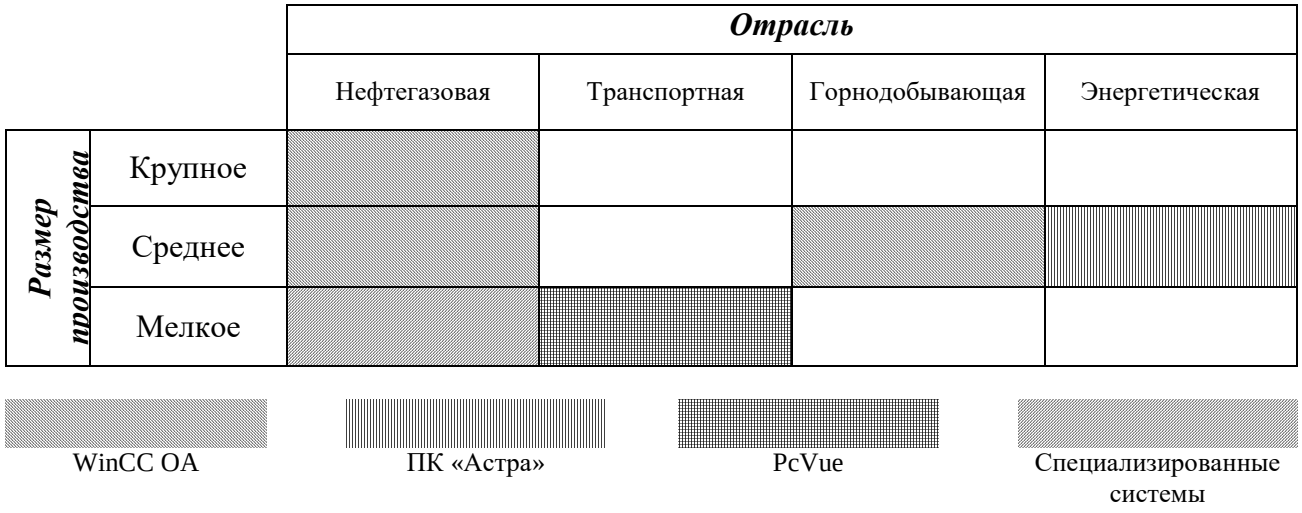


Рисунок 4.1 – Карта сегментирования отечественного рынка SCADA-систем с функциями ГИС

Анализ карты сегментирования показывает, что в целом на отечественном рынке уровень конкуренции низок. Существует большое количество свободных сегментов.

В результате сегментирования определены основные сегменты, на которые ориентирован разрабатываемый продукт, а именно это мелкие и средние предприятия и организации нефтегазовой, транспортной, горнодобывающей и энергетической отраслей.

4.2.2 Анализ конкурентных технических решений

Для анализа конкурентных технических решений выбраны SCADA-системы, в которых объедены функции контроля и сбора данных с функциями геоинформационных систем. В качестве конкурентных решений рассматриваются автоматизированный программный комплекс «Астра», WinCC OA и PcVue.

Результаты анализа, проведенного с помощью оценочной карты, приведены в таблице 4.1.

Таблица 4.1 – Оценочная карта сравнения конкурентных систем

Критерии оценки	**Вес крите-рия**	**Баллы**				**Конкурентоспособность**			
		$Б_ф$	$Б_{к1}$	$Б_{к2}$	$Б_{к3}$	$К_ф$	$К_{к1}$	$К_{к2}$	$К_{к3}$
1	2	3	4	5	6	7	8	9	10
Технические критерии оценки ресурсоэффективности									
1. Повышение производительности труда пользователя	0,09	4	4	4	3	0,36	0,36	0,36	0,27
2. Удобство в эксплуатации (соответствует требованиям потребителей)	0,1	4	5	4	3	0,40	0,50	0,40	0,30
3. Надежность	0,11	5	5	5	5	0,55	0,55	0,55	0,55
4. Безопасность	0,11	5	5	5	5	0,55	0,55	0,55	0,55

5. Потребность в ресурсах памяти	0,03	4	4	3	4	0,12	0,12	0,09	0,12
6. Функциональные возможности ГИС-модуля	0,08	3	4	3	3	0,24	0,32	0,24	0,24
7. Простота эксплуатации	0,07	5	3	4	5	0,35	0,21	0,28	0,35
8. Качество интеллектуального интерфейса	0,06	4	4	4	4	0,24	0,24	0,24	0,24
9. Возможность применения в различных отраслях	0,01	5	3	5	5	0,05	0,03	0,05	0,05
Экономические критерии оценки эффективности									
1. Конкурентоспособность продукта	0,05	5	4	5	3	0,25	0,20	0,25	0,15
2. Уровень проникновения на рынок	0,03	4	4	4	3	0,12	0,12	0,12	0,09
3. Цена	0,2	4	3	3	4	0,80	0,60	0,60	0,80
4. Послепродажное обслуживание	0,05	5	4	4	3	0,25	0,20	0,20	0,15
5. Срок выхода на рынок	0,01	4	4	3	3	0,04	0,04	0,04	0,04
Итого	**1**					**4,32**	**4,04**	**3,97**	**3,89**

Результаты экспертной оценки позволяют сделать вывод, что уязвимость программного комплекса (ПК) «Астра» связана с простотой эксплуатации, возможностью широкого применения, ценой и с обслуживаем во время эксплуатации.

Уязвимость WinCC OA связана с потреблением в ресурсах памяти, а также, аналогично ПК «Астра», с ценой и послепродажным обслуживанием.

Для PcVue критерии оценки, с которыми связана уязвимость следующие: повышение производительности труда пользователя, удобство в эксплуатации, конкурентоспособность продукта, уровень проникновения на рынок и послепродажное обслуживание.

Стоит также отметить преимущества конкурентных решений. ПК «Астра» превосходит удобством в эксплуатации и набором функций ГИС-модуля.

4.2.3 Диаграмма Исикавы

Диаграмма Исикавы, отображающая причинно-следственные связи представлена на рисунке 4.2. Данная диаграмма позволяет вывить факторы и причины, ставшие причиной неудовлетворенности клиента продуктом.

Рисунок 4.2 – Диаграмма Исикавы

4.2.4 QuaD – анализ

Описание качества новой разработки и ее перспективности на рынке выполнена с помощью технологии QuaD. Результат оценки, с учетом технических и экономических особенностей разработки приведен в таблице 4.2.

Таблица 4.2 – QuaD-анализ

Критерии оценки	**Вес критерия**	**Баллы**	**Макси-мальный балл**	**Относительное значение** (3/4)	**Средневзвешенное значение** (5x2)
1	2	3	4	5	
Показатели оценки качества разработки					

1. Повышение производительности труда пользователя	0,09	85	100	0,85	7,65
2. Удобство в эксплуатации (соответствует требованиям потребителей)	0,1	95	100	0,95	9,50
3. Надежность	0,11	90	100	0,90	9,90
4. Безопасность	0,11	85	100	0,85	9,35
5. Потребность в ресурсах памяти	0,03	70	100	0,70	2,10
6. Функциональные возможности ГИС-модуля	0,08	100	100	1,00	8,00
7. Простота эксплуатации	0,07	90	100	0,90	4,80
8. Качество интеллектуального интерфейса	0,06	80	100	0,80	4,80
9. Возможность применения в различных отраслях	0,01	90	100	0,90	0,90
Показатели оценки коммерческого потенциала разработки					
10. Конкурентоспособность продукта	0,05	95	100	0,95	4,75
11. Уровень проникновения на рынок	0,03	90	100	0,90	2,70
12.Цена	0,2	80	100	0,80	16,00
13.Послепродажное обслуживание	0,05	90	100	0,90	4,50
14.Срок выхода на рынок	0,01	75	100	0,75	0,75
Итого	**1**				**87,2 %**

В результате проведённого анализа, перспективность разработки составило 87%, из значения этого показателя можно сделать вывод, что данная разработка является перспективной, так как оно входит в диапазон от 80 до 100.

4.1.5 SWOT–анализ

Комплексный анализ, исследование внешней и внутренней среды проекта выполнен посредством SWOT-анализа. Результаты определения сильных и слабых сторон проекта, выявления возможностей и угроз для реализации проекта представлены в таблице 4.3.

Таблица 4.3 – Матрица SWOT

	Сильные стороны проекта: С1. Отсутствие отечественных аналогов С2. Кроссплатформенность С3. Простота эксплуатации С4. Централизованное хранение данных С5. Широкая область применения С6. Привычный пользовательский интерфейс	**Слабые стороны проекта:** Сл1. Необходимость поддержки системных администраторов Сл2. Необходимость приобретения сервера OPC UA Сл3. Малая функциональная мощность Сл4. Сложность внедрения Сл5. Поддержка одного картографического сервиса
Возможности: В1. Добавление новых функциональных возможностей с учетом потребностей заказчика В2. Добавление веб- и мобильного клиентов В3. Улучшение пользовательского интерфейса В4. Поддержка 3D-объектов В5. Поддержка новых картографических сервисов В6. Поддержка картографических данных в векторном формате	В1С2С5 – добавление новых функций позволит расширить функциональность для всех платформ и отраслей В2 С1С3С4С5С5 – добавление новых клиентов позволит улучшить конкурентоспособность, обеспечить простоту эксплуатации продукта, доступ к данным с различных устройств. В3С3С6 – улучшение пользовательского интерфейса позволит сделать его более интуитивно понятным, следовательно, сделает продукт более простым в эксплуатации. И это также позволит сохранить	В1В2Сл1Сл2 – добавлению новых функциональных возможностей может воспрепятствовать политика администрирования заказчика. В4В5В6Сл3 – поддержка возможностей увеличит стоимость продукта.

	привычные пользователю элементы управления. В4В5В6С1С5 – поддержка 3D, картографических сервисов и векторного формата позволят значительно повысить конкурентоспособность продукта и обеспечит возможность применения этих возможностей в различных отраслях автоматизации.	
Угрозы: У1. Отсутствие спроса на продукт У2. Появление на отечественном рынке аналогов разработки У3. Неверное выполнение инструкций пользователем У4. Несвоевременное финансирование У5. Изменение программного интерфейса картографического сервиса Open Street Map У6. Изменение спецификации протокола OPC UA	У1У2С1С2С3С4С5С6 – сильные стороны продукта позволят не допустить отсутствия спроса на продукт и при появлении конкурентов на рынке обеспечат конкурентоспособность. У3С3 – простота в эксплуатации позволит избежать неправильного выполнения инструкций пользователем. У6С4 – при изменении спецификаций потребуется коррективы лишь в одном месте проекта.	У1Сл2Сл3 – малая функциональная мощность, необходимость приобретения сервера OPC UA могут снизить спрос на продукт. У5Сл5 – изменение программного интерфейса картографического сервиса нарушит работоспособность продукта. У6Сл2 – изменение спецификации протокола OPC UA повлечет за собой необходимость покупки нового сервера.

Из результатов SWOT-анализа следует, что для повышения конкурентоспособности продукта необходимо в будущем расширить ее функциональность, создать новых клиентов, повысить эргономичность интерфейса пользователя. А также с целью исключения реализации известных рисков необходимо предусмотреть возможность гибкой смены источников данных.

4.3 Определение возможных альтернатив проведения разработки

Определение альтернатив проведения разработки выполнен с помощью морфологического подхода. Морфологическая матрица для составляющих реализации рассматриваемого проекта представлена в таблице 4.4.

Таблица 4.4 – Морфологическая матрица

	1	2	3	4
А. Технология разработки сервера	ASP.NET Core	Spring	Django	YII
Б. Технология разработки клиента	Windows Forms	Swing	PyQT	C++ QT
В. Протокол обмена данными с датчиками полевого уровня	OPC UA	OPC	SOAP	HTTP
Г. Система управления базами данных	PostgreSQL	MSSQL	MySQL	MirandaDB
Д. Источник картографических данных	Open Street Map	Google Maps	Yandex Maps	Bing Maps

Из составленной морфологической матрицы определены наиболее вероятные альтернативы разработки проекта:

1. Исполнение 1. А1Б1В1Г1Д1.
2. Исполнение 2. А2Б2В2Г2Д2.
3. Исполнение 3. А3Б3В3Г3Д3.
4. Исполнение 4. А4Б4В1Г4Д4.

В рамках ВКР реализуется первое исполнение.

4.4 Планирование научно-исследовательских работ

4.4.1 Организационная структура проекта

Планирование комплекса предполагаемых работ осуществляется в следующем порядке:

- определение структуры работ в рамках проекта;
- определение участников каждой работы;
- установление продолжительности работы;
- построение графика выполнения проекта.

Для выполнения проекта была сформирована рабочая группа, которая представлена в таблице 4.5.

Таблица 4.5 – Рабочая группа проекта

ФИО, должность	Должность	Функции
Айдаров Шукурдин Бахтиярович, студент группы 8ИМ6А	Исполнитель	1. Проектирование и разработка системы 2. Тестирование системы 3. Разработка технической документации
Ким Валерий Львович, д.т.н., профессор отделения информационных технологий	Научный руководитель	1. Координирование деятельности исполнителя проекта
Харин Сергей Сергеевич, начальник отдела разработки транспортных систем	Руководитель от предприятия	1. Формирование целей проекта 2. Координирование деятельности проекта 3. Консультирование по вопросам проектирования и разработки

Перечень этапов и работ в рамках выполнения проекта, а также распределение исполнителей по видам работ, представлен в таблице 4.6.

Таблица 4.6 – Перечень этапов работ и распределение исполнителей

Основные этапы	**№ раб**	**Содержание работ**	**Должность исполнителя**
Постановка задачи	1	Подбор и изучение материалов по теме	исполнитель
	2	Сбор и описание потребностей	исполнитель
	3	Обзор существующих решений	руководитель от предприятия, исполнитель
	4	Обзор источников картографических данных	исполнитель
	5	Обзор инструментов разработки ГИС	исполнитель
	6	Составление и утверждение технического задания	руководитель от предприятия, исполнитель
Проектирование веб-сервера	7	Проектирование архитектуры сервера	исполнитель
	8	Концептуальное проектирование базы данных	
	9	Проектирование REST API	
	10	Проектирование модулей сервера	
Проектирование клиента	11	Проектирование архитектуры клиента	исполнитель
	12	Проектирование модулей клиента	
	13	Разработка эскизов пользовательского интерфейса	
Разработка веб-сервера	14	Разработка компонентов слоя доступа к данным	исполнитель
	15	Разработка компонентов слоя предметной области	
	16	Разработка компонентов слоя сервисов	
Разработка клиента	17	Разработка компонентов пользовательского интерфейса	исполнитель
	18	Разработка компонентов логики клиента	
	19	Разработка агента REST API	
Тестирование системы	20	Интеграционное тестирование	исполнитель
	21	Функциональное тестирование	
	22	Исправление ошибок	

Оформление документации	23	Разработка документации к REST API	исполнитель
	24	Оформление пояснительной записки	
	25	Проверка работы	научный руководитель

4.4.2 Определение трудоемкости выполнения работ

Для определения ожидаемого (среднего) значения трудоемкости $t_{ожi}$ используется следующая формула:

$$t_{\text{ожі}} = \frac{3t_{\min i} + 2t_{\max i}}{5}, \quad (4.1)$$

где $t_{ожi}$ – ожидаемая трудоемкость выполнения *i*-ой работы чел.-дн.;

$t_{\min i}$ – минимально возможная трудоемкость, чел.-дн.;

$t_{\max i}$ – максимально возможная трудоемкость, чел.-дн.

Исходя из ожидаемой трудоемкости работ, определяется продолжительность каждой работы в рабочих днях $T_р$, учитывающая параллельность выполнения работ несколькими исполнителями.

$$T_{\text{р}_i} = \frac{t_{\text{ожі}}}{\text{Ч}_i}, \quad (4.2)$$

где $T_{рi}$ – продолжительность одной работы, раб. дн.;

$t_{ожi}$ – ожидаемая трудоемкость выполнения одной работы, чел.-дн.

Ч_i – численность исполнителей, выполняющих одновременно одну и ту же работу на данном этапе, чел.

Далее необходимо рабочие дни перевести в календарные. Для этого используется следующая формула:

$$T_{\text{кі}} = T_{\text{р}i} \cdot k_{\text{кал}}, \quad (4.3)$$

где $T_{кi}$ – продолжительность выполнения i-й работы в календарных днях;

$T_{рi}$ – продолжительность выполнения i-й работы в рабочих днях;

$k_{кал}$ – коэффициент календарности.

Коэффициент календарности определяется по следующей формуле:

$$k_{кал} = \frac{T_{кал}}{T_{кал} - T_{вых} - T_{пр}}, \quad (4.4)$$

где $T_{кал}$ – количество календарных дней в году; $T_{вых}$ – количество выходных дней в году; $T_{пр}$ – количество праздничных дней в году.

Результаты расчетов временных показателей приведены в таблице 4.7.

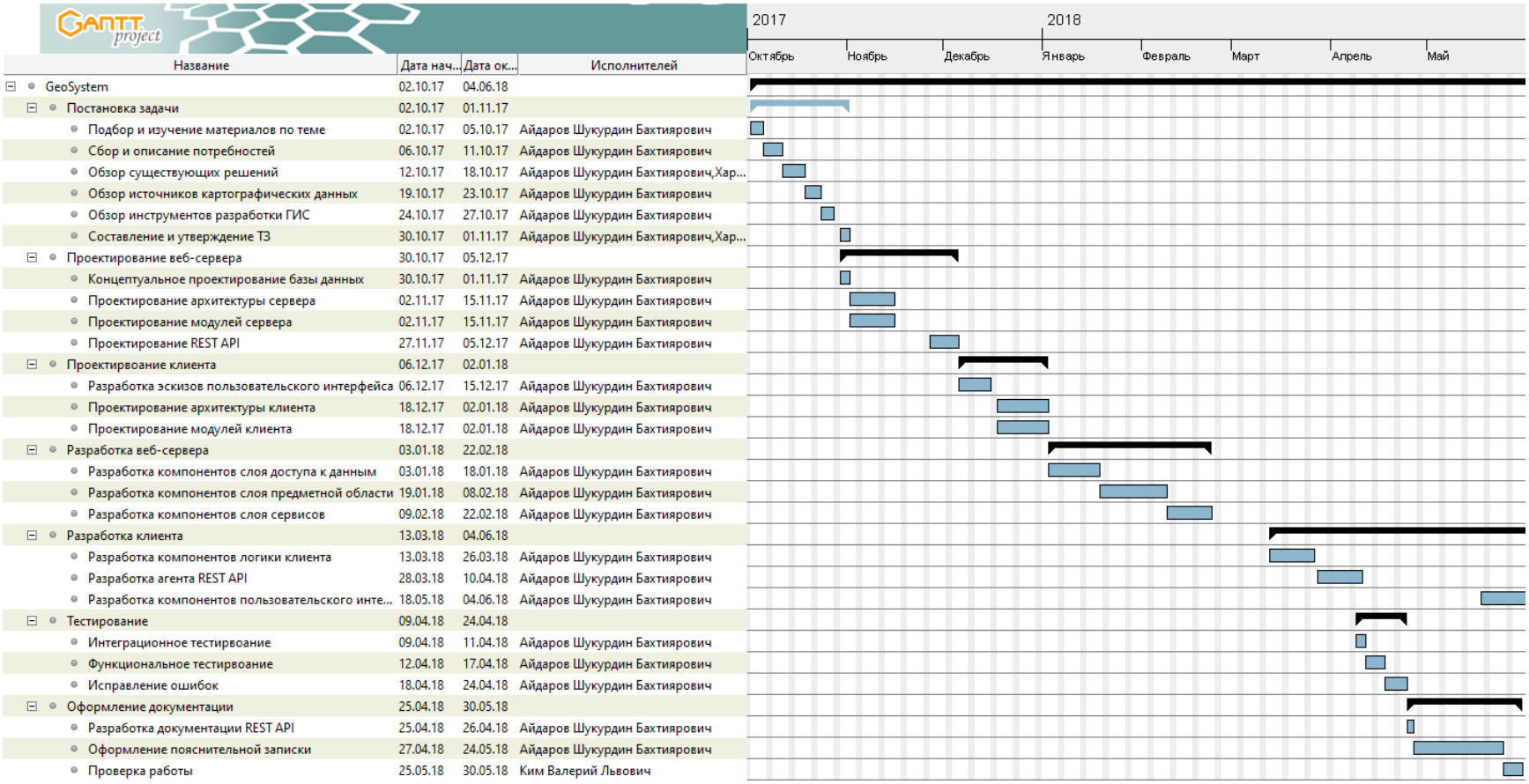
Диаграмма Гантта для текущего проекта представлена на рисунке 4.3.

Таблица 4.7 – Временные показатели выполнения проекта

Название работы	**Трудоёмкость работ**												**Исполнители**				**Длительность работ в рабочих днях**				**Длительность работ в календарных днях**			
	t_{min}, чел-дни				**t_{max},чел-дни**				**$t_{ож}$, чел-дни**															
	И1	И2	И3	И4	И1	И2	И3	И4	И1	И2	И3	И4	И1	И2	И3	И4	И1	И2	И3	И4	И1	И2	И3	И4
Подбор и изучение материалов по теме	2	2	2	2	4	4	4	4	3	3	3	3	1	1	1	1	3	3	3	3	4	4	4	4
Сбор и описание потребностей	2	2	2	2	4	4	4	4	3	3	3	3	1	1	1	1	3	3	3	3	4	4	4	4
Обзор существующих решений	4	4	4	4	6	6	6	6	5	5	5	5	2	2	2	2	3	3	3	3	4	4	4	4
Обзор источников картографических данных	1	1	1	1	3	3	3	3	2	2	2	2	1	1	1	1	2	2	2	2	3	3	3	3
Обзор инструментов реализации функциональности ГИС	2	2	2	2	4	4	4	4	3	3	3	3	1	1	1	1	3	3	3	3	4	4	4	4
Составление и утверждение технического задания	1	1	1	1	3	3	3	3	2	2	2	2	1	1	1	1	2	2	2	2	3	3	3	3
Проектирование архитектуры сервера	5	5	5	5	10	10	10	10	7	7	7	7	1	1	1	1	7	7	7	7	10	10	10	10
Концептуальное проектирование базы данных	2	2	2	2	3	3	3	3	2	2	2	2	1	1	1	1	2	2	2	2	3	3	3	3
Проектирование REST API	4	4	4	4	6	6	6	6	5	5	5	5	1	1	1	1	5	5	5	5	7	7	7	7
Проектирование модулей сервера	5	5	5	5	10	10	10	10	7	7	7	7	1	1	1	1	7	7	7	7	10	10	10	10
Проектирование архитектуры клиента	6	6	6	6	11	11	11	11	8	8	8	8	1	1	1	1	8	8	8	8	12	12	12	12

Продолжение таблица 4.7

Проектирование модулей клиента	6	6	6	6	11	11	11	11	8	8	8	8	1	1	1	1	8	8	8	8	12	12	12	12
Разработка эскизов пользовательского интерфейса	3	3	3	3	9	9	9	9	5	5	5	5	1	1	1	1	5	5	5	5	7	7	7	7
Разработка компонентов слоя доступа к данным	5	6	8	10	10	11	14	18	7	8	10	13	1	1	1	1	7	8	10	13	10	12	15	19
Разработка компонентов предметной области	8	8	8	8	12	12	12	12	10	10	10	10	1	1	1	1	10	10	10	10	15	15	15	15
Разработка компонентов слоя сервисов	5	6	8	10	12	14	16	18	8	9	11	13	1	1	1	1	8	9	11	13	12	13	16	19
Разработка компонентов логики клиента	5	5	5	5	10	10	10	10	7	7	7	7	1	1	1	1	7	7	7	7	10	10	10	10
Разработка компонентов пользовательского интерфейса	6	8	8	10	11	16	16	18	8	11	11	13	1	1	1	1	8	11	11	13	12	16	16	19
Разработка агента REST API	5	6	8	12	10	12	15	20	7	8	11	15	1	1	1	1	7	8	11	15	10	12	16	22
Интеграционное тестирование	1	1	1	1	4	4	4	4	2	2	2	2	1	1	1	1	2	2	2	2	3	3	3	3
Функциональное тестирование	2	2	2	2	5	5	5	5	3	3	3	3	1	1	1	1	3	3	3	3	4	4	4	4
Исправление ошибок	2	2	2	2	4	4	4	4	3	3	3	3	1	1	1	1	3	3	3	3	4	4	4	4
Разработка документации к REST API	1	1	1	1	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Оформление пояснительной записки	12	12	12	12	16	16	16	16	14	14	14	14	1	1	1	1	14	14	14	14	21	21	21	21
Проверка работы	3	3	3	3	6	6	6	6	4	4	4	4	1	1	1	1	4	4	4	4	6	6	6	6
Итого									**134**	**140**	**147**	**158**					**132**	**138**	**145**	**156**	**191**	**200**	**210**	**226**



Название	Дата нач...	Дата ок...	Исполнителей
GeoSystem	02.10.17	04.06.18	
Постановка задачи	02.10.17	01.11.17	
Подбор и изучение материалов по теме	02.10.17	05.10.17	Айдаров Шукурдин Бахтиярович
Сбор и описание потребностей	06.10.17	11.10.17	Айдаров Шукурдин Бахтиярович
Обзор существующих решений	12.10.17	18.10.17	Айдаров Шукурдин Бахтиярович,Хар...
Обзор источников картографических данных	19.10.17	23.10.17	Айдаров Шукурдин Бахтиярович
Обзор инструментов разработки ГИС	24.10.17	27.10.17	Айдаров Шукурдин Бахтиярович
Составление и утверждение ТЗ	30.10.17	01.11.17	Айдаров Шукурдин Бахтиярович,Хар...
Проектирование веб-сервера	30.10.17	05.12.17	
Концептуальное проектирование базы данных	30.10.17	01.11.17	Айдаров Шукурдин Бахтиярович
Проектирование архитектуры сервера	02.11.17	15.11.17	Айдаров Шукурдин Бахтиярович
Проектирование модулей сервера	02.11.17	15.11.17	Айдаров Шукурдин Бахтиярович
Проектирование REST API	27.11.17	05.12.17	Айдаров Шукурдин Бахтиярович
Проектирвоание клиента	06.12.17	02.01.18	
Разработка эскизов пользовательского интерфейса	06.12.17	15.12.17	Айдаров Шукурдин Бахтиярович
Проектирование архитектуры клиента	18.12.17	02.01.18	Айдаров Шукурдин Бахтиярович
Проектирование модулей клиента	18.12.17	02.01.18	Айдаров Шукурдин Бахтиярович
Разработка веб-сервера	03.01.18	22.02.18	
Разработка компонентов слоя доступа к данным	03.01.18	18.01.18	Айдаров Шукурдин Бахтиярович
Разработка компонентов слоя предметной области	19.01.18	08.02.18	Айдаров Шукурдин Бахтиярович
Разработка компонентов слоя сервисов	09.02.18	22.02.18	Айдаров Шукурдин Бахтиярович
Разработка клиента	13.03.18	04.06.18	
Разработка компонентов логики клиента	13.03.18	26.03.18	Айдаров Шукурдин Бахтиярович
Разработка агента REST API	28.03.18	10.04.18	Айдаров Шукурдин Бахтиярович
Разработка компонентов пользовательского инте...	18.05.18	04.06.18	Айдаров Шукурдин Бахтиярович
Тестирование	09.04.18	24.04.18	
Интеграционное тестирвоание	09.04.18	11.04.18	Айдаров Шукурдин Бахтиярович
Функциональное тестирвоание	12.04.18	17.04.18	Айдаров Шукурдин Бахтиярович
Исправление ошибок	18.04.18	24.04.18	Айдаров Шукурдин Бахтиярович
Оформление документации	25.04.18	30.05.18	
Разработка документации REST API	25.04.18	26.04.18	Айдаров Шукурдин Бахтиярович
Оформление пояснительной записки	27.04.18	24.05.18	Айдаров Шукурдин Бахтиярович
Проверка работы	25.05.18	30.05.18	Ким Валерий Львович

Рисунок 4.3 – Диаграмма Гантта

4.4.3 Бюджет научно-технического исследования

При планировании бюджета проекта рассматривается полное отражение всех видов расходов, связанных с его выполнением. В процессе формирования бюджета проекта используется следующая группировка затрат по статьям:

- материальные затраты;
- основная заработная плата исполнителей темы;
- дополнительная заработная плата исполнителей темы;
- отчисления во внебюджетные фонды (страховые отчисления);
- накладные расходы.

4.4.3.1 Расчет материальных затрат

Материальные затраты включают стоимость всех материалов, используемых при разработке проекта. Для выполнения работ проекта использован один персональный компьютер стоимостью 60000 руб. Мелкие расходы такие, как канцелярия и затраты на печать, могут быть отнесены к статье прочих расходов.

Материальные затраты проекта одинаковы для всех вариантов исполнения.

4.4.3.2 Расчет основной заработной платы исполнителей проекта

В данную статью включается основная заработная плата научного руководителя, руководителя от предприятия и студента, также премия, выплачиваемая ежемесячно из фонда заработной платы в размере 20 –30 % от тарифа или оклада. Расчет выполняется на основе трудоемкости выполнения

каждого этапа и величины месячного оклада исполнителя. Результаты расчета основной заработной платы представлен в таблице 4.8.

4.4.3.3 Расчет затрат по дополнительной заработной плате

Данная статья включает затраты по дополнительной заработной плате исполнителей проекта, учитывающие величину доплат, связанных с обеспечением гарантий и компенсаций.

Расчет дополнительной заработной платы ведется по формуле:

$$З_{доп} = k_{доп} \cdot З_{осн} \quad (4.5)$$

где $k_{доп}$ – коэффициент дополнительной заработной платы (на стадии проектирования принимается равным 0,15).

Результаты расчета представлены в таблицы 4.9.

Таблица 4.8 – Основная заработная плата исполнителей системы

Исполнитель	Оклад, руб./мес	Среднедневная ставка	Затраты времени, раб. Дни				Коэффициент	Фонд з/платы, руб.			
			И1	И2	И3	И4		И1	И2	И3	И4
Научный руководитель	33264,9	1738,4	4	4	4	4	1,3	6953.6	6953.6	6953.6	6953.6
Руководитель от предприятия	50000	2500,0	5	5	5	5	1,3	12500	12500	12500	12500
Исполнитель	25000	1250	132	138	145	156	1,3	165000	172500	181250	195000
Итого								184453	191953	200703	214453

Таблица 4.9 – Дополнительная заработная плата исполнителей системы

Исполнитель	Основная заработная плата , руб.				Коэффициент	Дополнительная заработная плата, руб.			
	Исп. 1	Исп. 2	Исп. 3	Исп. 4		Исп. 1	Исп. 2	Исп. 3	Исп. 4
Научный руководитель	6953	6953	6953.6	6953	0,12	834	834	834	834
Руководитель от предприятия	12500	12500	12500	12500		1500	1500	1500	1500
Студент	165000	172500	181250	195000		19800	20700	21750	23400
Итого						22134	23034	24084	25734

4.4.3.4 Расчет отчислений во внебюджетные фонды

В данной статье расходов отражаются обязательные отчисления по установленным законодательством Российской Федерации нормам органам государственного социального страхования (ФСС), пенсионного фонда (ПФ) и медицинского страхования (ФФОМС) от затрат на оплату труда работников.

На 2018 г. в соответствии с положениями ст.58.2 закона №212-ФЗ установлены следующие тарифы страховых взносов: ПФРФ – 0.22 (22%), ФСС РФ – 0.029 (2,9%), ФФОМС – 0,051 (5,1%). Результаты расчетов представлены в таблице 4.10.

Таблица 4.10 – Отчисления во внебюджетные фонды

Исполнитель	**Основная заработная плата, руб.**				**Дополнительная заработная плата, руб.**			
	Исп. 1	**Исп. 2**	**Исп. 3**	**Исп. 4**	**Исп. 1**	**Исп. 2**	**Исп. 3**	**Исп. 4**
Научный руководитель	6953	6953	6953	6953	834	834	834	834
Руководитель от предприятия	12500	12500	12500	12500	1500	1500	1500	1500
Студент								
Коэффициент ПФРФ	0,22							
Коэффициент ФСС	0,029							
Коэффициент ФФОМС	0,051							
Итого								
Исполнение 1	61976							
Исполнение 2	64496							
Исполнение 3	67436							
Исполнение 4	72056							

4.4.3.5 Расчет накладных расходов

Накладные расходы учитывают все затраты, не вошедшие в предыдущие статьи расходов: печать и ксерокопирование, оплата электроэнергии, оплата пользования услугами и пр.

Величину коэффициента накладных расходов можно принять в размере 16%. Результаты расчета накладных для каждого варианта исполнения представлены в таблице 4.11.

4.4.3.6 Формирование бюджета затрат проекта

Сумма затрат по всем статьям расходов представлена в таблице 4.11.

Таблица 4.11 – Бюджет затрат проекта

Статья затрат	**Сумма, руб.**			
	Исп. 1	**Исп. 2**	**Исп. 3**	**Исп. 4**
Материальные затраты НТИ	6000	6000	6000	6000
Основная заработная плата исполнителей темы	184453	191953	200703	214453
Дополнительная заработная плата исполнителей темы	22134	23034	24084	25734
Отчисления во внебюджетные фонды (страховые отчисления)	61976	64496	67436	72056
Накладные расходы	52570	54317	56355	59559
Итого	381134	393801	408580	431803

Из анализа данных таблицы 4.11 следует, что наименьший бюджет, необходимый для разработки проекта приходится на исполнение 1.

4.5 Определение ресурсной, финансовой, бюджетной, социальной и экономической эффективности исследования

Определение эффективности производится путем определения интегрального показателя эффективности научного исследования через нахождение величин финансовой и ресурсной эффективности.

Интегральный финансовый показатель определяется по следующе формуле:

$$I_{\text{финр}}^{исп.i} = \frac{\Phi_{\text{p}i}}{\Phi_{\max}} \tag{4.6}$$

где $I_{\text{финр}}^{\text{исп.i}}$ – интегральный финансовый показатель разработки; $\Phi_{\text{p}i}$ – стоимость *i*-го варианта исполнения; $\Phi_{\max}$ – максимальная стоимость исполнения научно-исследовательского проекта.

Интегральный показатель ресурсоэффективности определяется по формуле:

$$\text{I}_{\text{pi}} = \sum_{i}^{n} a_i b_i \tag{4.7}$$

где I_{pi} – интегральный показатель ресурсоэффективности для i-го варианта исполнения разработки; a_i – весовой коэффициент i-го варианта исполнения разработки; b_i^p – бальная оценка i-го варианта исполнения разработки, устанавливается экспертным путем по выбранной шкале оценивания; n – число параметров сравнения.

Результаты сравнительной оценки характеристик вариантов исполнения проекта приведен в таблице 4.12.

Таблица 4.12 – Сравнительная оценка характеристик вариантов исполнения проекта

Критерии	**Объект исследования**				
	Весовой коэффициент	**Исп.1**	**Исп.2**	**Исп.3**	**Исп.4**
1. Повышение производительности труда пользователя	0,09	4	4	4	4
2. Удобство в эксплуатации (соответствует требованиям потребителей)	0,1	5	4	3	3
3. Надежность	0,11	5	5	5	5
4. Безопасность	0,11	5	5	5	5
5. Потребность в ресурсах памяти	0,03	4	4	4	5
6. Функциональные возможности ГИС-модуля	0,08	4	4	4	4
7. Простота эксплуатации	0,07	4	4	4	4
8. Качество интеллектуального интерфейса	0,06	4	4	4	4
9. Возможность применения в различных отраслях	0,01	5	5	5	5
Экономические критерии оценки эффективности					
1. Конкурентоспособность продукта	0.05	5	3	3	4
2. Уровень проникновения на рынок	0.03	4	4	4	4
3. Цена	0.20	5	4	3	3
4. Послепродажное обслуживание	0.05	5	5	5	5
5. Срок выхода на рынок	0.01	5	5	5	5
Ресурсоэффективность		**4,64**	**4,24**	**3,94**	**4,02**

Интегральный показатель эффективности вариантов исполнения разработки определяется по формуле:

$$I_{исп.1} = \frac{I_{р-исп1}}{I_{финр}^{исп.1}} \quad (4.8)$$

После этого определяется сравнительная эффективность исполнений разработки, которая позволит определить самый выгодный вариант разработки с позиции финансовой и ресурсной эффективности:

$$Э_{ср} = \frac{I_{исп.1}}{I_{исп.2}} \quad (4.9)$$

Результаты сравнительной эффективности разработки представлена в таблице 4.13.

Таблица 4.13 – Сравнительная эффективность разработки

№ п/п	Показатели	Исп. 1	Исп. 2	Исп. 3	Исп. 4
1	Интегральный финансовый показатель разработки	0.88	0,91	0,95	1
2	Интегральный показатель ресурсоэффективности разработки	4,64	4,24	3,94	4,02
3	Интегральный показатель эффективности	5,27	4,66	4,15	4,02
4	Сравнительная эффективность вариантов разработки	1	0,88	0,79	0.76

По результатам таблицы 4.13 можно сделать вывод, что самым эффективным исполнением с позиции ресурсоэффективности и финансовой эффективности является первое исполнение. Наименее эффективным является четвертое исполнение, поскольку его интегральный показатель эффективности на 24% ниже, чем у первого исполнения.

Выводы по разделу:

Необходимость произведенного исследования обусловлена тем, что вероятность реализации организационных, финансовых и кадровых рисков при разработке программного продукта очень велика. Работы по оценке коммерческого

потенциала и перспективности, а также планирования работ позволяют уменьшить вероятность реализации рисков.

В ходе экономического исследования были определены организационная структура проекта и эффективный вариант исполнения, проведен комплексный анализ, а также рассчитан бюджет проекта.

5 СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ

Введение

В современном мире, вместе с развитием техники и технологий появляется все больше опасных факторов, влияющих на условия труда человека. Трудовая деятельность разработчика программного обеспечения тесно связана с вычислительными машинами, многочасовой работой в сидячем положении в производственном помещении. При этом на работника могут негативно воздействовать различные физические, биологические и психофизиологические вредные и опасные факторы.

В настоящем разделе рассматриваются вопросы производственной и экологической безопасности, безопасность в чрезвычайных ситуациях, а также правовые и организационные вопросы обеспечения безопасности, связанные с выполнением работ по разработке и эксплуатации проектируемого решения.

Объектом исследования является геоинформационное приложение для отображения технологических объектов, информации о параметрах и ходе технологического процесса на географической карте. Данный продукт расширит функциональность основного продукта компании ЭлеСи SCADA Infinity и обеспечит конкурентное преимущество над отечественными SCADA-системами.

5.1 Производственная безопасность

Поскольку все работы при разработке и эксплуатации решения выполняются в офисном помещении на территории АО «ЭлеСи», то основными источниками вредных и опасных факторов являются электронно-вычислительные устройства и элементы электрической сети этого помещения. Перечень вредных и опасных

факторов согласно ГОСТ 12.0.003-74, характерных для проектируемого решения представлены в таблице 5.1.

Таблица 5.1- Опасные и вредные факторы при выполнении работ по разработке геоинформационной подсистемы для SCADA Infinity

Источник фактора, наименование видов работ	Факторы (по ГОСТ 12.0.003-74)		Нормативные документы
	Вредные	Опасные	
Выполнение работ в офисном помещении	1. Повышенный уровень электромагнитных излучений	1. Электрический ток	1.Уровень электромагнитного излучения: СанПиН 2.2.2/2.4.1340-03 и СанПиН 2.2.4.1191-03

5.1.1 Повышенный уровень электромагнитных излучений

Основные работы, как было описано выше, связанные с разработкой и эксплуатацией проектируемого решения, выполняются в офисном помещении, где находятся различные технические приборы: персональные компьютеры, стационарные телефоны и факсы, принтеры, сканеры, мобильные устройства сотрудников, электрическая проводка и прочее. Все эти устройства являются источниками электромагнитных излучений, которые вызывают у человека функциональные нарушения нервной системы, слабость, раздражительность, быструю утомляемость, ослабление памяти, нарушение сна и многое другое.

Гигиенические требования к персональным электронно-вычислительным машинам и организации работы СанПиН 2.2.2/2.4.1340-03 определяют величину напряженности электромагнитного поля и плотность магнитного потока: в

диапазоне 5 Гц ÷ 2 кГц эти величины не должны превышать 25 В/м и 250 нТл соответственно; в диапазоне 2 кГц ÷ 400кГц - 2,5 В/м и 25 нТл.

Ниже перечислены способы, позволяющие уменьшить действие электромагнитного излучения на организм человека:

- следует установить монитор компьютера на расстоянии 60-80 см от глаз, но не менее 50 см. А системный блок компьютера на максимально возможное расстояние;
- необходимо делать 15-ти минутные перерывы во время работы за компьютером каждые 2 часа;
- по окончанию рабочего дня следует отключить от сети все возможные электрические устройства;
- мобильные телефоны стоит отложить на максимально возможное расстояние;
- оргтехнику нужно разместить на расстоянии не менее 1,5 м от рабочего места;
- в случае мощных электромагнитных излучений, следует использовать средства защиты, ограничивающие поступление электромагнитного излучения на рабочее место: специальных экранов, поглотителей излучений и других средств индивидуальной защиты.

Считается, что одним из самых эффективных способов, позволяющих избавится от действий электромагнитных излучений является прогулка по свежему воздуху. Работникам офисных помещений следует регулярно бывать в лесу, гулять в парке, ходить в походы и т.д.

5.1.2 Электрический ток

Поражение электрическим током является опасным производственным фактором и, поскольку выполнение работ по разработке и эксплуатации проектируемого решения требует использования электронно-вычислительных устройств, то вопросам электробезопасности необходимо уделить особое внимание. Нормы электробезопасности на рабочем месте регламентируются СанПиН 2.2.2/2.4.1340-03, требования к защите от поражения электрическим током приведены в ГОСТ Р 12.1.019-2009 ССБТ.

Вероятность поражения электрическим током увеличивается при наличии оголенных участков кабеля, нарушении изоляции электрических устройств, неисправности оборудования и т.д.

Электрический ток может стать причиной тяжелых несчастных случаев, большая часть которых происходит из-за пренебрежения к опасности, которую представляет собой электрический ток.

Для предотвращения возможности поражения электрическим током следует проводить организационные мероприятия по обеспечению безопасности, включающий инструктаж и обучение безопасным методам труда, проверку знаний правил безопасности и инструкций в соответствии с занимаемой должностью применительно к выполняемой работе. А также необходимо обеспечить соблюдение следующих мероприятий:

- при производстве монтажных работ необходимо использовать только исправный инструмент, аттестованный службой КИПиА;
- 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 должны быть запрещены;
- все работы по устранению неисправностей должен производить квалифицированный персонал;
- необходимо постоянно следить за исправностью электропроводки.

5.2 Экологическая безопасность

В ходе выполнения работ по разработке и эксплуатации проектируемого решения отсутствуют выбросы каких-либо вредных веществ в атмосферу, следовательно, загрязнение воздуха не происходит. Также не происходит сбросов в водоемы, поэтому не оказывается никакого влияния на гидросферу. Но непосредственно во время разработки системы образовались отходы, такие как использованные аккумуляторные батареи, канцелярские принадлежности, бумага и компоненты искусственного освещения.

Особое внимание необходимо уделить компоненту искусственного освещения, такому как люминесцентная лампа. Эти лампы, применяющиеся для искусственного освещения рабочих мест, также требуют особой утилизации, так как 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.

5.3 Безопасность в чрезвычайных случаях

Чрезвычайные ситуации бывают техногенного, природного, биологического, социального или экологического характера. При работе в офисном 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;
- 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.

Наиболее вероятной чрезвычайной ситуацией, которое может возникнуть при разработке и эксплуатации системы является чрезвычайная ситуация техногенного характера, а именно пожар. Данная ситуация может возникнуть по ряду причин: короткое замыкание в электрической проводке, являющееся следствием нарушения изоляции, электросоединений и электрораспределительных щитов; возгорание электрических устройств, по причине внутренней неисправности; возгорание мебели и устройств искусственного освещения. Также пожар может стать следствием нарушения правил пожарной безопасности и правил эксплуатации электрических устройств.

Технический регламент о требованиях пожарной безопасности помещения по пожарной и взрывной опасности относит офисное помещение, в котором выполнялись работы по разработке решения, в категорию Г – умеренная пожароопасность.

Существует ряд мероприятий, позволяющие уменьшить вероятность возникновения пожаров. К первым относятся эксплуатационные мероприятия, в основе которых лежит выбор и использование современных автоматических средств сигнализации, автоматических стационарных систем и первичных средств пожаротушения, разработка методов и применение устройств ограничения распространения огня и т.д.

Ко вторым относятся организационные мероприятия, направленные на обучение сотрудников правилам пожарной безопасности, разработку и реализацию норм и правил пожарной безопасности, инструкций эксплуатации рабочего оборудования, планов эвакуации и прочих.

5.4 Правовые и организационные вопросы обеспечения безопасности

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

Основными нормативными документами, устанавливающие требования к рабочему месту являются ГОСТ 12.2.032-78 и ГОСТ Р 50923-96.

Организация рабочего места программиста предполагает соблюдение ряда требований, устанавливаемых ГОСТ 12.2.032-78: оптимальное размещение оборудования, входящего в состав рабочего места и достаточное рабочее пространство, позволяющее осуществлять необходимые движения и перемещения. Взаимное расположение всех элементов рабочего места должно соответствовать

физическим и психологическим требованиям, мониторы персональных компьютеров должны быть расположение по отношению к источникам естественного света сбоку, преимущественно слева.

Основными элементами рабочего места является стол и кресло, рациональный подбор этих элементов позволяет создать комфортную рабочую обстановку.

Рабочий стол должен соответствовать следующим требованиям:

- высота стола должна быть выбрана с учетом возможности сидеть свободно, в удобной позе, при необходимости опираясь на подлокотники;
- в нижний части стола должно быть предусмотрено пространство для ног, высотой не менее 60 см, шириной – не менее 50 см, глубиной на уровне колен – не менее 45 см и на уровень вытянутых ног – не менее 65 см;
- поверхность стола не должна создавать бликов;
- конструкция стола должна предусматривать наличие выдвижных ящиков, для хранения документаций и канцелярских принадлежностей.

Рекомендуемая высота сиденья рабочего кресла над уровнем пола должна находится в пределах 42-50 см. поверхность сиденья мягкая, ширина и глубина поверхности сиденья не менее 40 см, передний край закругленный, а угол наклона спинки – регулируемый.

Стоит также отметить, что для комфортной и качественной работы на компьютере существенное значение имеют размеры знаков, плотность их размещения, контраст и соотношение яркостей символов и фона экрана. Для рекомендуемого расстояние от глаз работника до монитора, лежащего в диапазоне 60-80 см, высота знака должна быть не менее 3 мм, оптимальное соотношение

ширины и высоты знака составляет 3:4, расстояние между знаками – 15-20% их высоты и соотношение яркости фона экрана и символов должно находится в пределах от 1:2 до 1:15.

Рабочее место, за которым выполнялись работы по разработке проектируемого решения удовлетворяют требованиям в выше перечисленных нормативных документах.

Выводы по разделу:

В результате работы произведен анализ вредных и опасных факторов, которые могут возникнуть при выполнении работ по разработке и эксплуатации проектируемого решения, а именно выявлены следующие вредные и опасные факторы: превышение допустимого уровня электромагнитного излучения и опасность поражения электрическом током. Рассмотрено влияние работ по разработке и эксплуатации текущего решения на экологическую безопасность, определена наиболее вероятная чрезвычайная ситуация и предложены мероприятия, позволяющие снизить вероятность ее появления. А также рассмотрены правовые и организационные вопросы обеспечения безопасности.

ЗАКЛЮЧЕНИЕ

В данной работе рассмотрен процесс разработки геоинформационной системы для отображения технологических объектов и информации о состоянии технологического процесса на географической карте.

В ходе работы, был выполнен обзор существующих *SCADA*-систем, объединяющие функции мониторинга и управления технологическим процессом с функциями геоинформационных систем. Выполнен обзор картографических сервисов с целью определения потенциальных преград, которые могут возникнуть при использовании материалов этих сервисов. Также проведен анализ вариантов реализации систем, и определен наиболее эффективный вариант реализации. Кроме этого проделан краткий обзор программных платформ и инструментов, предоставляющие компоненты, в которых реализована функциональность, относящая к ГИС.

На этапе проектирования были определены цели архитектуры, архитектурный стиль системы, типы приложений, основные варианты использования и развертывания, учитывающие все ограничения. Созданы структурные схемы компонентов системы, диаграммы *UML*, эскизы пользовательского интерфейса и абстрактные модели, позволяющие упростить понимание системы и перейти к этапу реализации. Также спроектирован *REST* интерфейс, обеспечивающий взаимодействие компонентов системы.

На этапе реализации созданы структура базы данных и компоненты доступа к данным. Реализован веб-сервер, предоставляющий функциональность централизованного хранения и доступа к данным для совместного использования клиентами, а также компонент *ActiveX*, обеспечивающий интеграцию с программой *Infinity HMI*, следствием которого является организация единого автоматизированного рабочего места.

Результатом диссертации является геоинформационное приложение, обеспечивающий отображении технологических объектов и информации о состоянии технологического процесса на географической карте.

CONCLUSION

In this work, the process of developing a geoinformation system for displaying technological objects and information on the state of the technological process on a geographic map is considered.

The review of existing *SCADA*-systems, combining the functions of monitoring and control of the technological process with the functions of geoinformation systems, was performed. An overview of mapping services was performed to determine the potential barriers that may arise when using the materials of these services. Also, an analysis of the implementation options for the systems was carried out, and the most effective implementation option was identified. In addition, a brief overview of software platforms and tools providing components in which *GIS* functionality is implemented is provided.

At the design stage, the objectives of the architecture, the architectural style of the system, the types of applications, the main uses and deployments that took into account all the constraints were defined. The structural diagrams of the system components, *UML* diagrams, user interface sketches and abstract models have been created, which make it easier to understand the system and go to the implementation phase. A REST interface has been designed to ensure the interaction of system components.

At the implementation stage, a database structure and the data access components have been created. A web server has been implemented that provides the functionality of centralized storage and data access for customer sharing, and an *ActiveX* component that integrates with the *Infinity HMI* program, which results in a single automated workstation.

The result of the work is a geoinformation application that provides mapping of technological objects and information about the state of the technological process on a geographical map.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Руководство Microsoft по проектированию архитектуры приложений / Корпорация Майкрософт – 529 с.

2 Patterns of Enterprise Application Architecture / M. Fowler – 559 .c

3 Паттерны проектирования на платформе .NET / С. Тепляков – 320 с.

4 Геоинформационные системы: учебное пособие / Р.В. Ковин, Н.Г. Марков. - Томск: Изд-во Томского политехнического университета 2008 – 175 с.

5 Model-driven Design of Geo-Information Services / M. Javier, G. Morales – 218 с.

6 Документация Microsoft [Электронный ресурс] // - Режим доступа: https://msdn.microsoft.com (дата обращения 2.05. 2018).

7 Документация ASP.NET Core 2 [Электронный ресурс] // - Режим доступа: https://docs.microsoft.com/en-gb/aspnet/core (дата обращения 16.03. 2018).

8 Спецификация протокола OPC UA [Электронный ресурс] // - Режим доступа: https://opcfoundation.org/developer-tools/specifications-unified-architecture (дата обращения 5.04. 2018).

9 Официальный сайт OPC Foundation [Электронный ресурс] // - Режим доступа: https://opcfoundation.org (дата обращения 25.04. 2018).

10 Официальный сайт GISize [Электронный ресурс] // - Режим доступа: http://www.gisize.com (дата обращения 5.05. 2018).

11 Официальный сайт PcVue Solutions [Электронный ресурс] // - Режим доступа: https://www.pcvuesolutions.com (дата обращения 4.05.2018).

12 Сайт НПК Астра [Электронный ресурс] // - Режим доступа: http://www.astraeng.ru (дата обращения 4.05.2018).

13 Документация Google Maps API [Электронный ресурс] // - Режим доступа: https://developers.google.com (дата обращения 25.12. 2017).

14 Документация API Яндекс.Карт [Электронный ресурс] // - Режим доступа: https://tech.yandex.ru (дата обращения 25.12. 2017).

15 Документация API Bing Maps [Электронный ресурс] // - Режим доступа: https://www.microsoft.com/en-us/maps/documentation (дата обращения 5.04. 2018).

16 Документация картографического сервиса Open Street Map на русском языке [Электронный ресурс] // - Режим доступа: http://wiki.openstreetmap.org/wiki/RU:API_v0.6 (дата обращения 2.03. 2018).

17 Документация системы управления базами данных PostgreSQL [Электронный ресурс] // - Режим доступа: https://www.postgresql.org/docs (дата обращения 15.03. 2018).

18 Официальный сайт компании Pitney Bowes Software Inc, США. [Электронный ресурс] // - Режим доступа: https://www.pitneybowes.com (дата обращения 9.03.2018).

19 Официальный сайт компании ESRI Inc, США. [Электронный ресурс] // - Режим доступа: https://www.esri.com (дата обращения 19.02.2018).

20 Дистрибутор продуктов Pitney Bowes Software Inc [Электронный ресурс] // - Режим доступа: http://www.mapinfo.ru (дата обращения 19.02.2018).

21 Официальный сайт QGIS [Электронный ресурс] // - Режим доступа: https://www.qgis.org (дата обращения 25.05.2018).

22 Официальный сайт GMap.Net [Электронный ресурс] // - Режим доступа: https://greatmaps.codeplex.com/ (дата обращения 25.05.2018).

23 Официальный сайт DevExpress [Электронный ресурс] // - Режим доступа: https://www.devexpress.com/ (дата обращения 19.03.2018).

24 Документация Automapper [Электронный ресурс] // - Режим доступа: http://docs.automapper.org/en/stable/index.html (дата обращения 2.05. 2018).

ПРИЛОЖЕНИЕ А

(Обязательное)

Часть магистерской диссертации, выполненный на иностранном языке

ANALITICAL REVIEW

Студент:

Группа	ФИО	Подпись	Дата
8ИМ6А	Айдаров Шукурдин Бахтиярович		

Руководитель ВКР:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор ОИТ	Ким Валерий Львович	д.т.н., профессор		

Консультант школы отделения (НОЦ) ОИТ:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Мирошниченко Евгений Александрович	к.т.н., доцент		

Консультант-лингвист отделения иностранных языков ШБИП:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИЯ	Комиссарова Ольга Валентиновна	к.ф.н., доцент		

ANALYTICAL REVIEW

1 Review of existing solutions

Information about existing analogs, and a qualitative analysis of their merits and shortcomings in many cases can become the basis for the success of the project. This stage allows you to determine the successful concepts of products of the competitors that are worth implementing in the developed system. And also avoid some unfavorable decisions that caused unsatisfactory user feedback about the product or even their failures.

Below is a brief description of the most popular *SCADA*-systems, in which the functions of data collection and control with functions of geographic information systems (GIS) are united.

1.1 GISize Wonderware

The company Wonderware, which develops and delivers software solutions for managing technological operations, provides a modern GISize tool for developing solutions in the field of automation. GISize combines a rich set of functionalities with virtual reality technologies, 3D-visualization, mobile and geoinformation technologies.

A lot of GISize functions for working with geodata allow creating specialized layers for technological objects and defining objects with spatial and attributive data on them. Attributive data can be information about the state of the technological process, integration with the technological process is carried out by means of the InTouch OMI dispatch control platform [8], which relates to the development of Wonderware.

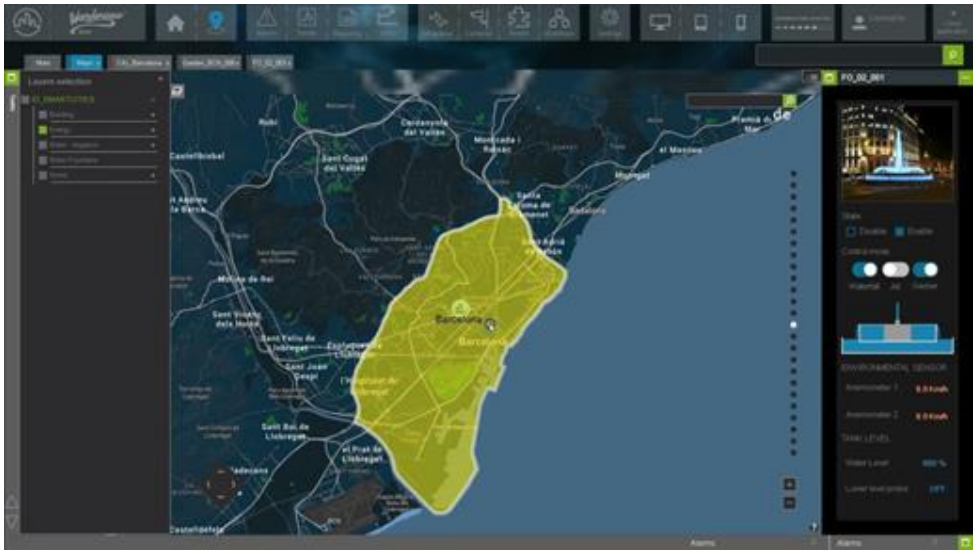


Figure 1.1 – Map in GISize

The functionality of GISize in the areas of virtual reality and 3D-visualization allows you to create virtual models of production equipment or premises. These environments can be used for a variety of purposes, such as simulating systems and extending the use of information in industrial contexts, interactively managing equipment using virtual reality devices, allowing you to display information related to the location of equipment or people, thereby improving their management (Figure 1.2).

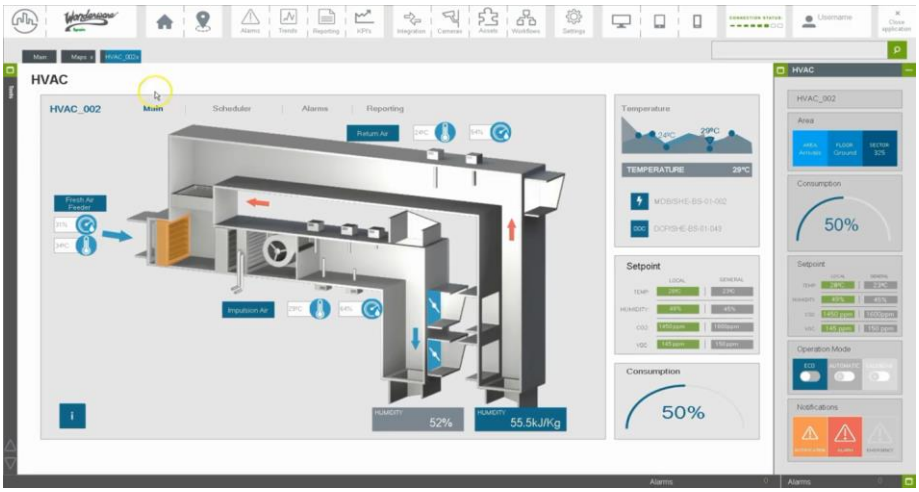


Figure 1.2 - Interactive equipment management

GISize provides a powerful integrated development environment (IDE) with a simple and user-friendly interface to create map layers and interactive projects, which is presented in Figure 1.3. The environment provides for the creation of various object templates, allowing to speed up the development process. With the help of this environment, you can create solutions of any scale for any industry.

It should also be noted that there are possibilities of GISize integration with various providers of geodata using standard WMS and REST services, and creating maps in vector format.

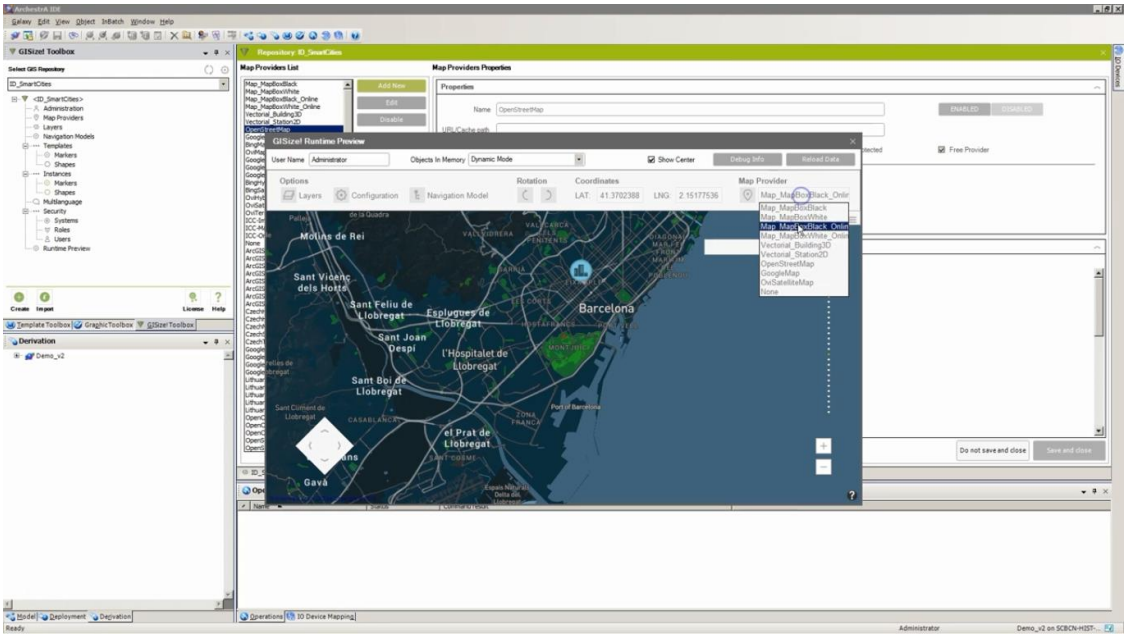


Figure 1.3 - GISize IDE

Simple and user-friendly interface, 3D-visualization and virtual reality, and a lot of GISize functionalities can certainly be attributed as merits of this product, but the functional power and quality of this product causes its high cost.

Another drawback is the dependence of the components of GISize with Microsoft products. The *InTouch* platform is designed for the Microsoft version of Windows 7 and later, is compatible with Microsoft's database since the release of SQL Server 2008 SP3.

1.2 PcVue geoinformation module

The software for data acquisition and dispatching control of technological processes *PcVue*, belongs to the development of the company *ARC Informatique* [9]. This software combines mapping technology with classical functions of *SCADA*-systems and includes a geoinformation module that allows displaying technological objects on a geographical map in the form of markers of various types. The GIS-oriented interface of the module is shown in Figure 1.4.

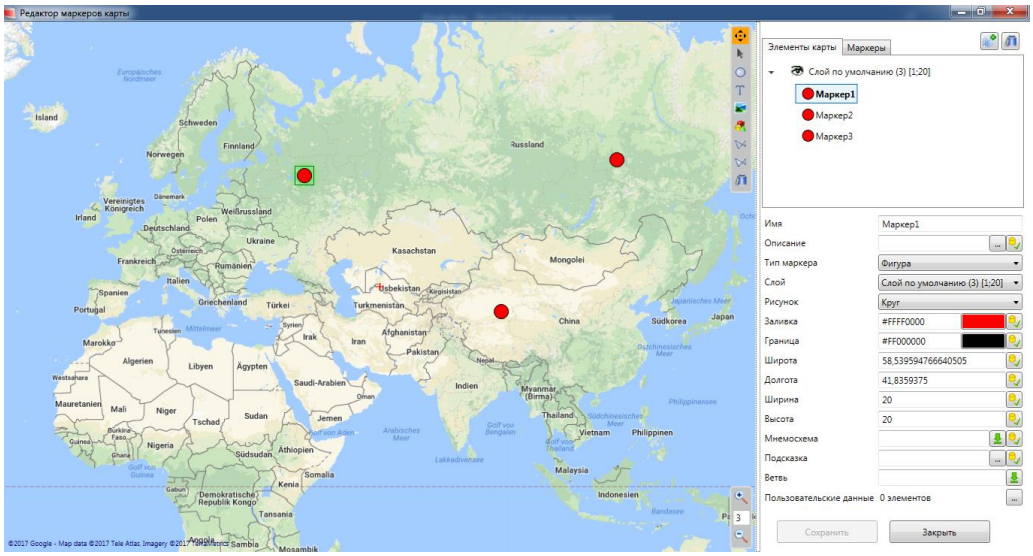


Figure 1.4 – PcVue GIS module

The following types of markers are defined in PcVue: text, symbol, figure, marker type "image", linear and polygon marker. The markers of the first four types are point geographic objects, the linear marker is a one-dimensional object described by two coordinates, and the polygon marker is a two-dimensional object that describes some area.

Each type of marker has a characteristic set of properties. The value of a specific property can be set in two ways. In the first way, the user determines the value of the property in the input field.

In the second way, as the source of the value, the user specifies the node in the data collection server's variable tree, and the system itself performs all operations associated with reading and setting the current value to the marker property. The collection server is a standard OPC server. The window for selecting a node from the variable tree is shown in Figure 1.5.

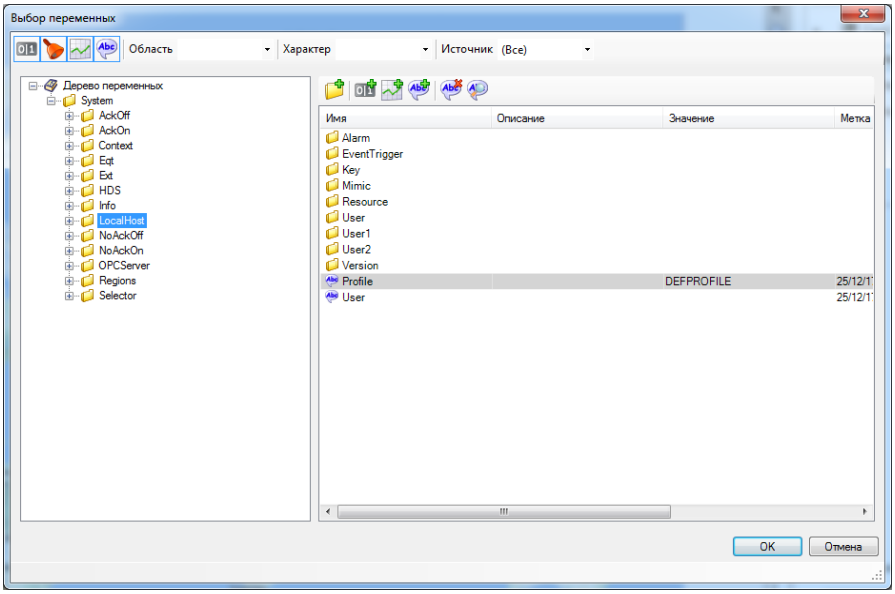


Figure 1.5 – Variable tree

One of the disadvantages is the inconvenience of the user interface for working with markers, the process of placing markers on the map requires maximum concentration. The reason for this was poorly designed cursors and marker controls. Another drawback is the lack of access to the geodata on the internal network, located behind the proxy server. But this disadvantage is compensated by the possibility of exporting geodata to a file that can be moved and used on a computer behind the firewall.

The merits of this product include a large number of supported cartographic services and ease of deployment. In the current version, PcVue 11 supports Google Maps, Bing Maps, OpenStreetMap and a number of other services.

1.3 Astra: GIS-SCADA

GIS-SCADA is software for building dispatching and automation systems for engineering networks, belongs to the development of the domestic research and production company Astra [10]. This software is a configured and optimized tool for heat supply, service and management companies, water utilities and municipal governments. The *GIS-SCADA* includes interactive maps with administrative and engineering layers, mnemonic schemas with the possibility of control, calculation and analytical modules, also an intuitive and simple editor of projects.

Geoinformation module *GIS-SCADA* contains standard layers of the administrative map in raster and vector formats with details for different scales, also specialized layers for the heat and water supply industries.

GIS-SCADA contains a predefined set of specialized templates to represent technological objects on a map, that are oriented to one of the above-listed industries. Integration with the technological process can be carried out using components that are part of GIS-SCADA or by means of standard OPC servers.

Astra offers ready-made products based on GIS-SCADA for solving problems of heat energy, gas, electricity and water, also other products.

Among the disadvantages of *GIS-SCADA* should be highlighted the complexity of adapting this product to solve problems in other industries, the complexity of deployment, also the lack of cross-platform. The components of *GIS-SCADA* are oriented on the Microsoft platform.

The main advantage of this product is low cost relative to other similar products.

2 Ways of solving the problem and their analysis

There are various ways to solve the problems of mapping technological information on a geographic map. One of the ways is the adaptation of a full-featured geoinformation system for solving the tasks posed, another way is the creation of information systems capable of extracting technological information and presenting them on a map.

2.1 The use of existing GIS

One of the ways is the adaptation of a universal, full-featured GIS to solve the problems of displaying technological objects and information about the state of the technological process on a geographic map.

During the adaptation process, it will be necessary to develop appropriate modules for the selected GIS and to find ways to integrate with the *Infinity HMI* program. Integration with Infinity HMI is one of the requirements for the system being developed, a complete list of technical requirements is given in Appendix 1.

Furthermore, modern universal GIS have a huge functional capacity, which is not required in the context of the problem.

It is also worth noting that such GIS require additional costs for their specific support, including installation, configuration, maintenance, etc. In view of the foregoing, this version of the system was rejected.

2.2 Developing a desktop application

Another way for creating the system is the development of a full-featured desktop application using software GIS components that provide tools for working with cartographic data. This way allows you to create a system with the necessary functionality and a specialized user interface for solving the problems of the current system. Also in

this approach, it is possible to implement integration with Infinity HMI through *ActiveX* technology.

Integration with *Infinity HMI* using *ActiveX* means creating a desktop application as an *ActiveX* control. The runtime for this control is the *ActiveX* container implemented in *Infinity HMI*. Since this control will implement all the functionality of the system, then it has a large amount of memory and processor resources. However, the presence of such components in *Infinity HMI* reduces its performance. This will create enormous inconvenience to the user and may break the operation of other components in the *Infinity HMI*. Therefore, this way of creating the system was also rejected.

2.3 Development of a desktop client, an ActiveX control, and a Web server

Given the integration requirements in Appendix 1 and restrictions of *Infinity HMI*, a third way for creating the system appeared, which was chosen. This way assumes splitting the system into three components. The first component is a lightweight ActiveX control, which focuses on the functionality associated with displaying information on a geographic map.

The second component is a full-featured desktop application, in which the functionality editor of markers and other additional functions are concentrated. This application is called the project development environment, because it creates markers and determines the information that will be presented on the map.

The third component is a web server, it is the main part of the system, which provides centralized storage and access to data to other components of the system.

The advantages of this way are as follows:

- the lightweight *ActiveX* control does not slow down or disrupt the operation of the *Infinity HMI*;

- the functionality necessary only for creating markers is concentrated in a separate application that does not need to be started up and does not need to spend computer resources;
- the presence of the web server allows you to flexible add new customers to the system, due to the standardized interface.

3 Review of cartographic data sources

This section provides a brief review of popular mapping services to determine the main source of geodata, the way to integrate with them, and possible restrictions.

3.1 Information retrieval service Google Maps

Information retrieval web service *Google Maps* provides access to a huge amount of cartographic material in vector, raster or hybrid modes [11].

Google offers a lot of tools and programming interfaces for the use of data and services of the *Google Maps*, oriented different programming languages and platforms. The most popular are the *JavaScript API*, *Static Maps API*, *Android API* and *SDK* for *IOS*, which allow you to integrate maps into applications for various platforms.

It should be noted that the Static Maps API is a more versatile interface, as it allows interaction over the standard *HTTP*(s) protocol and is not oriented to a specific platform [11].

The materials and services of *Google Maps* services are provided free of charge for non-commercial use, but with a number of restrictions on use. For corporate deployment, there is a specific licensing policy.

3.2 Yandex.Maps search and information map service

One of the geoinformation services of the Russian development is *Yandex.Maps* search-information cartographic service, related to the development of *Yandex*. This service provides geodata and allows you to search the map, get information about traffic jams, plot routes and panorama streets of major cities, etc. [12].

Like Google, Yandex provides a programming interface and tools for integration with the service. The programming interface forms a set of services that allow the use of Yandex cartographic materials and technologies in various applications. This interface consists of the JavaScript API and the HTTP API, the Organizational Search API and the Static API [12].

JavaScript API is a set of software components on the JavaScript language, allowing you to place interactive maps on the site pages. These components allow you to embed into the application a map with a search by the name of the geographical object and organizations, and use other functions available on Yandex.Maps.

The HTTP API provides access to individual features of the Yandex.Maps service, such as a geocoder, using the HTTP (s) protocol.

Static API allows to get a raster image of the desired fragment of the map, using *HTTP* (s) requests. By adding various parameters to the query and setting their values, can define the center of the map, its size and display area, mark the desired objects and display the traffic jams.

Yandex provides free access to cartographic materials for non-commercial purposes, but with some limitations.

In the free version of the *Yandex.Maps API*, one can receive images of maps with dimensions not exceeding 600x450 pixels. Such an image must be placed on a public site or in an application. Also, the free API cannot be used for transport monitoring and in closed systems. For commercial use, there is a paid version of the API with different tariffs [12].

3.3 Cartographic service Bing Maps

Bing Maps is a cartographic service company Microsoft [13]. Integration with Bing Maps can be implemented by means of REST services or by using software components.

REST services are divided into two groups. The first group includes *Spatial Data Services* with which can geocode, store and query geodata using standard *HTTP* (s) requests.

The second group of services is REST Services, providing such operations as the creation of static maps with markers, geocoded addresses and routes.

In addition to services, *Microsoft* offers a set of ready-made software components that allow to create desktop, mobile and web applications. *Bing Maps WPF Control* controls are provided for the development of desktop applications. They provide interaction with the Bing Maps service, visualization and management of cartographic data, adding markers, animations and various figures to the map, as well as a number of additional functions.

High-performance, convenient and multifunctional Version 8 Control allows you to create web applications. This set allows to perform clustering of markers, build heat zone maps, create animations, and also supports *GeoJSON* and *GeoXML* formats [13].

The use of cartographic data Bing Maps involves the conclusion of one of the licensing agreements Microsoft [13].

3.4 Cartographic service OpenStreetMap

OpenStreetMap is a popular non-commercial cartographic web service. OpenStreetMap map data are based on data from GPS devices, aerial photographs, video recordings, satellite imagery and street panoramas provided by some companies. At the

heart of data collection is the "Wikipedia" principle, which allows each registered user to make changes to cartographic data.

The advantage of the service is free access to geo-data, which became the main reason for its popularity. The cartographic materials of the service are distributed on the terms of the free Open Database License [14] license, which grants the right to freely distribute, modify and use these data in applications, including commercial too.

Like *Google Maps*, *Bing Maps* and *Yandex.Maps*, *OpenStreetMap* provides a *REST API* that allows other systems to integrate with it using standard *HTTP(s)* requests.

Among the drawbacks of OpenStreetMap the presence of inaccuracies should be highlighted, because of the principle of "Wikipedia" in cartographic data.

Conclusions on the section:

The review of existing solutions has made it possible to determine the ways of presenting technological objects and information about the state of the technological process in a *GIS*-oriented interface. As the main method, a way of displaying this information with the help of markers was chosen, since implementation of this method requires less resources.

When using cartographic materials for *Google Maps services*, *Yandex.Maps* and *Bing Maps* for commercial purposes, it is necessary to obtain a special license that allows to use the data of these services. As the main source of cartographic data, in view of its availability, *OpenStreetMap* is chosen.

ПРИЛОЖЕНИЕ Б

(Обязательное)
Технические требования

Термины и определения

Термин/Сокращение	Определение
АСУТП	Автоматизированная система управления технологическими процессами.
АРМ	Автоматизированное рабочее место.
Маркер	Представляет собой элемент пользовательского интерфейса, который связан с определенным местом на географической карте.
Группа маркеров	Представляет собой совокупность маркеров, для которых определен способ отображения их на карте. Группа определяет диапазон масштаба в пределах которого маркеры отображаются на карте.
Символ	Пользовательский элемент управления.
Infinity Server	Сервер обработки технологических данных.
Infinity HMI	Программа разработки и просмотра графических мнемосхем.
OPC UA	Промышленный протокол обмена данными.
Атрибуты маркера	Примитивные характеристики маркера. В отличие от свойств они жестко заданы для каждого типа маркера.
Свойства маркера	Дополнительные характеристики маркера, определяемые пользователем. В отличие от атрибутов они задаются пользователем.
OpenStreetMaps	Картографический веб-сервис.
VBA	Язык программирования.

1 Назначение и цели системы

1.1 Назначение

Система предназначена для отображения технологических объектов, информации о параметрах и ходе технологического процесса на географической карте.

1.2 Цели создания системы

- расширение функций программного комплекса *SCADA Infinity*;
- обеспечение конкурентного преимущества на отечественном рынке

Таблица 1 – Классы пользователей

Класс пользователей	Описание
Администратор	Должен иметь возможность гибкой настройки и разграничения прав доступа для пользователей системы.
Инженер	Выполняет работы по разработке, настройке проектов.
Оператор	Выполняет мониторинг и контроль за технологическим процессом.

Таблица 2 – Типы маркеров

Типы маркеров	Описание
Текст	Этот тип маркера предоставляет возможность отображения текстовой информации на карте.
Изображение	Этот тип маркера позволяет отобразить картинку на карте.
Фигура	Этот тип маркера позволяет отобразить геометрическую фигуру на карте. Используются следующие типы геометрических фигур: круг, квадрат, прямоугольник, треугольник, трапеция и ромб.

2 Требования к системе

Таблица 2.1 – Группы требований

Символ	Группа требований
F	Функциональные требования
I	Требования к интерфейсу пользователя
RD	Требования к надежности
PR	Требования к производительности
LR	Требования к лицензированию системы
LI	Требования к лингвистическому обеспечению
TS	Требования к техническому обеспечению
SR	Требования к программному обеспечению

WR	Требования к документированию

[F] Функциональные требования

Код требования	Требования	Примечания
F.01	**Общие требования**	
F.01.01	Система должна обеспечить отображение географической карты из веб-сервиса Open Street Map.	
F.01.02	Система должна предоставить возможность управления масштабом карты.	
F.01.03	Система должна обеспечить возможность перемещения по карте.	
F.01.04	Система должна обеспечить возможность отображения объектов АСУТП на карте.	
F.01.05	Отображение объекта АСУТП на карте должно быть задано с помощью маркера.	Список маркеров приведен в таблице 2.
F.01.06	Система должна обеспечить возможность отображения информации о состоянии объекта АСУТП.	
F.01.07	Система должна иметь среду разработки проектов.	
F.01.08	Система должна иметь библиотеки символов.	
F.01.09	Система должна иметь библиотеки изображений.	
F.01.10	Система должна обеспечить возможность обмена данными с сервером обработки данных Infinity Server.	По протоколу OPC UA.
F.01.10.01	Система должна обеспечить возможность обмена данными с 10 серверами обработки данных в одном проекте.	
F.01.11	Система должна иметь компонент ActiveX.	
F.01.12	Система должна обеспечить возможность авторизации пользователя.	
F.02	**Требования к Среде разработки проектов**	
F.02.01	Среда разработки должна обеспечить возможности добавления, редактирования и удаления группы маркеров.	
F.02.02	Среда разработки должна предоставить возможности добавления, редактирования и удаления маркеров.	
F.02.03	Среда разработки должна обеспечить возможность добавления, редактирования и удаления свойств маркеров.	
F.02.04	Среда разработки должна обеспечить возможность указания узлов сервера OPC UA в качестве источника значений свойств маркера.	

Код требования	Требования	Примечания
F.02.05	Среда разработки должна обеспечить возможность отмены выполненных операций.	Функции Undo и Redo.
F.02.06	Среда разработки должна обеспечить возможность определения ступеней масштаба для группы маркеров, в пределах которого эта группа видима пользователю.	
F.02.07	Среда разработки должна обеспечить возможность создания ограничения области просмотра карты в проекте.	
F.02.08	Среда разработки должна обеспечить возможность добавления, переименования, перемещения и удаления символов в/из библиотек символов.	
F.03.09	Среда разработки должна обеспечить импорт символов из файла проекта Infinity HMI.	
F.03.10	Среда разработки должна обеспечить возможность добавления, переименование, перемещения и удаления изображений в/из библиотек изображений.	
F.03.11.01	Среда разработки должна обеспечить возможность добавления изображений размером не более 100 Кб.	
F.03.12	Среда разработки должна обеспечить возможность экспорта фрагмента карты из источников, приведенных в таблице.	
F.03	**Требования к компоненту ActiveX**	
F.03.01	Компонент должен предоставить пользователю возможность выбора проекта.	
F.03.02	Компонент должен отобразить маркеры из выбранного проекта на географической карте.	
F.03.03	Компонент должен обеспечить отображение свойств маркера.	
F.03.04	Компонент должен обновлять значения свойств маркера с периодом в 1с.	
F.03.05	Компонент должен обеспечить отображение маркеров, меняющих свое географическое положение.	
F.03.06	Компонент должен предоставить программный интерфейс для взаимодействия с Infinity HMI.	
F.03.06.01	Компонент должен обеспечить возможность переключения между мнемосхемами.	
F.03.06.02	Компонент должен обеспечить возможность вызова скриптов VBA, определенных пользователем.	

Код требования	Требования	Примечания
F.03.06.03	Компонент должен обеспечить возможность вызова скриптов VBA с переменным числом аргументов.	
F.03.07	Компонент должен обеспечить возможность фильтрации маркеров на карте.	
F.02.10	Компонент должен иметь журнал со статической информацией о ее работе.	

[IS] Требования к информационной безопасности

Код требования	**Требования**	**Примечания**
IS.01	Доступ к Системе должен поддерживаться на основе установленных ролей пользователей.	Смотреть таблицу 1.

[I] Требования к интерфейсу пользователя

Код требования	Требования	Примечания
I.01	Элементы управления пользовательского интерфейса должны быть выполнены в едином графическом дизайне.	
I.02	Кнопки пользовательского интерфейса должны быть подписаны текстом или помечены соответствующим графическим значком.	

[PR] Требования к производительности

Код требования	Требования	Примечания
PR.01	Задержка отображения информации о состояние объекта АСУТП не должна превышать 5 с**.**	

[LR] Требования к лицензированию системы

Код требования	Требования	Примечания
LR.01	Система должна обеспечить возможность лицензирования средствами аппаратной защиты Sentinel HASP.	
LR.02	Система должна предоставить часовой демонстрационный период работы.	

[LI] Требования к лингвистическому обеспечению

Код требования	Требования	Примечания
LI.01	Все прикладное программное обеспечение подсистемы для организации взаимодействия с пользователем должно использовать русский язык.	

[TS] Требования к техническому обеспечению

Код требования	Требования	Примечания
TS.01	**Требования к персональному компьютеру клиентов**	
TS.01.01	2-х ядерный процессор 2 ГГц.	
TS.01.02	4 Гбайт оперативной памяти.	
TS.01.03	1 Гбайт дискового пространства.	
TS.01.04	Стандартная сетевая карта.	
TS.01.05	Стандартный VGA монитор.	
TS.01.06	Стандартная клавиатура.	
TS.01.07	Манипулятор «мышь».	

[SR] Требования к программному обеспечению

Код требования	Требования	Примечания
SR.01	**Требования к среде разработки**	
SR.01.01	Подсистема должна функционировать в операционных средах – Microsoft Windows (Windows 7 или более поздние версии), Unix (Linux).	
SR.02	**Требования к компоненту ActiveX**	
SR.02.01	Компонент должен функционировать в среде Infinity HMI.	
SR.03	**Требование к СУБД**	
SR.03.01	Поддержка мультиплатформенности.	
SR.03.02	Поддержка многопроцессорности.	

[RD] Требования к надежности

Код требования	Требования	Примечания
RD.01	Система должна обеспечить надежную работу 24 часа в сутки 365/366 дней в году.	
RD.02	Суммарное время выхода системе из строя не должно превышать 100 часов в год.	
RD.03	Время восстановления системы не должно превышать более четырех часов.	

[WD] Требования к документированию

Код требования	Требования	Примечания
WD.01	Документация должна быть оформлена в текстовом формате на русском языке.	
WD.02	В составе документации должно быть руководство пользователя.	

ПРИЛОЖЕНИЕ В

(Обязательное)
Основные варианты использования

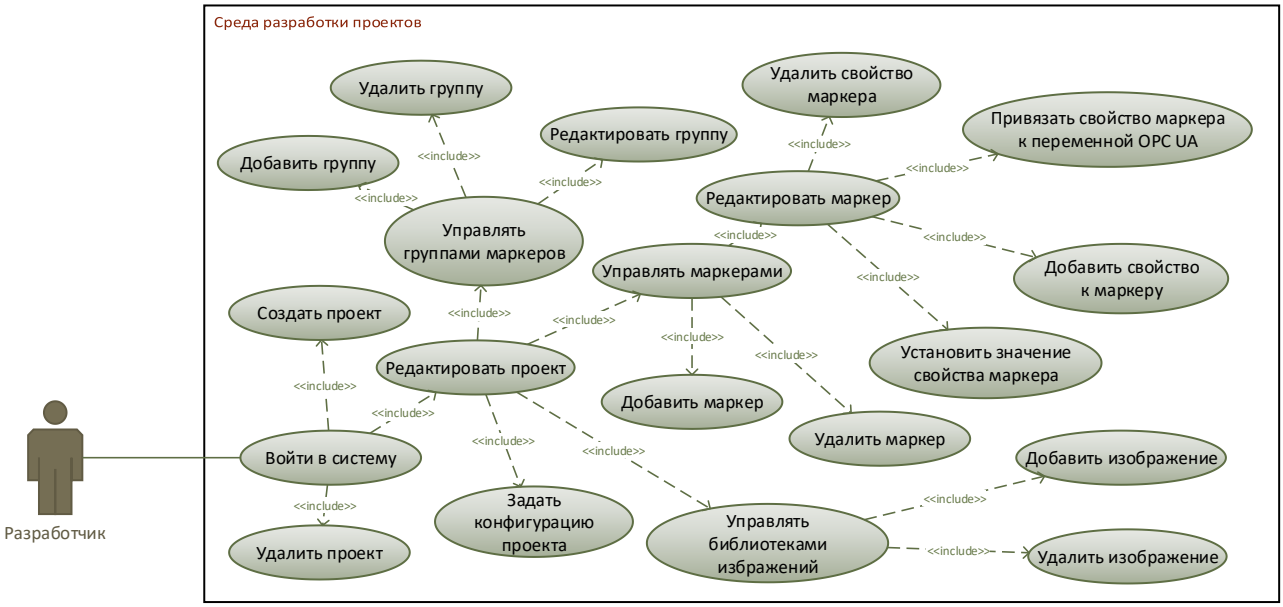


Рисунок В.1 – Варианты использования среды разработки проектов

Каткое описание вариантов использования среды разработки проектов приведено ниже.

ВИ «Войти в среду разработки проектов»

Цель: Перейти к рабочему окну программы.

Начальное состояние: Пользователь запустил приложение, перед ним находится форма входа.

Основной сценарий:

Пользователь вводит логин и пароль и отдает команду «ОК».

Если логин и пароль введены верно, а за пользователем закреплена роль «разработчик», то отображается окно выбора проектов.

ВИ «Создать проект»

Цель: Создать новый проект маркеров.

Начальное состояние: Пользователь авторизован и перед ним открыто окно выбора проектов.

Основной сценарий:

Пользователь отдает команду «Создать».

Программа отображает окно создания нового проекта.

Пользователь заполняет все поля и отдает команду «ОК».

Если все поля заполнены верно, то программа создает новый проект и отображает окно редактора проектов.

ВИ «Редактировать проект»

Цель: Открыть проект для редактирования.

Начальное состояние: Пользователь авторизован и перед ним открыто окно выбора проектов.

Основной сценарий:

Пользователь выбирает проект и отдает команду «Редактировать».

Программа отображает окно редактора проектов.

ВИ «Добавить группу»

Цель: Добавить новую группу маркеров.

Начальное состояние: Пользователь авторизован и перед ним открыто окно редактора проектов.

Основной сценарий:

Пользователь отдает команду «Добавить новую группу».

Программа отображает окно с полями для данных группы.

Пользователь заполняет необходимые поля и отдает команду «ОК».

Программа сохраняет введенные значения в базу данных и отображает окно редактора проектов.

ВИ «Удалить группу»

Цель: Удалить группу маркеров.

Начальное состояние: Пользователь авторизован и перед ним открыто окно редактора проектов.

Основной сценарий:

Пользователь в окне управления группами выбирает группу и отдает команду «Удалить группу».

Программа удаляет запись о группе из базы данных и отображает окно редактора проектов.

ВИ «Редактировать группу»

Цель: Редактировать группу маркеров.

Начальное состояние: Пользователь авторизован и перед ним открыто окно редактора проектов.

Основной сценарий:

Пользователь в окне управления группами выбирает группу и отдает команду «Редактировать группу».

Программа отображает окно с полями для данных группы.

Пользователь редактирует необходимые данные и отдает команду «ОК».

Программа сохраняет введенные значения в базу данных и отображает окно редактора проектов.

ВИ «Добавить маркер»

Цель: Добавить новый маркер.

Начальное состояние: Пользователь авторизован и перед ним открыто окно редактора проектов.

Основной сценарий:

Пользователь отдает команду «Добавить новый маркер».

Программа отображает окно с полями для ввода информации о технологическом объекте.

Пользователь заполняет необходимые поля и отдает команду «ОК».

Программа сохраняет введенные значения в базу данных и отображает окно редактора проектов. На карте появляется новый маркер.

ВИ «Удалить маркер»

Цель: Удалить информацию о технологическом объекте.

Начальное состояние: Пользователь авторизован и перед ним открыто окно редактора проектов.

Основной сценарий:

Пользователь выбирает в панели управления маркерами или на карте маркер и отдает команду «Удалить маркер».

Программа удаляет запись из базы данных и отображает окно редактора проектов.

ВИ «Редактировать маркер»

Цель: Редактировать информацию о технологическом объекте.

Начальное состояние: Пользователь авторизован и перед ним открыто окно редактора проектов.

Основной сценарий:

Пользователь выбирает в панели управления маркерами или на карте маркер и отдает команду «Редактировать маркер».

Программа отображает окно с полями для ввода информации о технологическом объекте.

Пользователь редактирует необходимые данные и отдает команду «ОК».

Программа сохраняет введенные значения в базу данных и отображает окно редактора проектов.

ВИ «Добавить свойство к маркеру»

Цель: Добавить информацию о свойстве технологического объекта к маркеру.

Начальное состояние: Пользователь авторизован и перед ним открыто окно редактора проектов.

Основной сценарий:

Пользователь выбирает в панели управления маркерами или на карте маркер и отдает команду «Редактировать маркер».

Программа отображает окно с полями для ввода информации о технологическом объекте.

Пользователь отдает команду «Дополнительные свойства маркера».

Программа отображает окно с дополнительными свойствами маркера.

Пользователь отдает команду «Добавить свойство». В списке свойств появляется новое свойство.

Пользователь задает имя свойству и отдает команду «ОК».

Программа сохраняет введенное значение в базу данных и отображает окно редактора проектов.

ВИ «Установить значение свойства маркера»

Цель: Установить источник статическое значение свойства маркера.

Начальное состояние: Пользователь авторизован и перед ним открыто окно редактора свойств маркера.

Основной сценарий:

Пользователь выбирает свойство из списка, задает значение и отдает команду «ОК».

Программа сохраняет введенное значение в базу данных и отображает окно редактора проектов.

ВИ «Привязать свойство маркера к переменной OPC UA»

Цель: Установить источник динамического значение свойства маркера.

Начальное состояние: Пользователь авторизован и перед ним открыто окно редактора свойств маркера.

Основной сценарий:

Пользователь выбирает свойство из списка и отдает команду «Привязать к переменной».

Программа отображает окно с деревом переменных сервера OPC UA.

Пользователь выбирает переменную и отдает команду «ОК» в окне редактора свойств маркера.

Программа сохраняет введенное значение в базу данных и отображает окно редактора проектов.

ВИ «Удалить свойство маркера»

Цель: Удалить информацию о свойстве технологического объекта.

Начальное состояние: Пользователь авторизован и перед ним открыто окно редактора свойств маркера.

Основной сценарий:

Пользователь выбирает свойство из списка и отдает команду «Удалить». Свойство удаляется из списка.

Пользователь отдает команду «ОК».

Программа удаляет записи из базы данных и отображает окно редактора проектов.

ВИ «Добавить изображение»

Цель: Загрузить изображение для общего использования.

Начальное состояние: Пользователь авторизован и перед ним открыто окно редактора проектов.

Основной сценарий:

Пользователь отдает команду «Открыть библиотеки».

Программа отображает окно с библиотеками изображений.

Пользователь выбирает библиотеку и отдает команду «Добавить изображение».

Программа отображает окно с поля для ввода информации об изображении и ее физическом расположении, и отдает команду «ОК».

Программа создает запись в базе данных и сохраняет изображение на файловом сервере.

ВИ «Удалить изображение»

Цель: Изъять изображение из общего доступа.

Начальное состояние: Пользователь авторизован и перед ним открыто окно библиотек изображений.

Основной сценарий:

Пользователь выбирает библиотеку и отдает команду «Удалить изображение».

Программа удаляет запись из базы данных и изображение из файлового сервера.

На рисунке В.2 представлена диаграмма вариантов использования компонента *ActiveX*. Описание вариантов использования приведены ниже.

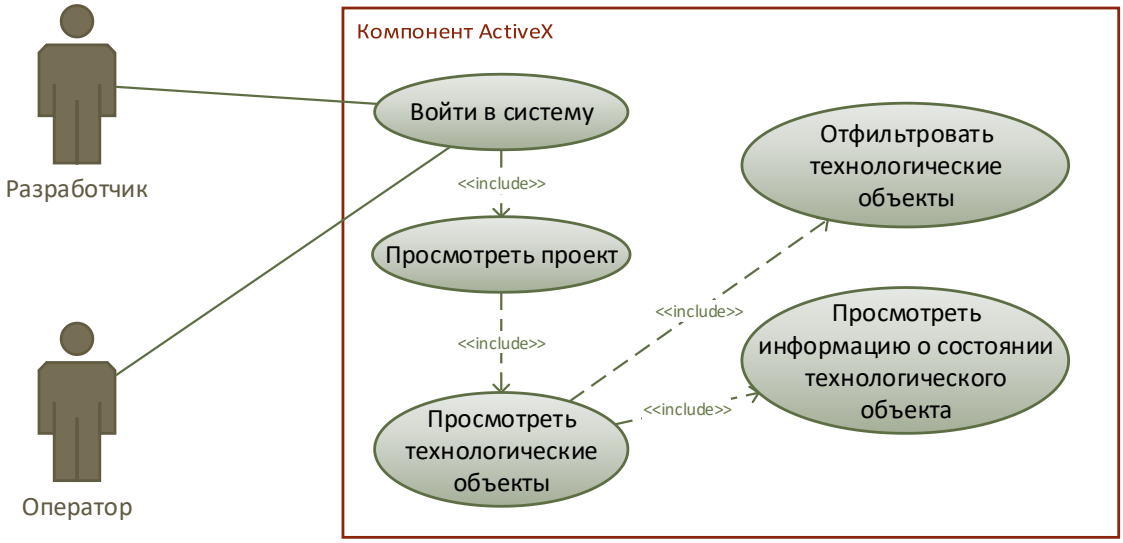


Рисунок В.2 – Варианты использования компонента ActiveX

ВИ «Вход в систему» обеспечивает авторизацию пользователя, аналогичен ВИ «Вход в среду разработки проектов».

ВИ «Просмотреть проект»

Цель: Просмотреть информации о технологических объекта и состоянии технологического процесса.

Начальное состояние: Пользователь прошел авторизацию и перед ним открыто окно выбора проекта для просмотра.

Основной сценарий:

Пользователь выбирает проект из списка и отдает команду «ОК».

Программа открывает основное окно, на котором размещены карта и маркерами, и периодически получает информацию о состоянии технологического процесса.

ВИ «Отфильтровать технологические объекты»

Цель: Отобразить на карте определенные объекты.

Начальное состояние: Пользователь прошел авторизацию и перед ним открыто основное окно.

Основной сценарий:

Пользователь открывает панель фильтров, задает фильтр и отдает команду «ОК».

Программа отображает на карте только те объекты, которые удовлетворяют фильтру.

ВИ «Просмотреть информацию о состоянии технологического объекта»

Цель: Узнать текущее состояние технологического объекта.

Начальное состояние: Пользователь прошел авторизацию и перед ним открыто основное окно.

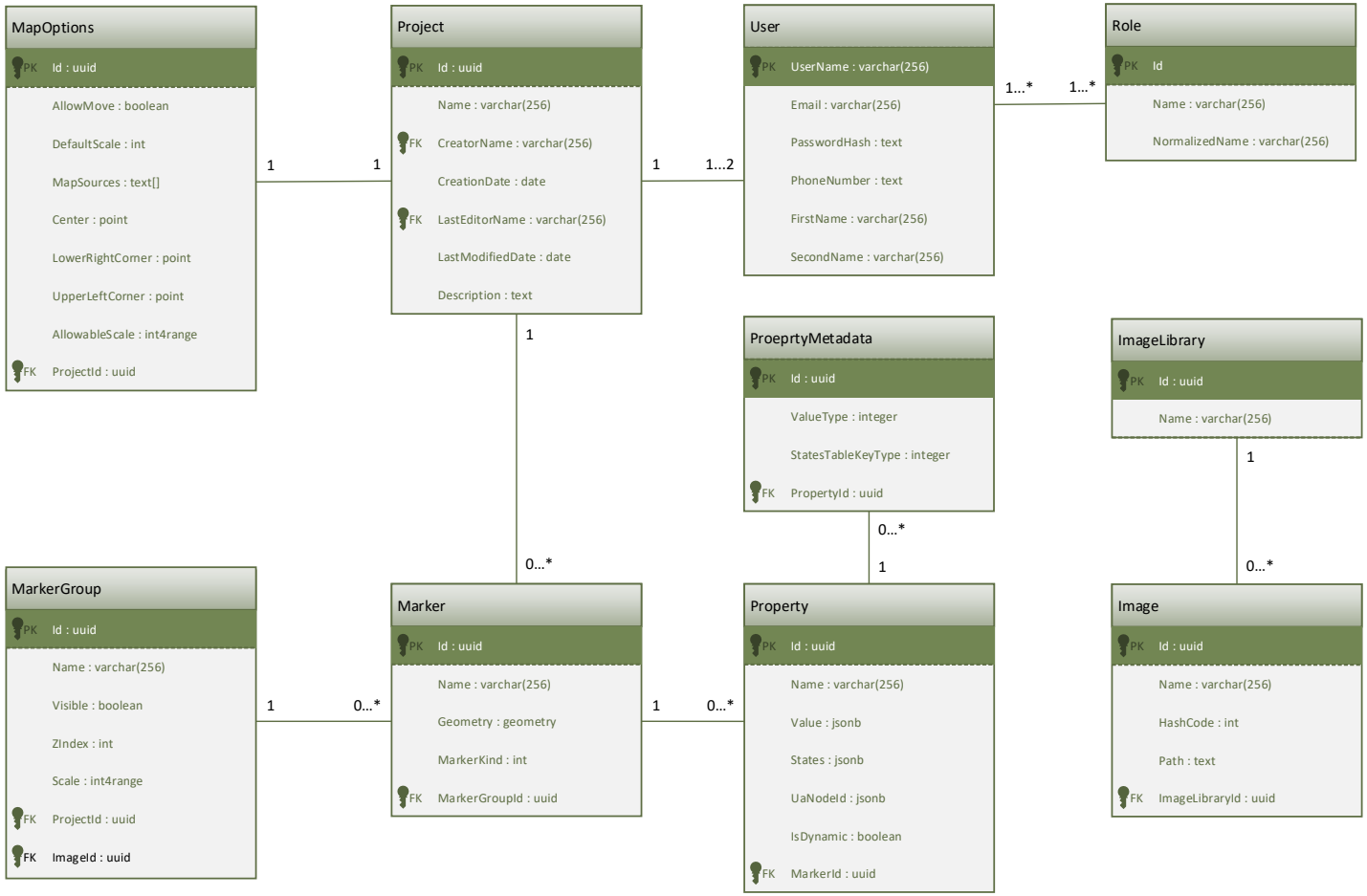
Основной сценарий:

Пользователь на карте левой кнопкой мыши нажимает на маркер, представляющий технологический объект.

Программа отображает всплывающее окно с информации о состоянии технологического объекта.

ПРИЛОЖЕНИЕ Г

(Обязательное)
Схема базы данных



ПРИЛОЖЕНИЕ Д

(Обязательное)
Диаграмма классов слоя доступа к данным

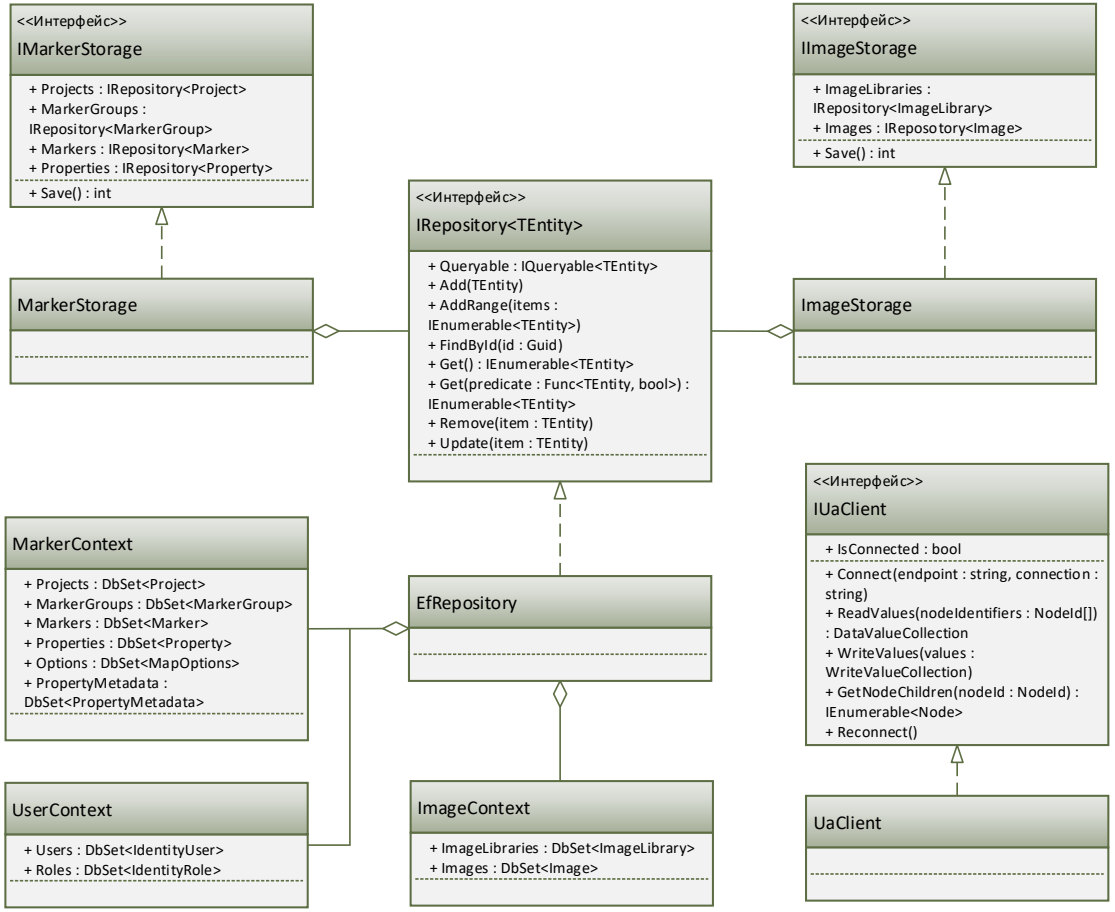


Рисунок Д.1 – Диаграмма классов-агентов доступа к данным

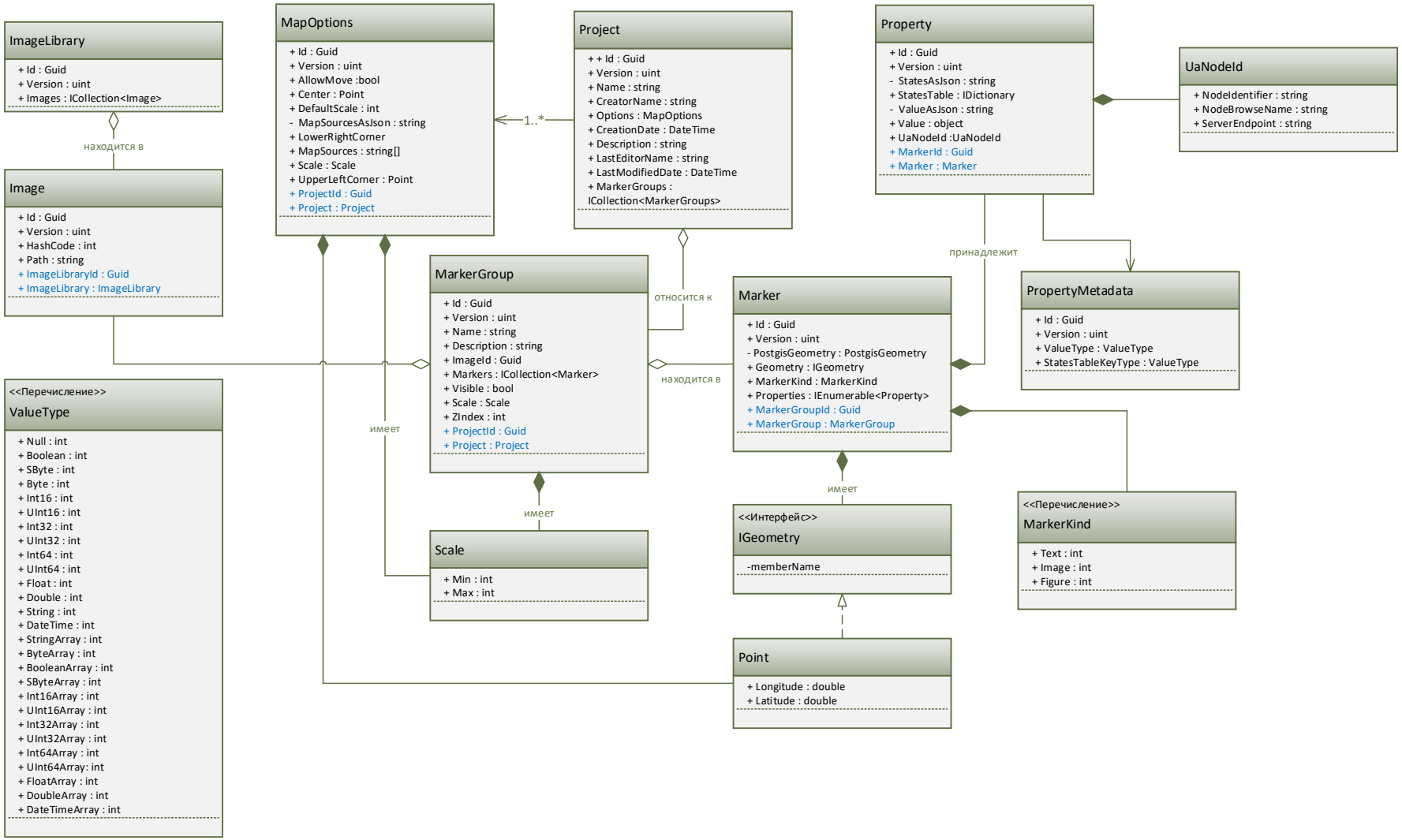


Рисунок Д.2 – Диаграмма классов-сущностей

ПРИЛОЖЕНИЕ Е

(Обязательное)

Диаграмма классов слоя предметной области

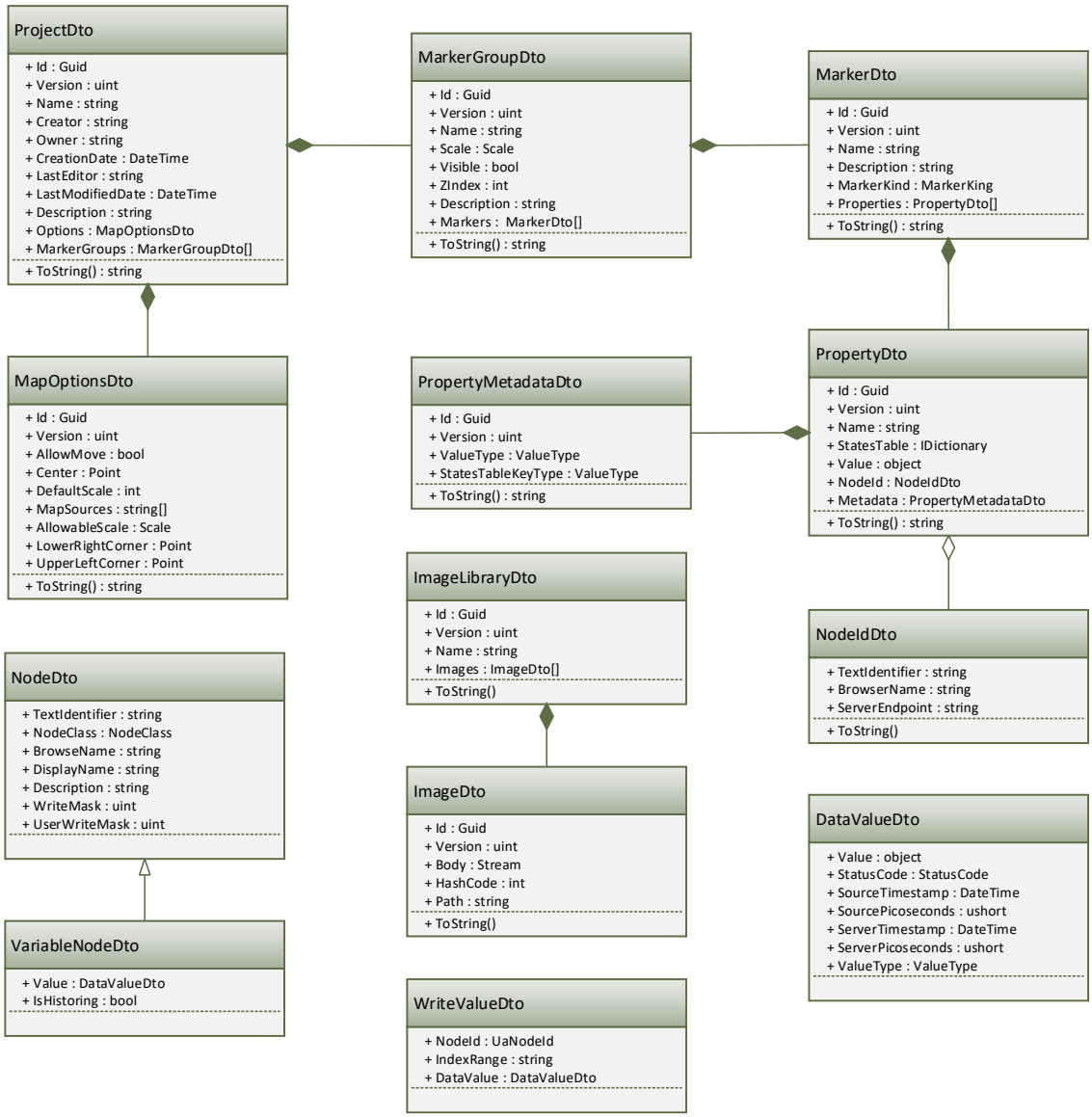


Рисунок Е.1 – Диаграмма классов *DTO*

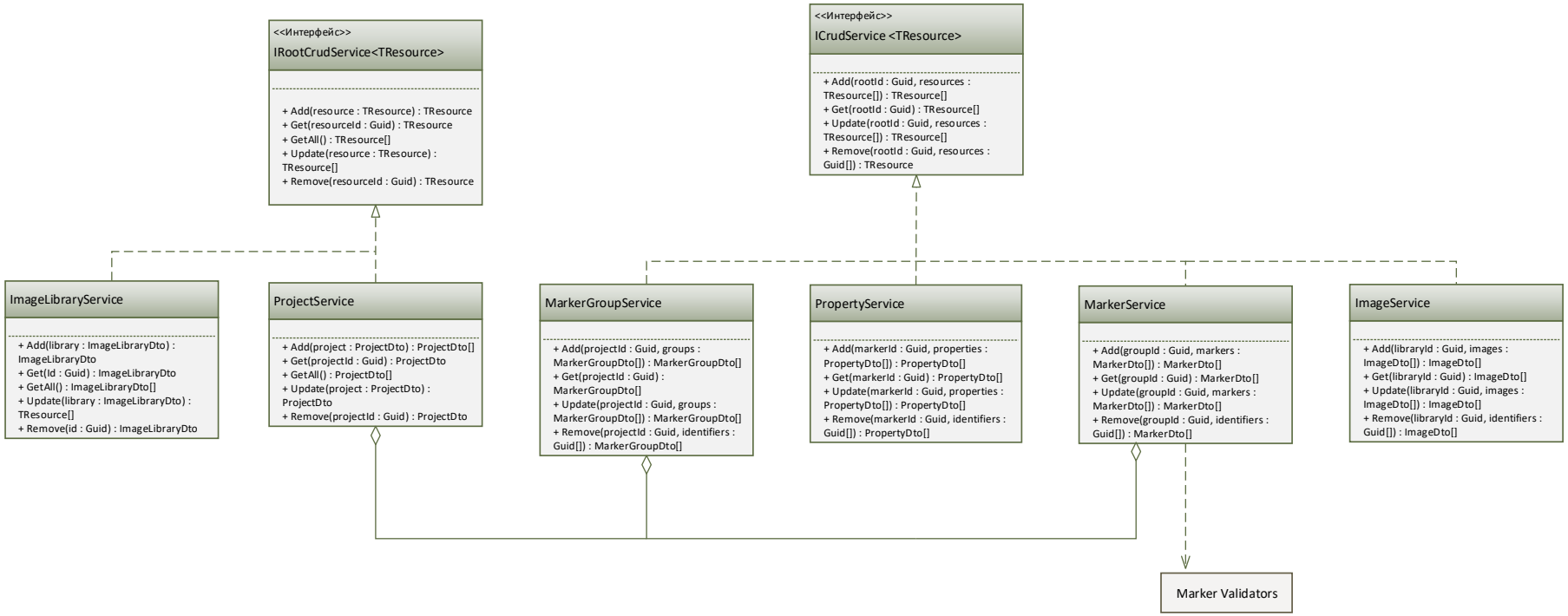


Рисунок Е.2 – Диаграмма классов, обеспечивающие управление данными

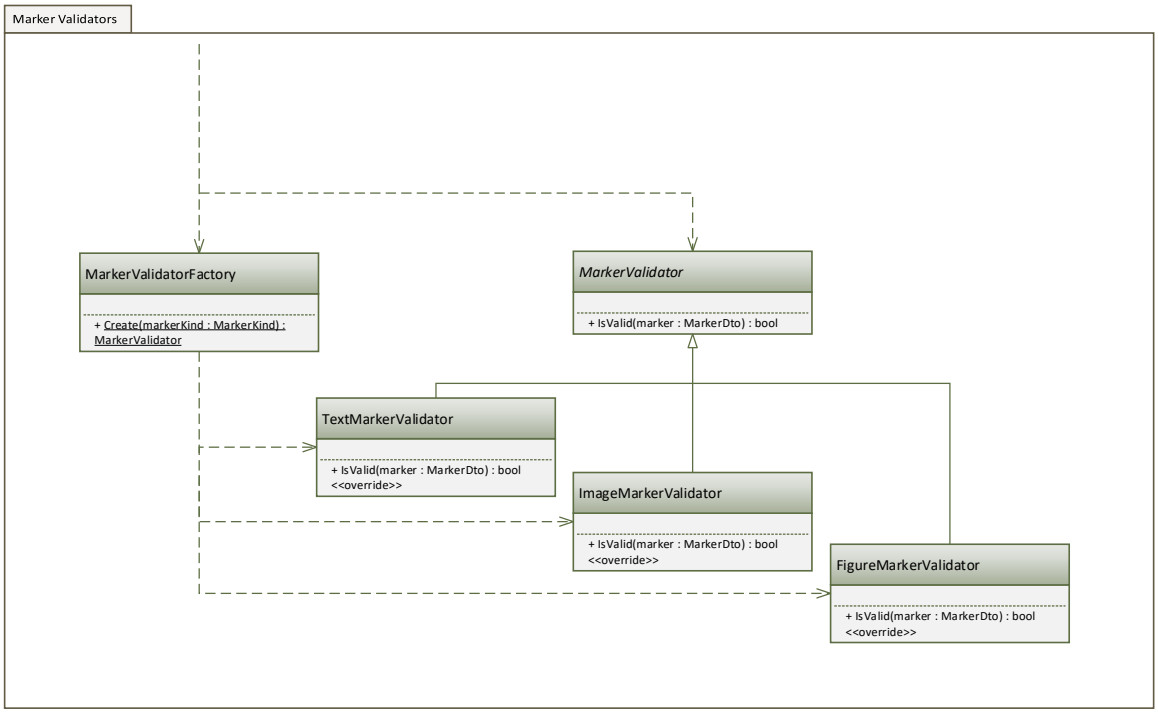


Рисунок Е.3 – Диаграмма классов-валидаторов маркера

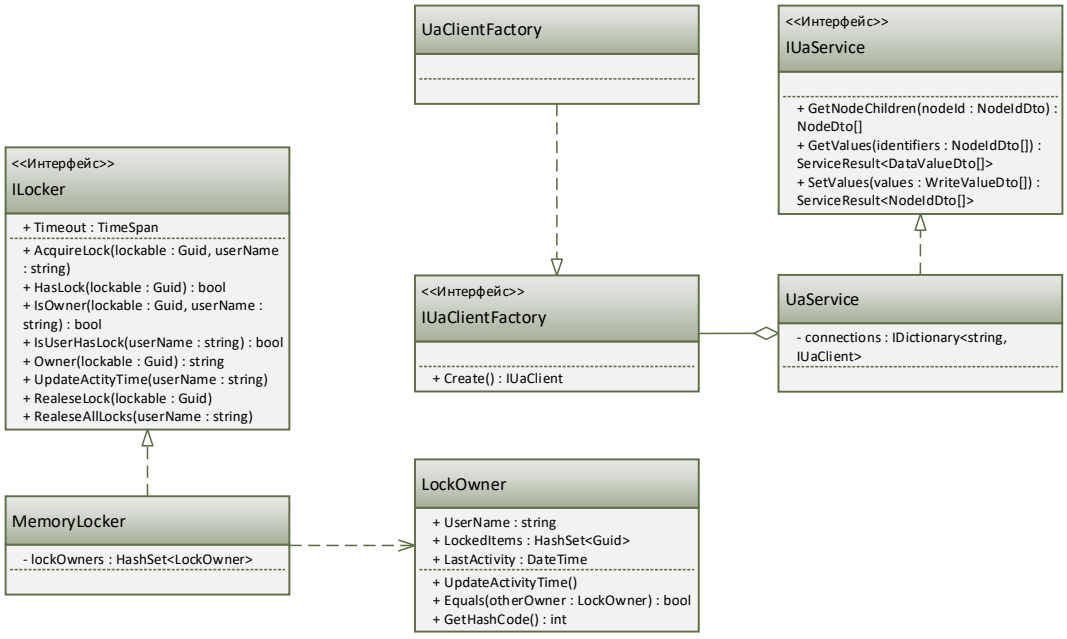


Рисунок Е.4 – Диаграмма классов, реализующих пессимистическую блокировку и управление пулом подключения к серверам *OPC UA*

ПРИЛОЖЕНИЕ Ж

(Обязательное)

Диаграмма классов слоя сервисов

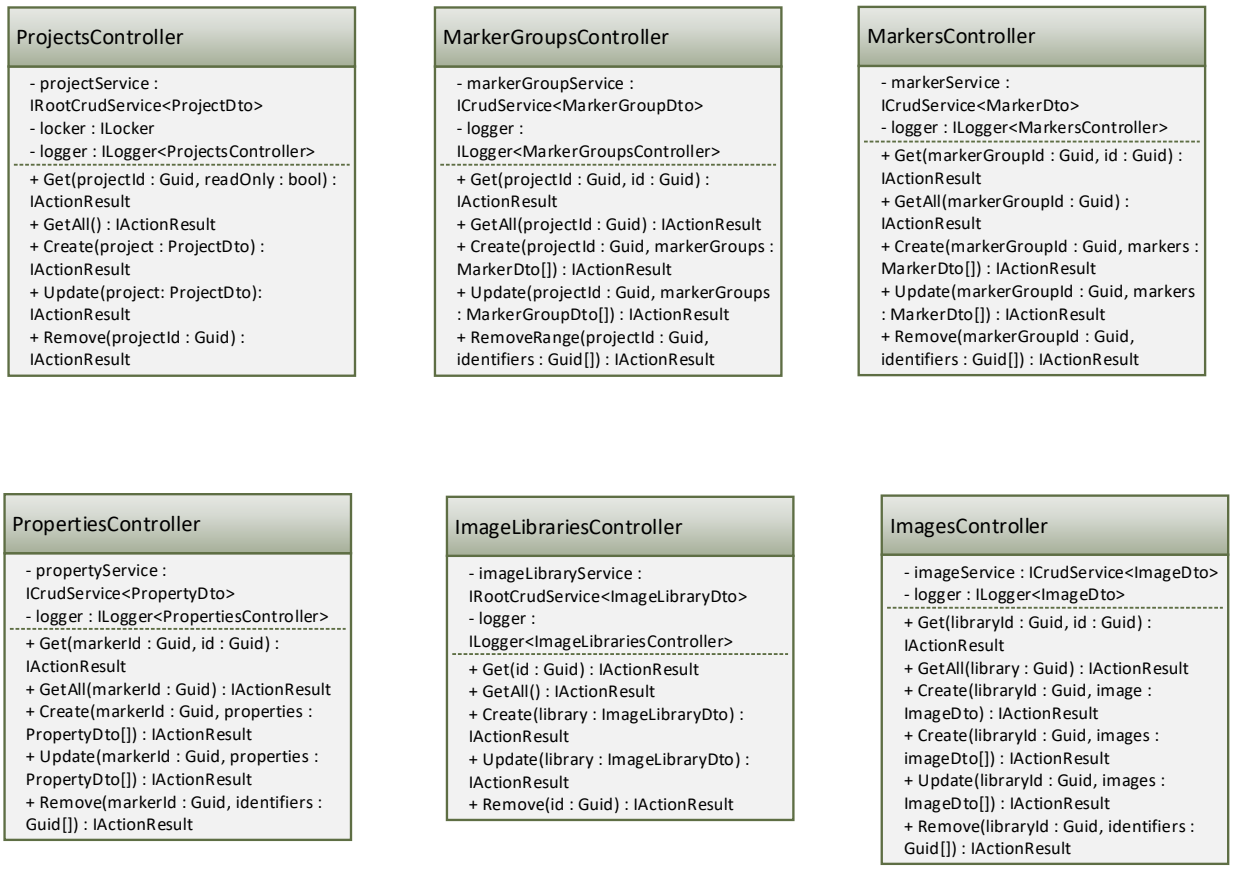


Рисунок Ж.1 – Диаграмма классов-контроллеров

ПРИЛОЖЕНИЕ З

(Обязательное)

Краткое описание REST API

Таблица 1 – Токен

Запрос	**Описание**
POST /token	Создать токен для пользователя, логин и пароль которого переданы в теле запроса. В ответ возвращается токен.

Таблица 2 –Проекты

Запрос	**Описание**
GET /api/v1/projects	Получить все проекты. В ответ возвращаются только проекты, связанные сущности не включены в ответ.
GET /api/v1/projects/{id}	Получить проект с заданным идентификатором ***id***. В ответ возвращается проект, его конфигурация и группы маркеров, определенные в этом проекте.
POST /api/v1/projects	Создать новый проект. В ответ возвращается созданный проект.
PUT /api/v1/projects	Обновить проект, который передан в теле запроса. В ответ возвращается обновленный проект.
DELETE /api/v1/projects/{id}	Удалить проект с заданным идентификатором ***id***. В ответ возвращается удаленный проект.

Таблица 3 – Группа маркеров

Запрос	**Описание**
GET /api/v1/{projectId}/markerGroups	Получить все группы маркеров проекта с идентификатором ***projectId***. В ответ возвращаются только группы, связанные с группой маркеры не включены в ответ.
GET /api/v1/markerGroups/{id}	Получить группу маркеров с заданным идентификатором ***id***. В ответ возвращается группа маркеров и сами маркеры, состоящие в этой группе.
POST /api/v1/{projectId}/markerGroups	Добавить группы маркеров, переданные в теле запроса, в проект с идентификатором ***projectId***. В ответ возвращаются добавленные группы маркеров.
PUT /api/v1/markerGroups	Обновить группы маркеров, переданные в теле запроса. В ответ возвращаются обновленные группы маркеров.

Запрос	Описание
DELETE /api/v1/marker-groups	Удалить группы маркеров, переданные в теле запроса. В ответ возвращаются удаленные группы маркеров.

Таблица 4 – Маркеры

Запрос	Описание
GET /api/v1/{projectId}/markerGroups /{groupId}/markers/	Получить все маркеры, состоящие в группе с идентификатором ***groupId***. В ответ возвращаются маркеры, свойства (Property) и действия (Action) маркеров не включены в ответ.
GET /api/v1/{projectId}/markerGroups /{groupId}/{id}	Получить маркер с указанным идентификатором ***id***. В ответ вместе с маркером возвращаются его свойства и действия.
POST /api/v1/{projectId}/markerGroups /{groupId}//markers	Добавить маркеры, переданные в теле запроса, в группу с идентификатором ***groupId***. В ответ возвращаются добавленные маркеры.
PUT /api/v1/{projectId}/markerGroups /{groupId}/markers/	Обновить маркеры, переданные в теле запроса. В ответ возвращаются обновленные маркеры.
DELETE /api/v1/{projectId}/markerGroups /{groupId}/markers/	Удалить маркеры, переданные в теле запроса. В ответ возвращаются удаленные маркеры.

Таблица 5 – Свойства маркеров

Запрос	Описание
GET /api/v1/{projectId}/markers /{markerId}/properties	Получить все свойства маркера с указанным идентификатором ***markerId***.
GET /api/v1/{projectId}/markers /{markerId}/properties/{id}	Получить свойство с указанным идентификатором ***id***.
POST /api/v1/{projectId}/markers /{markerId}/properties	Добавить свойства, переданные в теле запроса, к маркеру с идентификатором ***markerId***. В ответ возвращаются добавленные свойства.
PUT /api/v1/{projectId}/markers /{markerId}/properties	Обновить свойства, переданные в теле запроса. В ответ возвращаются обновленные свойства.
DELETE /api/v1/{projectId}/markers /{markerId}/properties	Удалить свойства, переданные в теле запроса. В ответ возвращаются удаленные свойства.

Таблица 6 – Библиотеки изображений

Запрос	**Описание**
GET /api/v1/imageLibraries	Получить все библиотеки изображений. В ответ возвращаются библиотеки без изображений.
GET api/v1/imageLibraries/{id}	Получить библиотеку изображений идентификатором ***id***. В ответ возвращается библиотека с его изображениями.
POST /api/v1/imageLibraries	Добавить библиотеки изображений, переданные в запросе. В ответ возвращаются добавленные изображения.
PUT /api/v1/imageLibraries	Обновить библиотеки изображений, переданные в теле запроса. В ответ возвращаются обновленные библиотеки.
DELETE /api/v1/imageLibraries/{id}	Удалить свойство маркера с указанным идентификатором ***id***. В ответ возвращается удаленное свойство.

Таблица 7 – Изображение

Запрос	**Описание**
GET api/v1/imageLibraries /{libraryId}	Получить все изображения из библиотеки с идентификатором ***libraryId***.
GET api/v1/imageLibraries /{libraryId}/images/{id}	Получить изображение с идентификатором ***id***.
POST api/v1/imageLibraries /{libraryId}/images	Добавить изображения, переданные в теле запроса, в библиотеку с идентификатором ***libraryId***. В ответ возвращаются добавленные изображения.
PUT api/v1/imageLibraries /{libraryId}	Обновить изображения, переданные в теле запроса. В ответ возвращаются обновленные изображения.
DELETE api/v1/imageLibraries /{libraryId}/images	Удалить изображения, переданные в теле запроса. В ответ возвращаются удаленные изображения.