МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий

Кафедра «Прикладная математика и информатика»

09.03.03 ПРИКЛАДНАЯ ИНФОРМАТИКА
(код и наименование направления подготовки, специальности)

Бизнес-информатика
(направленность (профиль) / специализация)

БАКАЛАВРСКАЯ РАБОТА

на тему Разработка информационной системы управления энергетическими ресурсами предприятия (на примере АО «ННК-Печоранефть»)

Обучающийся Д.Е. Маслов
(И.О. Фамилия) (личная подпись)

Руководитель К.п.н., доцент, О.Ю. Копша
(ученая степень, звание, И.О. Фамилия)

Тольятти 2022

Аннотация

Название дипломной работы – «Разработка информационной системы управления энергетическими ресурсами предприятия» (на примере АО «ННК-Печоранефть»)».

Данная дипломная работа посвящена созданию автономной информационной системы, работающей в режиме реального времени, обеспечивающей контроль и управление технологическими процессами.

Целью работы является проектирование и разработка децентрализованной, многоуровневой системы контроля и управления, которая предназначена для мониторинга состояния объектов промышленного предприятия в режиме реального времени.

В дипломной работе подробно описана эффективность эксплуатации автономных месторождений на примере Колвинского месторождения АО "ННК-Печоранефть".

Целью создания ИС является разработка модели и принципов построения системы сенсорного управления в виде распределенной многоуровневой расширяемой вычислительной системы, состоящей из автономных исполнительных модулей, которые интегрированы в единую сеть с центральным транспортным модулем.

Прототип системы управления стендом демонстрирует преимущества и эффективность предлагаемой модели по сравнению с существующей. Особенностью разрабатываемой системы является то, что управление исполнительным уровнем может осуществляться через Интернет.

Это позволяет разместить датчики контроля и рабочее место с управляющей программой далеко друг от друга.

В заключение отметим, что актуальность данной работы заключается в применении современных компьютерных технологий в процессе мониторинга технологических процессов управления ресурсами.

Одним из применений таких технологий является разработка автоматизированного рабочего места (АРМ), которое включает в себя как средства оперативного наблюдения, так и средства передачи команд от программ верхнего уровня на устройство.

Значимость работы обоснована тем, что результаты исследования могут быть использованы при разработке различных распределенных систем управления физическими объектами.

Abstract

The title of the graduation work is *Development of an information system for managing energy resources of an enterprise*.

This graduation work is devoted to the creation of an autonomous information system operating in real time, providing control and management of technological processes.

The aim of the work is to design and develop a decentralized, multi-level control and management system, which is designed to condition monitoring of industrial enterprise facilities in real time.

The graduation work describes in detail the efficiency of the operation of autonomous fields on the example of the Kolvinskoye field of AO NNK-Pechoraneft. The creation of the IS is to develop a model and principles for building a sensor control system in the form of a distributed multi-level extensible computing system consisting of autonomous executive modules that are integrated into a single network with a central transport module.

The prototype of the stand control system demonstrates the advantages and effectiveness of the proposed model compared to the existing one. A feature of the system being developed is that the control of the executive level can be performed via the Internet. This allows you to place the monitoring sensors and the workplace with the control program far from each other.

In conclusion, the relevance of this work consists in the application of modern computer technologies in the process of monitoring the technological processes of resource control. One of the applications of such technologies is the development of an automated workplace (APM), which includes both means of operational surveillance and means of transmitting commands from top-level programs to the device. The significance of the work is justified by the fact that the results of the study can be used in the development of various distributed control systems for physical objects.

Содержание

Введение

Целью выполнения выпускной квалификационной работы (ВКР) является разработка системы автоматизированного контроля и учета энергоресурсов промышленного предприятия АО «ННК-Печоранефть» г. Усинск.

Основное назначение системы - передача данных и команд управления от датчиков контроля до программы верхнего уровня и от программы верхнего уровня до исполнительных устройств.

Актуальность работы состоит в применении современных компьютерных технологий в процессе наблюдения за технологическими процессами контроля ресурсов. Одним из приложений таких технологий является разработка автоматизированного рабочего места (АРМ), включающих в себя как средства оперативного наблюдения, так и средства передачи команд от программ верхнего уровня к устройству.

Особенностью работы разрабатываемой системы является то, что управление исполнительным уровнем предполагается выполнять через сеть Интернет. Это позволит разнести датчики контроля и рабочее место с управляющей программой верхнего уровня территориально как угодно далеко друг от друга.

Для достижения поставленной цели необходимо было решить следующие задачи:

- обоснование необходимости разработки;
- разработка общих принципов построения системы;
- описание платформы реализации;
- постановка задачи на разработку;
- описание алгоритмов реализации;
- проектирование структуры транспортной базы данных;
- разработка web-сервиса обмена данными;

- реализация резидентной программы транспортного модуля;
- реализация резидентной программы исполнительного модуля;
- тестирование и устранение ошибок и недостатков.

Объектом исследования являются система автоматизированного учета и контроля энергоресурсов.

Предметом исследования являются способы прогнозирования электропотребления промышленного предприятия.

Областью исследования является исследование работоспособности и качества функционирования электротехнических комплексов и систем в различных режимах, при разнообразных внешних воздействиях.

Практическая значимость работы. Проведенные исследования позволяют ускорить процессы прогнозирования электропотребления и разрабатывать энергосберегающие мероприятия на предприятии.

Научная новизна проекта заключается в разработке модели и принципов построения системы управления датчиками контроля как распределенной многоуровневой расширяемой вычислительной системы, состоящей из автономных исполнительных модулей, объединенных в единую сеть с центральным транспортным модулем, который в свою очередь имеет выход во «внешний мир». Разработанный в результате выполнения проекта прототип системы управления стендом, построенной по этим принципам, наглядно продемонстрирует преимущества и эффективность предложенной модели.

Теоретическая значимость работы обоснована тем, что результаты исследования могут быть использованы при разработке различных распределенных систем управления физическими объектами на базе платформы Arduino.

Практическая значимость работы заключается в разработке действующего прототипа системы предназначенной для управления работой датчиков контроля.

1 Исследовательский раздел

1.2 Анализ и характеристика предметной области

1.1.1 Обобщенная характеристика предметной области

Объектом исследования в данной выпускной квалификационной работе выступает АО «ННК-Печоранефть». АО «ННК-Печоранефть» – агент и оператор по разработке Северо-Харьягинского, Лекхарьягинского и Колвинского нефтяных месторождений. С 1 января 2003 года в промышленной эксплуатации находится Средне-Харьягинское месторождение, промышленную разработку которого АО «ННК-Печоранефть» осуществляет в соответствии с лицензией НРМ 111 86 НЭ.

1.1.2 Характеристика предприятия АО ННК «Печоранефть»

Полное юридическое наименование: открытое акционерное общество по геологии, поискам, разведке и добыче нефти «Печоранефть»

Сокращенное название компании: АО ННК «Печоранефть».

Руководитель – и.о. генерального директора Машорин Владимир Александрович

Дата регистрации – 28.09.1998.

Основной вид деятельности:

- Добыча нефти и нефтяного (попутного) газа.

Компания осуществляет вспомогательные виды деятельности:

- Производство электромонтажных работ.
- Ремонт машин и оборудования.
- Строительство междугородних линий электропередачи и связи.
- Распиловка и строгание древесины.
- Образование профессиональное дополнительное.
- Деятельность вспомогательная прочая, связанная с перевозками.
- Транспортирование по трубопроводам нефти и нефтепродуктов.

- Производство нефтепродуктов.
- Деятельность по дополнительному профессиональному образованию прочая, не включенная в другие группировки.
- Работы геолого-разведочные, геофизические и геохимические в области изучения недр и воспроизводства минерально-сырьевой базы.
- Производство машин и оборудования для добычи полезных ископаемых и строительства.
- Строительство жилых и нежилых зданий.
- Строительство коммунальных объектов для обеспечения электроэнергией и телекоммуникациями.
- Строительство инженерных коммуникаций для водоснабжения и водоотведения, газоснабжения.
- Разведочное бурение.
- Консультирование по вопросам коммерческой деятельности и управления.
- Работы по монтажу стальных строительных конструкций.

Открытое акционерное общество по геологии, поискам, разведке и добыче нефти «Печоранефть» было создано в 1994 году на базе Печорской нефтегазоразведочной экспедиции ГГП «Ухтанефтегазгеология». В дальнейшем большинство из них было передано для освоения нефтегазодобывающим предприятиям. В настоящее время АО «ННК-Печоранефть»» – дочерняя нефтяная компания независимой компании WSR, осуществляющей свою деятельность в России. Концепция построения бизнеса компании включает: добычу нефти, разработку месторождений и геологоразведочные работы. Компания работает в Западной Сибири, Тимано-Печорской нефтегазоносной провинции и Волго-Уральском регионе. Подтвержденные, прогнозные и потенциальные запасы нефти на 31 декабря 2006 года составляли 370,1 миллиона баррелей.

АО «ННК-Печоранефть» – агент и оператор по разработке Северо-Харьягинского, Лекхарьягинского и Колвинского нефтяных месторождений. С 1 января 2003 года в промышленной эксплуатации находится Средне-Харьягинское месторождение, промышленную разработку которого АО «ННК-Печоранефть» осуществляет в соответствии с лицензией НРМ 111 86 НЭ [21].

1.1.3 Составление организационной структуры компании

Успешная деятельность каждой организации во многом определяется ее внутренней организационной структурой, а также формированием схемы органов управления, которые могли бы оперативно принимать управленческие решения.

Организационная структура АО ННК «Печоранефть» определяется составом его подразделений и их характером взаимосвязи и подотчетности.

На рисунке 1 рассматривается организационная структура компании.

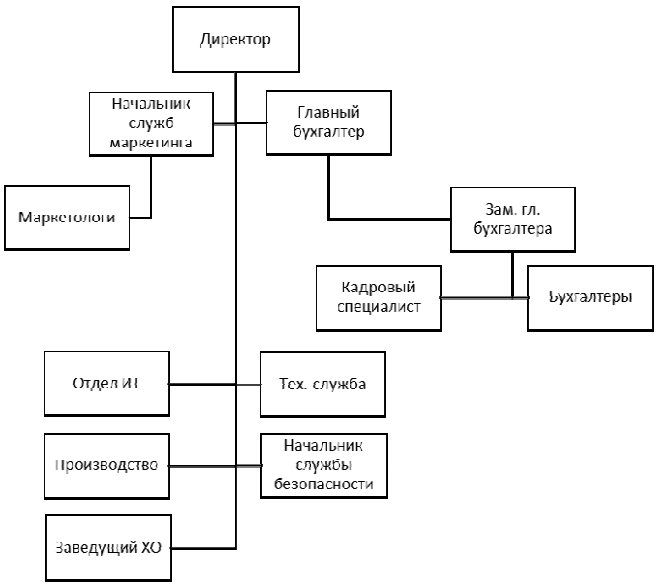


Рисунок 1 – Организационная структура АО ННК «Печоранефть»

Наибольший интерес для нас представляет в этой структуре Отдел ИТ. Отдел ИТ обеспечивает работоспособность всего парка компьютерного оборудования и ПО.

В работе будет выполняться разработка ПО силам персонала ИТ-отдела АО ННК «Печоранефть». На рисунке 2 рассматривается организационная структура ИТ-отдела АО ННК «Печоранефть».

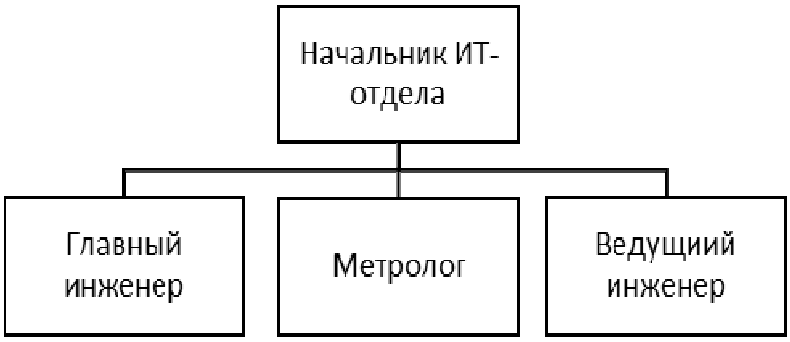


Рисунок 2 – Организационная структура ИТ-отдела АО ННК «Печоранефть»

Основными функциями всего ИТ-отдела являются:

- помощь в осуществлении электронного документооборота;
- проверка и регулировка точности работы измерительных аппаратов и приспособлений;
- приведение измерительных приборов в полное соответствие установленным стандартам;
- поддержка работоспособности компьютерной техники, которая используется в АО ННК «Печоранефть»;
- поддержка работоспособности сети Интернет и локальной вычислительной сети;
- обеспечение функционирования информационных систем бухгалтерии и других подразделений;
- разработка надстроек для ИС с целью максимально автоматизировать рабочий процесс АО ННК «Печоранефть».

Поскольку в каждом из отделов АО ННК «Печоранефть». используется компьютерная техника, то данное структурное подразделение выполняет поддержку деятельности и документооборота для каждого из них.

1.1.4 Техническое и аппаратное обеспечение

Рассмотрим далее описание программного и аппаратного обеспечения организации.

Основой ИТ-архитектуры является мощный сервер «Dell R440». Данный сервер является современным техническим решением, и служит для обеспечения доступа пользователей к общим данным, принтерам, файлам, а также к другим общим сервисам. Сервер «Dell R440» имеет следующие характеристики:

- Процессор: 2 × Intel Xeon Silver 4208 (8C 11M Cache 2.10 GHz);
- Оперативная память: 4x DDR4 RDIMM 2133MHz REF 32 Гб, всего 128 Гб;
- Контроллер RAID: Dell H330 (ZM);
- Модуль удаленного управления iDRAC 9 Enterprise;
- Интегрированная сетевая карта Dell 2port 1Gb;
- Жесткий диск: Dell 1.2TB SAS 10k 2.5" G14
- Блок питания: 2x Dell 550w Hot Plug.

Этот сервер выполняет в комендатуре функции контролера домена, сервера электронной почты, контроля доступа в сеть «Интернет», «файл-сервера», сервера приложений 1С и сервера баз данных (БД). Он функционирует под управлением Windows Server Standard 2014 OEM. Для его бесперебойного питания служит ИБП «APC Smart-UPS RT 1000VA 230V».

Началом локальной сети «Ethernet» комендатуры является управляемый коммутатор «Fast Ethernet» уровня 3 «D_Link DES-3552P». Компьютеры сотрудников подразделений подключаются к главному коммутатору либо непосредственно, либо через коммутаторы «D-Link» типа «Des 1008A» и «DES 1016», установленные в подразделениях.

Для печати, копирования и сканирования офисных документов в комендатуре используется сетевое многофункциональное устройство (МФУ)

«Xerox Phaser 3635 MFP/S». МФУ установлено в канцелярии комендатуры, подключено через сетевой порт к локальной сети предприятия и используется всеми сотрудниками совместно.

На рабочих местах сотрудников отделов установлены персональные компьютеры (ПК) архитектуры «x86» или «x86-64». Типовой ПК сотрудников имеет характеристики:

– процессор – «Intel I3»;
– оперативная память: 4 Gb;
– жесткий диск 500 Гб.

Техническая архитектура ИС изображена на рисунке 3.

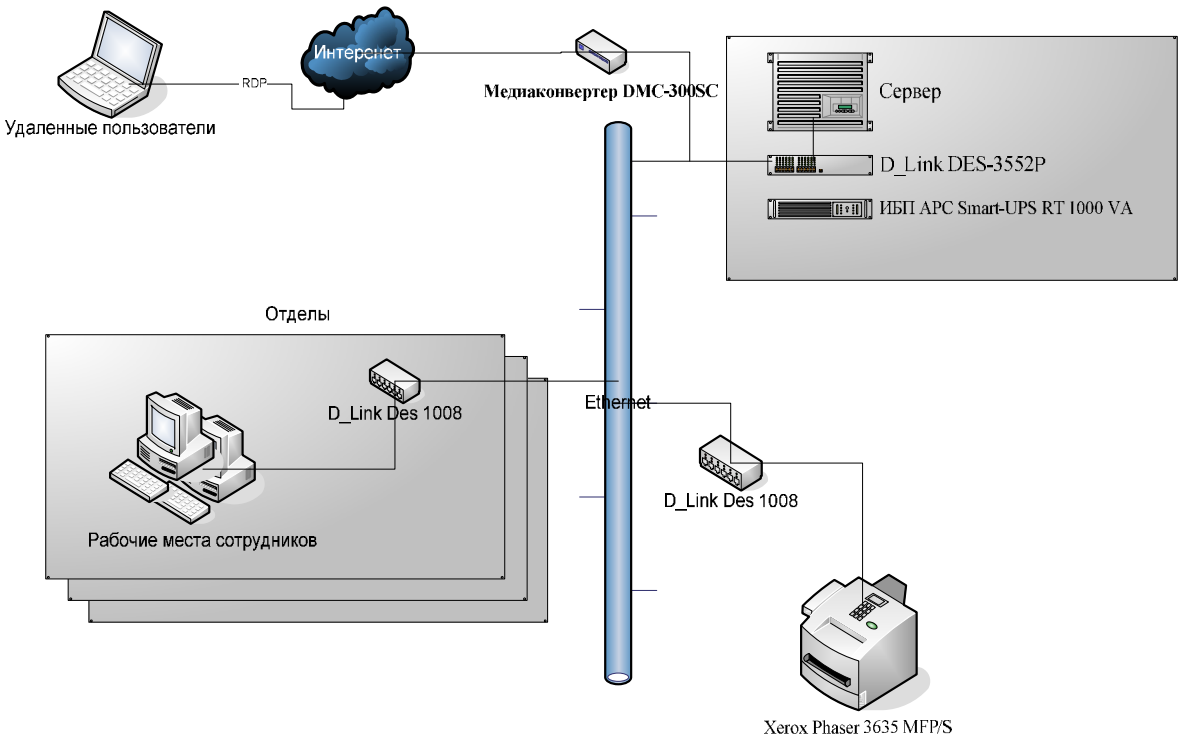


Рисунок 3 – Техническая архитектура ИС предприятия

На сервере установлены и успешно функционируют следующие ПП.

«1С:Бухгалтерия 8 (редакция 3.0)» на базе платформы «1С:Предприятие 8.3». Данный ПП является системой комплексной

автоматизации деятельности финансовой части. Возможностей этой системы достаточно для того чтобы вести учет персонала и учет заработной платы.

«1С:Бухгалтерия 8 (редакция 3.0)» используется в клиент-серверном варианте. На физическом сервере «Dell R440» функционирует сервер приложений 1С, который обращается к СУБД «Microsoft SQL Server 2014», который используется для хранения данных. «Microsoft SQL Server 2014» развернут на том же физическом сервере «Dell R440». На рабочих местах сотрудников финансовой части установлена клиентская часть программы «1С:Бухгалтерия 8».

В качестве антивирусной защиты служит «Symantec Endpoint Security». Система установлена на сервере, а на рабочих местах пользователей установлена клиентская часть.

На ПК сотрудников установлена операционная система (ОС) «Windows 7» или «Windows 10». Эта ОС позволяет подключаться к сети и использовать все ее ресурсы. На всех компьютерах установлено следующее специальное ПО: «Symantec Endpoint Security» клиентская часть, позволяющая обеспечить защиту ПК от вирусов; «Microsoft Office 2015» – ПП для создания и ведения электронных документов; «Mozilla Firefox» – «Интернет-браузер»; «Outlook Express» – почтовый клиент.

Файл-сервер предприятия является хранилищем данных учета происшествий, в том виде, в котором он ведется сейчас. Весь учет ведется в папке «Учет» и представляет собой набор «word» и «excel»-файлов. Все участники процесса учета происшествий на основе шаблонов заносят в папку «Учет» на файл-сервере свои файлы, касающиеся учета происшествий. В Качестве файл-сервера используется сервер Dell R440. В таблице 1 представлен анализ аппаратного и программного обеспечения в организации.

Таблица 1 – Анализ аппаратного и программного обеспечения в организации.

№	Техническое/программное обеспечение	Требует обновления
Техническое обеспечение		
1	Сервер Dell R440	Нет
2	ИБП APC Smart-UPS RT 1000VA 230V.	Нет
3	D_Link DES-3552P	Нет
4	D_Link DES 1008A	Нет
5	D_Link DES 1016	Нет
6	Xerox Phaser 3635 MFP/S	Нет
7	АРМ HP Pavilion 23	Нет
Программное обеспечение		
1	MS Windows Server Standard 2014 OEM	Да
2	1С:Предприятие 8.3	Да
3	1С:Бухгалтерия 8	Да
4	MS Windows 7	Да
5	MS Windows 10	Да
6	Symantec Endpoint Security	Да
7	Mozilla Firefox	Да
9	Outlook Express	Да

1.2 Обоснование необходимости разработки системы

Как известно, устойчивое функционирование предприятия необходимо для нормальной работы всего механизма экономики. На развитие хозяйствующих субъектов оказывает большое влияние высокая доля энергетических затрат в издержках производства. Ввиду быстрого увеличения цен на энергоносители, в связи с большим моральным и физическим износом основного оборудования и значительными потерями при транспортировке энергетических ресурсов, затраты на них в промышленности выросли многократно, и только в себестоимости промышленной продукции составляют от 5 до 40 %. Для этого необходимо внедрение на предприятии энергосберегающих мероприятий [13].

Энергосбережение на предприятии характеризуется ориентацией на долгосрочную перспективу, учитывая его отдельные инвестиционные приоритеты. Долгосрочная политика предполагает выработку определенных типов действий на изменение внешней и внутренней среды, которые

зафиксированы в определенных стратегиях. Принципы управления энергоэффективностью опираются на используемые подходы к энергосбережению, важнейшими из которых являются: технократический, системный и инновационный. Процесс разработки стратегии начинается с выработки принципов, которые заданы существующими условиями внешней и внутренней среды: уровнем технологического развития отрасли, стандартов энергоэффективности, государственной политикой на региональном и федеральном уровнях и т.п. Процесс управления энергопотреблением не является одномоментным, поэтому его невозможно рассматривать в отрыве от протяженности во времени. Общая стратегия развития компании должна, так или иначе, включать долгосрочную стратегию энергосбережения. Создание стратегии энергопотребления помогает предприятию избежать рисков и получить конкурентное преимущество относительно других компаний, представляющих свою продукцию или услуги на рынке [9].

1.3 Анализ предметной области с целью выявления объекта автоматизации

Таким образом, обобщив все вышеперечисленное, можно сказать, что внедрение и развитие системы энергосбережения на предприятии, является одним из важнейших факторов его конкурентоспособности, которое обеспечивает экономию топливно-энергетических ресурсов, системность в планировании работы производственных систем предприятия, формирование положительного имиджа компании и повышения энергоэффективности предприятия в целом.

Для этого стоит разработать прототип децентрализованной, распределенной, многоуровневой системы контроля и управления (СКУ), которая будет предназначена для мониторинга и регулирования в реальном

времени технического состояния различных объектов контроля и управления (ОКУ) на АО ННК «Печоранефть».

В качестве ОКУ могут выступать совершенно любые системы или процессы, состояние которых может быть описано с помощью того или иного набора датчиков, а параметры и режимы работы (или окружающей среды, в которой они функционируют) изменены при помощи того или иного набора регулирующих (реализующих) механизмов (РМ). Примеры возможных вариантов ОКУ приведены на рисунке 4.

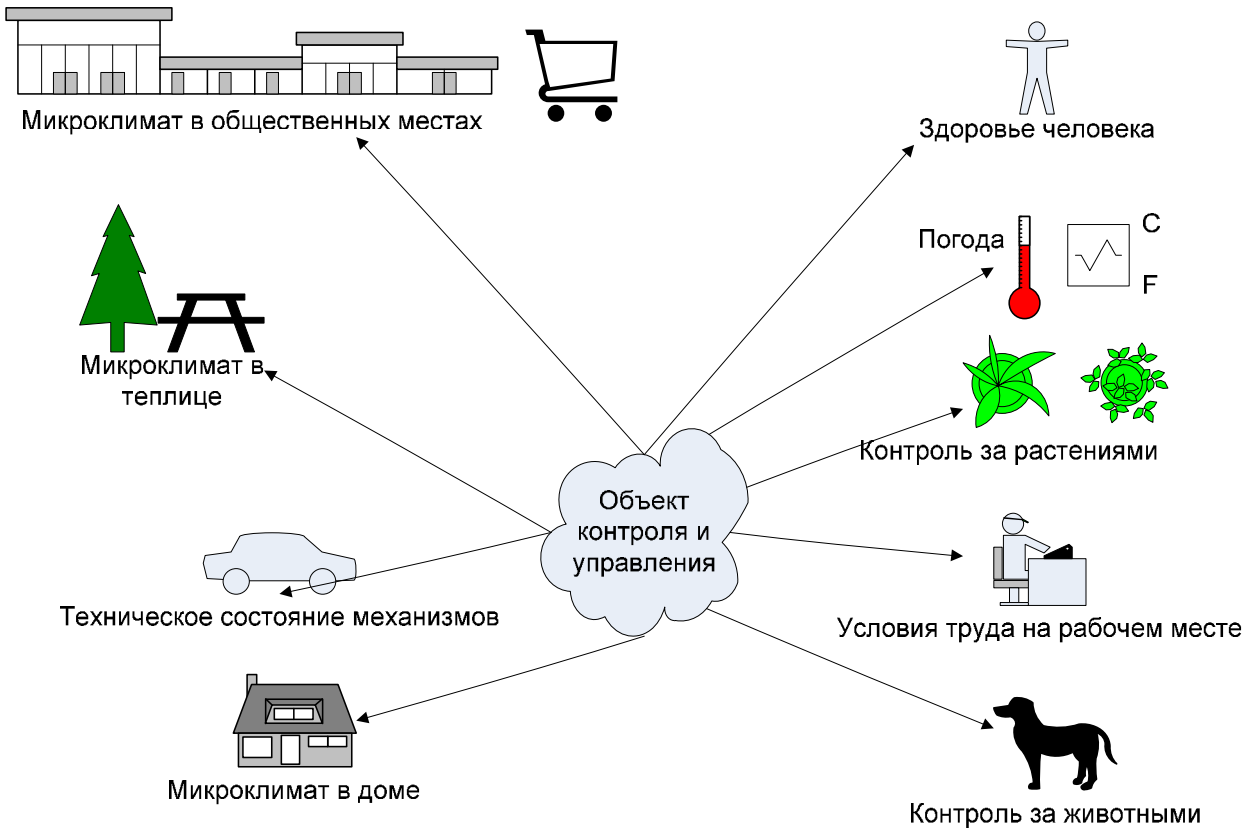


Рисунок 4 – Варианты применения разрабатываемой СКУ

Важным принципом разрабатываемой СКУ является использование программируемых микроконтроллеров, которые осуществляют непосредственный мониторинг состояния ОКУ при помощи датчиков и

управление ими (или окружающей средой, в которой находится ОКУ) при помощи РМ.

Основная идея построения СКУ – разнесение функций мониторинга и управления ОКУ по разным уровням. На верхнем уровне (ВУ) находится управляющая программа, которая при помощи графического интерфейса в удобном формате предоставляет возможность контроля и управления всеми ОКУ. На нижнем уровне (НУ) находятся микроконтроллеры, которые непосредственно считывают данные с датчиков и управляют ОКУ при помощи РМ. Передача команд управления от верхнего уровня на нижний уровень производится через специальный web-сервис (средний уровень, СУ), который функционирует в сети Интернет или в локальной сети организации. Передача команд и данных осуществляется по протоколу http. Web-сервис имеет собственную транспортную базу данных, которая хранит передаваемые между верхним и нижним уровнем команды и данные, что гарантирует сохранность данных. Обобщенная схема работы СКУ приведена на рисунке 5).

Рисунок 5 – Обобщенная схема работы СКУ

Основное назначение разрабатываемой СКУ - мониторинг и регулирование в реальном времени состояния различных ОКУ.

В качестве ОКУ в нашей работе будут выступать системы инженерных коммуникаций и датчики контроля ресурсов:

– эл энергия;
– газ;
– тепло;
– водоснабжение;

В результате анализа предметной области была разработана функциональная модель системы к данному проекту. Представим модель нашей системы помощью методологии IDEF0.

Построим диаграмму верхнего уровня, и представим на рисунке 6. Главной компонентой верхнего уровня будет являться процесс «Системы контроля и управления» (СКУ).

Входами для реализации этого процесса являются:

– данные контролеров ресурсов (данные с датчиков контроля теплоснабжения, водоснабжения, газа, электросети);
– данные о расположении контролеров ресурсов (данные о локации расположения датчиков)

Управление осуществляют:

– устав компании АО «ННК-Печоранефть» (в частности правило оформления отчетности данных контролеров);

Механизмами являются:

– персонал компании (в частности этим занимается Отдел ИТ: начальник отдела или метролог);
– офисное ПО, которое применяется для учета обращений.

Выходами являются:

– отчет показаний датчиков (список датчиков с их описанием и возможностью передачи или приема данных);
– отчет по расположению контролеров ресурсов (список датчиков с их расположением);
– список датчиков контроля (список датчиков с их описанием);
– журнал обмена данными (журнал полученных от контролера данных и команд, а также заданных команд датчикам контролеров, которые они получают или передают).

Все отчеты данных формируется посредством программного продукта «MS Office». Оформление происходит согласно и учетной политики организации.

Функциональная декомпозиция «СКУ» проводится также на основе методологии IDEF0.

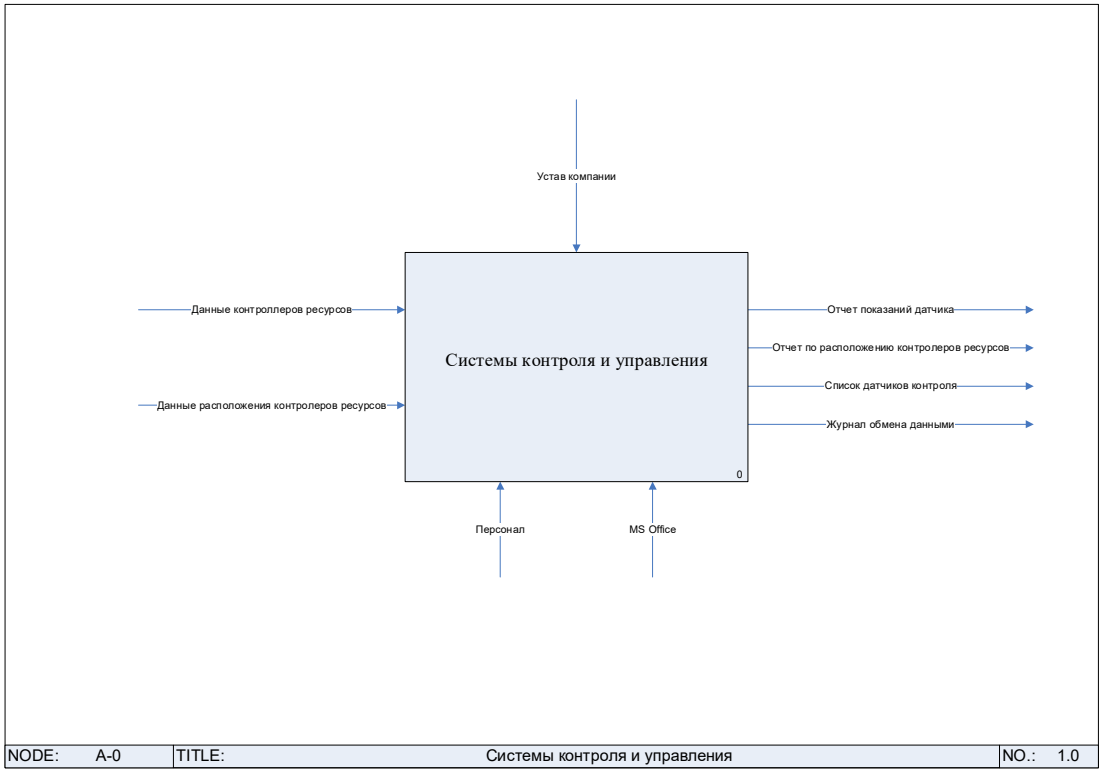


Рисунок 6 – Контекстная диаграмма «СКУ».

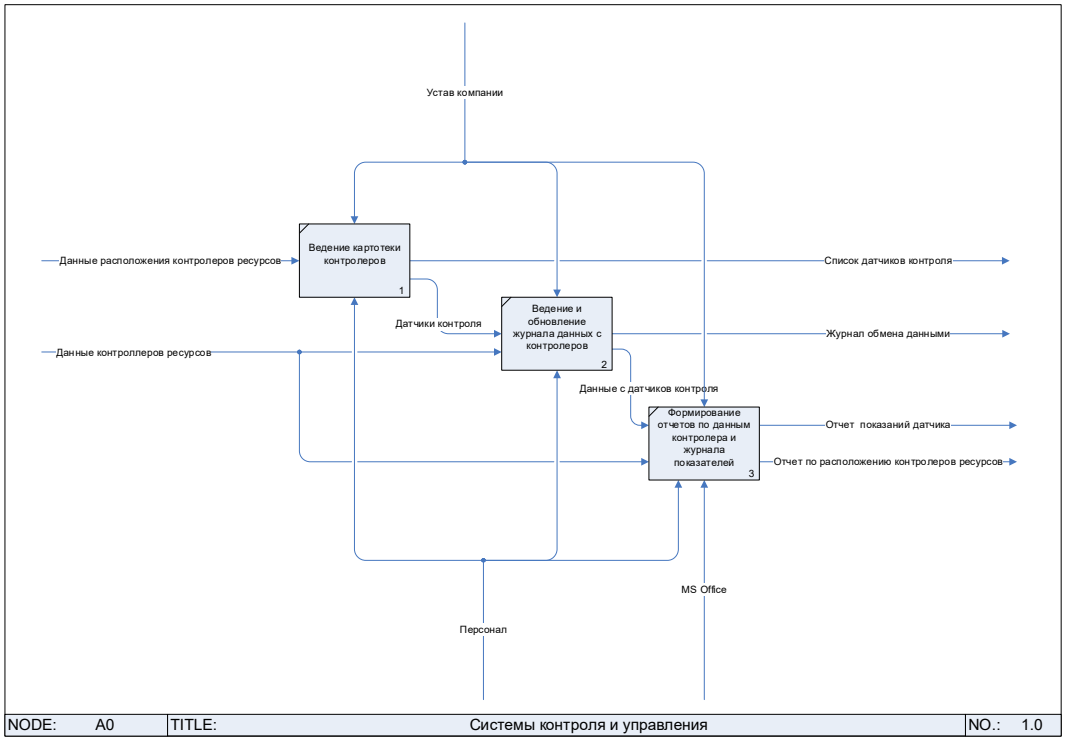


Рисунок 7 – Диаграмма декомпозиции «СКУ»

Контекстная диаграмма и декомпозиция «СКУ», указанная на рисунках 6-7, представим в виде трёх процессов: «Ведение картотеки контролеров», «Ведение и обновление журнала данных с контролеров» и «Формирование отчетов по данным контролера и журнала показателей».

Ведение картотеки контролеров.

Входной информацией для выполнения БП «Ведение картотеки контролеров» является данные контролеров ресурсов (данные с датчиков контроля теплоснабжения, водоснабжения, газа, электросети).

Управление регламентируется уставом компании АО «ННК-Печоранефть».

БП осуществляется персоналом компании (в частности этим занимается метролог).

Выходной информацией является список датчиков контроля (список датчиков с их описанием).

Ведение и обновление журнала данных с контролеров.

Входной информацией для выполнения БП «Ведение и обновление журнала данных с контролеров» является данные контролеров ресурсов (данные с датчиков контроля теплоснабжения, водоснабжения, газа, электросети) и список датчиков контроля (список датчиков с их описанием).

Управление регламентируется уставом компании АО «ННК-Печоранефть».

БП осуществляется персоналом компании (в частности этим занимается Отдел ИТ: начальник отдела или метролог).

Выходной информацией является журнал обмена данными и журнал обмена данными (журнал полученных от контролера данных и команд, а также заданных команд датчикам контролеров, которые они получают или передают) и данные с датчиков (данные с датчиков теплоснабжения, водоснабжения, газа, электросети).

Формирование отчетов по данным контролера и журнала показателей.

Входной информацией для выполнения БП «Формирование отчетов по данным контролера и журнала показателей» является данные с датчиков (данные с датчиков теплоснабжения, водоснабжения, газа, электросети) и данные контролеров ресурсов (данные с датчиков контроля теплоснабжения, водоснабжения, газа, электросети)

Управление регламентируется уставом компании АО «ННК-Печоранефть».

БП осуществляется персоналом компании (в частности этим занимается метролог).

Выходной информацией является отчет показаний датчиков (список датчиков с их описанием и возможностью передачи или приема данных) и отчет по расположению контролеров ресурсов (список датчиков с их расположением);

В результате рассмотрения бизнес-процесса необходимо разработку СКУ, в функции которой будет включено: мониторинг и регулирование в реальном времени состояния различных ОКУ для компании АО «ННК-Печоранефть», так как настоящая технология реализации данного процесса имеет следующие недостатки:

- ручное формирование отчетности;
- ручное снятие данных с датчиков (контроля теплоснабжения, водоснабжения, газа, электросети)
- отсутствие автоматизации в данной области предприятия;
- низкий уровень защиты информации и другие.

Так как количество ОКУ, находящихся на нижнем уровне может быть весьма большое, то вполне возможна ситуация, что подключить их все к одному микроконтроллеру окажется невозможно. В этом случае нужно использовать несколько микроконтроллеров, объединенных в сеть. Один из них является главным и выполняет обмен данными с верхним уровнем через web-сервис и передачу этих данных на остальные контроллеры. Остальные контроллеры нижнего уровня являются исполнительными, они непосредственно считывают данные с датчиков и управляют работой регулирующих механизмов. Все контроллеры нижнего уровня должны быть связны с главным контроллером по различным проводным и беспроводным каналам передачи данных, наиболее распространенных на текущий момент.

Главный микроконтроллер должен получить доступ к web-сервису обмена данными. Для этого он должен сначала получить доступ к локальной сети, функционирующей в данной организации и уже через эту локальную сеть получить доступ к сети Интернет и к web-сервису, функционирующему на каком-либо web-ресурсе.

Web-сервис обмена данными может быть расположен как на web-ресурсе где-то в сети Интернет, так и быть развернут на web-сервере в данной локальной сети. В последнем случае главному микроконтроллеру

нижнего уровня доступ к сети Интернет не нужен, нужен только доступ к локальной сети. Подключение к локальной сети, доступ к web-сервису, протоколы обмена данными никак не зависят от того где физически располагается web-сервис. Еще одним важным принципом построения системы является использование в протоколах всех уровней текстовых АТ-команд. Это дает возможность их очень простой интерпретации и восприятия, так как текстовые команды могут быть просмотрены в любом текстовом редакторе. Особенно это важно на этапе отладки.

Основные принципы построения системы приведены на рисунке 8.

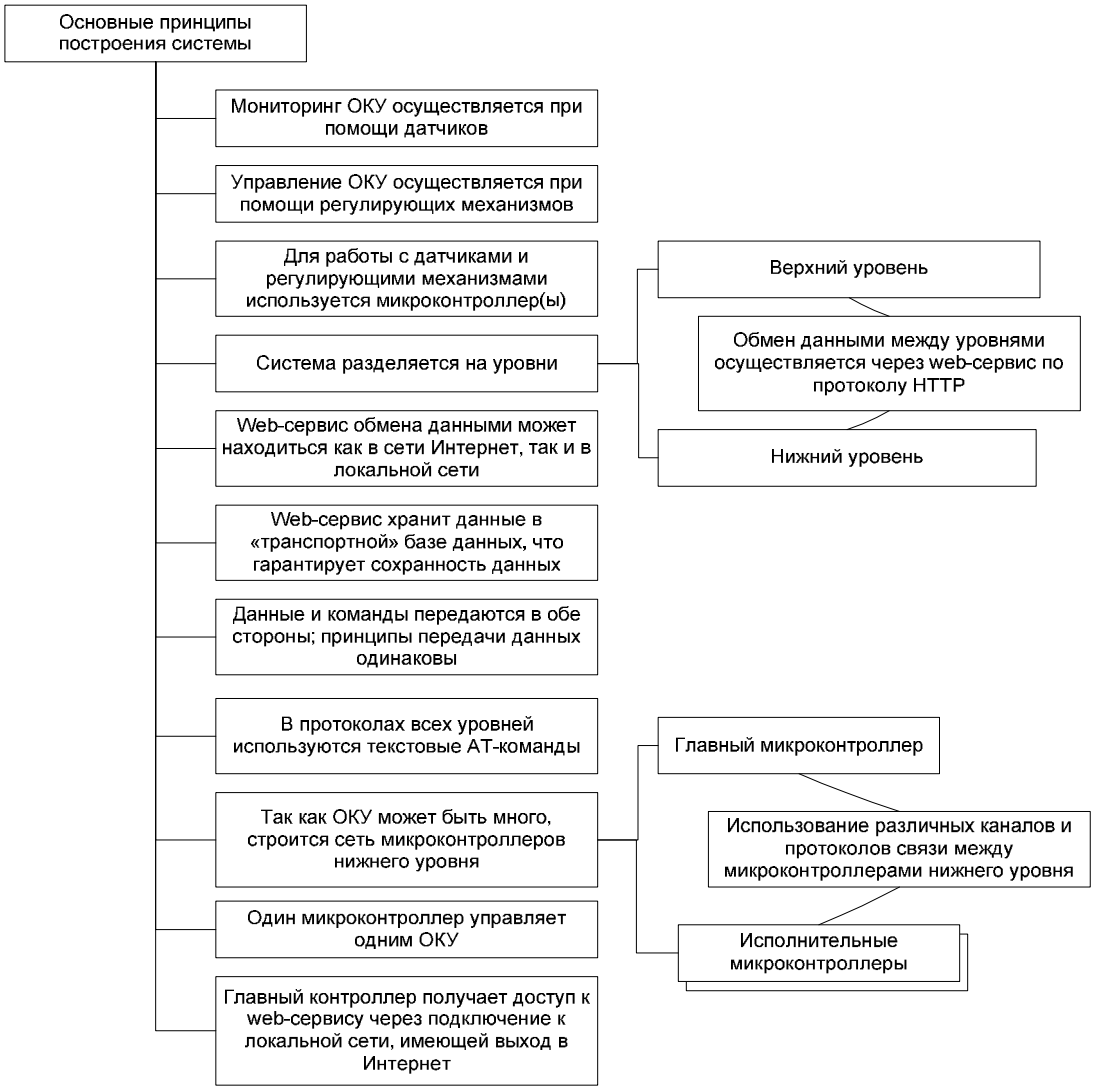


Рисунок 8 – Основные принципы построения системы

1.4 Постановка задачи на разработку

Основное назначение разрабатываемой СКУ - мониторинг и регулирование в реальном времени состояния различных ОКУ.

Одним из основных принципов проектирования СКУ является разнесение ее функций по уровням.

Структура уровней разрабатываемой системы приведена на рисунке 9.

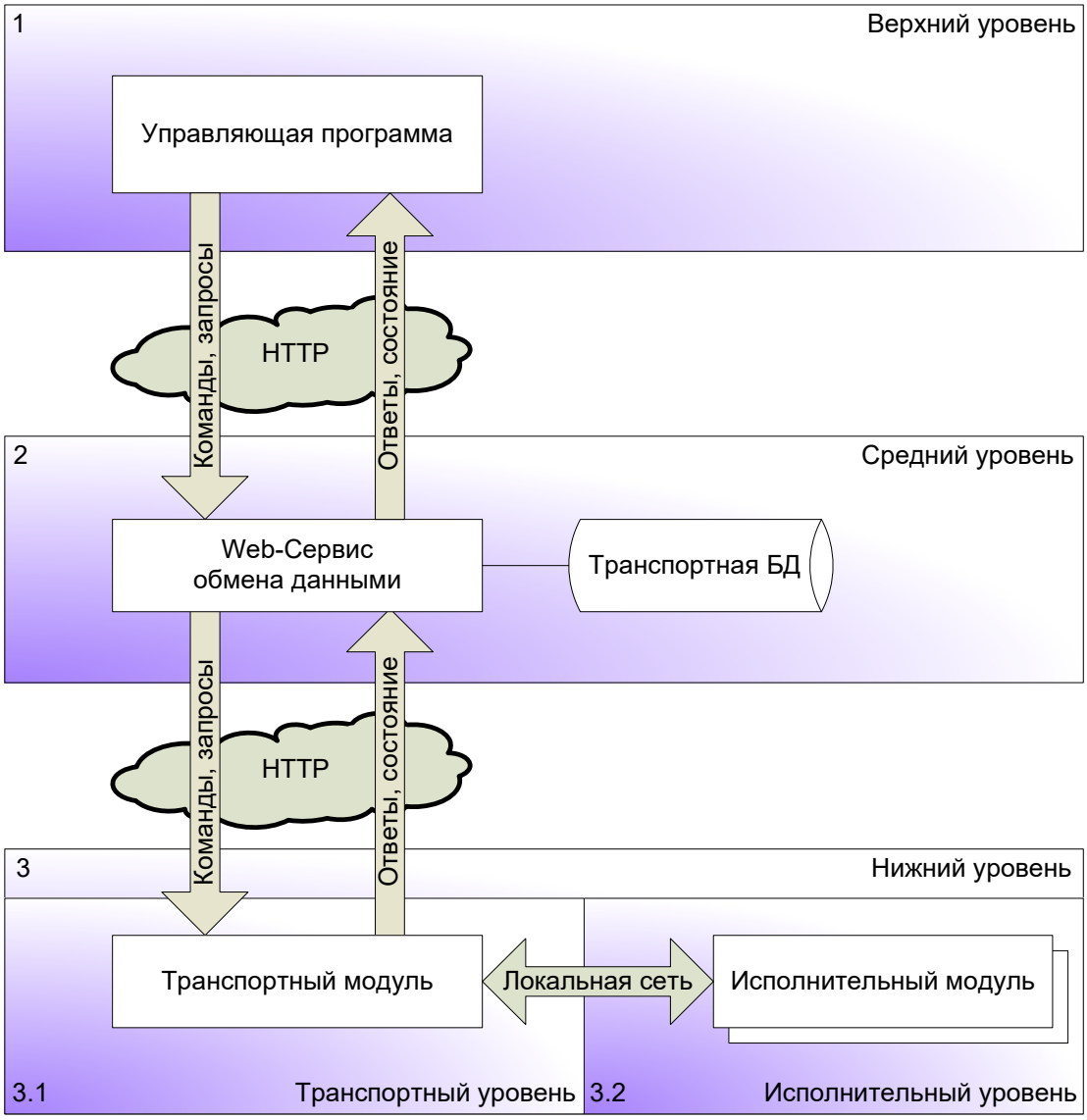


Рисунок 9 – Уровни, на которые делится система

На верхнем уровне работает специальная управляющая программа верхнего уровня (ПВУ). Программа работает на ПК и для хранения данных использует базу данных под управлением СУБД. Основное назначение ПВУ - отображать в удобном для пользователя виде состояние ОКУ и выдавать команды для его управления. Состояние ОКУ описывается показаниями набора датчиков, подключенных к исполнительным модулям. Управление осуществляется через набор команд, отправляемых на регулирующие механизмы. Программа ведет список контролируемых ОКУ, список подключенных к каждому ОКУ датчиков и РМ, а также привязку ОКУ к помещениям. По каждому ОКУ ведется история показаний, подключенных к нему датчиков. Кроме того, выводится текущие показания датчиков ОКУ и состояния РМ. Вывод данных организован в удобном для пользователя виде.

ПВУ может отдать команду (через СУ) на любой подключенный к ОКУ РМ. Состав команд определяется набором и характером РМ. ПВУ периодически запрашивает данные с ОКУ (через СУ), записывает их в свою базу данных и отображает на экране.

На среднем уровне функционирует специальный web-сервис, передающий команды и данные между ПВУ и НУ в обе стороны. Это универсальный транспорт, позволяющий передавать данные в обоих направлениях и взаимодействовать программам ВУ и НУ независимо от их местонахождения [12].

Web-сервис публикуется на каком-либо локальном или удаленном web-сервере. Это может быть web-хостинг в сети Интернет или web-сервер организации, функционирующий в локальной сети. Последний вариант целесообразно использовать для отладки системы. Размещение web-сервиса на каком-либо web-сервере в сети Интернет дает возможность размещать ОКУ в любой точке мира. То же самое касается и ПВУ.

Взаимодействие программ уровней ВУ и НУ с данным web-сервисом, передача команд и данных производится при помощи http-запросов. Формат

(протокол) данных, передаваемых внутри http-запроса, разрабатывается в рамках данного проекта. Данные передаются в виде текстовых срок, представляющих из себя АТ-команды. Формат передачи данных между уровнями ВУ-СУ и СУ-НУ одинаков.

Нижний уровень - это аппаратно-программный комплекс непосредственной работы с ОКУ и приема-передачи данных с СУ.

Нижний уровень разрабатываемой СКУ делится на подсистемы. Каждая подсистема – это автономный исполнительный модуль (ИМ) мониторинга и управления работой одного ОКУ. ИМ работает независимо и автономно и выполняет все, какие нужно функции по мониторингу и управлению контролируемым им ОКУ. Состав подключаемых к ИМ датчиков и РМ, а также логика его работы, полностью определяются тем, каким конкретно ОКУ он управляет. Постановка задачи и разработка каждого ИМ может быть выполнена в будущем в отдельных проектах, но должны соблюдаться общие принципы работы ИМ, касающиеся сетевого взаимодействия между уровнями. В данной работе мы не будем планировать и детально проектировать, какие ИМ будут участвовать в системе, какие датчики или РМ к ним подключать, какую логику работы в них закладывать. В данной работе будет сформулирована постановка задачи и разработка только одного ИМ.

Нижний уровень делится на два подуровня:

- транспортный подуровень (ТУ), на котором работает транспортный модуль (ТМ), обеспечивающий связь между СУ (web-сервис) и подуровнем непосредственной работы с ОКУ;
- исполнительный подуровень (ИУ) – непосредственная работа с ОКУ: считывание данных с датчиков и управление РМ.

Транспортный модуль является главным модулем (главным контроллером) НУ и выполняет обмен данными между всеми ИМ, находящимися на ИУ, с ВУ через web-сервис (СУ) в обе стороны. Остальные

контроллеры НУ являются исполнительными, они непосредственно управляют работой ИМ.

Все контроллеры нижнего уровня должны быть связаны с главным контроллером различными проводными и беспроводными каналами связи со своими интерфейсами. Совокупность этих каналов связи образует локальную сеть (ЛС) НУ. Для связи ИМ и ТМ по локальной сети также необходимо будет разработать специальный протокол в рамках, используемых в этой локальной сети интерфейсов. Варианты проводных и беспроводных интерфейсов связи ИМ и ТМ приведены на рисунке 10.

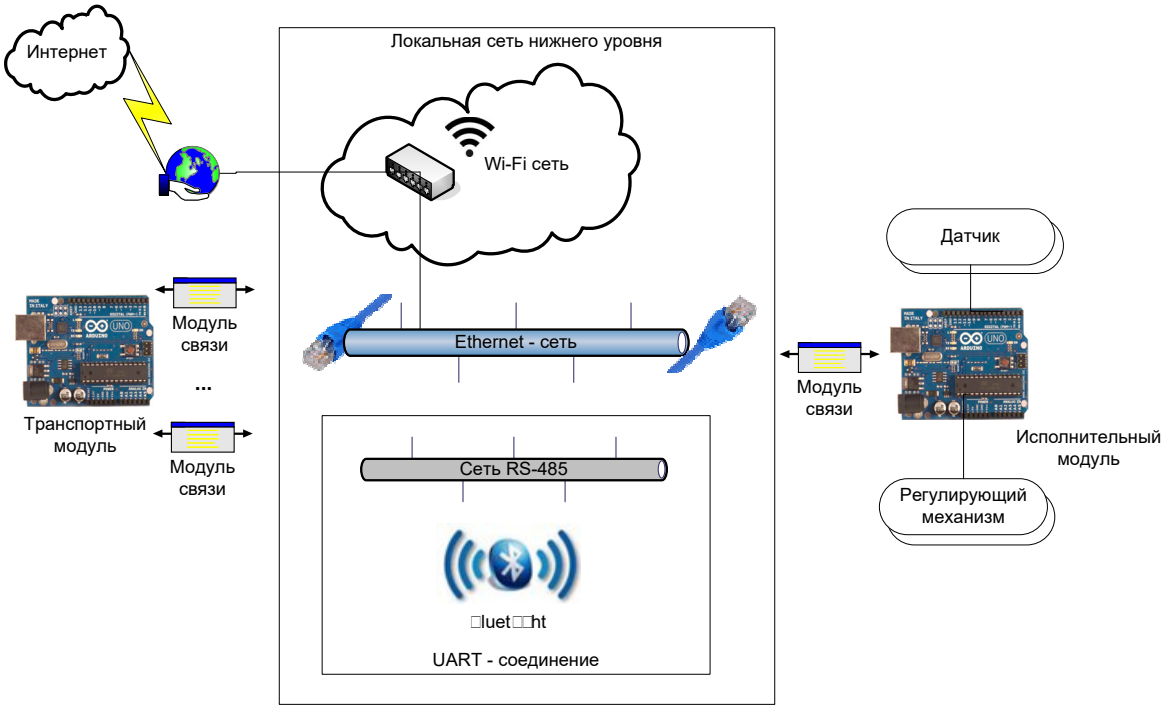


Рисунок 10 – Локальная сеть нижнего уровня

Транспортный модуль должен постоянно прослушивать соединение с СУ для приема команд от web-сервиса и в случае их получения – интерпретировать их и рассылать в локальную сеть всем ИМ, независимо от того, при помощи каких интерфейсов ИМ подключены к локальной сети. Параллельно ТМ должен прослушивать соединение с локальной сетью со

стороны ИУ, и в случае получения от ИМ этого уровня каких-либо данных (также независимо от интерфейса подключения ИМ к локальной сети) – переправлять их web-сервису на СУ.

Исполнительный модуль должен постоянно прослушивать соединение с ТМ для приема от него команд и в случае их получения – интерпретировать их. Если команда адресована данному ИМ, ИМ должен выполнить ее. Если команда требует ответа от ИМ, то ИМ должен сформировать и передать этот ответ в ТМ. ИМ может передать данные в ТМ и по собственной инициативе в зависимости от логики работы ИМ, например, при наступлении в ОКУ определенного события или при превышении показаний какого-либо из датчиков порогового значения.

Протокол передачи данных между ТМ и ИМ зависит от выбранного интерфейса передачи данных и физической реализации модуля связи. Однако формат передаваемых команд и данных не должен зависеть от протокола. Кроме того, этот формат должен совпадать с форматом передачи данных между ТМ и web-сервисом, находящимся на СУ, и web-сервисом и ПВУ, находящейся на ВУ. Команды и данные должны передаваться в виде АТ-строк для их удобной интерпретации.

2 Проектный раздел

Система разрабатывается как распределенная, децентрализованная система. Локальные контроллеры нижнего уровня, непосредственно управляющие ОКУ, работают абсолютно автономно. При этом в системе существует центр управления, позволяющий осуществлять общий мониторинг и управление всеми ОКУ. Выход из строя или потеря связи с этим центром никак не влияют на работу локальных контроллеров, непосредственно управляющих ОКУ. В качестве центра управления выступает программа верхнего уровня, однако частично управление локальными контроллерами берет на себя главный контроллер [7].

Причины построения системы по распределенному принципу приведены на рисунке 11.

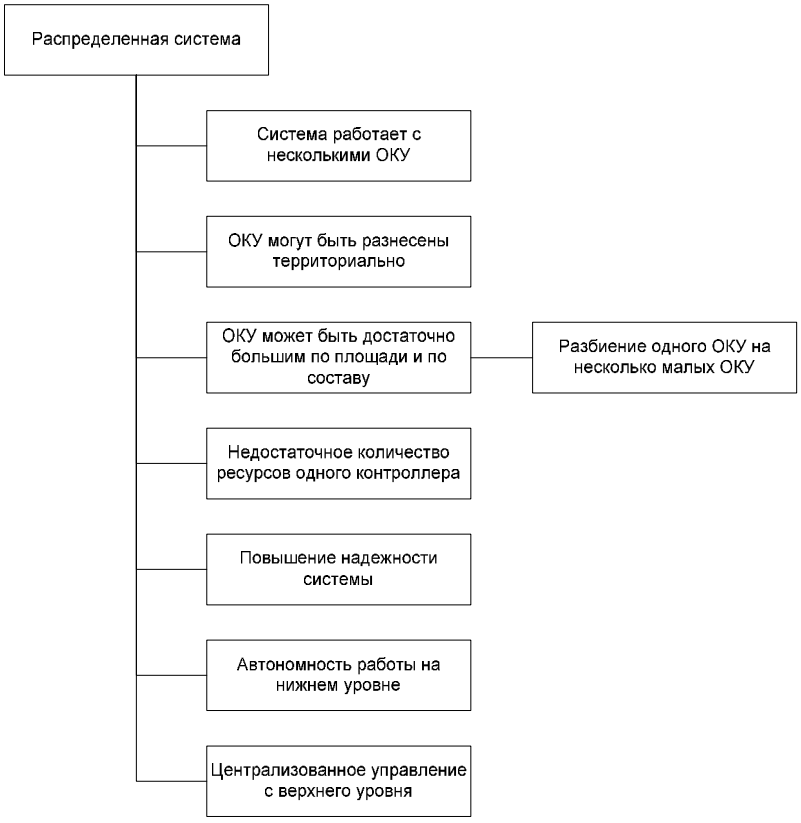


Рисунок 11 – Причины построения системы по принципу распределенности

Второй основной принцип, заложенный в разрабатываемую систему - распределение ее функций по уровням. Детализируем идею разделения системы на уровни в виде схемы уровней, представленной на рисунке 12.

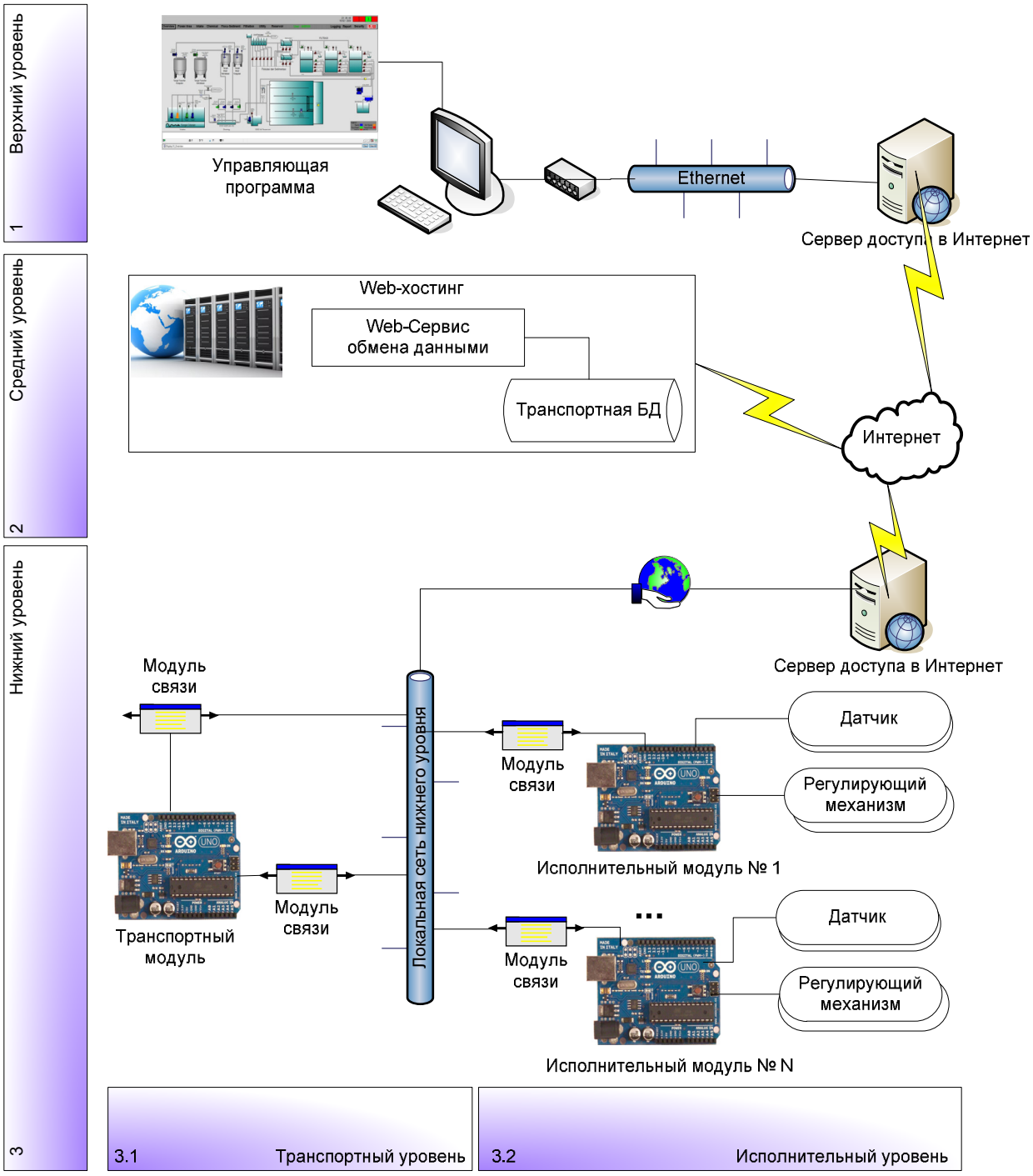


Рисунок 12 – Детализация разбиения системы на уровни

На верхнем уровне в нашем проекте разрабатывается программа верхнего уровня, которая через web-сервис передает команды на нижний уровень и получает от него данные, хранит их в базе данных, организует просмотр этих данных в удобном для пользователя виде.

В рамках работы над средним уровнем будет разработан web-сервис обмена данными между верхним и нижним уровнем.

Разработка программ верхнего и среднего уровня будет вестись на языке программирования C# в среде MS Visual Studio. База данных будут разработана в формате MS SQL Server.

В рамках работы над нижним уровнем требуется разработать целый комплекс: транспортный модуль и его резидентную программу, исполнительный модуль и его резидентную программу, обмен данными между всеми модулями в локальной сети нижнего уровня, по любым каналам и интерфейсам передачи команд и данных, единый формат передачи команд и данных.

2.1 Проектирование функциональной модели верхнего уровня

2.1.1 Построение логической модели верхнего уровня

Диаграмма вариантов использования применяется для отображения основных актеров, которые применяют участие в работе программного продукта, а также их основные функции – варианты использования программного продукта каждым актером.

Для оператора (пользователя) системы вся система состоит фактически только из одного уровня - верхнего. Интерфейсно он может общаться только с программой, управление датчиками контроля, поэтому с точки зрения оператора функции системы - это функции программы верхнего уровня:

- конфигурирование устройств (микроконтроллеров) нижнего уровня;
- конфигурирование мест установки устройств;

– конфигурирование датчиков и регулирующих механизмов каждого устройства;
– просмотр показаний с датчиков;
– ведение журнала сообщений (показаний, событий) пришедших от устройств нижнего уровня;
– ручная отправка запросов на передачу показаний с датчиков;
– ручная отправка команд на включение и выключение регулирующих механизмов;
– авторизация пользователя.

На рисунке 13 приведен пример диаграммы вариантов использования.

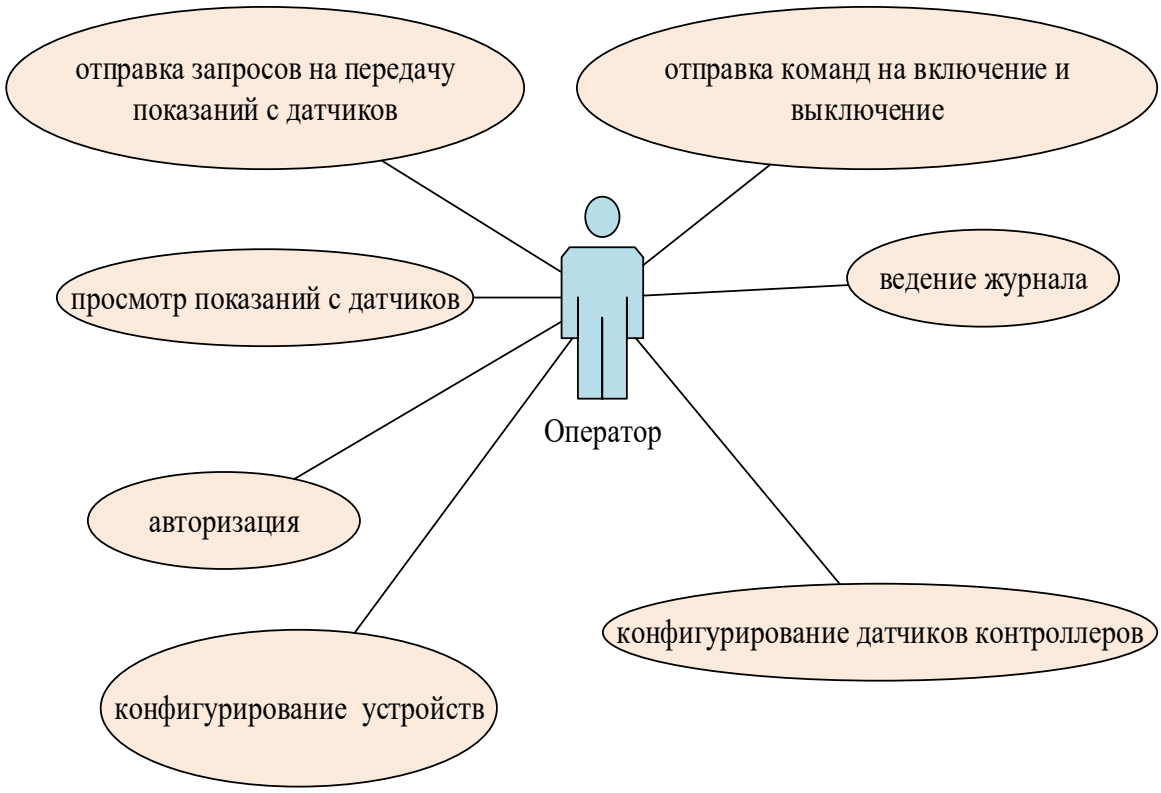


Рисунок 13 – Диаграмма вариантов использования

Для описания элементного аспекта предметной области выполним создание диаграмм классов представлена рисунке 14.

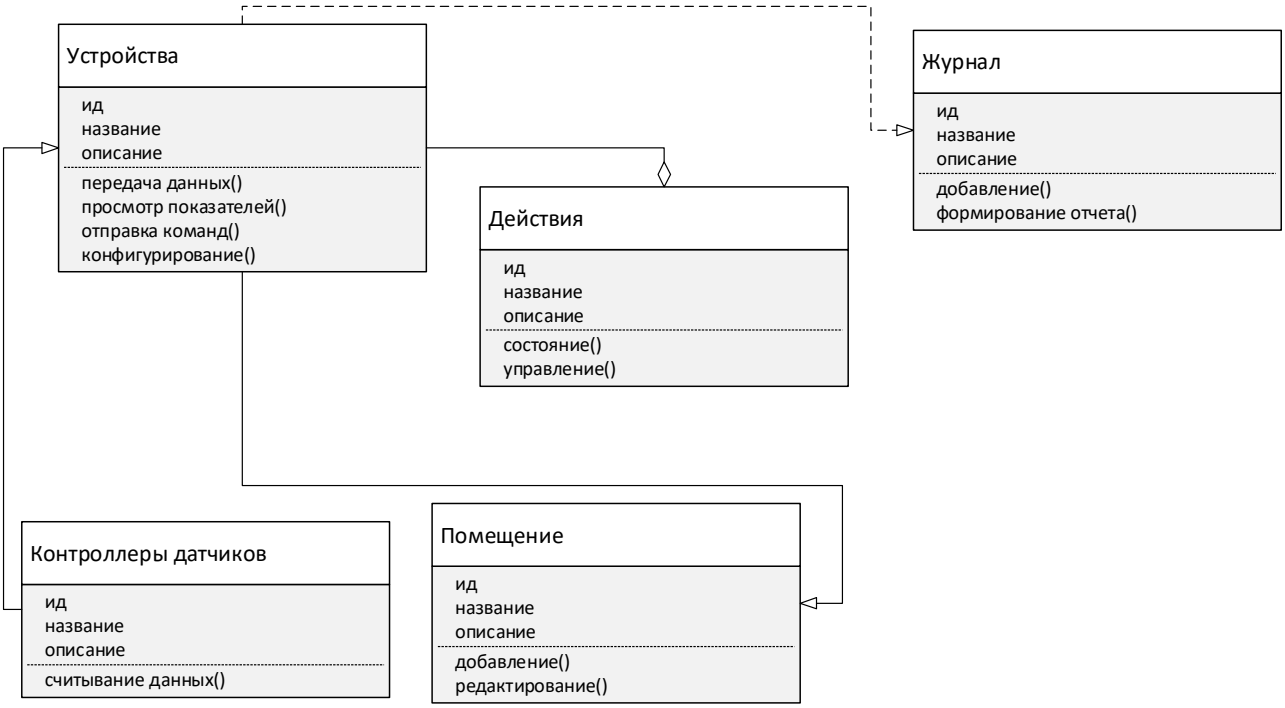


Рисунок 14 – Диаграмма классов

2.1.2 Построение логической модели базы данных

После анализа исходных данных надо выделить совокупность используемых объектов базы данных:

– журнал;

– помещение;

– контроллеры(датчики);

– устройства;

– команды;

– действия.

Опишем далее атрибутный состав для указанных выше объектов в таблице 2.

Таблица 2 – Описание атрибутов

Название	Тип	Ключевое поле
Журнал		
Код журнала	Число	Да
Код действия	Число	Да
Дата	Дата	
Время выполнения	Дата	
Наименование	Строка (текст)	
Описание	Строка (текст)	
Действия		
Код действия	Число	Да
Код устройства	Число	Да
Код команды	Число	Да
Код контроллера	Число	Да
Дата	Строка (текст)	
Наименование	Строка (текст)	
Описание	Строка (текст)	
Результат	Число (от -128 до 127)	
Команды		
Код команды	Число	Да
Код устройства	Число	Да
Наименование	Строка (текст)	
Описание	Строка (текст)	
Код	Строка(длинна 2)	
Устройство		
Код устройства	Число	Да
Код контроллера	Число	Да
Наименование	Строка (текст)	
Описание	Строка (текст)	
Код	Строка (длинна 2)	
Контроллеры		
Код контроллера	Число	Да
Код помещения	Число	Да
Наименование	Строка (текст)	
Описание	Строка (текст)	
Код	Строка (длинна 2)	

Рассмотрим ER-диаграмму, показанную на рисунке 15 и опишем определенные связи между объектами.

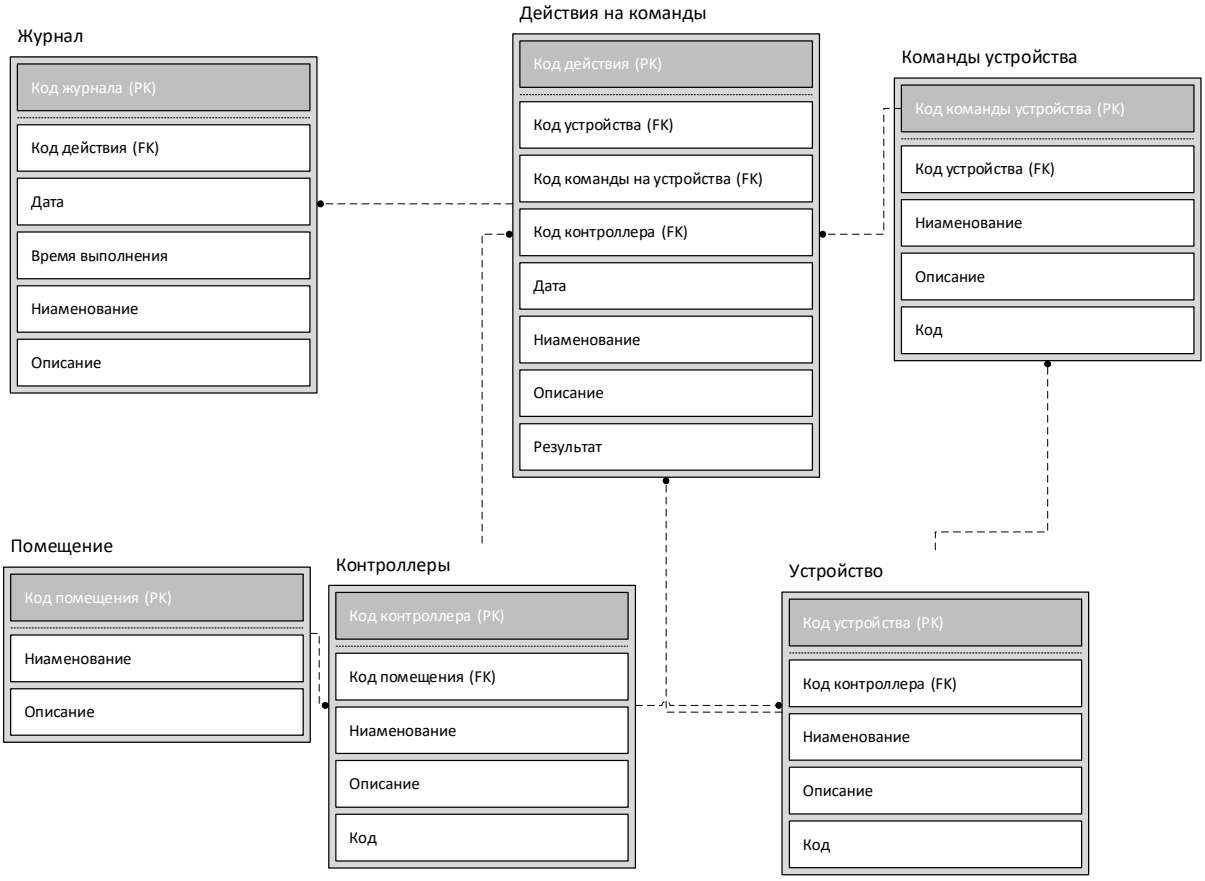


Рисунок 15 – ER-модель

Между объектами «Помещение» и «Контролеры» присутствует бинарная связь «один-ко-многим», так как одно помещение может иметь несколько контроллеров.

Между объектами «Контролеры» и «Устройство» есть связь «один-ко-многим», так как одно устройство может содержать в себе несколько контроллеров.

Между объектами «Устройство» и «Команды устройства» будет присутствовать связь «один-ко-многим», поскольку на одном устройстве можно запустить несколько команд.

Между объектами «Журнал» и «Действия» используется связь «один-ко-многим», так как один журнал может содержать несколько результатов действий.

Между объектами «Команды устройства» и «Действия» используется связь «один-ко-многим», так как одна команда может произвести несколько результатов действий.

Между объектами «Действия» и «Устройство» используется связь «один-ко-многим», так как одно устройство может произвести несколько действий.

Между объектами «Действия» и «Контроллеры» используется связь «один-ко-многим», так как один контроллер может произвести разные действия.

Проверим условия нормализации БД.

Для того, чтобы БД соответствовала 1НФ нужно, чтобы все реквизиты в ее таблицах были атомарными.

С рассмотренной диаграммы «сущность-связь» можно сделать вывод, что это условие выполняется для все разрабатываемых таблиц.

Для того, чтобы БД соответствовала 2НФ, нужно соответствие таблиц 1НФ и отсутствие аномалий ввода и удаления. Стоит отметить, что все таблицы соответствуют данному требованию.

БД соответствует 3НФ, если она соответствует 2НФ и для ее сущностей отсутствуют транзитивные зависимости. Указанное условие также выполняется, то есть, БД соответствует 3НФ.

2.2 Требования к технической и программной архитектуре нижнего уровня

2.2.1 Требования к исполнительным модулям

На ИУ функционирует несколько исполнительных модулей. Исполнительный модуль – это аппаратно-программный модуль, выполненный на базе МК, с подключенными к нему датчиками и РМ, выполняющий мониторинг и управление одним ОКУ [22].

Исполнительные модули обеспечивают, во-первых, автономное функционирование ИМ, а во-вторых взаимодействие с ТУ. Количество ИМ на ИУ может достигать 99 единиц. Логика работы каждого ИМ, состав подключаемых к нему датчиков и РМ определяются задачами по мониторингу и управлению ОКУ, которым управляет данный ИМ, характером самого ОКУ. ИМ должен продолжить полноценную автономную работу по мониторингу и управлению ОКУ при потере связи между подуровнями ИУ и ТУ или ТУ и СУ.

ИМ реализуется при помощи плат «Arduino UNO», «Arduino Nano» или «Aduino Mega». К плате подключены датчики, которые снимают показания состояния ОКУ и окружающей его среды. Так как плата «Arduino» имеет в своем составе различные интерфейсы связи, это позволяет подключать к ней разнообразные датчики и считывать с них показатели по различным распространенным интерфейсам. К плате «Arduino» подключены различные РМ, которые позволяют влиять на работу ОКУ или на параметры окружающей среды, в которой функционирует ОКУ. Как правило, в качестве РМ выступает реле, которое включает регулирующие и реализующие устройства и механизмы. Эти устройства, как правило, являются законченными механизмами, работающими по собственной логике от сети 220В. В большинстве случаев ИМ может только включить или выключить это устройство. Однако, если устройство имеет возможность управления им

при помощи интерфейсов, поддерживаемых платой «Arduino», достаточно включить в состав команд данного ИМ команды управления устройством и ИМ получит возможность управлять данным устройством [23].

ИМ в своем составе обязательно должен иметь хотя бы один модуль связи (МС) для выхода в локальную сеть нижнего уровня, реализующий один из четырех интерфейсов: Ethernet, Wi-Fi, RS-485, Bluetooth.

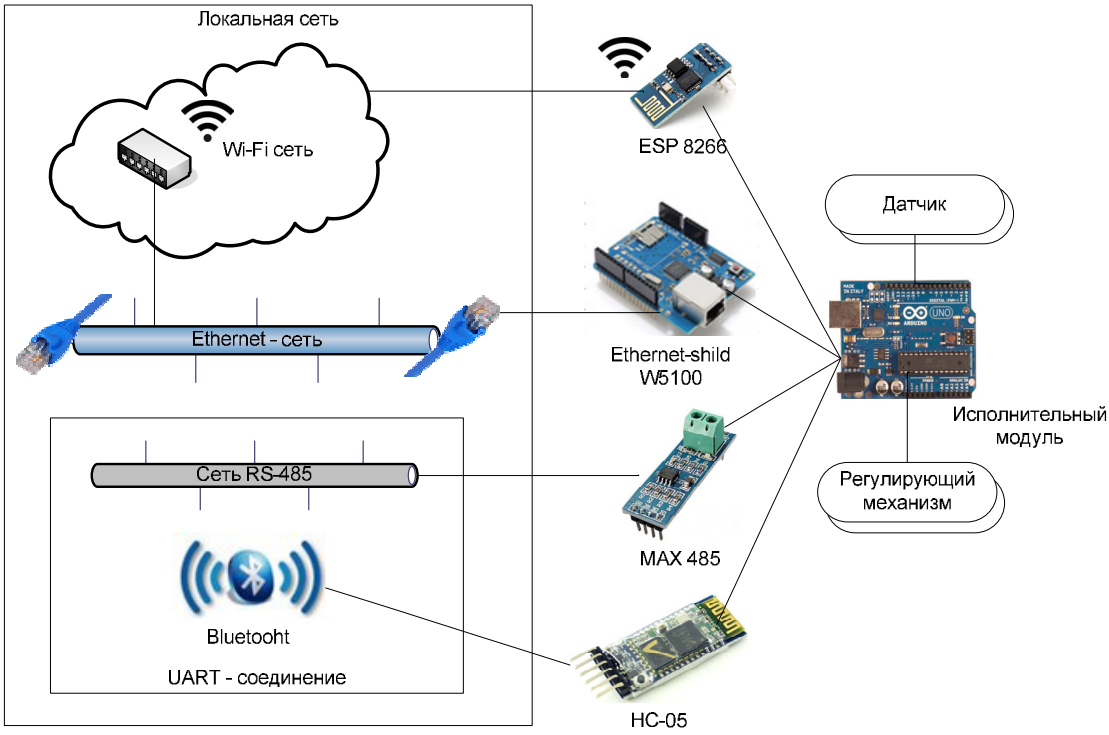


Рисунок 16 - Варианты реализации модуля связи

Варианты устройств, которые могут использоваться в качестве МС, приведены на рисунке 16.

2.2.2 Требования к транспортному модулю

ТМ реализуется на плате «Arduino UNO». К ТМ могут быть подключены несколько МС, реализующие различные интерфейсы связи с ИМ. Один из МС, подключенных к ТМ, обязательно должен обеспечить подключение к локальной компьютерной сети и через это подключение соединяться с web-сервисом, который находится на СУ. Web-сервис может

функционировать на web-сервере в этой же локальной сети или где-то в сети Интернет. Соединение с локальной сетью, предоставляемое ТМ, должно иметь все необходимые разрешения для доступа к web-сервису на СУ. Например, если для этого нужно разрешение для выхода в сеть Интернет – это разрешение должно быть[10].

Транспортный модуль подключается к компьютерной локальной сети с двумя целями:

– Обеспечить связь с ИМ, подключающихся к ТМ через компьютерную локальную сеть;

– Обеспечить связь с web-сервисом, функционирующим на СУ.

Транспортный модуль подключается к компьютерной локальной сети одним из вариантов: при помощи платы расширения Ethernet Shield W5100 или при помощи микроконтроллера ESP 8266.

Если на НУ присутствуют ИМ, принимающие и передающие данные по интерфейсу UART, то ТМ должен иметь в своем составе МС, подключающийся к этому интерфейсу. Если для связи с ИМ используется сеть RS-485, то ТМ должен подключаться к ней при помощи преобразователя интерфейсов UART-RS-485, такого как MAX485. Если используется Bluetooth-соединение по ТМ должен иметь в своем составе Bluetooth-модуль [18][19].

Транспортный модуль в некоторых случаях может самостоятельно управлять исполнительными модулями. При получении данных от какого-либо ИМ транспортный модуль проверяет пришедшие данные по специальной таблице кросс-команд. Если пришедшая строка данных присутствует в кросс-таблице, то ТМ отправляет в локальную сеть нижнего уровня кросс-команду. Данный механизм позволяет управлять различными РМ, подключенных к некоторым ИМ при наступлении событий на других ИМ. Причем это управление происходит без участия среднего и верхнего уровней СКУ.

2.2.3 Локальная сеть нижнего уровня и варианты подключения к ней модулей

Локальная сеть нижнего уровня делится на две физических подсети: традиционная компьютерная локальная сеть и сеть RS-485.

Традиционная компьютерная локальная сеть реализуется при помощи стандартного коммутатора, маршрутизатора или роутера. Структура сети может быть различной, от самой простой сети, реализованной при помощи одного коммутатора, до доменной сети со сложной организацией, состоящей из сотен устройств. Физически компьютерная локальная сеть может быть реализована как проводная сеть Ethernet или беспроводная сеть стандарта Wi-Fi или оба этих вариант вместе.

Если исполнительный модуль подключается к компьютерной локальной сети при помощи проводного Ethernet-соединения, то в качестве модуля связи в этом случае выступает плата расширения для Arduino UNO/Mega - Ethernet Shield W5100. Этот вариант подключения приведен на рисунке 17.

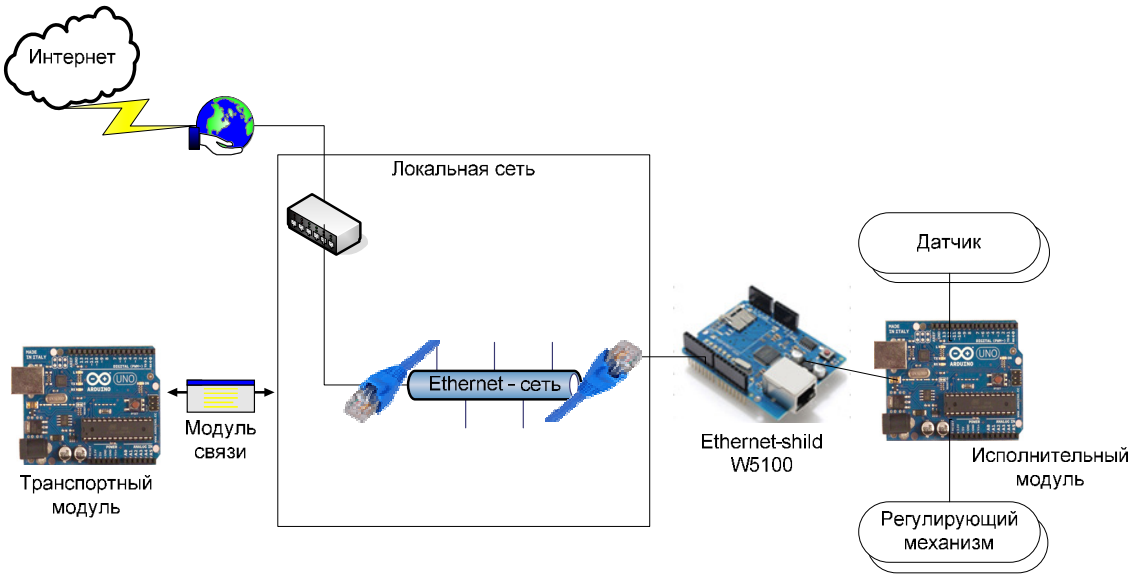


Рисунок 17 – Вариант подключения ИМ к локальной сети по Ethernet

Если исполнительный модуль подключается к компьютерной локальной сети при помощи беспроводного Wi-Fi-соединения, то в качестве модуля связи в этом случае выступает отдельный микроконтроллер ESP 8266. Этот вариант подключения приведен на рисунке 18Рисунок.

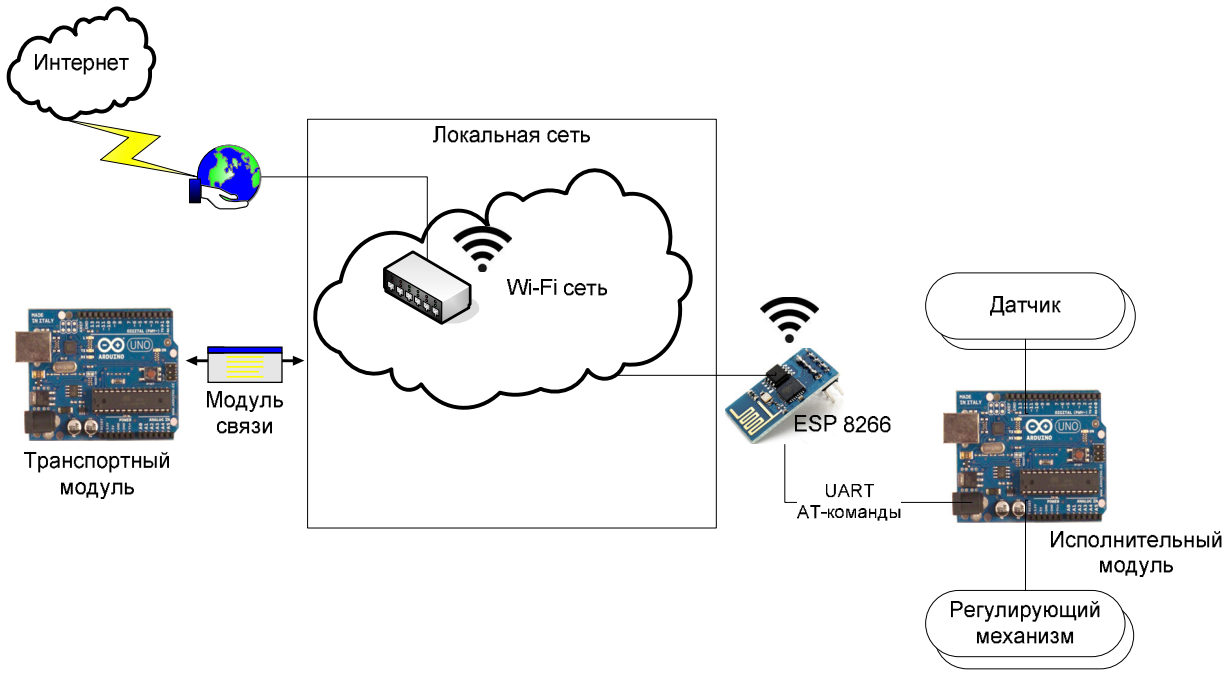


Рисунок 18 – Вариант подключения ИМ к локальной сети по Wi-Fi

Транспортный модуль подключается к компьютерной локальной сети одним из этих вариантов.

В любом случае исполнительный модуль при подключении к компьютерной локальной сети выступает в качестве клиента. Транспортный модуль должен обязательно на своей стороне обеспечить работу транспортного сервера, к которому подключаются ИМ-клиенты.

Взаимодействие ТМ и ИМ может быть выполнено и по последовательному интерфейсу. В этом случае ИМ подключаются к ТМ при помощи аппаратного интерфейса UART, который обязательно присутствует на любой плате Arduino. Для объединения ИМ и подключения их к ТМ

может использоваться стандарт RS-485. Аналогично варианту подключения к компьютерной локальной сети ИМ должен постоянно принимать команды от ТМ и отдавать данные в него по запросу или при наступлении определенных событий. ТМ на своей стороне должен обеспечить работу UART-сервера, к которому подключаются UART-клиенты. Этот вариант подключения приведен на рисунке 19.

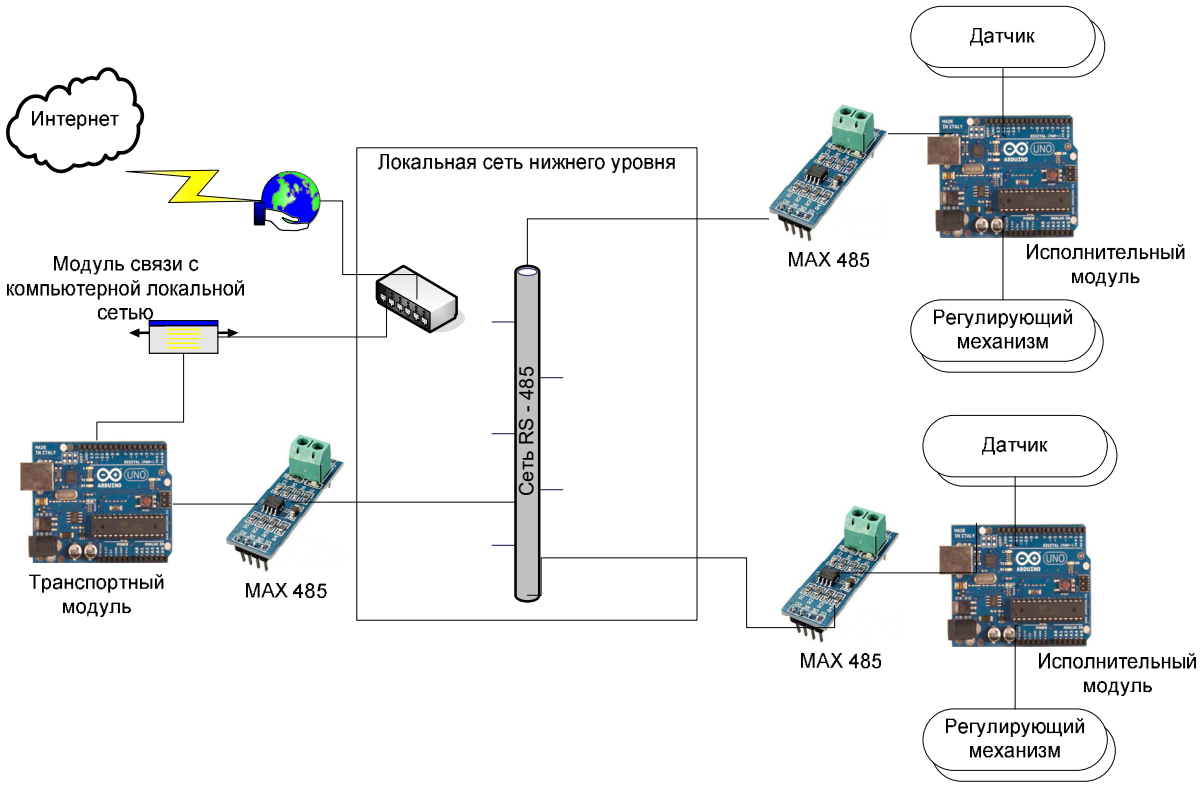


Рисунок 19 - Вариант подключения ИМ к локальной сети по RS-485

В качестве модуля связи в этом варианте выступает преобразователь интерфейса UART в RS-485, например, микросхема MAX-485. ТМ в данном варианте должен иметь как минимум два модуля связи: преобразователь UART - RS-485 для связи с ИМ и модуль связи для с компьютерной локальной сетью для связи с web-сервисом, находящемся на СУ.

Одним из вариантов UART-соединения является соединение по протоколу Bluetooth. В этом случае в качестве модуля связи выступает Bluetooth-модуль HC-05 (06). Прием и передача данных осуществляется

абсолютно аналогично варианту подключения по RS-485, но сетью передачи данных является не сеть RS-485, а Bluetooth-соединение. В этом случае транспортному модулю также нужен модуль связи с компьютерной локальной сетью для связи с web-сервисом. Этот вариант подключения приведен на рисунке 20Рисунок Рисунок.

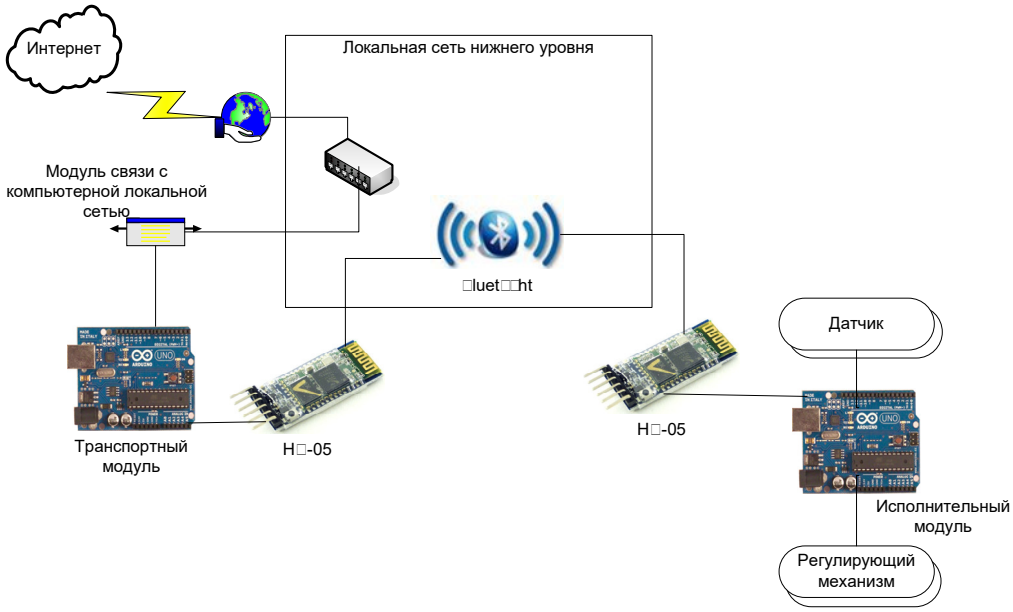


Рисунок 20 – Вариант подключения ИМ к локальной сети по Bluetooth

2.2.4 Клиент-серверная архитектура нижнего уровня

Взаимодействие ТМ и всех ИМ нижнего уровня построено по технологии клиент-сервер. Сервером выступает программное обеспечение ТМ. Клиентом выступает программное обеспечение ИМ. Схема клиент-серверной работы модулей нижнего уровня приведена на рисунке 21.

Одновременно транспортный модуль выступает клиентом по отношению к web-сервису среднего уровня.

Технология клиент-сервер выбрана не случайно. По условию СКУ должна стоится как распределенная система, а именно использование клиент-серверной технологии лучше всего подходит для этого [17].

На транспортном модуле работает два сервера:

– TCP-сервер для прослушивания соединений от TCP-клиентов, подключающихся к ТМ по компьютерной локальной сети с использованием Ethernet или Wi-Fi соединения;

– Serial-сервер (или UART-сервер) для прослушивания соединений от Serial-клиентов, подключающихся к ТМ по сети RS-485 или с использованием Bluetooth-соединения.

Кроме того, в программное обеспечение транспортного модуля включен TCP-клиент связи с web-сервисом среднего уровня СКУ.

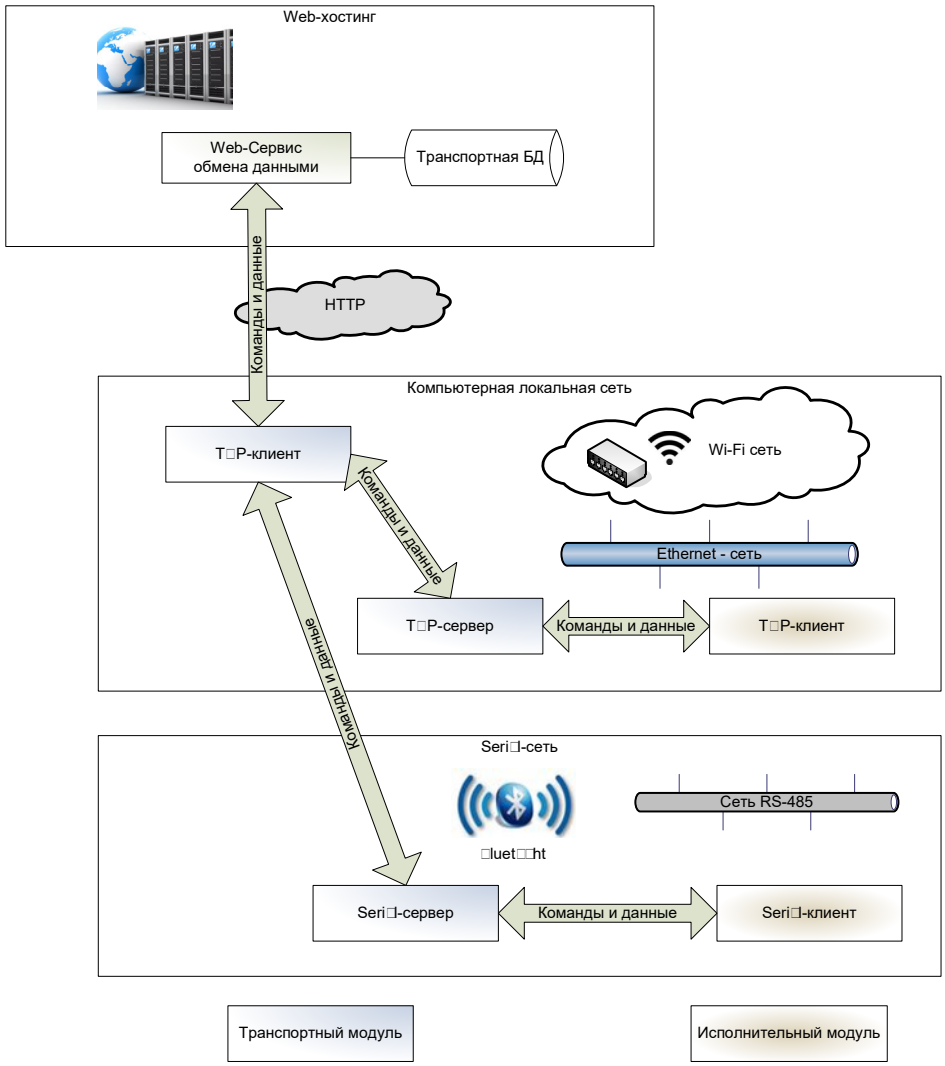


Рисунок 21 – Клиент-серверная система работы модулей нижнего уровня

TCP-сервер и TCP-клиент транспортного модуля реализуются при помощи программного обеспечения, которым организуется работа модуля связи транспортного модуля. Если модуль связи ТМ реализован при помощи платы расширения Ethernet W5100, то сервер и клиент создаются и работают при помощи библиотеки Ethernet.h. Если модуль связи ТМ реализован при помощи микроконтроллера ESP 8266, то сервер и клиент создаются и работают при помощи программного обеспечения этого микроконтроллера.

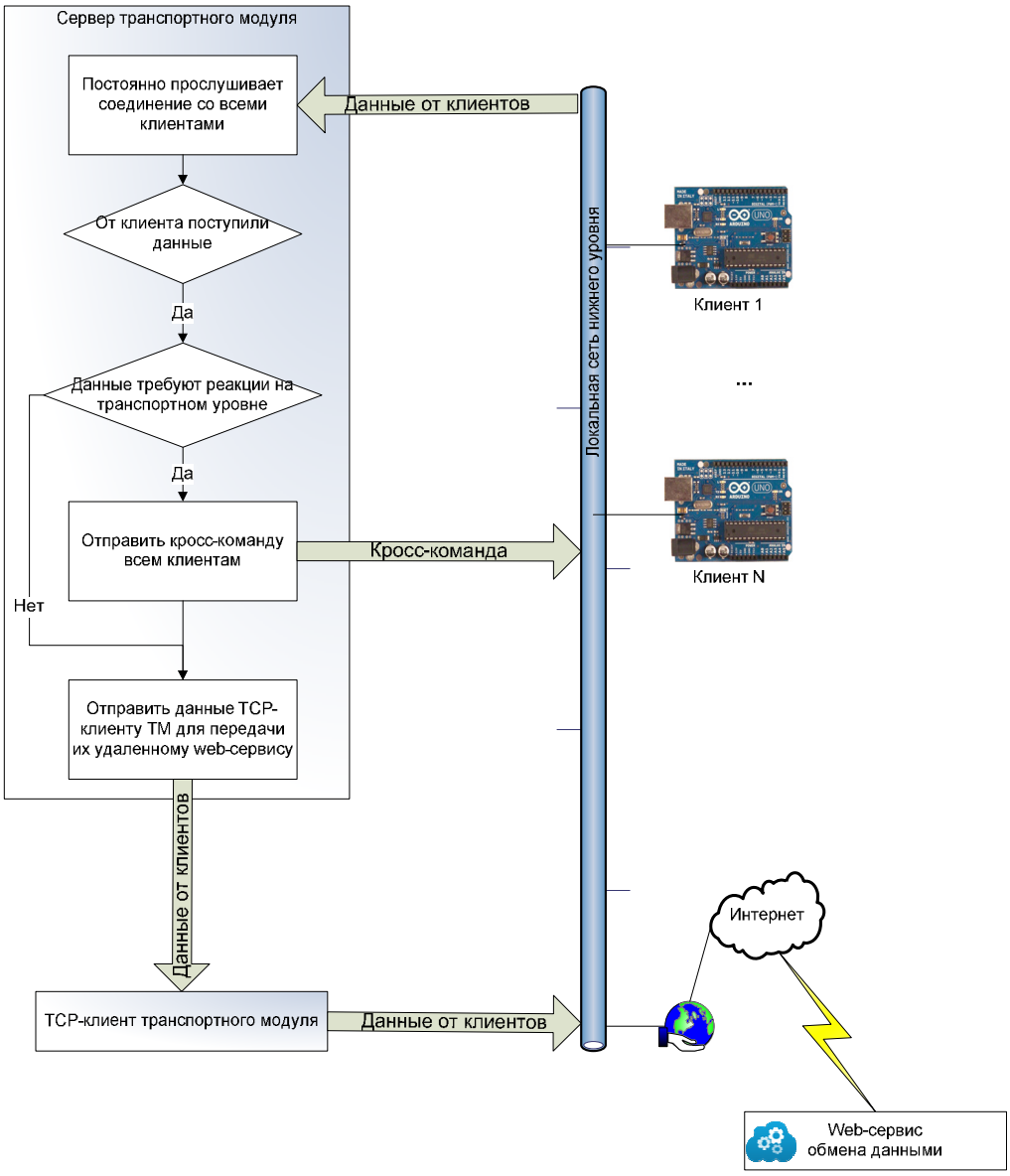


Рисунок 22 – Алгоритм работы сервера транспортного модуля

Для того чтобы запустить TCP-сервер, и TCP-клиент на транспортном модуле необходимо сначала подключиться к компьютерной локальной сети нижнего уровня. Это делается также при помощи программного обеспечения модуля связи ТМ.

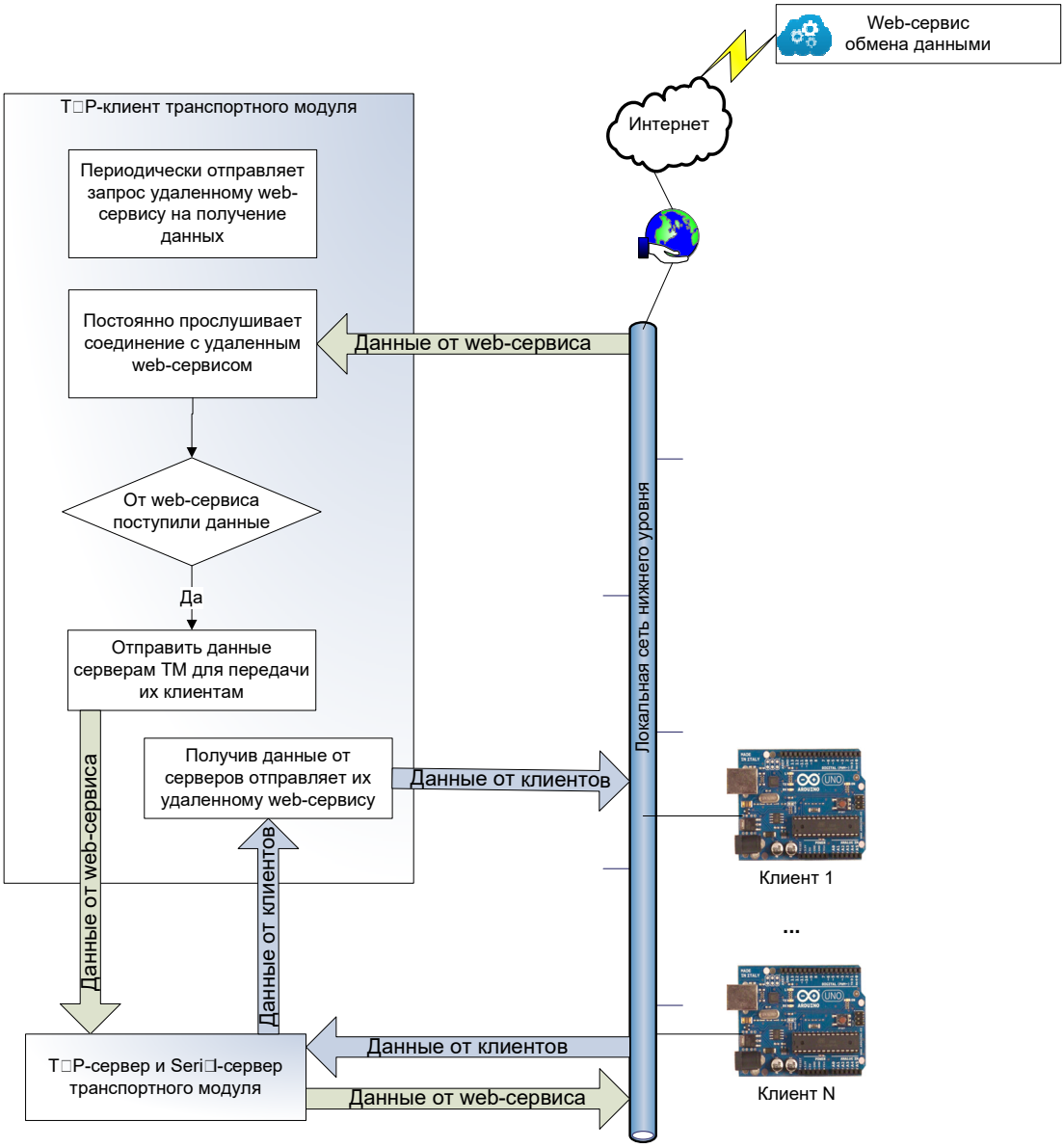


Рисунок 23 – Алгоритм работы TCP-клиента транспортного модуля

Serial-сервер транспортного модуля организуется при помощи подключения к последовательному UART-порту и работы с ним. Название

«Serial-сервер» и «Serial-клиент» в данной работе используются условно. Никаких реальных программных серверов и клиентов не создается, работа осуществляется при помощи стандартных средств чтения-записи в последовательный порт. Термины «сервер» и «клиент» используются для проведения параллели с работой TCP-сервера и TCP-клиента.

Независимо от способа организации сервер ТМ выполняет алгоритм, представленный на рисунке 22. NCP-клиент транспортного модуля выполняет алгоритм, приведенный на рисунке 23.

2.2.5 Подключение модулей по интерфейсу UART

Как уже говорилось выше локальная сеть нижнего уровня состоит из двух сетей: традиционной компьютерной сети и сети RS-485.

Топологию компьютерной локальной сети мы рассматривать не будем. Она может быть, как простейшей, так и сколь угодно сложной. Нам важно только то, что все подключающиеся к этой сети устройства получали необходимые разрешения, а пакеты данных доставлялись до адресатов. Протокол TCP/IP с этим благополучно справляется.

Сеть RS-485 организована по топологии «общая шина». Такое решение выбрано благодаря преимуществам такой топологии по сравнению с другими, например, радиальными топологиями. Преимущества топологии «общая шина» приведены на рисунке 24. Основные параметры интерфейса RS-485 приведены в таблице 3.

Таблица 3 – Параметры интерфейса RS-485

Параметр	Значение
Топология сети	Общая шина
Линия связи	Витая пара
Гальваническая развязка	Не оговаривается стандартом
Режим обмена данными	Полудуплекс
Способ передачи данных	Дифференциальные сигналы
Число абонентов сети	До 32 (можно увеличить за счет повторителей)
Максимальная длина линии связи	1200 м
Скорость передачи данных	10 Мбит/сек

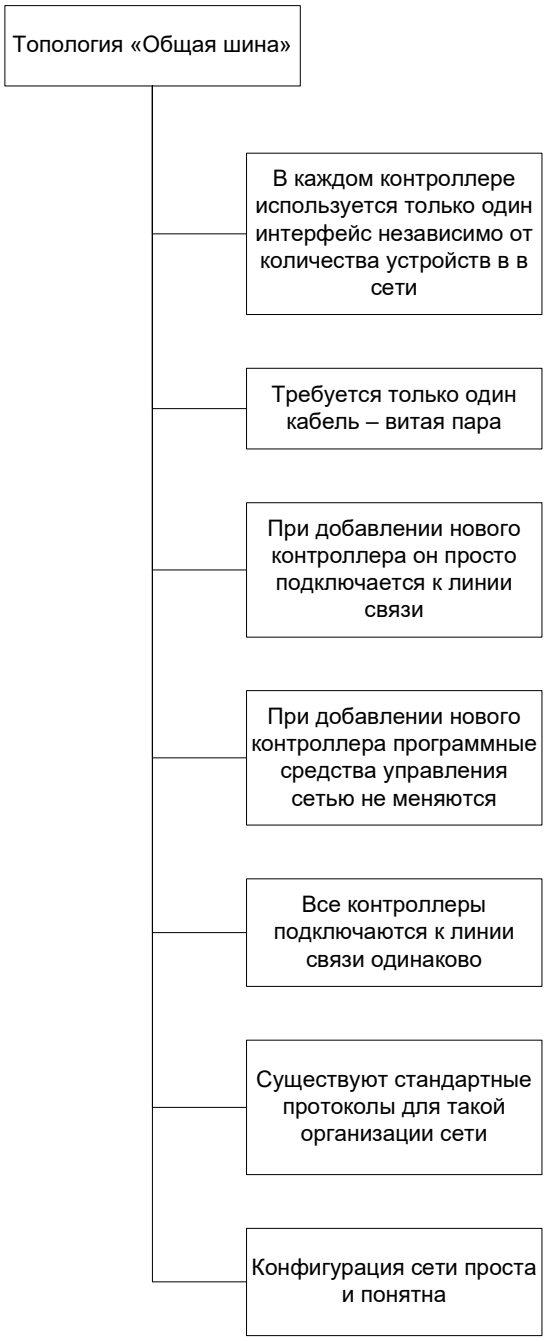


Рисунок 24 – Преимущества топологии «Общая шина»

Аппаратно интерфейс RS-485 реализуется при помощи дифференциальных передатчиков и приемников. Примером такого устройства (драйвера RS-485) является, например, микросхема MAX485. Схема подключения микросхемы к плате Arduino приведена на рисунке 25.

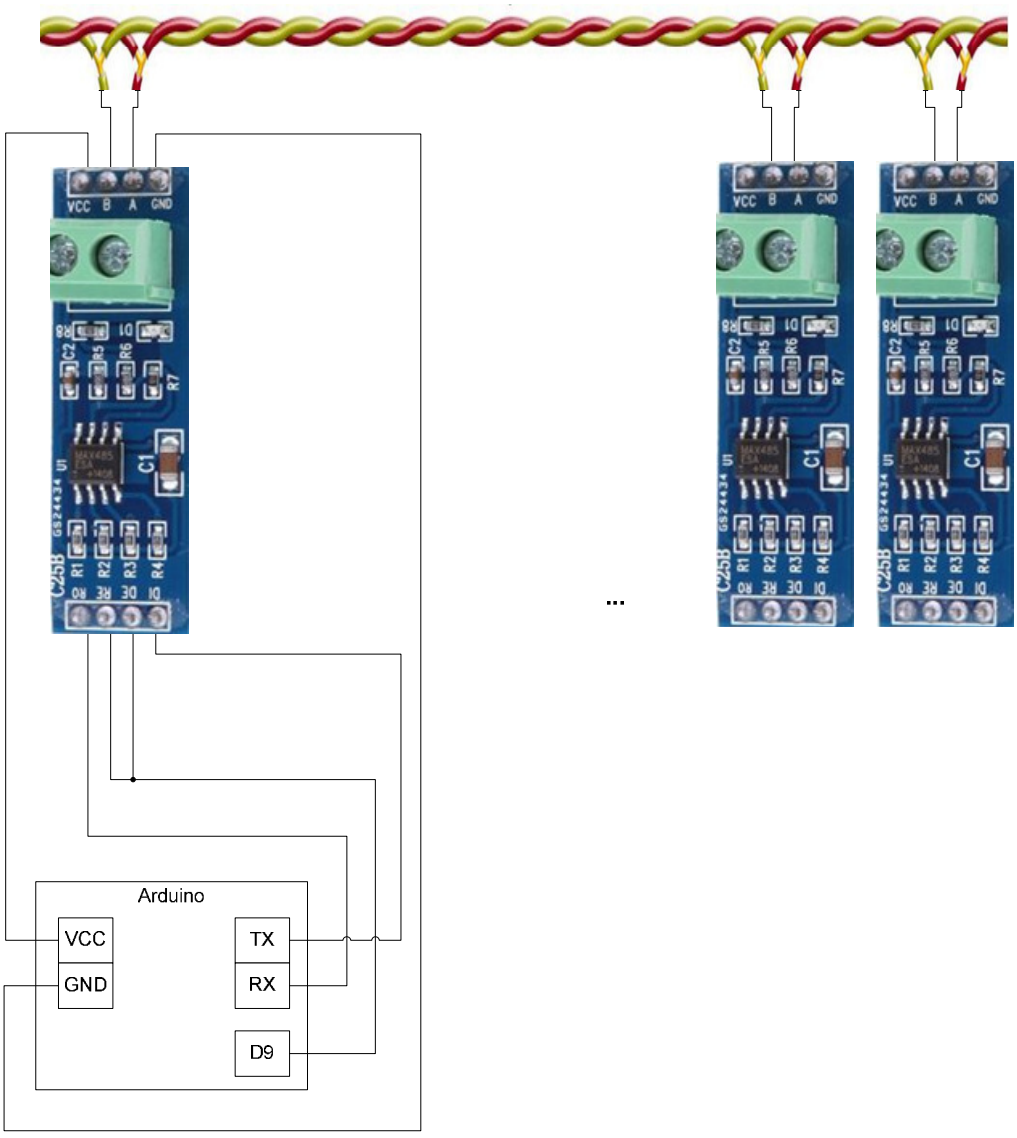


Рисунок 25 – Подключение устройств к шине RS-485 через драйвер MAX485

Здесь важно то, что все контроллеры подключаются к шине абсолютно одинаково. То же самое касается подключения любого нового устройства: никакой физической или программной перестройки сети делать не нужно, достаточно просто подключить новое устройство к шине аналогично уже действующим устройствам.

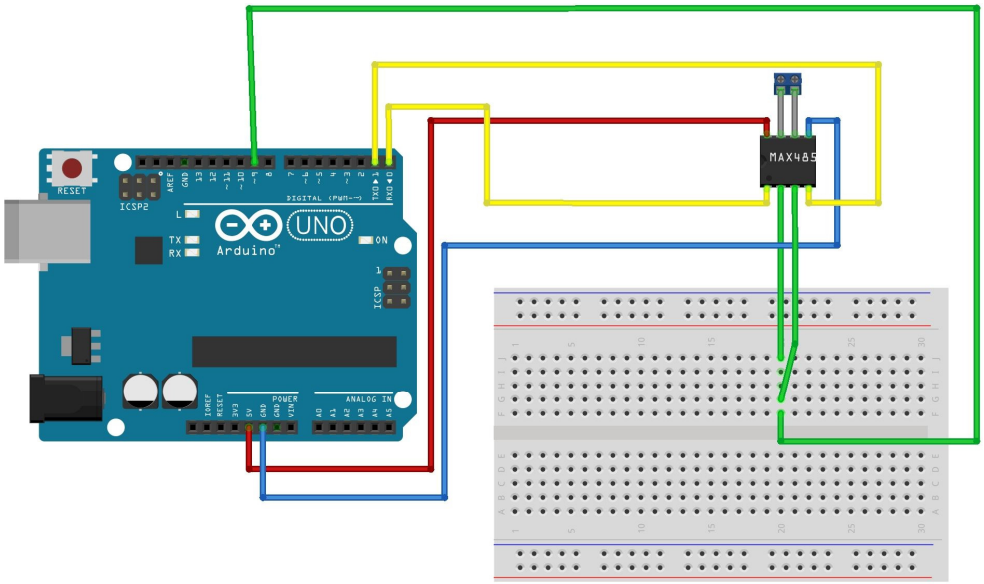


Рисунок 26 – Подключение драйвера MAX485 к плате Arduino UNO на макетной плате

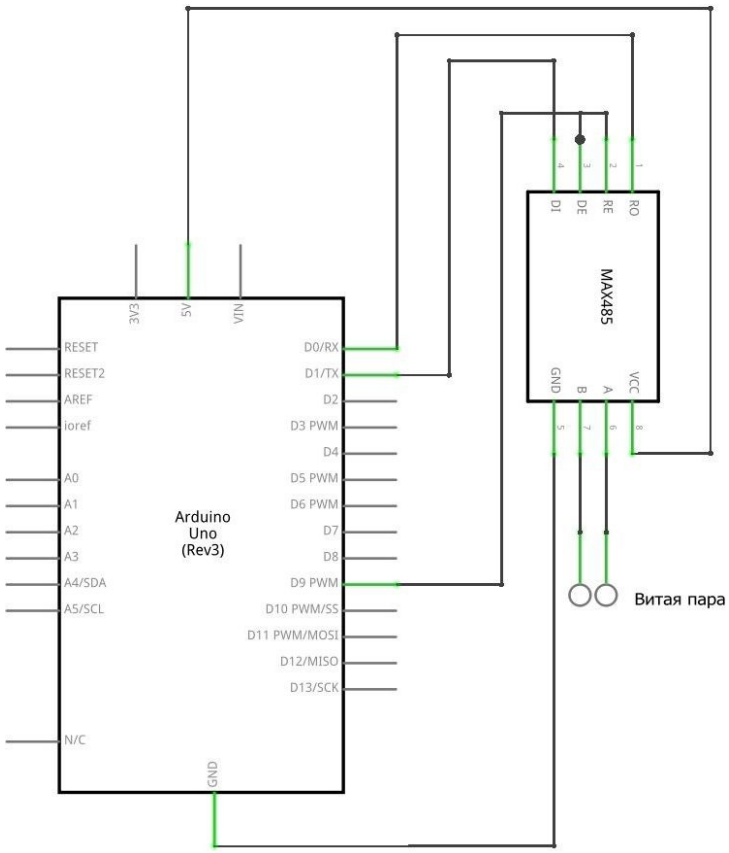


Рисунок 27 – Принципиальная схема подключения драйвера MAX485 к плате Arduino UNO

При подключении драйвера MAX485 к плате Arduino ее выходы DE и RE необходимо подключить к какому-либо выводу Arduino (на рисунке это

вывод D9) для управления включением/выключением передатчика и приемника.

Схема соединения драйвера MAX485 к плате Arduino UNO показана на рисунке 26, а принципиальная схема подключения – на рисунке 27.

2.3 Требования к технической архитектуре среднего уровня

2.3.1 Основные алгоритмы работы web-сервиса

Web-сервис, принимает команды и данные между программами верхнего уровня (ПК) и транспортным модулем нижнего уровня, реализованного на плате Arduino и подключенной к ней плате Ethernet Shield. Web-сервис представляет из себя универсальный транспорт, позволяющий передавать данные в обоих направлениях и взаимодействовать программам независимо от их местонахождения. Алгоритм работы web-сервиса представлен на рисунке 28.

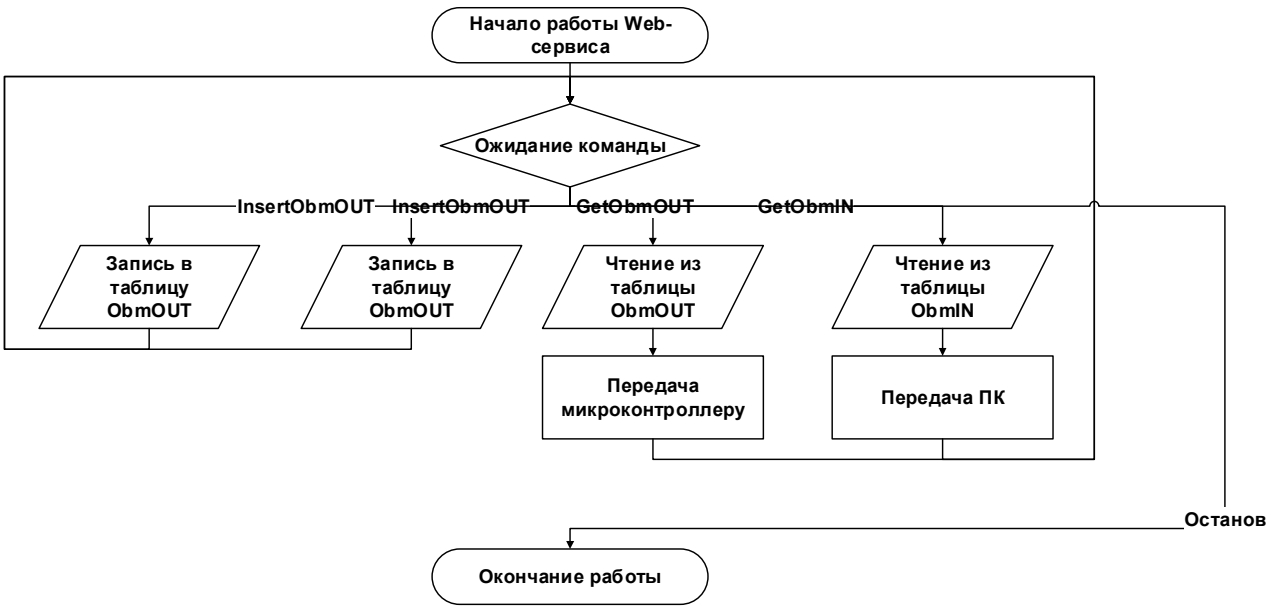


Рисунок 28 – Алгоритм работы web-сервиса

Web-сервис публикуется на каком-либо локальном или удаленном сервере. Это может быть web-сервер организации, бесплатный web-хостинг или web-сервер, функционирующий в локальной сети. Последний вариант целесообразно использовать для отладки системы. Размещение web-сервиса на каком-либо web-сервере в сети интернет дает возможность размещать точки сбора в любой точке мира [11, с. 201].

2.3.2 Обмен данными между web-сервисом и контроллером

Web-сервис обрабатывает четыре команды: получение данных/команд контролером, получение данных/команд ПК, запись данных/команд для контроллера и запись данных/команд для ПК.

Запрос от ПК для контроллера имеет вид:

«http://www.fsjoker.somee.com/Obm.svc/xml/InsertObmOUT?port=tert2&Data=AT0102031111», где InsertObmOUT название функции, AT0102031111 – собственно сама команда для контроллера. «AT» признак AT-команды, 01 - № ардуино, 02 - № устройства, 03 – код команды, 1111 – данные.

Запрос на получение команд для контролера имеет вид:

«http://www.fsjoker.somee.com/Obm.svc/xml/GetObmOUT», с помощью данного запроса главный контроллер получает команды для устройств.

Запрос от контроллера для ПК имеет вид:

«http://www.fsjoker.somee.com/Obm.svc/xml/InsertObmIN?port=tert2&Data=AT0102031111», где InsertObmIN название функции, AT0102031111 – собственно данные и от кого они. «AT» признак AT-команды, 01 - № ардуино, 02 - № устройства, 03 – код команды, 1111 – данные для ПК.

Запрос на получение данных/команд для ПК имеет вид:

«http://www.fsjoker.somee.com/Obm.svc/xml/GetObmIN», с помощью данного запроса ПК получает данные/команды от устройств.

3 Экспериментальный раздел

3.1 Обоснование выбора средств реализации

3.1.1 Обоснование и выбор технологии и платформы разработки программного обеспечения АИС

«Одним из самых «старых» и используемых языков программирования (ЯП) является «C++». В данном языке объекты представляются «классами». «Класс» расписывает, как именно функционируют объекты определенного типа, их «жизнедеятельность» от создания до удаления. Классы позволяют создавать программные модули более понятными» [7, с. 106].

««IDE «C++ Builder» — программный продукт, инструмент быстрой разработки приложений (RAD), интегрированная среда программирования (IDE), система, используемая программистами для разработки программного обеспечения на языках программирования Си и C++»» [10, с. 5].

««Eclipse» — свободная интегрированная среда разработки модульных кроссплатформенных приложений. Развивается и поддерживается «Eclipse Foundation»» [14][21].

Более новым языком, наследующим от C++ является детище фирмы Microsoft «C#» [1]. «C#» применяется для работы с библиотеками «.NET Framework» [17, с. 177]. Применение данного языка позволяет значительно ускорить разработку в среде программ Microsoft [5][15].

В связи с явными преимуществами средой разработки выбрана «Microsoft Visual Studio», а языком программирования – «C#».

Сравнение сред программирования представлены в таблице 4.

Задача СЭД не нуждается в сохранении гигантских объемов информации, так как организация не применяет большого объема используемой информации.

СУБД является объединенным множеством различных программ для возможности использования информации в различных БД множеством пользователей. С возможностью обеспечения целостности данных и их безопасного хранения. Наиболее известными СУБД в мире являются «MySQL» и «MS SQL Server» [6].

Таблица 4 – Сравнения сред программирования

IDE	Лицензирование	Кроссплатформенный	Применение отладчика	Разработка графического интерфейса	Автодополнение	Анализатор кода	Дерево классов
«RAD C++ Builder»	«проприетарная»	-	+	+	+	+	+
«Microsoft Visual Studio C#»	«проприетарная»	+	+	+	+	+	+
«Eclipse Java»	«EPL»	+	-	-	-	-	-

Для разработки выбрана СУБД «Microsoft SQL Server». Единый разработчик СУБД и среды разработки значительно облегчает совмещение БД и интерфейса работы с ней [4].

3.2.2 Обоснование и выбор выбранной платформы разработки

Опишем несколько предлагаемых для проекта контроллеров это Arduino или Raspberry Pi.

Микроконтроллеры могут одновременно исполнять всего одну задачу и отлично с этим справляются. А одноплатные компьютеры исполняют программы в рамках операционной системы (чаще всего Linux), обладают большей производительностью и широкими мультимедийными возможностями.

И в одном, и в другом лагере можно найти специализированные платы, которые сильно выделяются среди прочих какой-нибудь особенностью, но

сравнить возможности среднестатистических микроконтроллеров и компьютеров поможет таблица 5.

Таблица 5 – Сравнения микроконтроллеров

	Микроконтроллер Arduino	Одноплатный компьютер Raspberry PI
Производительность	Минус 1 ядро Малый объём оперативной памяти	Плюс 1 -4 ядер, большой объём оперативной памяти
Многозадачность	Минус Можно эмулировать.	Плюс Да.
Удобство работы с интернетом	Плюс Возможно подключить дополнительно сетевую плату.	Плюс Сетевой модуль присутствует.
Длительность работы от батареек	Плюс 0.33 Вт Возможны недели работы от батареек.	Минус 10 Вт Потребляет сотни-тысячи мА. Заряда большого аккумулятора хватит от силы на десяток часов.
Скорость реакции в проектах, критичных к времени	Плюс 100% контроль над временем и длительностью подачи сигналов.	Минус Из-за многозадачности критический процесс может проспать своё время.
Управление контактными цепями	Плюс Возможно.	Минус Невозможно.
Подключение аналоговых датчиков	Плюс Возможно.	Минус Только через дополнительные платы расширении.

Из таблицы 5 нужно понимать не только как сравнение конкретно платформ Raspberry PI и Arduino, но и в общем смысле проведения параллелей между одноплатными компьютерами и микроконтроллерами. Arduino, конечно, немного проигрывают в мощностях, но набирают свое за

счет экономного использования энергии и большого спектра подключаемых дополнительных плат и дешевизны.

Требование программной настройки функций и алгоритмов работы устройства приводит нас к необходимости использования программируемого микроконтроллера, который будет отвечать за всю работу устройства. В качестве такого контроллера выбран контроллер семейства «Arduino». Контроллер позволяет загружать в него программное обеспечение, написанное на языке высокого уровня «C++» в специальной среде разработке «Arduino IDE» [3] [8].

«Ардуино» (Arduino) это аппаратно-программный комплекс для создания простых систем автоматики и робототехники. «Ардуино» – это инструмент, предназначенный для создания систем плотно взаимодействующих с окружающей физической средой. Это платформа, реализующая идею «physical computing» с открытым программным кодом.

Программная часть платформы «Ардуино» состоит из интегрированной программной среды разработки (IDE), позволяющей создавать и компилировать программы, после чего загружать их в аппаратуру. Язык программирования устройств «Ардуино» основан на языке программирования «C/C++». Он прост в освоении, и на данный момент «Arduino» — это, пожалуй, самый удобный способ программирования устройств на микроконтроллерах [20].

Аппаратная часть платформы представлена электронной платой, главной частью которой является микроконтроллер, дополненный сопутствующими элементами (стабилизатор питания, кварцевый резонатор, блокировочные конденсаторы и т.п.), портом для связи с ПК, разъемами для сигналов ввода-вывода и т.п. Плата позволяет принимать сигналы от различных цифровых и аналоговых датчиков, которые могут быть подключены к нему, и управлять различными исполнительными устройствами [7].

Существует несколько версий платформ «Arduino». Самым распространенным вариантом является, пожалуй, «Arduino UNO R3» на рисунке 29. Технические характеристики приведены в таблице Таблица 6.

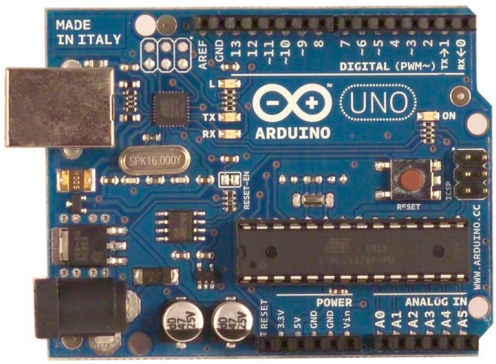


Рисунок 29 – Arduino UNO R3

Таблица 6 – Основные характеристики Arduino Uno R3

Тип микроконтроллера	ATmega328P
Напряжение питания микроконтроллера	5 В
Рекомендуемое напряжение питания платы	7 – 12 В
Предельно допустимое напряжение питания платы	6 – 20 В
Цифровые входы-выходы	14 (из них 6 поддерживают ШИМ)
Выходы ШИМ модуляции	6
Аналоговые входы	6
Постоянный ток через вход/выход	40 мА
Допустимый ток цифровых выходов	20 мА
Допустимый ток выхода 3,3 В	50 мА
Объем флэш памяти (FLASH)	32 кБ
Объем оперативной памяти (SRAM)	2 кБ
Объем энергонезависимой памяти (EEPROM)	1 кБ
Частота тактирования	16 мГц
Длина платы	68,6 мм
Ширина платы	53,4 мм
Вес	25 г

«Arduino Uno» построен на микроконтроллере «ATmega328». Платформа имеет 14 цифровых вход/выходов (6 из которых могут использоваться как выходы ШИМ), 6 аналоговых входов, кварцевый

генератор 16 МГц, разъем USB, силовой разъем, разъем ICSP и кнопку перезагрузки. Для работы необходимо подключить платформу к компьютеру посредством кабеля USB, либо подать питание при помощи адаптера AC/DC или батареи [4].

3.3 Физическое проектирование базы данных

Физическая модель данных строиться на основе логической модели. Результатом построения является физическая модель данных, содержащая всю информацию, которая необходима для определения всех объектов в базе данных.

Таблица 7 – Применение таблиц ИС

Таблица	Назначение таблицы
«Class»	Простые таблицы, состоят из полей наименование и описания
«ClassArr»	Характеристики для различных справочников
«ClassName»	Описание полей, созданных оператором и при создании БД простых таблиц
«Commands»	Возможные команды устройств
«Devices»	Устройства
«DevicesRez»	Возможные действия на команды с устройств
«Equip»	Контроллеры
«FieldsJornal»	Скрытые поля документов неотображаемые в шапке открытого в диалоговой форме документа
«JornalData»	Журнал полученных от устройств данных и команд
«Peoples»	Справочник «Физические лица»
«Prop»	Дополнительные свойства
«PropValue»	Значения дополнительных свойств справочников
«Rooms»	Помещения где расположены контроллеры
«SaveDocs»	Хранилище документов
«TypeDocs»	Типы документов
«Tree»	Структура папок для всех справочников
«Users»	Операторы ИС и их права доступа на работу с программой
«UsersAccess»	Права доступа пользователей

Использование таблиц БД показано в таблице 7. Состав таблиц БД и их связи указаны на рисунке 30.

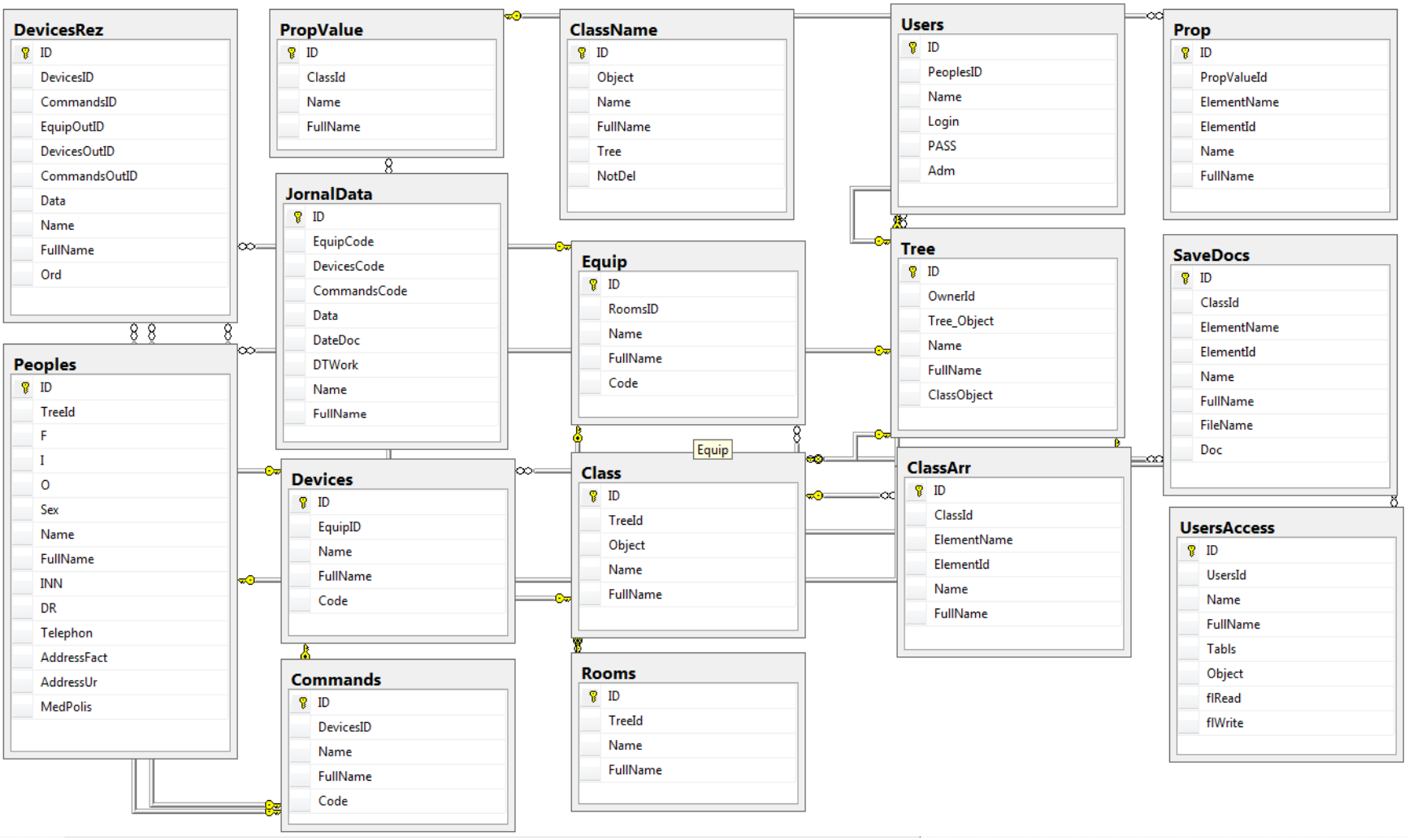


Рисунок 30 – Физическая модель данных

БД web-сервиса состоит из трех таблиц: «ObmIN», «ObmOUT», «JornalObm» они показаны на рисунке 31.

	Таблица	Столбец	Тип	Макс. Длина байт	М.б. NULL	ИД
1	JornalObm	ID	int	4	0	1
2	JornalObm	DT	datetime	8	1	0
3	JornalObm	Port	varchar	5	1	0
4	JornalObm	Name	varchar	255	1	0
5	JornalObm	FullName	varchar	255	1	0
6	JornalObm	Data	varchar	255	1	0
7	ObmIN	ID	int	4	0	1
8	ObmIN	DT	datetime	8	1	0
9	ObmIN	Port	varchar	5	1	0
10	ObmIN	Data	varchar	255	1	0
11	ObmOUT	ID	int	4	0	1
12	ObmOUT	DT	datetime	8	1	0
13	ObmOUT	Port	varchar	5	1	0
14	ObmOUT	Data	varchar	255	1	0

Рисунок 31 – Состав БД web-сервис

«ObmIN» содержит данные для ПК, «ObmOUT» содержит данные для контроллера, «JornalObm» хранит архив обменов. «ObmIN» содержит данные для ПК, «ObmOUT» содержит данные для контроллера, «JornalObm» хранит архив обменов.

3.4 Реализация аппаратного обеспечения

3.4.1 Реализация программы для контроллера исполнительного модуля

Резидентная программа для контроллера исполнительного модуля нижнего уровня системы написана на принципах объектно-ориентированного программирования. Все исполнительные устройства,

подключенные к контроллеру, объявляются, в программе как объекты соответствующих классов, которые также были разработаны в проекте для моделирования соответствующих устройств.

Все исполнительные устройства, подключаемые к исполнительному модулю, являются устройствами Led-типа для включения и выключения которых нужно подать сигнал высокого (5 В) или низкого (0 В) уровня. Простейшим примером является светодиод, но принципиального отличия между светодиодом и, например, сиреной, работающей от переменного напряжения 220В нет – сирена включается через реле при подаче на него «включающего» сигнала. Для моделирования устройств такого типа в программе разработан класс TLed, инкапсулирующий в себе все нюансы работы с Led-устройством. Так, например, если для включения устройства на него нужно подать инвертированный сигнал (низкого уровня), то это указывается в параметрах конструктора класса.

Библиотека MsTimer2.h, подключаемая к программе, содержит объявление класса и объекта класса MsTimer2, позволяющего организовать работу со вторым аппаратным таймером контроллера Arduino UNO. Задается период срабатывания таймера и процедура-обработчик срабатывания. В процедуре-обработчике выполняются следующие действия:

– Приращение счетчика программного таймера интервала приема данных из последовательного порта UART;

Благодаря использованию классов, основная программа контроллера становится совсем простой – всю «кухня» работы с устройствами инкапсулирована в методах этих классов. Основной программе остается лишь объявить объекты классов и вызывать соответствующие методы в соответствующих секциях программы.

Отдельно остановимся на программной реализации обмена данными с сервером по протоколу UART. Обмен по последовательному интерфейсу UART реализуется при помощи стандартной библиотеки Serial среды

разработки Arduino IDE. В ней объявлен объект Serial, позволяющий в параллельном процессе по прерыванию от контроллера UART организовать прием данных из последовательного порта, а при помощи метода write() этого класса – запись данных в порт.

Процедуры обмена помещены в отдельный модуль SerialNet.ino, включенном в проект. Процедура NetUpdate() вызывается в основном цикле loop() с частотой выполнения этого цикла для обеспечения непрерывного прослушивания порта. В случае обнаружения в порту данных программа проверяет их валидность: соответствие наличие стартовых символов, предназначенность пакета данных конкретно данному контроллеру и одному из устройств, подключенных к данному контроллеру. В случае успешной проверки вызывается процедура NetParseCommand(), выполняющая разбор команды. В зависимости от того, какому устройству предназначена команда вызывается процедура выполнения команды для этого устройства: NetCommandLed() – для устройств Led-типа. В зависимости от того, какая команда поступила от сервера вызывается тот или иной метод класса TLed. Это может быть включение, выключение или запуск «циклической» команды для Led-устройств. Список возможных команд приведен на рисунке 32.

```
// команды, которые приходят по сети для подключенных устройств и датчиков
#define NET_COMMAND_ON            1 // включить в т.ч. на заданное время
#define NET_COMMAND_OFF           2 // выключить
#define NET_COMMAND_BLINK         3 // мигать (включаться и выключаться) с заданным периодом
#define NET_COMMAND_QESTION       4 // состояние (вопрос или ответ или срабатывание)
```

Рисунок 32 – Список команд при обмене между сервером и контроллером

Резидентная программа контроллера исполнительного модуля состоит из следующих файлов:

Gl_ExchOptions.h – глобальные настройки обмена данными между программами верхнего и нижнего уровня системы;

ExchOptions.h – настройки данного конкретного исполнительного модуля: задание адреса платы Arduino, объявление подключенных к плате исполнительных устрйоств;

SerialClient.h – заголовочный файл проекта;

SerialClient.ino – основной файл проекта, функции setup() и loop();

SerialNet.h – заголовочный файл подпрограммы обмена данными с сервером, объявление констант;

SerialNet.ino – процедуры обмена данными с сервером.

3.4.2 Реализация программы для контроллера транспортного модуля

Резидентная программа контроллера транспортного модуля нижнего уровня разрабатываемой системы в основном работает с локальной сетью и удаленным сервером. Связь с ними осуществляется при помощи модуля Ethernet Shield W5100, подключенному к плате Arduino. Работа с Ethernet Shield W5100 осуществляется при помощи стандартной библиотеки Ethernet.h, входящей в программное обеспечение платформы Arduino. При этом Ардуино может выступать как в роли сервера, принимающего входящие соединения, так и клиентом, соединяющимся с удаленным сервером. Библиотека поддерживает до 4 одновременных соединений (входящих, исходящих, либо и тех, и других).

Библиотека содержит в себе набор классов (Ethernet, IPAddress, Server, Client) которые предоставляют достаточный набор методов для работы устройства в сети Ethernet, а также для подключения к удаленным серверам и работы с удаленными сервисами.

Сначала в программе создается клиент для работы в локальной Ethernet:

EthernetClient ENetClient;

Далее при помощи методов этого класса осуществляется подключение к локальной сети и работа с удаленным сервером. Если удалось осуществить

соединение, то дальше работа заключается в периодической отправке на удаленный сервер http-запроса «GetObmOUT» разработанного протокола обмена данными с web-сервисом и прослушивания от него ответа. Если ответ от удаленного сервера пришел, программа должна в нем обнаружить тэг <Data> и </Data>, в котором согласно протокола передаются команды и данные от web-сервиса исполнительным модулям нижнего уровня. Если такой тэг обнаружен, то его содержимое должно быть передано по последовательному интерфейсу UART всем исполнительным модулям, включенным в сеть модулей нижнего уровня системы.

Резидентная программа транспортного модуля состоит из следующих файлов: Adress.h - файл настроек подключения к локальной сети и удаленному серверу; WebClient.ino - основной файл, содержащий в себе функции setup() и loop() и остальные функции программы.

3.4.3 Реализация web-сервиса

Web-сервис является приложением службы WCF, которое размещается на IIS. Модули web-сервиса «WcfObmen» изображены на рисунке 33.

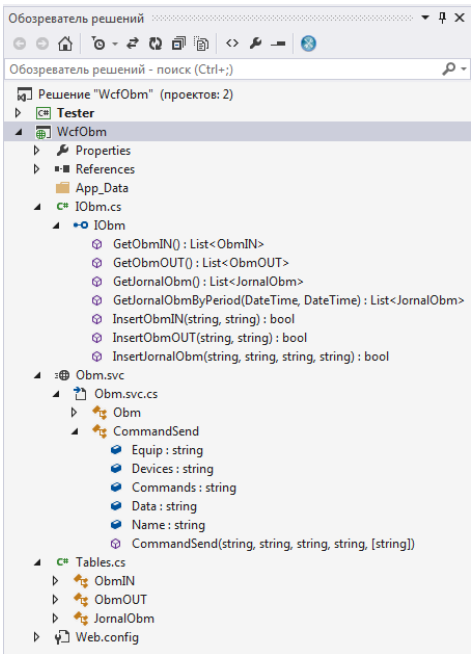


Рисунок 33 – Модули web-сервиса «WcfObmen»

Данные передаваемые между контроллером и программой верхнего уровня хранятся в БД. Описание классов таблиц представлено на рисунке 34.

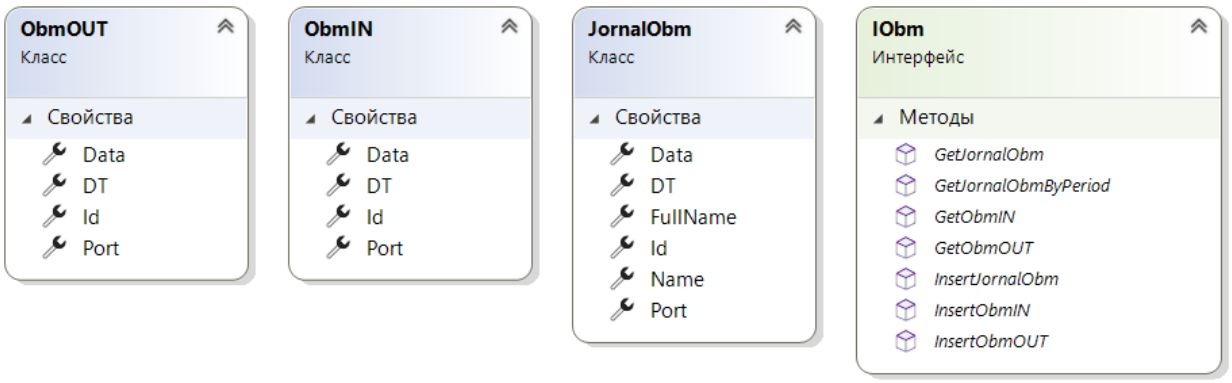


Рисунок 34 – Описание классов web-сервиса

Описание интерфейса функций, доступных извне производится в модуле «IObm.cs».

В модуле «Obm.cs» описываются сами функции, обеспечивающие обмен между контроллером и программой верхнего уровня. Web-сервис работает на сервере постоянно.

3.4.4 Реализация обмена данными с web-сервисом

Web-сервис, принимает команды и данные между программами верхнего уровня (ПК) и микроконтроллером (Ардуино). Универсальный транспорт, позволяющий передавать данные в обоих направлениях и взаимодействовать программам независимо от их местонахождения.

Для передачи команды в контролер выполняется команда:

«http://www.fsjoker.somee.com/Obm.svc/xml/InsertObmOUT?port=tert&Data=AT0102031», после чего в браузере отображается информация о выполнении, пример выполнения представлен на рисунке 35.

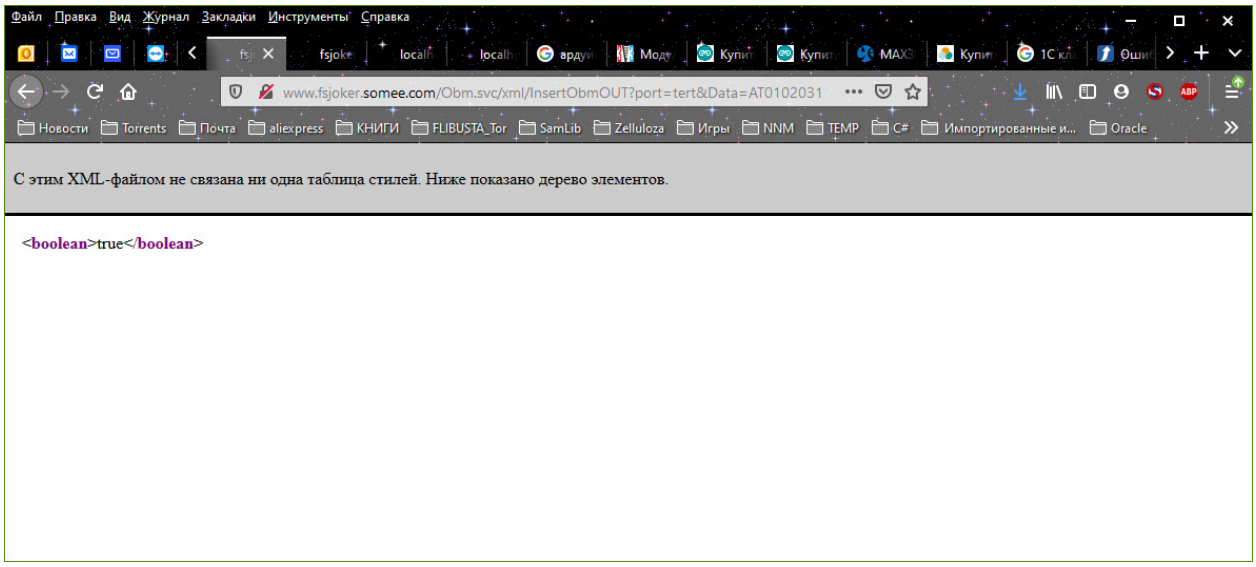


Рисунок 35 – Передача команды для контроллера

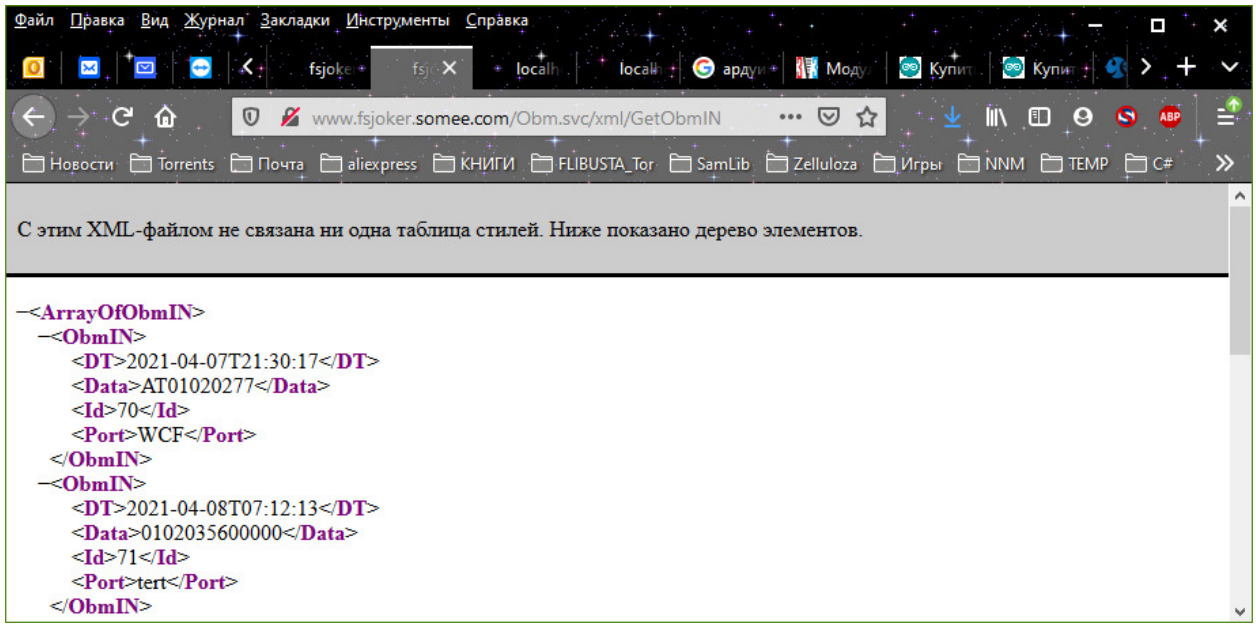


Рисунок 36 – Данные от контроллера

Получить данные можно выполнив запрос:

«http://www.fsjoker.somee.com/Obm.svc/xml/GetObmIN», после чего в браузере отображается Данный в формате XML, пример выполнения представлен на рисунке 36.

3.4.5 Реализация обмена данными между контроллерами на нижнем уровне

Реализация обмена данными по разработанному протоколу между транспортным модулем и исполнительными модулями нижнего уровня выполняется по интерфейсу UART.

Для работы с интерфейсом UART в Ардуино есть встроенный класс Serial. Использование методов этого класса значительно облегчает разработку процедур обмена данными по последовательному порту. Класс считывает данные с контроллера UART и записывает их в буфер. Это происходит по прерыванию от контроллера UART, незаметно для основной программы. Передача данных также происходит под управлением класса Serial. Для этого достаточно загрузить данные в программный буфер – дальше контроллер UART все сделает сам.

Из всех методов класса Serial в программе использованы следующие.

void begin(long speed)

Разрешает работу порта UART и задает скорость обмена в бод (бит в сек). Для задания скорости передачи данных рекомендуется использовать стандартные значения. В системе для обеспечения максимальной скорости обмена устанавливается значение 115200 бод.

print()

Передает данные через последовательный порт как ASCII текст.

println()

Передает данные через последовательное соединение как ASCII текст с следующим за ним символом переноса строки.

int available()

Функция получает количество байт (символов) доступных для чтения из порта UART. Это те байты, которые уже поступили и записаны в буфер последовательного порта. Буфер может хранить до 64 байт.

int read()

Считывает очередной доступный байт из буфера последовательного соединения.

Программа транспортного модуля получив от web-сервера обмена данными ответ на запрос команд и обнаружив в этом ответе команду заданного формата отправляет эту команду в последовательный порт. Ее получат все исполнительные модули сети контроллеров нижнего уровня.

Программа исполнительного модуля постоянно прослушивает последовательный порт. Обнаружив в нем команду заданного формата, программа определяет – адресована ли эта команда данному исполнительному модулю и одному из устройств, подключенных к данному исполнительному модулю и, если эта проверка прошла - выполняет данную команду.

3.4.6 Разработка программы верхнего уровня

В среде «Visual Studio C#» основной частью программного модуля является форма. Форма включает в себя элементы интерфейса оператора и программный код. «Visual Studio C#» формирует для формы класс, который включает в себя компоненты, находящиеся на форме, а также свойства и методы[16].

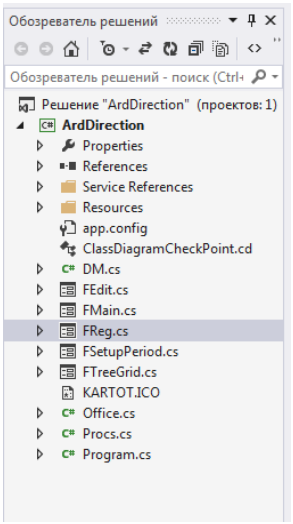


Рисунок 37 – Модули приложения

В разработанной программе имеется главная форма (ГФ) «FMain» - контейнер MDI, «FReg» - форма авторизации, «FTreeGrid» - форма отображения и редактирования таблиц, «FEdit» - форма редактирования данных в диалоговой форме и «FSetupPeriod» - форма для выбора периода отображения журнала. Разработанные модули представлены на рисунке 38.

Главная форма программы - «FMain», из нее вызываются все другие формы. Форма включает в себя главное меню и панель с кнопками «быстрого запуска». Макет главной формы представлен на рисунке 38.

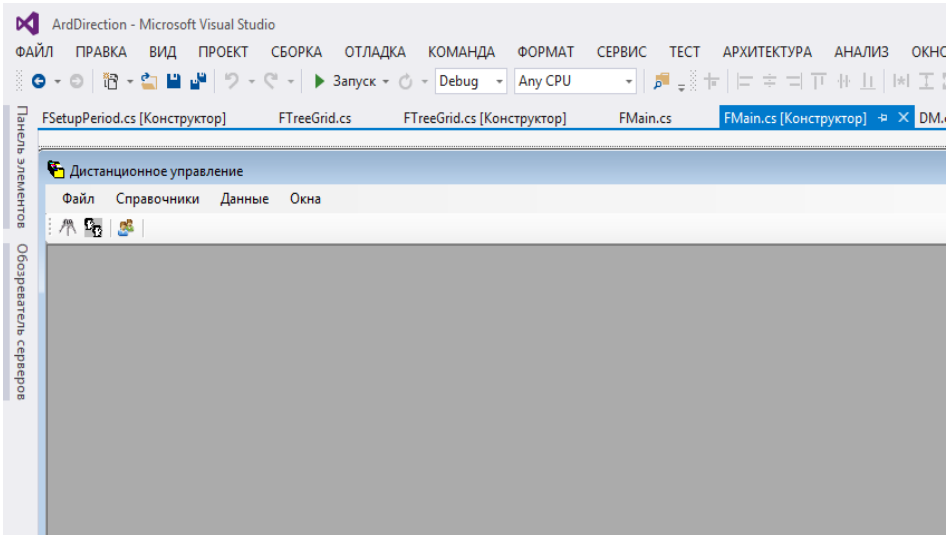


Рисунок 38 – Макет «FMain»

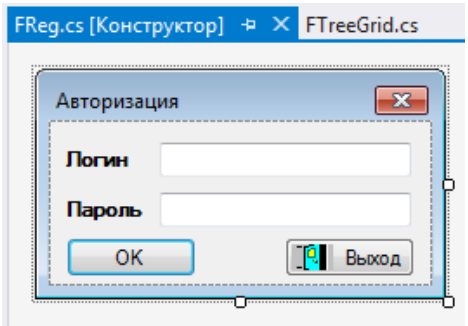


Рисунок 39 – Форма «FReg»

В форме авторизации пользователь вносит свои учетные данные для работы с ИС (по умолчанию 1, 1), модуль проверяет право доступа к ИС. Макет формы авторизации представлен на рисунке 39.

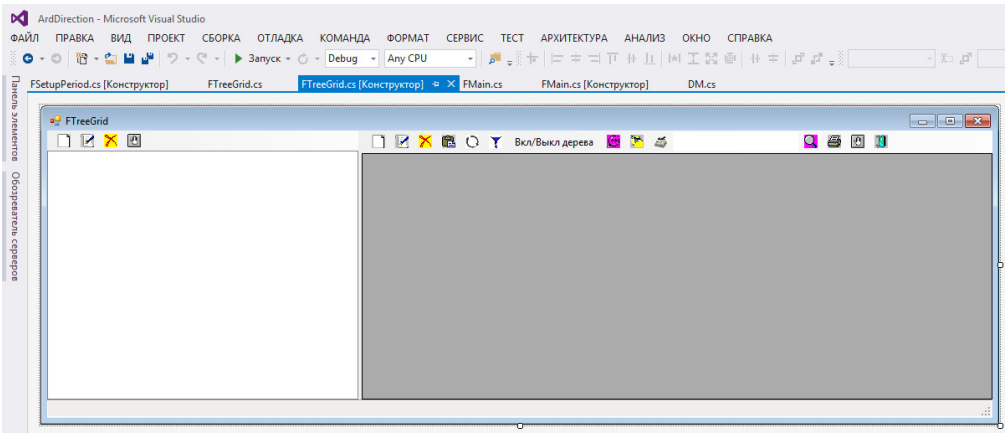


Рисунок 40 – Форма «FTreeGrid»

```
ссылок 1/9
public class ClassTable
{
    public string ClassObject;  //Для работы с таблицей Class если простая таблица имеет дерево

    public string BeginSQLText;         //текст запроса выполняющегося при загрузки формы (LOAD)

    public bool FlReadOnly;               // только для чтения
    public bool NotAllowUserToAddRows;  // ЗАПРЕТИТЬ добавлять новые строки в ГРИДЕ
    public bool NotAllowUserToDeleteRows;  // ЗАПРЕТИТЬ удалять строки в ГРИДЕ

    public bool FlTree;             // наличие дерева (FALS ЕСТЬ ДЕРЕВО)
    public bool FlSelect;           // true - форма для выбора
    public bool FlSelectTree;           // true - форма для выбора дерева
    public bool MinusTree;          // Настройка списка в ГРИДЕ весь или по группам (TRUE - весь)

    public string ConnectString;    // подключение к БД

    public string TextForm;         // заголовок формы
    public string TableName;        // название основной таблицы
    public string SQLText;          //текст запроса
    public string PrimaryKeyName;   // имя ключа таблицы

    public int LocateID;            // для позиционирования на элемент по ID
    public string LocateColumn;       // активный столбец после позиционирования
'/ ПОИСК
    public string FindStr;          // строка для поиск
    public string FindColumn;       // название столбца для поиска


    public string Pref;             // префикс таблицы ( C.TreeId - префикс C.)

    public string StrWhereGlobal;    // строка глобального условия (без where)
    public string StrWhereLocal;     // строка локального условия (без where)
    public string StrWhereTree;      // строка локального условия для вхождение в дерево
    public string StrWherePeriod;    // строка условия по периоду (DateDoc)
```

Рисунок 41 – Класс «ClassTable»

«FTreeGrid» является формой просмотра таблиц БД она представлена на рисунке 40. В качестве главного параметра, получаемого формой, является ссылка на класс «ClassTable», представленный на рисунке 41.

```
public ArrayList ArrInsertType;

// Массив с подчиненными, данной таблице, справочниками ...
public ArrayList ArrTables;
public ArrayList ArrTablesFK; //массив с именем внешнего ключа (ClientsID)
ссылок 0
public ClassTable()...
ссылок 15
public ClassTable(string sqltext)...
ссылок 61
public ClassTable(ClassTable Src)...

ссылок 0
public void Dispose()...
  public void SetField(string Name, string NameRus, int Width, bool flUpd, bool ReadOnly, bool Visible) ...
// записать поле в массив полей таблицы
ссылок 88
public void SetField(ClassField cf)...
// записать SQL DELETE (UPDATE) запрос в массив
ссылок 0
public void SetSQLDelete(string str)...
// записать SQL проверки перед удалением
ссылок 0
public void SetSQLCheckDelete(string str)...
// записать часть запроса SQL INSERT
ссылок 33
public void SetSQLInsert(string strName, string strValue, Type type)...
// очистить массив
ссылок 29
public void ClearSQLInsert()...

//удалить или обнулить ссылки на удаляемый элемент
ссылок 2
public bool SQLDeleteEx(int ID)...

//Проверка перед удалением,  true можно удалять
ссылка 1
public bool SQLCheckDelete(int ID)...
// Запись массива подчиненных таблиц
ссылок 13
public void SetTables(ClassTable ct, string FN)...
// Поиск поля типа ClassField по наименованию
ссылок 22
public ClassField FindField(string Name)...
// Поиск поля типа ClassField по NameIDOwner ( поиск подчиненного справочника)
ссылок 4
public ClassField FindFieldOnNameIDOwner(string NameIDOwner)...

ссылок 27
public int FindFieldNumber(string Name)...
```

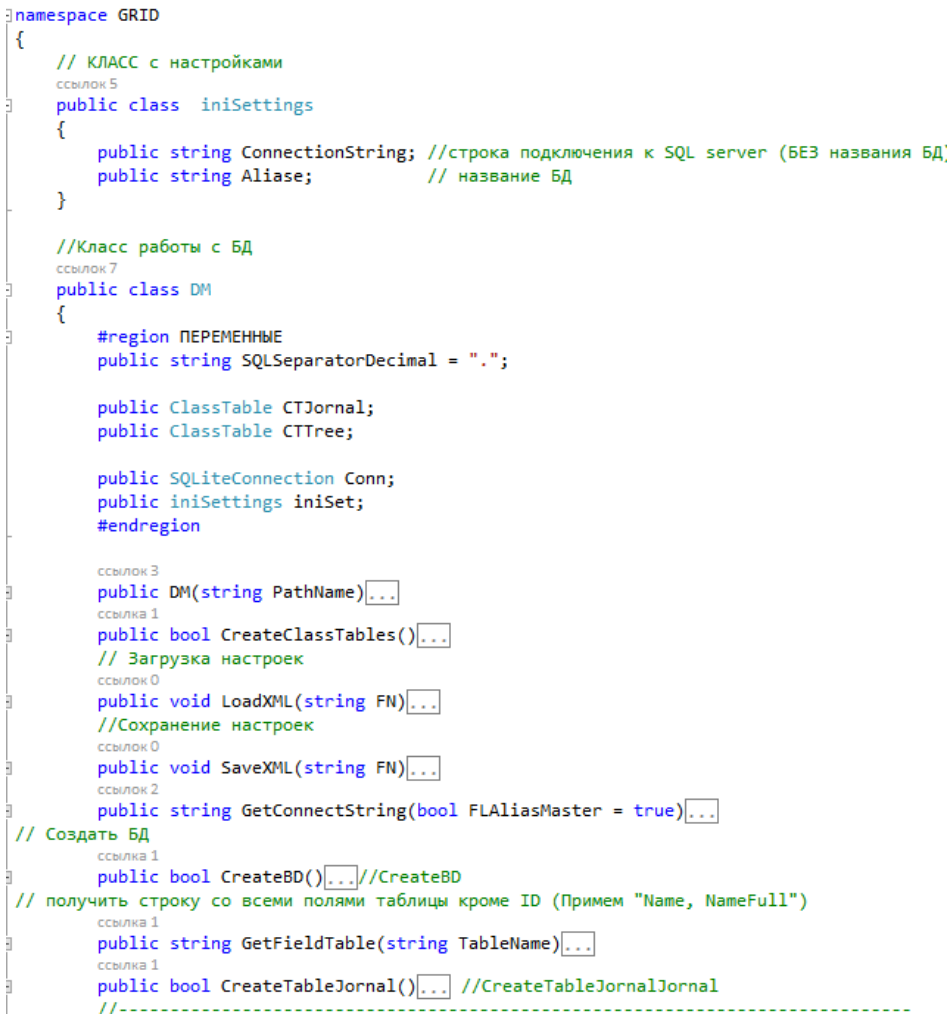
Рисунок 42 – Класс «ClassTable» продолжение

Класс «ClassTable» включает в себя полное описание таблицы, ее полей, а также содержит методы для работы с таблицами, представлен на рисунке 42.

Класс «ClassField» содержит описание полей таблицы, а также методы для работы с ними представлен на рисунке 43.

```
#region КЛАСС ПОЛЯ ТАБЛИЦЫ
ссылок 139
public class ClassField
{
    public string Name;          //название поля как в SQL - запросе
    public string NameRus;       //название поля на русском
    public int Width;          //ширина столбца
    public bool flUpd;          //признак изменения поля в БД
    public bool ReadOnly;       //ReadOnly для столбца DGV
    public bool Visible;        //видимость столбца

    public string ToolTipText;  // подсказка для поля

    public int Butt;           // наличие кнопки в колонке 1 - кнопка, 2- CheckBox

    public string NameIDEx;           // Поле в основной таблице для связи с подчиненной

    public string NameIDOwner;      // Код владельца если в одной форме ред-я 2-а связанных справочника
                                    // например шапка документа поля Организации и адреса ...
    public ClassTable ClTableEx;    // описание под-таблицы
    public Type type;
    public bool flInfField;         // true - информационное поле, отсутствует в таблице. не может изменять БД
                                    // используется для вывода информации или связи с владельцем ...

    public bool flNotPref; //ненужен префикс( таблица связана с основной через другую таблицу
    ссылок 0
    public ClassField(string Name,
                      string NameRus,
                      int Width,
                      bool flUpd,
                      bool ReadOnly,
                      bool Visible)...
    ссылок 88
    public ClassField()...

}
#endregion
```

Рисунок 43 – Класс «ClassField»



```
namespace GRID
{
    // КЛАСС с настройками
    ссылок 5
    public class  iniSettings
    {
        public string ConnectionString; //строка подключения к SQL server (БЕЗ названия БД)
        public string Aliase;           // название БД
    }

    //Класс работы с БД
    ссылок 7
    public class DM
    {
        #region ПЕРЕМЕННЫЕ
        public string SQLSeparatorDecimal = ".";

        public ClassTable CTJornal;
        public ClassTable CTTree;

        public SQLiteConnection Conn;
        public iniSettings iniSet;
        #endregion

        ссылок 3
        public DM(string PathName)...
        ссылка 1
        public bool CreateClassTables()...
        // Загрузка настроек
        ссылок 0
        public void LoadXML(string FN)...
        //Сохранение настроек
        ссылок 0
        public void SaveXML(string FN)...
        ссылок 2
        public string GetConnectString(bool FLAliasMaster = true)...
// Создать БД
        ссылка 1
        public bool CreateBD()...//CreateBD
// получить строку со всеми полями таблицы кроме ID (Примем "Name, NameFull")
        ссылка 1
        public string GetFieldTable(string TableName)...
        ссылка 1
        public bool CreateTableJornal()... //CreateTableJornalJornal
        //-----------------------------------------------------------------------------
```

Рисунок 44 – Модуль «DM»

В классе «DM» выполняется проверка наличия БД и таблиц и при необходимости создаются новые БД и таблицы. В классе «DM» выполняется выборка данных из таблиц БД и задается описание всех полей он представлен на рисунке 44.

Модель классов представлена на рисунке 45.

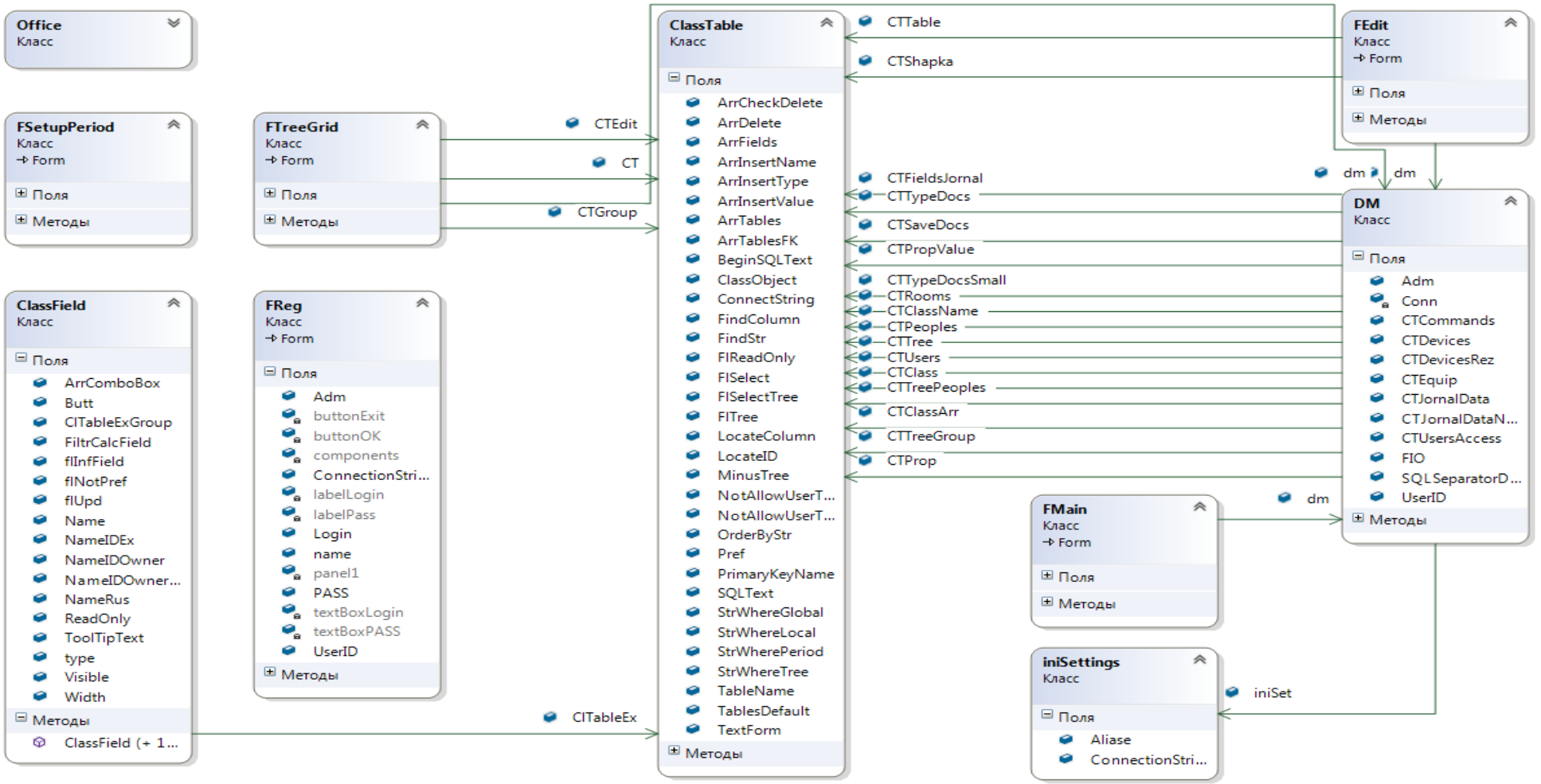


Рисунок 45 – Модуль классов

3.5 Тестирование и описание функциональности АИС

3.5.1 Инструкция по настройке и запуску транспортного модуля нижнего уровня

Транспортный модуль нижнего уровня предназначен для обмена данными между Web-сервисом Web-уровня и исполнительными модулями нижнего уровня.

Транспортный модуль нижнего уровня системы представляет из себя аппаратно-программный комплекс, состоящий из платы Arduino UNO R3, Ethernet-модуля и резидентной программы управления, реализующей протокол и логику обмена.

Обмен данными с web-сервисом производится по протоколу http, путем отправки http-запросов к web-сервису и получения от него ответов. Web-сервис может размещаться в локальной сети, на каком-либо ее сервере, а может быть опубликован в сети Интернет, например, на каком-либо бесплатном web-хостинге. В любом случае транспортный модуль обмена данными должен быть подключен к сети, чтобы получить доступ к web-сервису.

Выполнив физическое подключение Ethernet Shield к плате Arduino и подсоединив его к ближайшему коммутатору или роутеру локальной сети, необходимо записать в память платы Arduino транспортного уровня программу (предварительно произведя в ней специальные настройки) и запустить ее в работу. Программа находится в файле WebClient.ino.

Программные настройки соединения с локальной сетью и web-сервером прописаны в файле Adress.h проекта WebClient. Их нужно изменить в соответствии с настройками сети, в которой будет функционировать транспортный модуль.

Соединение с локальной сетью может происходить как с использованием сервера DHCP, так и без него. Если предполагается

использовать сервер DHCP, то нужно раскомментировать строку «//#define DHCP» в файле Adress.h. В этом случае Ethernet-модуль получит IP-адерс динамически. Если не предполагается использовать сервер DHCP и работать со статическим IP-адресом, то нужно оставить строку закомментированной. В этом варианте нужно прописать значения статического IP-адреса, адреса DNS-сервера, шлюза и маски подсети в константах EthrNet_IP, EthrNet_DNS, EthrNet_GATEWAY и EthrNet_SUBNET. В любом случае должны быть обеспечены все необходимые разрешения доступа Ethernet-модулю для подключения к серверу, на котором находится web-сервис. В частности, если web-сервис находится в сет Интернет должен быть разрешен выход в Интернет с этого IP-адреса.

За настройку расположения web-сервиса в сети Интернет или в локальной сети отвечает константа #define INTERNET. Если web-сервис находится в локальной сети – необходимо закомментрировать эту строку. В противном случае предполагается что web-сервис находится в сети Интернет.

Имя сервера, на котором опубликован web-сервис, отвечающий за обмен данными в системе, прописан в константе RemoteServerName. а порт соединения – в константе RemoteServerPort. Необходимо прописать в этих константах реальное имя сервера и порт, на котором вы разместите web-сервис обмена данными. Это нужно сделать для варианта работы в сети Интернет или в локальной сети. Текст http-запроса на получение команд от web-сервиса прописан в константе GetObmOUT_Request. Вместо этого теста можно прописать текст другого запроса, но при нормальной работе системы этого делать не нужно.

Еще один параметр, который настраивается при работе с web-сервисом прописан в константе CONNECTION_KEEP_ALIVE и зависит от настроек работы сервера, на котором располагается web-сервис обмена данными. Отсутствие комментария в этой строке говорит о том, что программа будет поддерживать постоянное соединение с сервером, после того как удастся его

установить. Если политика сервера требует обязательно закрывать соединение после каждого запроса – необходимо закомментировать эту строку.

В любом случае после запуска программы на исполнение она выдает в последовательный порт отладочную информацию об успешном или неуспешном соединении с локальной сетью и удаленным сервером. Эту информацию можно прочитать в мониторе порта среды Arduino IDE. Это может понадобиться для устранения проблем, если они возникнут в процессе соединения.

3.5.2 Инструкция по настройке и запуску исполнительного модуля нижнего уровня

Исполнительный модуль нижнего уровня представляет из себя аппаратно-программный комплекс, состоящий из платы Arduino UNO R3 или Arduino Mega и резидентной программы управления, реализующей протокол и логику обмена с транспортным модулем нижнего уровня и непосредственно управление исполнительными устройствами, имитирующими работу стенда (светодиодами).

К одной плате Arduino UNO может быть подключено 18 светодиодов, в том числе, если задействовать аналоговые входы как выходы. Чтобы подключить большее количество светодиодов необходимо использовать несколько плат Arduino UNO или использовать плату Arduino Mega, имеющую большее количество выходов.

Для адресации плат Arduino исполнительного подуровня нижнего уровня в разрабатываемой системе они должны быть пронумерованы в диапазоне 1-99. Номер задается в настройках резидентной программы управления платы Arduino исполнительного модуля.

Исполнительные устройства, подключенные к исполнительному модулю, должны быть описаны и идентифицированы. Это также делается

настройками резидентной программы управления платы Arduino исполнительного модуля.

Нумерация плат Arduino исполнительного подуровня нижнего уровня и нумерация исполнительных устройств, подключенных к этим платам нужны для того чтобы четко адресовать передаваемые из управляющей программ верхнего уровня команды включения и отключения исполнительных устройств нижнего уровня.

Резидентная программа исполнительного модуля нижнего уровня находится в папке SerialClient и называется так же. Настройки прописаны в файле ExchOptions.h. При конфигурировании каждой платы Arduino исполнительного модуля этот файл должен быть изменен под конкретную плату, после чего программа должна быть скомпилирована и загружена в память микроконтроллера платы Arduino.

Для адресации платы Arduino исполнительного модуля предусмотрена константа «#define THIS_ARDUINO». Необходимо задать порядковый номер конфигурируемой платы в этой константе.

Далее необходимо описать все исполнительные устройства, подключенные к данной плате Arduino. Для этого нужно сначала объявить все эти устройства. Так как все исполнительные устройства в нашем проекте — это светодиоды, то их объявление сводится к созданию объектов специального класса (библиотеки) TLed. Пример объявления светодиода выглядит так:

TLed Led1(LED1_PIN);

Параметр конструктора – это номер вывода платы Arduino, к которой подключен светодиод.

Описание исполнительных устройств задается массивом ThisDevices. Необходимо прописать для каждого устройства его порядковый номер (адрес) в пределах данной платы Arduino в диапазоне 1-99, тип устройства (TYPE_LED для всех) и ссылку на экземпляр класса устройства (ранее

созданный объект класса TLed). Пример описания исполнительного устройства выгладит так:

{01, TYPE_LED, &Led1}

Количество исполнительных устройств, подключенных к данной плате Arduino, задается константой #define ArduinoDeviceCount.

После задания настроек конфигурируемой платы Arduino необходимо скомпилировать программу и загрузить ее в память контроллера платы Arduino. Во время загрузки плата должна быть отключена от последовательного интерфейса, при помощи которого она соединена с транспортным модулем нижнего уровня.

Если на нижнем уровне используется несколько исполнительных модулей, то после настройки первого модуля и загрузки в него программы, необходимо настроить и загрузить программу во второй и последующие модули. При этом каждый раз нужно менять конфигурационные настройки: адрес платы Arduino и описание подключенных к ней исполнительных устройств.

3.5.3 Процедура загрузки резидентной программы в память платы Arduino

Программа для транспортного модуля и исполнительного модуля нижнего уровня должна быть загружена в память контроллера Arduino, выполняющего роль соответствующего модуля. Загрузка производится путем подключения платы Arduino к персональному компьютеру через последовательный интерфейс.

«Любая плата Ардуино имеет, как минимум, один аппаратный последовательный интерфейс UART, через который производится в том числе загрузка программы. Для этого к сигналам интерфейса UART (RX и TX) подключены соответствующие выводы микросхемы ATmega16U2 - преобразователя интерфейса USB/UART» [2, с. 3].

На стороне персонального компьютера должен быть установлен специальный драйвер ch341ser.exe, создающий в операционной системе Windows виртуальный COM-порт на базе USB-порта. Через него и происходит обмен.

Физически соединение происходит по USB-кабелю. Со стороны платы Arduino имеется USB-разъем, который через преобразователя интерфейса USB/UART позволяет загрузить программу в память контроллера. Со стороны ПК провод должен быть подключен к разъему USB, на который «повешен» виртуальный COM-порт. Важно понимать, несмотря на то, что плата подключена к компьютеру через USB порт, все программы обмениваются данными через виртуальный COM порт, не подозревая, что порт виртуальный.

Загрузка программы производится из программного обеспечения Arduino IDE. В этой среде необходимо выбрать тип платы, которая подключена к ПК (меню «Инструменты / Плата») и порт, через который производится загрузка (меню «Инструменты / Порт»). Для загрузки необходимо нажать кнопку «Загрузка». На момент загрузки последовательное соединение через интерфейс UART всех модулей нижнего уровня должно быть отключено, иначе загрузка программы не будет выполнена.

3.5.4 Инструкция по работе с программой верхнего уровня

ИС хранит данные в СУБД «MS SQL Server». При первом запуске ИС выполняется создание БД. Информация для подключения к СУБД записывается в файл «ArdDirection.xml». Настройки подключения к СУБД представлены на рисунках 46-48.

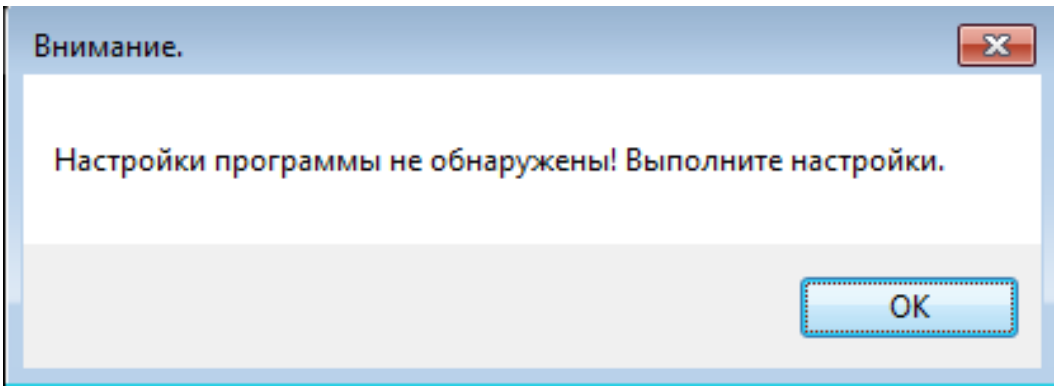


Рисунок 46 – Необходимость настроить подключение

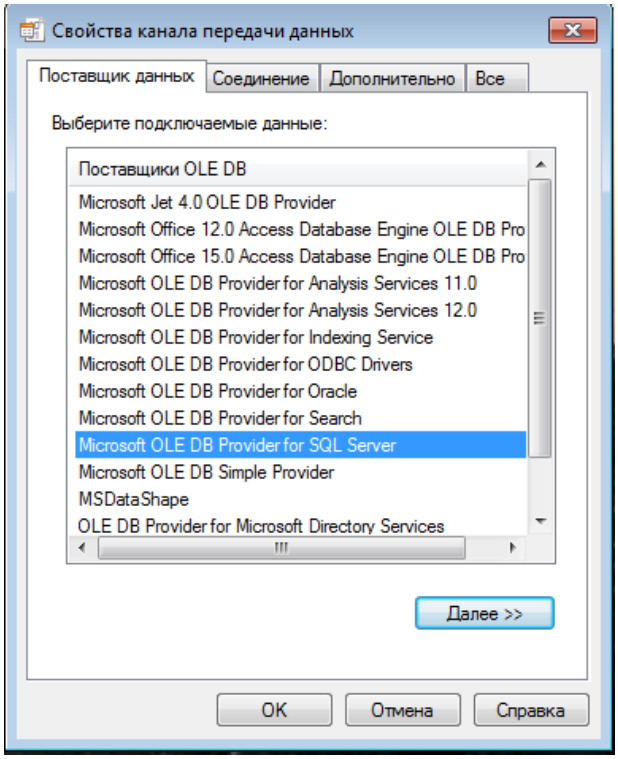


Рисунок 47 – Выбор поставщика

Для разработанной ИС поставщик «MS SQL Server».

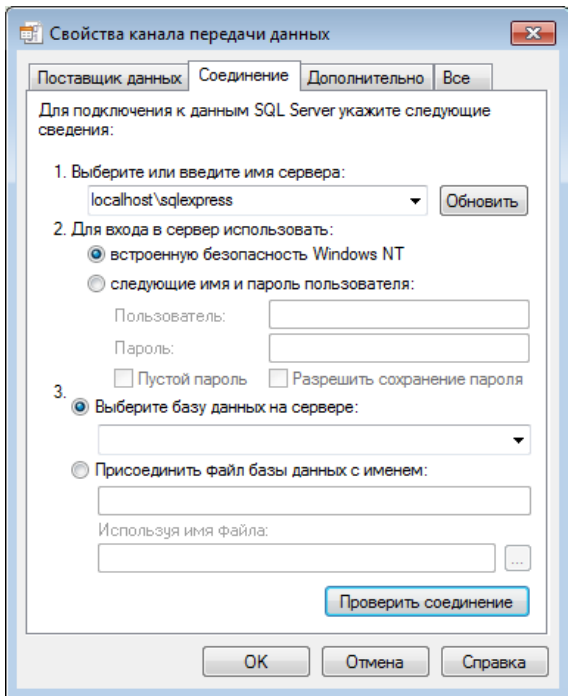


Рисунок 48 – Соединение с СУБД

У пользователя ИС должны быть «административные» права для СУБД и ОС. Заполнив настройки требуется выполнить проверку подключения. Выполнение ИС начинается с запуска «DOCS.exe». На экран выводится окно для авторизации оператора (рисунок 49). Настроечные данные авторизации: «логин» «1», пароль «1».

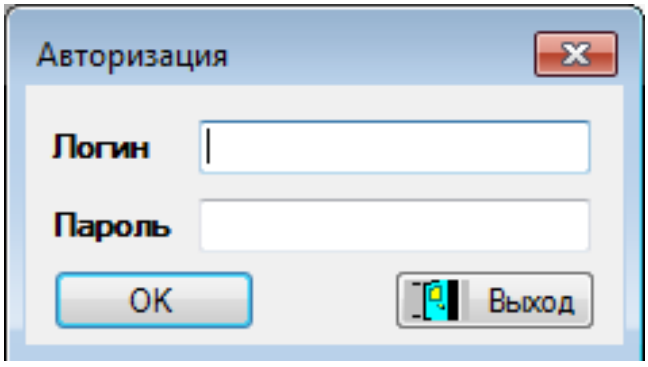


Рисунок 49 – Окно авторизации

После проверки доступа открывается главное окно ИС (рисунок 50).

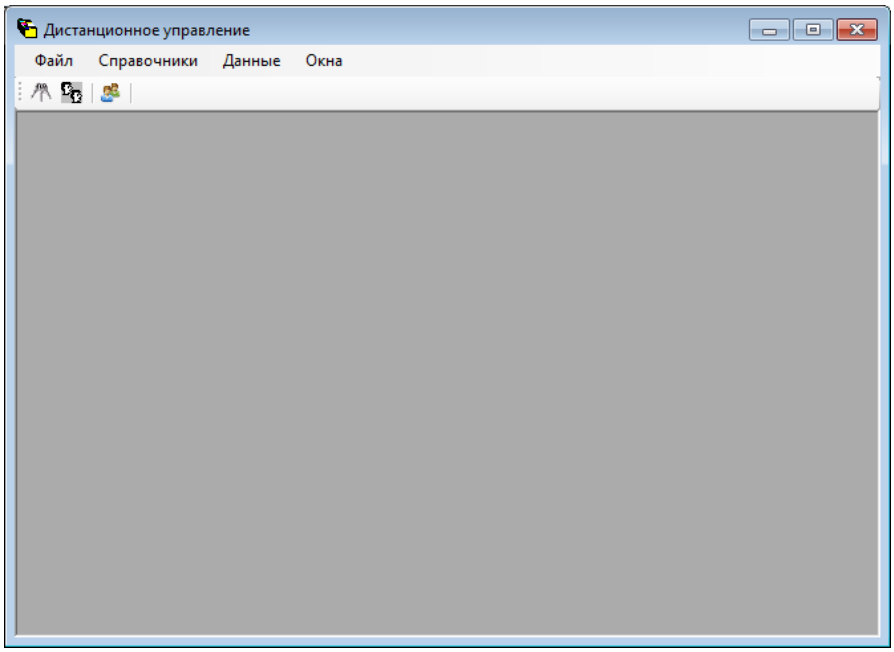


Рисунок 50 – Главное окно ИС

Главное меню ИС позволяет пользователю использовать все функции ИС и включает в себя пункты: «Файл», «Справочники», «Данные» и «Окна».

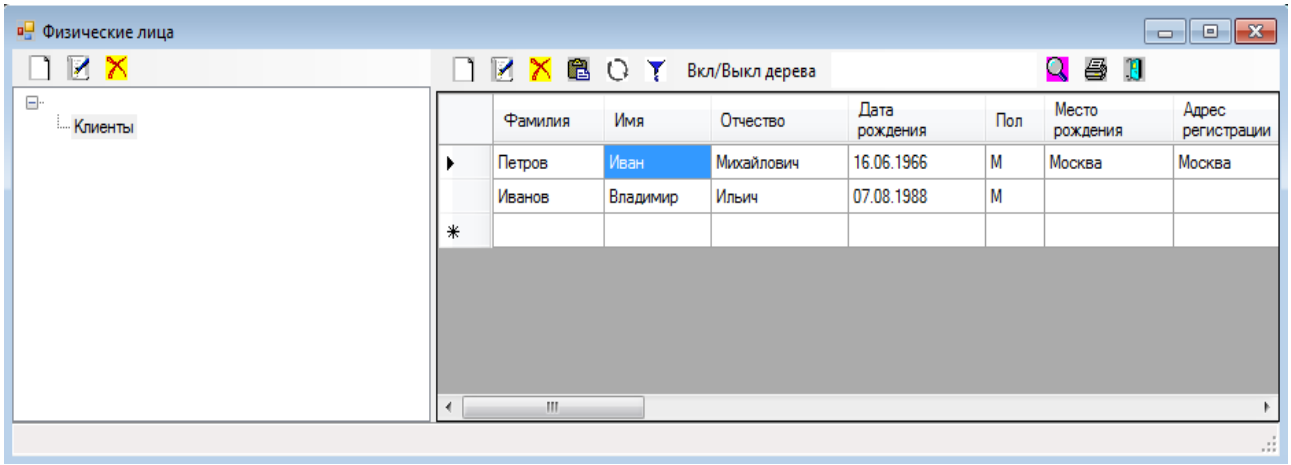


Рисунок 51 – Окно данных о физических лицах

Справочник «Физические лица» доступен через пункт меню «Справочники \ Физические лица». Вследствие чего открывается форма для просмотра и редактирования списка таблицы (рисунок 51).

Данная форма универсальная для просмотра всех таблиц ИС. Опишем ее командные элементы, кнопки:

- «Новый» применяется при вводе нового элемента. Редактирование может производится как в самом списке, так и в отдельном окне, если это предусмотрено для данного справочника.
- «Изменить» изменение элемента.
- «Удалить» применяется при удалении элементов.
- «Копировать» позволяет копировать текущий элемент.
- «Обновить» запрос данных у СУБД.
- «Фильтр» фильтрация списка элементов по заданным пользователем параметрам.
- «Включение/Выключение дерева» отображение элементов согласно иерархии справочника.
- «Подчиненные таблицы» отображает подчиненные справочники для текущего элемента.
- «Поиск» применяется при необходимости найти элемент по определенной строке.
- «Печать» экспортирует текущий список элементов в приложение «MS Excel».
- Сортировка элементов списка выполняется двойным «щелчком» манипулятора «мышь» по заголовку требуемого столбца.
- «Выход» закрывает текущее окно.

Объект контекстного меню «Подчиненные таблицы» «Характеристики» дает возможность увидеть характеристики текущего элемента. Для «основных» таблиц ИС можно вести описание «дополнительных» свойств. За это отвечает пункт «Подчиненные таблицы /

Дополнительные свойства». Любой список ИС с настроенным фильтром и сортировкой можно экспортировать в «MS Excel», благодаря существующей кнопке «Печать». Выполнять настройку «простейших» справочников нужно через пункт: «Справочники / Дополнительные справочники / Настройки простых справочников» (рисунок 52).

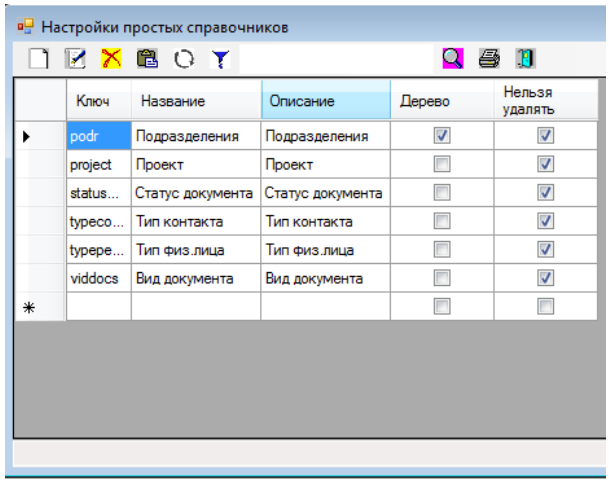


Рисунок 52 – Данные «простейших» таблиц ИС

Настройку контроллеров можно выполнить через пункт «Справочники \Настройки \ Управляющие устройства» (рисунок 53).

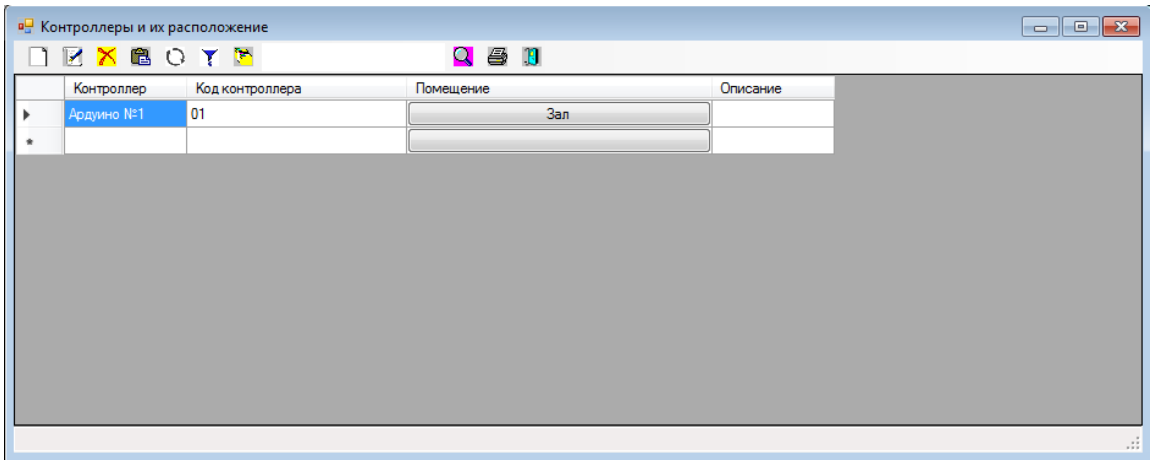


Рисунок 53 – Контроллеры

В данном списке задаются все управляющие устройства (контроллеры), а также принадлежащие им устройства и датчики (рисунок 54).

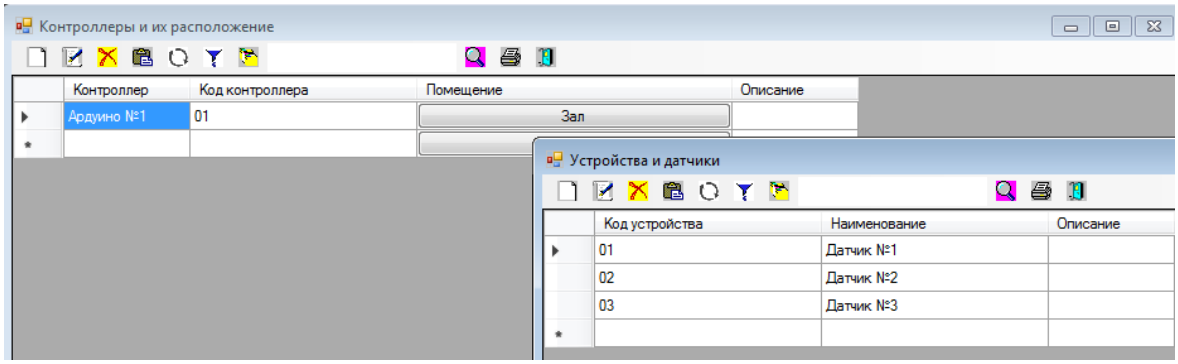


Рисунок 54 – Устройства и датчики

Для датчиков и устройств можно задать команды, которые буду передавать или получать информацию контролёра (рисунок 55).

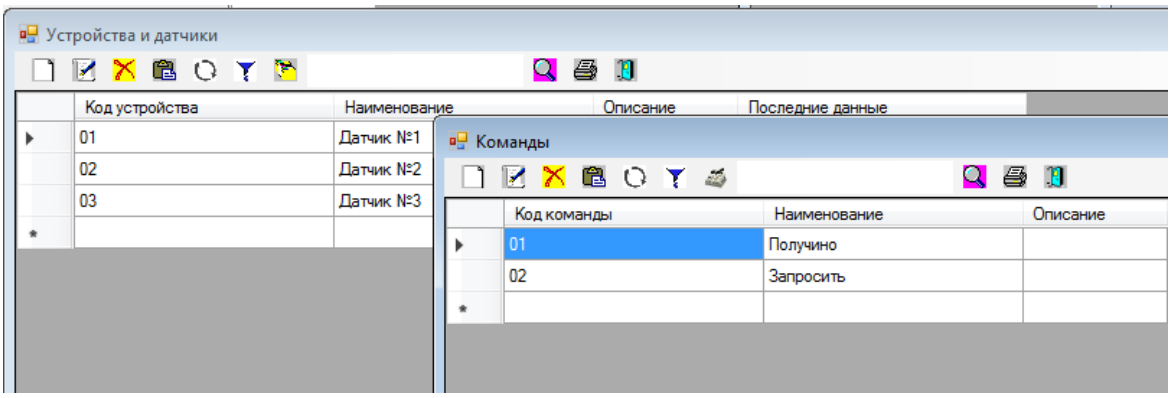
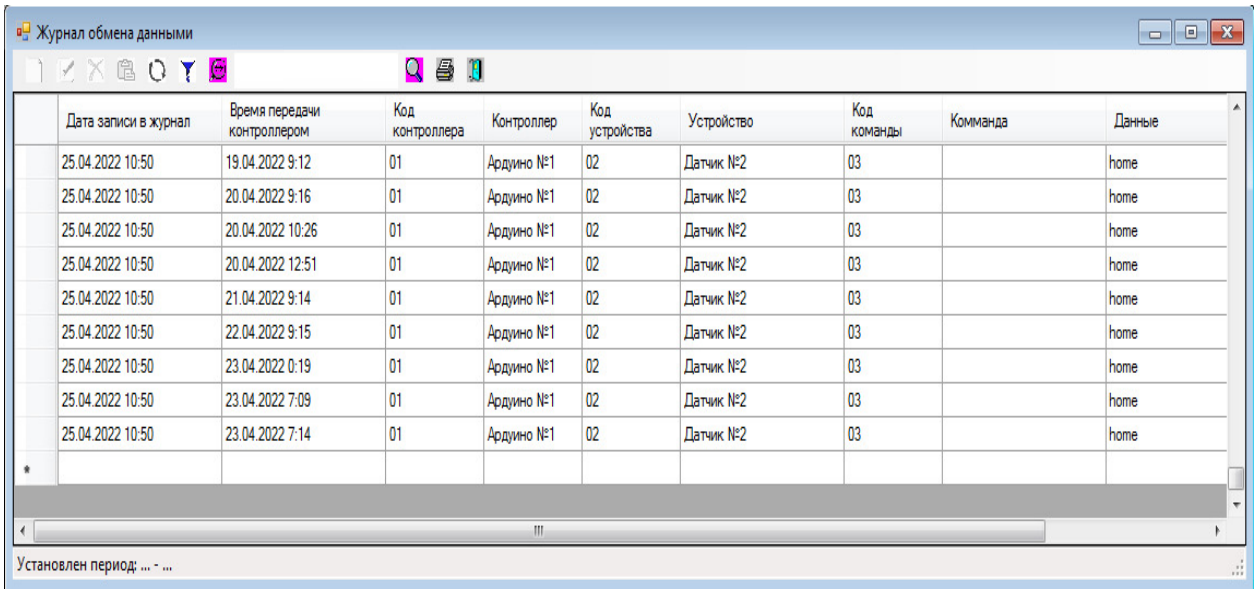
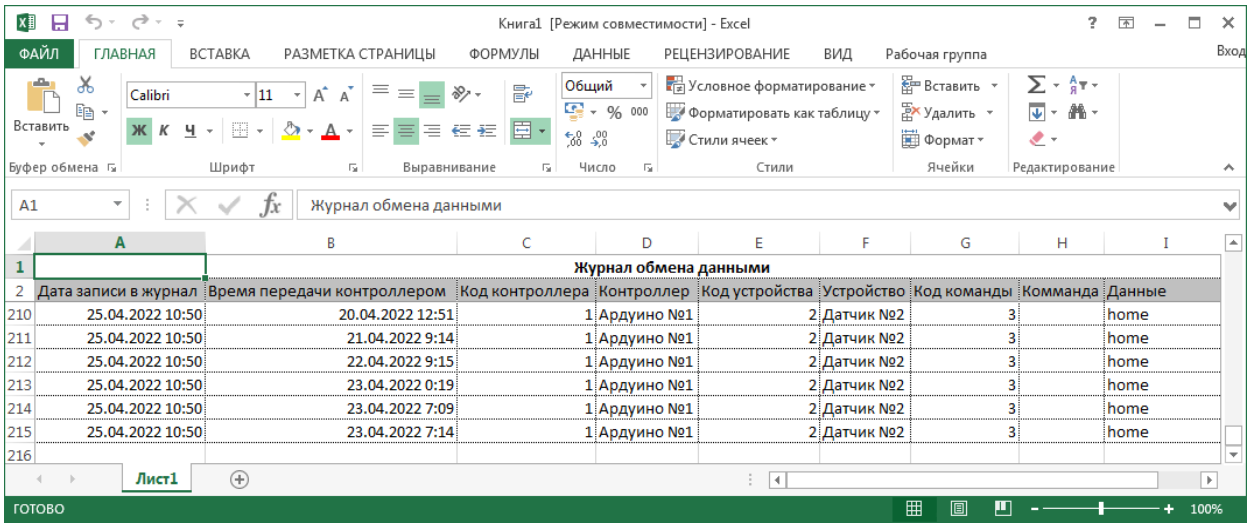


Рисунок 55 – Команды



Дата записи в журнал	Время передачи контроллером	Код контроллера	Контроллер	Код устройства	Устройство	Код команды	Комманда	Данные
25.04.2022 10:50	19.04.2022 9:12	01	Ардуино №1	02	Датчик №2	03		home
25.04.2022 10:50	20.04.2022 9:16	01	Ардуино №1	02	Датчик №2	03		home
25.04.2022 10:50	20.04.2022 10:26	01	Ардуино №1	02	Датчик №2	03		home
25.04.2022 10:50	20.04.2022 12:51	01	Ардуино №1	02	Датчик №2	03		home
25.04.2022 10:50	21.04.2022 9:14	01	Ардуино №1	02	Датчик №2	03		home
25.04.2022 10:50	22.04.2022 9:15	01	Ардуино №1	02	Датчик №2	03		home
25.04.2022 10:50	23.04.2022 0:19	01	Ардуино №1	02	Датчик №2	03		home
25.04.2022 10:50	23.04.2022 7:09	01	Ардуино №1	02	Датчик №2	03		home
25.04.2022 10:50	23.04.2022 7:14	01	Ардуино №1	02	Датчик №2	03		home

Рисунок 56 – Окно журнала обменов данными



	A	B	C	D	E	F	G	H	I
1		Журнал обмена данными							
2	Дата записи в журнал	Время передачи контроллером	Код контроллера	Контроллер	Код устройства	Устройство	Код команды	Комманда	Данные
210	25.04.2022 10:50	20.04.2022 12:51	1	Ардуино №1	2	Датчик №2	3		home
211	25.04.2022 10:50	21.04.2022 9:14	1	Ардуино №1	2	Датчик №2	3		home
212	25.04.2022 10:50	22.04.2022 9:15	1	Ардуино №1	2	Датчик №2	3		home
213	25.04.2022 10:50	23.04.2022 0:19	1	Ардуино №1	2	Датчик №2	3		home
214	25.04.2022 10:50	23.04.2022 7:09	1	Ардуино №1	2	Датчик №2	3		home
215	25.04.2022 10:50	23.04.2022 7:14	1	Ардуино №1	2	Датчик №2	3		home
216									

Рисунок 57 – Экспорт журнала

Журнал обменов доступен в «Данные / Журнал обмена» (рисунок 56). Для печати журнала согласно установленным фильтрам и сортировкам нужно выбрать кнопку «Печать» и список будет экспортирован в программу «Excel» Журнал обменов доступен в «Данные / Журнал обмена» (рисунок 57).

Операторов ИС можно добавлять и изменять в «Файл / Настройки / Пользователи». В данном списке можно не только редактировать учетные записи пользователей, но и давать им права доступа на объекты ИС через кнопку «Подчиненные таблицы» как показано на рисунке 58.

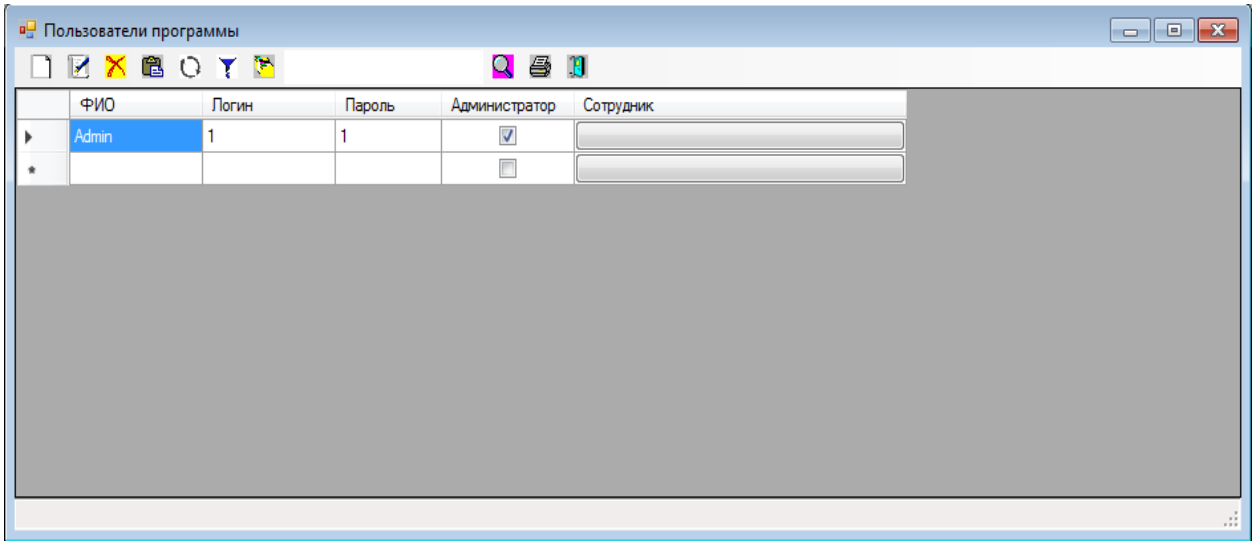


Рисунок 58 – Окно пользователей

Завершение работы ИС выполняется через пункт «Файл \ Выход».

Заключение

Целью ВКР является разработка системы автоматизированного контроля и учета энергоресурсов промышленного предприятия АО «ННК-Печоранефть» г. Усинск.

На начальном этапе при написании ВКР был проведен анализ общих принципов построения АРМ, доказана необходимость разработки системы, а также проведено обоснование выбора и описана платформа на которой проведена разработка.

В процессе работы были изучены основные особенности платформы Ардуино, а также проведен анализ и изучены характеристики контроллера «Arduino».

Изучены особенности работы с цифровыми и аналоговыми входами данного контроллера и работы с аналогово-цифровым преобразователем этого контроллера.

Для обеспечения обмена данными был изучен «Ethernet» модуль и способы выхода в локальную и глобальную сети.

Все элементы системы были сведены в единую схему.

Далее для этой схемы была разработана программа в среде «Arduino IDE».

Ряд функций выведен в отдельный параллельный процесс, периодически запускаемый по прерыванию от аппаратного таймера.

В основной программе происходит общее управление программным процессом.

Большинство устройств в программе реализованы при помощи классов, которые включают в себя все особенности работы с этими объектами.

Классы предоставляют основной программе программный интерфейс по работы с ними.

Для разработки web-сервиса и программы верхнего уровня выбран ПП «Visual Studio C#».

Выбор обоснован надежностью ПП, прочной связью с ОС, гибкостью платформы, являющейся основой ПП, значительным перечнем возможностей, предоставляемых ПП для решения самого обширного круга задач. В качестве СУБД для web-сервиса и программы верхнего уровня был выбран ПП «MS SQL Server».

Проведенное тестирование показывает, что разработанное автоматизированное рабочее место полностью выполняет поставленную перед нами задачу, работает стабильно, обладает гибкой функциональностью и продуманным интерфейсом как программных, так и аппаратных частей системы.

Список используемой литературы

1. Абрамов А.М., Никулин О.Ю., Петрушин А.Н. Системы управления доступом. М.: ОБЕРЕГ – РБ, 2018. 192 с.

2. Аппаратная платформа Arduino. [Электронный ресурс]: Официальный сайт компании Arduino. URL: http://arduino.ru/ (дата обращения: 21.04.2022).

3. Барсуков В.С., Водолазкий В.В. Современные технологии безопасности. М.: Нолидж, 2017. 496 с.

4. Волхонский В.В. Системы контроля и управления доступом. Модель, структура, методы идентификации // БДИ. Безопасность, достоверность, информация. 2018. № 4. С. 18 – 20.

5. Волхонский В.В. Системы охранной сигнализации. СПб.: Экополис и культура, 2019. 164 с.

6. ГОСТ Р 51241-98. Средства и системы контроля и управления доступом. Классификация. Общие технические требования. Методы испытаний: утв. постановлением N 472 от 29.12.1998 // Консультант плюс: справочно-правовая система.

7. Давиденко Ю.Н., 500 схем для радиолюбителей. Современная схемотехника в освещении. Эффективное электропитание люминесцентных, галогенных ламп, светодиодов, элементов Умного дома. М.: Наука и техника, 2018. 470 с.

8. Дементьев А.В, Умный дом XXI века. М.: Издательские решения, 2019. 139 с.

9. Зиборов В.В., Visual C# на примерах. СПб.: БХВ-Петербург, 2018. 475 с.

10. Крахмалев А.К., Системы контроля доступа // Системы безопасности, связи и телекоммуникаций. 2018. № 32. С. 26 – 29.

11. Культин Н. Б. «Microsoft Visual C# в задачах и примерах.» —

СПб.: БХВ-Петербург, 2019. 320 с.

12. Петин В.А., Arduino и Raspberry Pi в проектах Internet of Things. М.: НТ Пресс, 2017. 338 с.

13. Петин В.А., Проекты с использованием Arduino М.: БХВ, 2018. 132 с.

14. Поиск, разведка и добыча. [Электронный ресурс]: компании АО ННК-Печоранефть. URL: http://pechoraneft.ru/about/history/ (дата обращения: 21.04.2022).

15. Ревич Ю.В. Азбука электроники. Изучаем Arduino – М.: АСТ Кладезь, 2017. 224 с.

16. Тесля Е.В., Умный дом своими руками. М.: Питер, 2020. 717 с.

17. Шнайдер Д.А., Подход к оперативному анализу эффективности теплоснабжения зданий // ВестникЮУрГУ. Серия Компьютерные технологии, управление и радиоэлектроника. 2019. № 13. С. 70–73.

18. Brian Evans. Arduino programming notebook – Creative Commons, 2007. – 14 p.

19. Brian Kernighan, Dennis Ritchie. C programming language – Prentice-Hall, 2008. – 86 p.

20. Bjorn Stroustrup. C++ programming language, 4th Edition – Addison-Wesley Professional, 2013. – 132 p. – ISBN-10 0321563840; ISBN-13 978-0321563842.

21. Michael McRoberts. Beginning Arduino – Apress, 2013. – 319 c.

22. Massimo Banzi. Getting Started with Arduino – Maker media 2014. – 149 p.

23. Simon Monk. – Programming the Arduino. Professional work with sketches – McGraw Hill TAB, 2013. – 74 p.