Министерство науки и высшего образования Российской Федерации
«Южно-Уральский государственный университет»
Высшая школа электроники и компьютерных наук
Кафедра «Инфокоммуникационные технологии»

Допустить к защите
Руководитель направления
_____С.Н. Даровских
«___» _____2020г.

Рецензент
_____
«___» _____2020г.

Разработка системы мониторинга в сфере сельского хозяйства на основе технологии интернета вещей

Направление 11.04.02 **«Инфокоммуникационные технологии и системы связи»**

магистерская программа **«Системы мобильной связи»**

ЮУрГУ – М 11.04.02.2020.577.00 ПЗ ВКР

Научный руководитель
Николаев А.Н. _____

“_____” _____2020 г.

Магистрант
студент группы КЭ-223
Горшков А.В._____

“_____”_____2020 г.

Нормоконтролер
Спицына В.Д._____

“_____” _____2020г.

Челябинск
2020

РЕФЕРАТ

Горшков А.В. Разработка системы мониторинга в сфере сельского хозяйства на основе технологии Интернета вещей - Челябинск: ЮУрГУ, ВШЭКН, 2020, илл.41, с.117 – Библиографический список 38 – наименований, 6 плаката формата А1

В данной выпускной квалификационной работе рассмотрены основные теоретические и практические аспекты создания системы мониторинга на основе технологии интернета вещей. В первой главе выявлена актуальность поднятой проблемы. Во второй главе описывается концепция технологии интернета вещей, развитие данной технологии, описание архитектуры технологии, а так же приведена архитектура системы мониторинга на основе интернета вещей. В следующих главах анализируются конкретные компоненты и способы реализации проекта, теоретически исследованы разные технологии, которые применяются про создании системы мониторинга.

В работе кратко описаны эксперименты с имеющемся оборудованием. Проведены настройки оборудования некоторых технологий и проведены примеры их реализации. Показано организация передачи данных от разных компонентов системы. Сделаны выводы по работе и описаны дальнейшие планы по разработке системы мониторинга сельского хозяйства.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Продовольственная проблема – это глобальная проблема всего человечества. Дефицит продуктов сопровождал человечество на всем протяжении его развития. В связи с развитием мировой торговли и транспорта эта проблема несколько ослабла, но не исчезла. Причем, современная мировая продовольственная ситуация трагична. По прогнозам ООН, к 2050 году будет необходимо производить на 70 % больше продуктов питания, чем сейчас, чтобы прокормить растущее население Земли. [1] Для сельского хозяйства это означает регулярный и постоянно растущий спрос на сельскохозяйственную продукцию, а также появление ряда новых вызовов и принципиально новых требований к уровню производительности в целом. Многие страны возлагают большие надежды на цифровизацию экономик, понимая под этим различные элементы автоматизации. Одним из наиболее эффективных инструментов в достижении нового уровня цифровизации может стать «Интернет вещей» (Internet of Things, IoT) и появление «Умного сельского хозяйства», которое будет ставить перед собой цель максимально автоматизировать сельскохозяйственную деятельность, повысить урожайность и качество продукции .

Интернет вещей — сегодня этот термин можно услышать чуть ли не на каждом шагу. Интернет вещей (Internet of Things, IoT) — это единая сеть физических объектов, способных изменять параметры внешней среды или свои, собирать информацию и передавать её на другие устройства[2].Если обратиться к википедии в поисках определения для термина “интернет вещей”, можно увидеть следующее: Интернет вещей (англ. Internet of Things, IoT) — концепция вычислительной сети физических предметов («вещей»), оснащённых встроенными технологиями для взаимодействия друг с другом или с внешней средой, рассматривающая организацию таких сетей как явление, способное перестроить экономические и общественные процессы, исключающее из части действий и операций необходимость участия человека.

Сегодня Интернет вещей состоит из слабо связанных между собой разрозненных сетей, каждая из которых была развернута для решения своих специфических задач. Интернет вещей проникает в самые разные устройства и приложения, которые могут использоваться во всех отраслях промышленности и таким образом соединит огромное количество устройств. За последние пять лет, в качестве устройства связи, лидируют смартфоны, которые помогают решать повседневные задачи. Как только Интернет-вещей станет обыденным, легко вообразить целые категории потребительских приложений и бизнес-приложений, задействованных в этой новой системе, которую можно описать понятием “подключенная жизнь”. Это включает строительную и бытовую автоматизацию, системы отопления и кондиционировании воздуха, управление дорожным движением (например, умный светофор), организацию заботы о пожилых людях, системы безопасности, а также подключенные к интернету автомобили и наружную рекламу[3].

С одной стороны, повсеместно развернутые сети датчиков открывают новый мир полезных приложений. Вместе с тем все эти новшества создают целый ряд проблем с передачей и обработкой информации, а так же появляется возможность попадания сети датчиков в руки злоумышленника. Такое особенно вероятно, если сеть напрямую управляет физическими устройствами с помощью коммуникаций "машина-машина" (machineto-machine, M2M). [3] Таким образом, при подключении сети IoT, нам необходимо создать качественное соединение всех компонентов системы, обеспечить своевременную и доверительную передачи информации и обезопасить их взаимодействия. Этого можно добиться путем проверки и анализа передаваемых устройствами параметров, проведения сравнения существующих систем дистанционного контроля устройств Интернета вещей и разработку системы контроля и мониторинга различной информации.

Выпускная квалификационная работа направлена на исследования вопроса использования технологии IoT в современных проблемам сельского

хозяйства. В ходе написания работы были описаны архитекторы подсистем и всей системы мониторинга. Проведен анализ конечных устройств, существующих технологий беспроводной передачи данных, создание программного обеспечения и базы данных, способы реализации клиентской части. Проведен обзор аналогичных систем мониторинга сельского хозяйства. Предложены этапы разработки программного обеспечения и пользовательского интерфейса. В ходе работы выявлены преимущества и недостатки различных технологий, и предложены наработки IoT подсистем для системы мониторинга.

1 Постановка задачи

Интернет вещей позволит создать динамические сети, состоящие из миллиардов и триллионов таких вещей, коммуницирующих между собой. Таким образом, обеспечится сплав цифрового и физического миров, для которого приложения, сервисы, компоненты связующего ПО и конечные устройства — это вещи.

Я выбрал сферу связанную с выращиванием сельскохозяйственных культур. Развитость и продуктивность сельхозпроизводства влияет на сбалансированность экономики государства, политическую обстановку в нём, его продовольственную независимость. При этом сельское хозяйство в условиях рыночной экономики не способно полноценно конкурировать с другими отраслями, поэтому уровень и эффективность его поддержки со стороны государства соотносится с благополучием самого государства.[1] В отличие от промышленности технологический процесс в сельском хозяйстве тесно связан с природой, где земля выступает в роли главного средства производства. Поэтому стоит большой вопрос правильного использование земли, мониторинг роста культур, сохранение семян и последующая обработка растений.

Проблемы сельского хозяйства в стране всегда будут актуальны, особенно в нашем регионе. Например для выращивании хорошего урожая картофеля целесообразно начинать после устойчивого перехода температуры почвы на глубине 8-10 см через +8°C и проводить в кратчайшие сроки. [4] Для соблюдения данных критериев помогут датчики и их мониторинг.

Другой пример, длительность хранения жизнеспособных семян, прежде всего относятся температура и влажность. Обычно чем ниже температура и меньше содержание влаги в хранилище, тем дольше семена остаются жизнеспособными. Некоторые исследователи считают, что продолжительность хранения удваивается при понижении количества влаги на 1% или температуры на 5°C[4].

Данная выпускная квалификационная работа носит исследовательский характер, поэтому главной целью работы является анализ существующих систем дистанционной передачи информации для интернета вещей, результаты которого, будут положены в основу разработки собственной системы для работы с устройствами IoT. Ранее предлагалась разработать прототип данной системы мониторинга параметров окружающей среды для сельского хозяйства в рамках программы поддержки коммерчески ориентированных научно-технических проектов молодых ученых «УМНИК».

Планируемая к разработке система должна быть конкурентоспособна относительно существующих коммерческих систем устройств IoT за счет своей дешевизны, применение новых актуальных технологий, и обширности решаемых задач для частных фермерских хозяйств. Появление конкурентоспособной IoT системы мониторинга фермерского хозяйства для увеличения количества и качество урожая, что может привести к увеличению прибыли. Поэтому главной целью проекта является контроль текущего состояния земли.

Исходя из этого, можно выделить несколько основных задач для реализации проекта, которому я дал название SMART-АГРО:

– изучить основные проблемы концепции Интернета вещей;

– выбрать архитектуру системы мониторинга;

– провести анализ всех подсистем, а именно конечные устройства, технология передачи данных, программное обеспечение и пользовательский интерфейс;

– провести обзор аналоговых систем мониторинга в сфере сельском хозяйстве;

– выбрать основные компоненты системы и описать наработки своего учебного проекта SMART-АГРО .

2 Технологии Интернета вещей

2.1 Архитектура интернета вещей

Интернет вещей – это способы взаимодействия физических объектов, устройств и систем между собой и с окружающим миром с применением различных технологий связи и стандартов соединения [2].

Одним из первых подключенных к сети устройств стал вендинговый аппарат по продаже Coca-Cola, установленный в Университете Карнеги — Меллон в 1982 году. Так, аппарат имел возможность передавать данные о количестве содержащихся в нем бутылок и о своем состоянии в целом. Периодом активных обсуждений сетей, которые смогли бы обеспечить межмашинное взаимодействие стали 1990-е годы. Например, руководитель исследовательских работ в Xerox PARC (исследовательском центре компании Xerox) Марк Вейзер предложил концепцию повсеместного компьютинга, предполагавшую массовое внедрение компьютеров и организацию связи между ними, благодаря которой машины самостоятельно бы решали повседневные задачи пользователя .[5]

Периодом бурного развития Интернета вещей стали 2000-е. Тогда как в 1990-х вся деятельность, связанная с IoT, носила в основном теоретический характер – концепции, обсуждения, отдельные идеи, в 2000-х и 2010-х стали массово появляться и запускаться успешные IoT-проекты в реальности.[5] Так, было разработано множество пользовательских устройств, относящихся к Интернету вещей – от фитнесс-трекеров до умных ламп и умных дверей. Кроме того, начали развиваться масштабные проекты, основанные на технологиях IoT – умные города, умное производство, умный транспорт, беспилотные автомобили и многое другое. Не в последнюю очередь это стало возможным благодаря активному прогрессу в сфере информационных технологий – повсеместному распространению беспроводного соединения, повышению

пропускной способности интернет-связи, возникновению энергоэффективных сетей дальнего радиуса действия и др.[3]

В настоящее время прослеживается отставание России от ведущих европейских стран в данном вопросе. Появление новых инструментов и проектов Интернета Вещей могут стать одним из факторов, способных сократить данное отставание. Так же ключевым фактором, препятствующим развитию Интернета Вещей в России, является отсутствие единых стандартов IoT. В России функционируют различные структуры, развивающие концепцию Интернета Вещей. Однако если они будут действовать разрозненно, то это замедлит ее реализацию. В связи с этим, необходима разработка единых стандартов и требований к исследованиям, технологиям, их безопасности и эксплуатации. Важно предусмотреть совместимость новых технологий с уже существующими IT–системами. При возникновении разногласий все участники понесут материальные и временные затраты.

Еще одна проблема, многие до сих пор многие путают IoT архитектуру с архитектурой автоматизации, где главной задачей является получение информации с датчиков, и на их основе осуществляется управление исполнительными механизмами.[6]

При обсуждении рынка Интернета Вещей часто происходит отождествление этого технологического явления с решениями, поддерживающими межмашинное взаимодействие (machine-to-machine, M2M), такими как телеметрия или наблюдения за состоянием производственных объектов. Решения в этой области существуют достаточно давно и активно используются в машиностроении, транспорте, энергетике, добыче полезных ископаемых, торговле и логистике. Технологии M2M используются в системах физической безопасности и наблюдения. Эти решения имеют ярко выраженную индустриальную принадлежность и представляют собой закрытые системы, часто реализуемые на специальном оборудовании со встроенным программным обеспечением. M2M-решения являются важной частью рынка Интернета

Вещей. Уже сегодня компании, использующие их, демонстрируют результаты оптимизации своих бизнес-процессов. Это является хорошей стартовой площадкой для развития в компаниях полноценных решений, основанных на технологиях IIoT. Использование M2M позволяет получить достаточное количество надежной информации для принятия решений, но требует человеческого участия для осуществления последующих операций [7].

Процесс перехода от M2M к IoT подразумевает, что информация, полученная в ходе интеллектуального анализа данных, позволит быстрее и надежнее принимать решения, влиять на процессы без привлечения человека. Именно аналитика большого количества данных, которые создаются различными устройствами, выводит оптимизацию процесса на другой уровень. Аналитическая система в составе IoT проводит анализ данных и понимает, какое действие нужно предпринять. Большое количество рутинных процессов (например, мониторинг данных с объекта и осуществление действий на основании этих данных) может происходить автоматически и существенно влиять на производительность и оптимизацию операционной деятельности. Интернет Вещей сегодня представляет собой сеть слабо связанных между собой разрозненных сетей, каждая из которых имеет свое специфическое назначение [7]. Для каждого типа взаимодействия между сетями используются свои стандарты передачи данных (например, CoAP, ETSI SmartM2M, MQTT, LWM2M). По мере развития Интернета Вещей разрозненные сети будут объединяться в связную сеть, и стимулировать унификацию и стандартизацию протоколов и коммуникационных решений [6].

Интернет-вещей концептуально принадлежит к сетям следующего поколения, поэтому его архитектура во многом схожа с архитектурой NGN, это мультисервисные сети связи, ядром которых являются опорные IP-сети, поддерживающие полную или частичную интеграцию услуг передачи речи, данных и мультимедиа. [6] Данный сети строятся на уже известных структурах и единых топологиях. Топология IoT отличается от

обычной уровневой модели, такой как OSI. Это не линейный и более сложный граф потоков. Некоторые компоненты являются необязательными и могут отсутствовать в конкретном классе решений. Архитектура IoT не имеет единой топологии, так как состоит из набора различных инфокоммуникационных технологий, обеспечивающих функционирование Интернет-вещей. Архитектура IoT включает четыре функциональных уровня, которые показаны на рисунке 1.

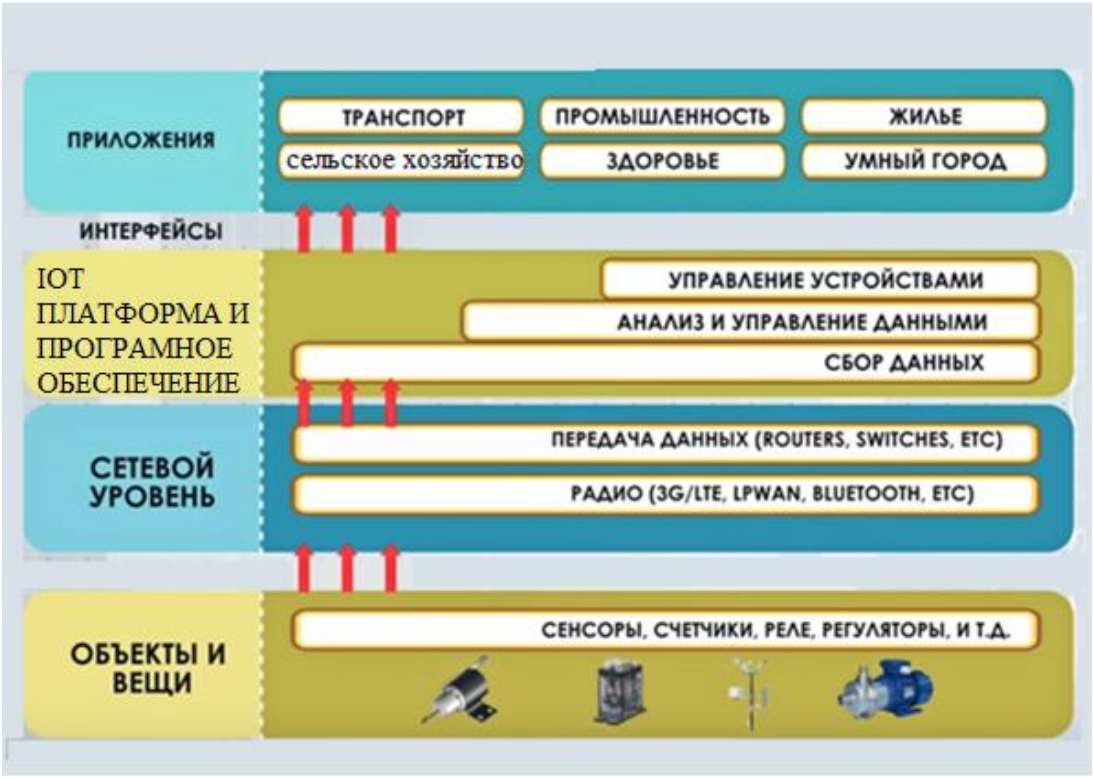


Рисунок 1 —– Архитектура Интернета вещей

IT архитектура включает в себя две, на первый взгляд несовместимые вещи: с одной стороны — это большое число периферийных устройств с малыми вычислительными мощностями, низким энергопотреблением, высокой скоростью реакции на события, а с другой стороны — облачные сервера с высокой вычислительной мощностью для обработки большого массива данных, их хранения и классификации, часто с элементами машинного интеллекта и аналитики. Эти два мира используют совершенно разные принципы построения и внутренней архитектуры.[3]

Самый нижний уровень архитектуры IoT состоит из «умных» объектов, интегрированных с сенсорами (датчиками). Сенсоры реализуют соединение физического и виртуального (цифрового) миров, обеспечивая сбор в реальном масштабе времени. Миниатюризация, приведшая к сокращению физических размеров аппаратных сенсоров, позволила интегрировать их непосредственно в объекты физического мира.[8] Большинство сенсоров требуют соединения с агрегатором сенсоров (шлюзом), которые могут быть реализованы с использованием локальной вычислительной сети (LAN, Local Area Network), такие как Ethernet, Wi-Fi или персональной сети (PAN, Personal Area Network). Для сенсоров, которые не требуют подключения к агрегатору, их связь с серверами/приложениями может предоставляться с использованием глобальных беспроводных сетей WAN, таких как GSM, GPRS и LTE. Сенсоры, которые характеризуются низким потреблением энергии и низкой скоростью передачи данных, образуют широко известные беспроводные сенсорные сети WSN. [9]

Большой объем данных, воздаваемых на первом уровне IoT многочисленными сенсорами, требует надежной и высокопроизводительной проводной или беспроводной инфраструктуры в качестве транспортной среды данный уровень состоит из конвергентной сетевой инфраструктуры, которая создается путем интеграции разнородных сетей в единую сетевую платформу. Сервисный уровень содержит набор информационных услуг, которые автоматизируют технологические бизнес операции в IoT: поддержки операционной и бизнес деятельности, различной аналитической обработки информации, хранения данных, обеспечения информационной безопасности, управления бизнес процессами и др.[3]

На данном уровне существуют различные типы приложений для соответствующих промышленных секторов и сфер деятельности (энергетика, транспорт, торговля, медицина, образование, сельское хозяйство и др.). Приложения могут быть «вертикальными», когда они являются

«специфическими» для конкретной отрасли промышленности, а также «горизонтальными», которые могут использоваться в различных секторах экономики. [10]

Составной частью интернета вещей является Веб вещей (WEB of Things, WoT). Термин «Веб Вещей» (Web of Things, англ., WoT — аббр.) возник в конце XX века для обозначения концепции, включающей принципы, архитектурные стили и программные шаблоны, которые позволяют объектам реального мира стать частью Всемирной паутины. Он обеспечивает взаимодействие различных интеллектуальных объектов («вещей») с использованием стандартов и механизмов интернет, таких как унифицированный единообразный идентификатор ресурса URI (Uniform Resource Identifier), протокол передачи гипертекста HTTP (HypperText Transfer Protocol), стиль построения архитектуры распределенного приложения REST (Representational State Transfer) и др. Фактически WoT предусматривает реализацию концепции IoT на прикладном уровне с использованием уже существующих архитектурных решений, ориентированных на разработку web приложений.[11]

Для реализации проекта SMART-АГРО предлагается использовать данную архитектуру и использовать все четыре функциональных уровня. На рисунке 2 представлена архитектура проекта. В следующих разделах подробно описываются и анализируются технологии всех уровней.

2.2 Конечные устройства (Объекты и вещи)

Весьма предсказуемо, что все начинается с датчиков. Более того, чем лучше подходит датчик для выполнения своей задачи, тем эффективней будет система дальше. Это своего рода “краеугольный камень” системы.[10] Важно отметить, что датчик регистрирует изменение окружающей среды, а не ее статическое состояние. Датчики делятся на активные — излучающие сами сигналы и принимающие отражение; и пассивные — работающие только на

прием. Естественно, что последние значительно выигрывают по параметрам энергопотребления. Большинство датчиков основано на приеме волн — звуковых, ультразвуковых, световых различного диапазона, тепловых. Однако есть категория датчиков, основанных на изменении их физических характеристик, таких как индуктивность, емкость, давление. Хорошие результаты получаются от комбинации нескольких датчиков, например PIR детектор и емкостной датчик для определения движения. Базовые элементы делятся на несколько типов: сенсоры, актуаторы и гейты.[12]

Рисунок 2 — Архитектура проекта SMART-АГРО

Сенсоры

Сенсор, или датчик (sensor) - это устройство для преобразования некоторой физической величины в электрический сигнал. Сенсоры являются по сути нервной системой робота и служат для обеспечения обратной связи между контроллером и окружающим миром. Оно ничем не отличается от стандартных: разнообразные термометры, микрофоны, камеры и десятки прочих, менее распространённых устройств. Поскольку на поведение автоматической системы могут оказываться влияние многие физические факторы, существует и множество различных сенсоров способных эти факторы фиксировать.[12] На рисунке 3 представлены разнообразные сенсоры разного функционала.

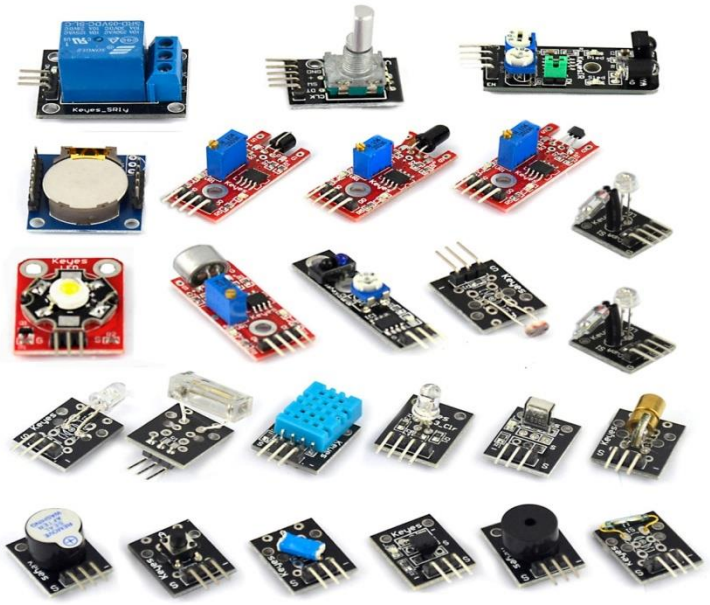


Рисунок 3 —— Разнообразные сенсоры

Актуаторы

Когда говорят "актуаторы" имеем ввиду компактные и маломощные механизмы, которые могут использоваться не только в промышленных установках, но и для бытовых целей - например для открывания дверей или багажника автомобиля, для распашных или раздвижных ворот и т.п. Любой актуатор преобразует вращательное движение вала в поступательное

перемещающее движение штока. В основном перемещает с малым усилием на небольшие расстояния.[8]

Данный тип элементов предназначается для того, чтобы воздействовать на окружающую среду, или на определённый объект в ней. Эту роль могут выполнять самые разнообразные устройства: от сервоприводов и динамиков до замков (конечно, электронных) с осветительными приборами.

Актуаторы находят широкое применение практически повсеместно. Их можно задействовать в разнообразных устройствах, к примеру, для регулировки положения телевизионного приемника, для перемещения пандуса, в станках, компрессорах, игрушках, самолетах, подводных лодках, пароходах и космических кораблях и т.п.[12]

Гейты

Это устройства, на которые обычно возлагают логику поверхностного анализа информации, поступающей от подключенных к ним сенсоров. В определённых ситуациях, анализ данных может требовать малого количества вычислительных ресурсов, так что гейты вполне способны принимать некоторые решения самостоятельно. Принимая такие решения, они отправляют определённые команды управления на актуаторы, которые, в свою очередь, выполняют уже свои функции.[13]
Если же обработка иформации требует больших затрат, или эта информация подлежит сбору, гейты отправляют её на сервера, где с ней и производится дальнейшая работа. Вполне себе вероятно использование в роли гейтов микрокомпьютеров или микропроцессоров, которые показаны на рисунке 4.

Для того, чтобы построить мониторинговую систему, достаточно будет использования лишь сенсоров и некоторые функции гейтов. Например, благодаря сенсору температуры можно без особых усилий организовать систему слежение за температурой почвы.

Но для использование датчиков в сфере сельского хозяйства требуется конструктивно датчики расположить в защищенных корпусах, которые

предохраняют конечные устройство от влияние внешней среды. Система позволяет в любой момент произвести замену датчиков в случаи повреждения.

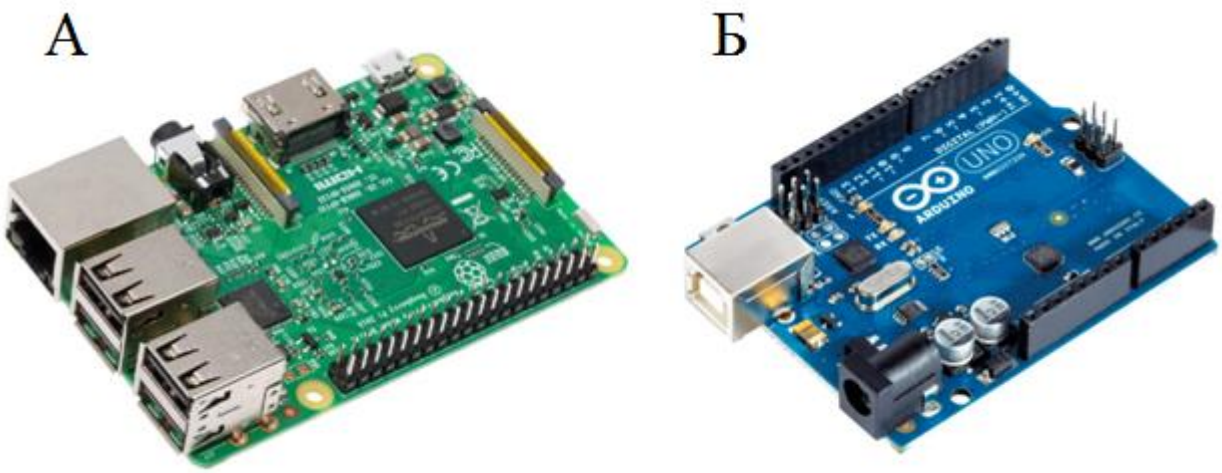


Рисунок 4 —– Микрокомпьютер (А) и микропроцессор (Б)

На выбор датчиков влияет большое количество факторов. Каждый раз нужно формулировать требования, учитывать все факторы и выбирать оптимальный датчик из десятков и сотен образцов с похожими названиями и разными характеристиками. Факторы, которые влияют на выбор датчика:

- габариты;
- точность измерения;
- диапазон измерений;
- энергоэффективность;
- и другие.[12]

И так подведем вывод данного раздела. В реализации проекта предлагается использовать сенсоры, которые будут подключены к простому микроконтроллеру. Требуется температурный датчик для контроля температуры почвы и датчик влажности, чтобы выполнить качественную посадку и сбор урожая. Предлагается воспользоваться такие же датчиками для мониторинга помещения хранения семян. Все датчики должны работать на одинаковых протоколах и иметь среднюю стоимость по рынку. Датчики будут иметь батареи питания, так как будут находиться в местах, где провести кабель

питание будет затруднительно. Все компоненты конечных устройств будут выбраны в последующих разделал.

2.3 Классификация беспроводных технологий

Проект требует технологию, которая будет выполнять передачу небольших сообщений в конкретные промежутки времени. Информация будет собираться с большой территории, на которой располагается ферма. В таких условиях проводные технологии уступают беспроводным, потому что кабели будут мешать при работе на фермерского хозяйства.

Интернет вещей радикальным образом увеличивает объем собираемых данных, что является следствием огромного количества источников информации. Ячейки, состоящие из вышеперечисленных типов устройств, находятся в периферии и для коммуникации должны использовать специальные протоколы взаимодействия. Такие протоколы имеют разные беспроводные технологии.[10]

На текущий момент беспроводные технологии представлены огромным многообразием, разделенным в основном по сферам применения. Однако существует много других способов их разделения по классам:

- дальность связи; (рисунок 5)
    1. WPAN (беспроводные персональные сети);
    2. WLAN (беспроводные локальные сети);
    3. WMAN (беспроводные сети масштаба города);
4. WWAN (беспроводные глобальные сети);

- топология;
    1. двухточечные (точка-точка);
    2. многоточечные (точка-многоточка);
- сфера применения;
    1. корпоративные;
    2. операторские.[14]

Рассмотрим категории беспроводных технологий более подробно на наиболее актуальном примере, взяв за критерий дальность связи.

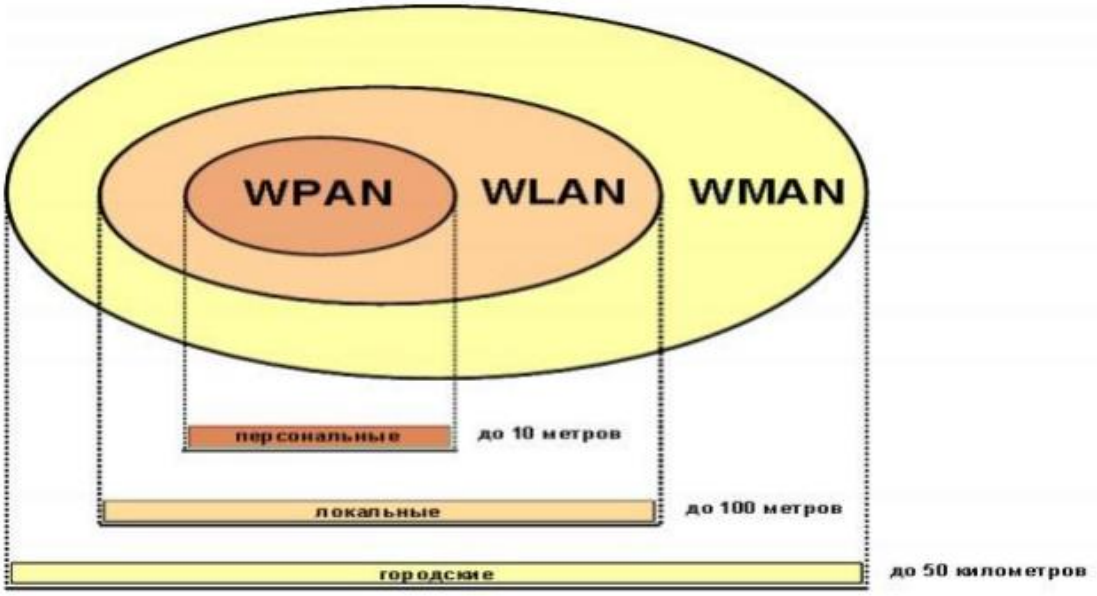


Рисунок 5 – Радиус действия беспроводных сетей

2.3.1 WPAN – Wireless Personal Area Networks

Беспроводные персональные сети применяются для связи различных устройств в условиях ограниченного пространства. Рабочей группой IEEE описан стандарт 802.15 излагающий способы функционирования таких сетей. В качестве актуального примера возьмем технологию Bluetooth. Стандарт Bluetooth является компромиссным с точки зрения соотношения параметров экономичность дальность скорость. Основная идея его создания заключалась в создании надежного, универсального и очень дешевого радиоинтерфейса беспроводного доступа. Технология позволяет обеспечивать сопряжение с множество профессиональным и бытовым оборудованием в режимах передачи речи, данных и мультимедиа. Принцип действия исходит из взаимодействия радиоволн. Радиосвязь происходит в нелицензируемом ISM диапазоне от 2.4 до 2.483 ГГц. Используется метод расширения спектра FHSS со скачкообразной перестройкой частоты.[15] К достоинствам стандарта можно отнести :

- высокий уровень стандартизации и совместимости между устройствами разных производителей;
- защита передаваемых данных;
- низкая стоимость;
- универсальность и большое разнообразие модулей под разные задачи.

В качестве недостатков можно упомянуть относительно высокое энергопотребление и невысокую скорость обмена данными.
Сформировавшаяся область применения представлена:

- автомобильная электроника;
- системы удаленного управления и телеметрии;
- компьютерная техника и телекоммуникационное пользовательское оборудование.

2.3.2 WLAN – Wireless Local Area Network

Беспроводные локальные сети используются для связи различных устройств в локально-вычислительную сеть без использования кабельных технологий. Рабочая группа IEEE на примере стандарта 802.11 описала нормы работы таких устройств, включающих в себя более 20 сертификаций. В качестве примера рассмотрим наиболее актуальную технологию Wi-Fi. Этот стандарт был спроектирован под нужды создания ЛВС из нескольких компьютеров. Сети, построенные с помощью кабелей, выделяются необходимостью множества сопроводительных работ связанных с прокладкой проводов внутри рабочих территорий. Беспроводные сети Wi-Fi лишены этих недостатков. Все устройства можно подключать с использованием минимального количества ресурсов. .[16] Рабочие диапазоны и способы модуляции различаются в зависимости от используемого стандарта и более подробно будут рассмотрены далее. К плюсам технологии относятся:

- высокая скорость передачи данных;
- компактность;

– большое разнообразие модулей под различные задачи;
– высокий уровень стандартизации и совместимости;
– в качестве недостатков можно выделить сравнительно большое энергопотребление и плохую защиту от взлома.

Области применения, продиктованные особенностями стандарта Wi-Fi:

– промышленность;
– общественные места;
– компьютерная и офисная техника;
– системы удаленного управления и телеметрии;
– частные беспроводные сети.

2.3.3 WMAN – Wireless Metropolitan Area Networks

Беспроводные сети масштаба города подразумевают под собой широкополосный доступ к сети с использованием радиоканала. Рабочей группой IEEE был создан стандарт 802.16 в котором описаны основные условия взаимодействия таких устройств. Как пример возьмем наиболее актуальную технологию WiMAX. Перед разработчиками всегда стояла проблема «последней мили» (канал, соединяющий оборудование пользователя с узлом доступа провайдера).WiMax обеспечивает доступ и соединяет между собой точки Wi-Fi, позволяет создавать точки удаленного доступа без привязки к географическому положению, обеспечивает системами удаленного мониторинга и т.д. [16] Способы модуляции и рабочих диапазонов сильно варьируются в зависимости от стандартов и будут указаны ниже в сравнительной таблице. К достоинствам стандарта относятся:

– легкость подключения;
– охват территории;
– мобильность;
– качество передачи;
– высокий уровень стандартизации;
– гибкость.

В качестве недостатков нужно упомянуть неподготовленность законодательной базы, дефицит частот и трудности внедрения технологии. В основном технология применяется для предоставления услуг высокоскоростного доступа в Интернет для бизнес-структур и частных лиц.

2.3.4 WWAN – Wireless Wide Area Networks

Беспроводные глобальные сети будет представлены наиболее актуальным стандартом беспроводной высокоскоростной передачи данных для мобильных телефонов и различных устройств LTE. Этот стандарт был разработан консорциумом 3GPP и является закономерным способом модернизации для операторов сетей GSM/UMTS. В дословном переводе «долгосрочное развитие». Использование нового метода цифровой обработки сигнала и модуляции позволило значительно увеличить пропускную способность и скорость передачи данных в мобильных сетях. Рабочая частота технологии находится в промежутке от 800Мгц до 3.5 ГГц. Используется три вида модуляции QPSK, 16QAM, 64QAM.[14] Преимущества стандарта:

- низкое значение задержки;
- высокая скорость;
- повышенная стабильность;
- доступность.

К недостаткам можно отнести возможные расхождения рабочих частот в разных странах. Технология LTE используется в основном для предоставления доступа к сети Internet посредством протокола IP и мобильной связи.

2.4 Сравнительный анализ стандартов беспроводной связи

В этом разделе за счет анализа стандартов следует выбрать какой стандарт подходит проекту SMART–АГРО. Для удобства анализа и восприятия информации ниже в таблице 1 приведены сравнительные характеристики основных современных беспроводных технологий.

Таблица 1 – Характеристики известных стандартов беспроводной связи

Технология	Стандарт	Область применения	Пропускная способность, Мбит/с	Дальность связи	Модуляция, доступ к среде	Частотный диапазон, ГГц
Bluetooth	802.15.1	WPAN	до 0,7	до 10 м	FHSS	2,4
	V4.0		до 2.3	до 60 м	GMSK	2,4
	V5.0		до 5	до 100 м	FGMSK	2.4
Wi-Fi	802.11g	WLAN	до 54	до 140 м	DQPSK	2,4
	802.11n		до 300	до 250 м	64QAM	2,4 или 5
	802.11ac		до 1000	до 500 м	256QAM	5-6
WiMAX	802.16d	WMAN	до 75	до 50 км	OFDM	1,5-11
	802.16e	WWAN	до 40	до 5 км	OFDM	2-13
	802.16m		до 1000	до 100 км	COFDM	2-66
LTE	LTE	WWAN	до 100	до 15 км	OFDM	0,8-3,5
	Advanced		до 1000	до 100 км	COFDM	0,8-3,5

В таблице перечислены основные технологии беспроводной связи, проведем краткое описание их.

Bluetooth – это технология беспроводной передачи данных на небольшое расстояние (несколько десятков метров максимум) между различными типами устройств: мобильные телефоны, наушники, автомобили, медицинские аппараты и мн. др. Сегодня эта технология нашла очень широкое распространение. Bluetooth является универсальной и недорогой технологией. Версия с низким энергопотреблением Bluetooth LE выпущенная в декабре 2015 года, наиболее существенным достоинством которой является сверхмалое пиковое энергопотребление, среднее энергопотребление и энергопотребление в режиме простоя.[15]

Wi-Fi — это аббревиатура, которая произошла от английского словосочетания Wireless Fidelity, что означает «беспроводная передача данных» или «беспроводная точность». Это система короткого действия, покрывающая

десятки метров и которая использует не лицензированные диапазоны частот для обеспечения доступа к сети. Это протокол и стандарт на оборудование для широкополосной радиосвязи, предназначенной для организации локальных беспроводных сетей. Другими словами, Wi-Fi — это современная и перспективная беспроводная технология, которая использует радиоканалы для передачи данных. Данная технология предполагает наличие точки доступа/маршрутизатора Wi-Fi (стандарты 802.11a/b/g/n), которая обеспечивает стабильный доступ к сети из некоторой области радиусом до 45 метров в помещении и 90 метров на открытом пространстве.[16]

Телекоммуникационная технология WiMax разработана с целью предоставления универсальной беспроводной связи на больших расстояниях для широкого спектра устройств (от рабочих станций и портативных компьютеров до мобильных телефонов). WiMax — это аббревиатура расшифровывается как Worldwide Interoperability for Microwave Access, что дословно в переводе означает «Международное взаимодействие для микроволнового доступа». Стоит сказать что WiMax не является более опасным для здоровья, чем обычная сотовая связь. Технология использует высокую степень защиты для передачи данных, что идеально подходит для ведения бизнеса. Сеть на базе этой технологии строится на основе базовых и абонентских станций и оборудования, связывающего между собой базовые станции, с поставщиком Интернета и других сервисов. Используемый рабочий диапазон от 1,5 до 11 ГГц. Скорость теоретически может достигать 70 Мбит/с. Не требуется прямая видимость между базой и приемником.[16]

LTE беспроводная сеть на основе стандарта четвертого поколения — 4G. LTE стандарт беспроводной высокоскоростной передачи данных для мобильных телефонов и других терминалов, работающих с данными. LTE является естественным обновлением как для операторов с сетью GSM/UMTS, так и для операторов с сетью CDMA2000. В разных странах используются различные частоты и полосы для LTE, что делает возможным подключать к

LTE-сетям по всему миру только многодиапазонные телефоны. Теоретически, LTE может работать на скорости 300 мбит/сек. На практике, все зависит от качества покрытия, расстояния до оборудования оператора и выбранного тарифного плана. Под работу сети выделено 40 диапазонов частот (так называемых, band). В каждой стране, в большей степени, используется свой band.[14]

Для фермерского хозяйства, которое имеет большую площадь, требуются базовые станции с радиусом покрытия в несколько километров. Таким радиусом обладают стандарты категории WMAN и WWAN. Из таблицы можно выбрать две технологии из этих категорий WiMAX и LTE.

Одним из главных является их энергоэффективность. Дело в том, что идея интернета вещей заключается в создании среды устройств, коммуницирующих между собой без участия человека. Стоит заметить, что в некоторых случаях полностью избежать вмешательства человека избежать не удастся. [10] Например, в системе подсчёта количества прошедших человек есть сенсор движения. Ему, как и любому другому электрическому устройству, необходимо питание. Проводить провода с питанием к каждому такому сенсору (если их больше 5 и они сильно разбросаны в пространстве) кажется не лучшей идеей. Соответственно, работать они будут от батареек или аккумуляторов. Если потребление заряда будет чрезмерным, элементы питания им нужно будет менять довольно часто. А это приведёт к тому, от чего стремится уйти интернет вещей — нужно же будет кому-то заменять эти батарейки. А вот если сенсоры будут энергоэффективны, то достаточно будет просто вставить батарейку и забыть об этом на год, два, пять и т. д.

Ещё одним преимуществом этих сетей является высокая помехоустойчивость. Каждый бит информации в этих сетях отправляется отдельным радиосигналом, поэтому его довольно просто выделить на фоне эфирного шума.[9]

После сравнения более всего распространены стандартов рассмотрим рынок более новых технологий, которые в основном используются для Интернета вещей, и имеют свою категорию LPWAN. LPWAN это беспроводная технология передачи небольших по объёму данных на дальние расстояния, разработанная для распределённых сетей телеметрии, межмашинного взаимодействия и интернета вещей. LPWAN является одной из беспроводных технологий, обеспечивающих среду сбора данных с различного оборудования: датчиков, счётчиков и сенсоров.[17] На рисунке 6 показан анализ технологий по скорости передачи данных и дальности передачи данных. В таблице 2 представлены некоторые технологии LPWAN и сравниваются с технологией мобильной связи.

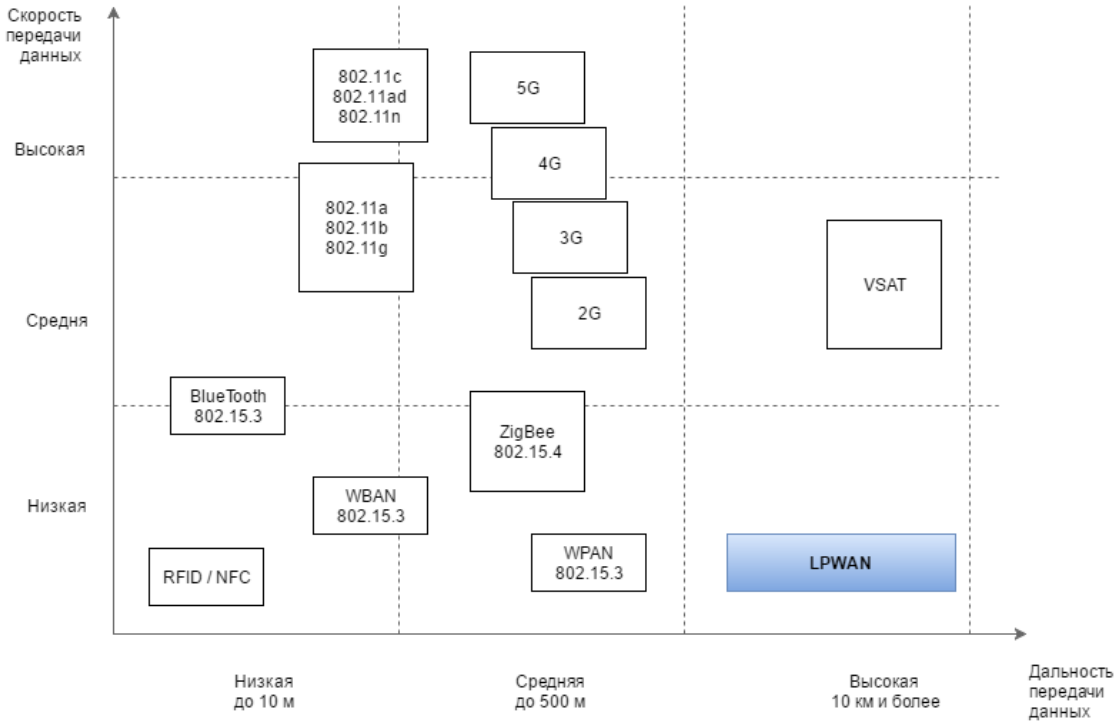


Рисунок 6 – Функциональное соотношение протоколов

В таблице приведены только некоторые технологии, но проблема не в том, что нет технологий подходящей для проекта, а том что многие системы не распространены в нашей стране. Например МТС и Мегафон на основе технологии NB-IoT .[17]

Таблица 2 – Характеристики систем связи дальнего радиуса действия

Технические характеристики	LoRa	SIGFOX	Weightless P	GSM/ GPRS
Метод модуляции	CSS	–	FDMA / TDMA	FDMA / TDMA
Скорость	0,3–50 кбит/с	100 бит/сек	0,2–100 кбит/сек	До 384 кбит/с
Полоса	Широкополос. до 500 кГц	Узкполос. 100 кГц	Узкполос. 12,5 кГц	Широкополос. от 200 кГц
Время автономии	> 10 лет	3 года	3–5 лет	–
Дальность	До 5 км в городе, до 45 км вне города	До 10 км в городе, до 50 км вне города	До 2 км в городе	от 400 м до 35 км
Чувствительность	до -148 дБм	до -100 дБм	–	-103 дБм

Одновременно сравнивая стандарты беспроводной сеты, были выделены технологии LoRaWAN и ZigBee. Обе эти сети являются очень медленными в сравнении, например, с 4G или даже с 3G, большой радиус покрытия и высокой помехоустойчивостью. Все три технологии имеют разность между собой, которая показана в таблице 3.

Как можно заметить, LoRa имеет большую дальность и меньшую скорость передачи данных. В данном проекте не требуется большие потоки информации, а требуется большая дальность базовой станции, для того чтобы на одно хозяйство требовалось как можно меньше базовых станций.

Делая вывод по данному разделу, анализ беспроводных технологий показал, что к проекту START–АГРО больше всего подходит технология LoRaWAN.

LoRaWAN: предназначенный для беспроводных устройств с батарейным питанием в региональной, национальной или глобальной сети, LoRaWAN нацелен на ключевые требования IoT для обеспечения безопасной,

маломощной, двунаправленной связи с мобильностью и услугами локализации на широкой территории. LoRaWAN обеспечивает бесшовную и долгосрочную совместимость между устройствами IoT без необходимости поддержки локальной сети.[6]

Таблица 3 –Сравнение характеристик развивающихся технологий

	sigfox	LoRa	NB-IOT
Выход на рынок, г.	**2009**	**2012**	**2017**
Количество соединений	9 млн.	Нет данных	10 тыс.
Диапазон частот	Нелицензируемый в области 900 МГц	До 900 МГц	Лицензируемый диапазон частот LTE, защитная полоса частот, индивидуальная полоса
Максимальная скорость передачи данных, кбит/с	0,1	0,3–50	170–250
Стратегия управления сетями	Сеть, управляемая Sigfox	Сеть, которая может управляться кем угодно	Сеть, которая может управляться любой телекоммуникационной компанией
Стратегия использования оборудования	Открыто для всех поставщиков оборудования	Открыто для всех поставщиков оборудования, но только с лицензионными компонентами	Открыто для всех поставщиков оборудования
Публичная/частная сеть	Публичная	Публичная и частная	Публичная
Типичный пример использования	Связанные «умные» мусорные баки	Точное земледелие/удаленная поливка	Умный мониторинг приборов учета

2.5 Программное обеспечение и пользовательский интерфейс

Для корректной работы любой вычислительной машины или системы требуется объединить компоненты не только аппаратно, но и программно. В 1958 году статье журнала American Mathematical Monthly математик из Принстонского университета Джон Тьюки первый раз упомянул понятие «software».Собственно сам термин «программное обеспечение» вошёл в широкий обиход с начала 1960-х годов, когда стало актуальным разграничение команд, управляющих компьютером, и его физических компонентов. Тогда же и началось становление индустрии программного обеспечения, как самостоятельной отрасли.[18] Сейчас ПО понимается как программа или множество программ, используемых для управления компьютером. Если говорить более техническим языком программное обеспечение — это совокупность программ, позволяющих осуществить в системе

автоматизированную обработку информации. Программное обеспечение делится на системное (общее) и прикладное (специальное).

Системное программное обеспечение — комплекс программ, которые обеспечивают управление компонентами системы. Например, персональный компьютер обладает такими компонентами как процессор, оперативная память, устройства ввода-вывода, сетевое оборудование, выступая как «межслойный интерфейс». Прикладная программа или приложение — программа, предназначенная для выполнения определённых задач и рассчитанная на непосредственное взаимодействие с пользователем. В отличие от прикладного программного обеспечения, системное не решает конкретные практические задачи, а лишь обеспечивает работу других программ, предоставляя им сервисные функции, абстрагирующие детали аппаратной и микропрограммной реализации вычислительной системы, управляет аппаратными ресурсами вычислительной системы и системы.[18]

При разработки системы основанной на технологии интернета вещей требуется объединить различные компоненты в одну систему единой сетью, например как взаимодействуют компоненты персонального компьютера. Технология интернета вещей тоже имеет подобное ПО, и образует единую экосистему IoT. Она представляет собой промежуточный уровень между уровнем вещей (уровнем сенсоров) и уровнем приложений. IoT платформа и программное обеспечение предназначены для подключения интернет вещей (датчиков, контроллеров и других устройств) к серверу и удаленного доступа к ним с помощью сети интернет. По сути – это ключевой инструмент разработки IoT-приложений и сервисов, объединяющий физические объекты и клиента. [6]

Для понимания архитектуры проекта разделим термины платформа IoT и программное обеспечения. Платформа является ПО, но ее главная задача объединить все уровни системы мониторинга с помощью единого протокола передачи данных. Но кроме этого требуется программные надстройки системы, предназначенные для упрощения общения пользователя с базой данных,

которая хранит и обрабатывает информацию от датчиков. Все эти задачи выполняют специальные сервисные программы.[3] В основном любое программное обеспечение аппаратно находится на сервере, из за этого сложно сказать какое ПО отвечает за какую либо задачу. Поэтому в выпускной работе позволим разделение на платформу IoT и другого ПО.

Стоит отметить, что на рынке понятие платформы IoT–это технологии, которые обеспечивают бесшовную интеграцию различных аппаратных средств, используя протоколы связи, применяя различные типы топологии. Таким образом, можно использовать от разных фирм различные датчики, сенсоры, базовые станции и микроконтроллеры и объединить их с помощью единой платформы. IoT платформы используются поставщиками и производителями умных устройств для оснащения своих продуктов функциями дистанционного управления, мониторинга в режиме реального времени, настраивания предупреждений и уведомлений, интеграции со смартфонами и другими устройствами.

Очень много областей применения подобных IoT платформ, например в промышленном секторе (так называемый IIoT), так же используются при создании систем умного города для предоставления различных услуг частным и государственным компаниям. Самыми популярными программными IoT платформами являются: Microsoft Azure IoT, Amazon Web Services (AWS) IoT, Google Cloud, ThingWorx IoT, IBM Watson, Artik от Samsung Electronics, Cisco IoT Cloud Connect, Salesforce IoT Cloud и многие другие. Описание подобной системы встречается в следующей главе.[10]

Все эти инструменты, которые были описаны ранее, позволяют обеспечивать передачу информации между пользователем-человеком и аппаратными компонентами системы мониторинга. Но клиент должен получить уже конечных результат мониторинга. Для этого на клиентской части или приложении требуется создание понятного интерфейса. Пользовательский

интерфейс обеспечивает передачу информации человеку от программно-аппаратных компонентов системы.

Клиент может не только получать цифры, графики и звуковые сообщения, но и вводить запросы. По наличию тех или иных средств ввода, интерфейсы разделяются на типы — жестовый, голосовой, нейрокомпьютерный интерфейс, и т. д., возможны смешанные варианты. Средства эти должны быть необходимыми и достаточными, быть удобными и практичными, расположенными и скомпонованными разумно и понятно, соответствовать физиологии человека, не должны приводить к негативным последствиям для организма пользователя.[11]

Для простого управления системой мониторинга любому пользователю хочется получить удобный и понятный интерфейс. Поэтому при разработке проекта следует проанализировать какой вид интерфейса следует выбрать, его дизайн, а так же возможность конкуренции на рынке.

Одной из главных задач проекта является серьезное изучение программного обеспечение, построение серверов и его управление, создание правильной логики анализа данных, создание безопасной связи всех уровней архитектуры и анализ платформ IoT .

2.6 Выводы по главе

В данной главе описывается концепция технологии интернета вещей, развитие данной технологии, описание архитектуры технологии, а так же была предложена архитектура системы мониторинга на основе интернета вещей. Далее были описаны и проанализированы основные компоненты архитектуры технологии интернета вещей, а именно конечные устройства, технологии беспроводной связи, программное обеспечение и пользовательский интерфейс.

В конце главы получили следующие выводы. Для реализации проекта мониторинга на основе технологии интернета вещей предлагается использовать архитектуру, которая имеет четыре функциональных уровня:

1.Объекты и сенсоры, которые собирают информацию.

2.Базовые станции какой либо технологии беспроводной связи.

3.IoT платформа для обработки и анализа информации.

4.Устройства и приложения для мониторинга информации.

В реализации проекта предлагается использовать в качестве конечных устройств только сенсоры. Все датчики должны работать на одинаковых протоколах и иметь среднюю стоимость по рынку. Датчики будут иметь батареи питания, так как будут находиться в местах, где провести кабель питание будет затруднительно.

Анализ беспроводных технологий показал, что к проекту больше всего подходит технология LoRaWAN. LoRaWAN нацелен на ключевые требования IoT для обеспечения безопасной, маломощной, двунаправленной связи с мобильностью и услугами локализации на широкой территории.

Важно выбрать удобную платформу IoT для связи всех компонентов системы с общим протоколом передачи данных. Платформа должна выполнять обработку информацию для передачи ее пользователю. Проект требует создания удобного и простого интерфейса для конкуренции на рынке.

3 Этапы построения системы мониторинга с использованием технологии интернета вещей

3.1 Выбор инструментов реализации

Как говорилось в прошлой главе для реализации проекта требуется объединить основные компоненты системы мониторинга на основе технологии интернета вещей. Для реализации проекта потребуется знания в разных направлениях, например программирование конечных устройств, знание серверного языка, умение в создание сети с множеством устройств, креативное подход для создания клиентского интерфейса.[2]

На данном этапе следует выбрать конкретные компоненты и способы реализации проекта, поэтому в данной главе опишем какими решениями и инструментами будет создаваться проект. Воспользуемся наработками курсах IoT Академия Samsung, которые я проходил около полу года. На данных курсах кратко знакомились с рынком компонентов технологии интернета вещей, проходили программирование конечных устройств и программного ПО, а как же учились объединять все уровни в единую систему на основе технологии интернета вещей.

Для выполнения поставленной задачи нам будет необходимо реализовать:

– обширную сеть конечных устройств: температурный датчик и датчик влажности. Подобрать мини метеостанцию для отслеживания изменений в климате.

– применить технологию беспроводной передачи данных LoRaWAN, которая обеспечивает бесшовную и долгосрочную совместимость между устройствами IoT без необходимости поддержки локальной сети.

– выбрать подходящею IoT платформу для общего сбора информации.

– серверную часть, содержащую основную часть логики приложения.

– клиентскую часть, позволяющую пользователю в удобном виде просматривать информацию.

Для поиска и сбора данных требуется выбрать датчики с конкретными характеристиками, поэтому требуется изучить рынок конечных устройств.

Сбор и передачу данных будет осуществляться благодаря технологии LoRaWAN, значит следует ознакомится с характеристиками предлагаемых базовых станций, а так же следует изучить протоколы передачи.

На курсах IoT Академия Samsung предлагалась одна из IoT платформ, таким образом требуется более углубленно изучить подобную технологию.

Для реализации серверной и клиентской частей необходимо выбрать язык программирования и используемую базу данных. Возможные варианты выбора серверного языка программирования: PHP, ASP.NET, Java, Python (Django). А для клиентской части подойдут следующие языки CSS, HTML, JavaScript. Далее рассмотрим более подробно каждый выбор инструментов реализации проекта, проведем исследование рынка компонентов, проанализируем сетевые проколы и выберем удобные языки програмирования.

3.2 Выбор конечных устройств и их установка

В прошлой главе был сделан вывод, что для создания системы мониторинга потребуется только сенсоры, так как требуется только сбор информации. Другие конечные устройства могут быть установлены для дополнительных функций, например различные реле, светодиоды и другие. Сенсоры должны собирать информацию и передавать ее по радиоканалам на базовую станцию беспроводной связи. Таким образом, надо подобрать датчики для мониторинга температуры, влажности, давления и освещения, а так же выбрать радиомодуль для передачи данных по беспроводной технологии.

Начнем анализ рынка конечных устройств с Unwired Range— это семейство устройств с поддержкой технологии LoRa.[19] Unwired Range обеспечивает и настраивает модули для Интернета вещей, а так же разрабатывает прошивку на сенсоры, которые подходят для IoT.Данная фирма

знакома из куров IoT Академии Samsung. Изучая предложения Unwired Range были выбраны следующие датчики:

- bme280 Датчик температуры, влажности и давления воздуха;
- lm75 Внешний датчик температуры;
- sht21 Датчик температуры и влажности воздуха;
- lps331 Датчик барометрического давления;
- opt3001 Датчик внешней освещённости;
- lmt01 Четыре внешних датчика температуры.

По проекту датчики передают информацию об температуре и влажности почвы и помещений, давление в посещениях, а так же информация об освещенности. Все выбранные датчики подходят для сбора подобной информации, но для данного проекта желательно задействовать небольшое количество датчиков. Это объясняется тем, что датчики не должны занимать много места и быть компактными. На одном фермерском участке может находится больше сотни датчиков, поэтому прилагается как можно меньше видов датчиков, для того чтобы система была простой в использовании и обслуживании. Таким образом, предлагается рассмотреть сначала два датчика bme280 и opt3001 более подробно.

Bosch Sensortec BME280 представляет собой комбинированный цифровой датчик влажности, давления и температуры, у которого основанные принципы это считывание информации. Сенсорный модуль размещен в чрезвычайно компактном корпусе с металлической крышкой, с размером всего 2,5×2,5 мм² при высоте 0,93 мм. Его небольшие размеры и его низкий энергопотребление позволяет реализовать в устройствах с батарейным питанием. BME280 достигает высокой производительности во всех системам, требующих измерения влажности, температуры и давления.[20]

Датчик влажности обеспечивает минимальное время отклика для быстрой обработки в системе и высокая общая точность во всем диапазоне. Датчик

давления представляет собой датчик абсолютного барометрического давления с чрезвычайно высокой точностью и низким уровнем шума.

Встроенный датчик температуры был оптимизирован для минимального шума и самого высокого разрешения. Его выход используется для температурной компенсации датчиков давления и влажности и может также использоваться для оценки температуры окружающей среды.[20]

Рабочий диапазон:

– давление: 300 …1100 гПа ;

– температура: минус 40…плюс 85°С;

– напряжение питания VDD минус 1,71…плюс 3,6 В;

– средний потребляемый ток (Частота обновления данных 1Гц) - 1,8 мкА 1 Гц (H, T) - 2,8 мкА 1 Гц (P, T) - 3,6 мкА 1 Гц (H, P, T) T = температура, H = влажность, P = давление;

– средний потребляемый ток в режиме ожидания 0.1 мкА.

Датчик влажности: Время отклика 1s Допуск точности ± 3% относительной влажности Гистерезис 2% относительной влажности

Датчик давления: RMS шума 0,2 Па (экв. до 1,7 см) Ошибка чувствительности ±0,25% (экв. до 1 м при 400 м изменении высоты)

Температурный коэффициент смещения ±11,5 Па/К (экв. до ±12,6 см при изменении температуры на 1°С).[20]

Если данный датчик подойдет для проекта SMART–АГРО, тогда всю остальную подробную информацию можно найти в паспорте BME280 комбинированный датчик влажности, температуры и давления.

Цифровой оптический датчик OPT3001 от Texas Instruments позволяет измерять освещенность в соответствии со спектральной характеристикой человеческого глаза и не реагирует на инфракрасные источники излучения. OPT3001 измеряет освещенность в пределах до 0,01 до 3886 люкс, что примерно соответствует диапазону от лунной ночи до солнечной освещённости при чистом небе. OPT3001 предназначен для тех приложений, где необходимо

реагировать на уровень освещенности именно в том виде, как это воспринимается человеком.[21]

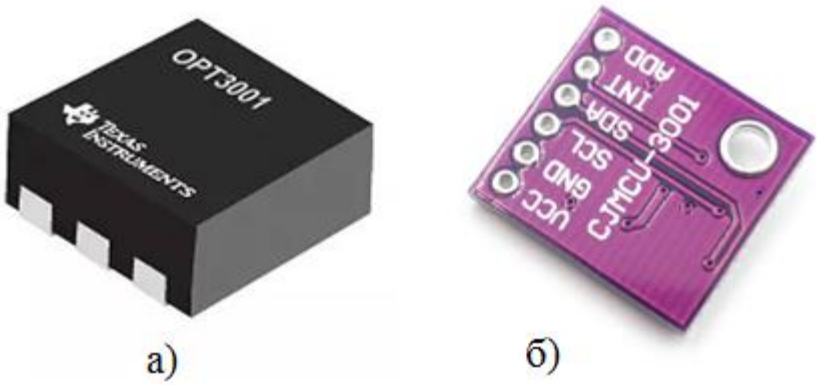
а) в корпусе

б) без корпуса

Рисунок 7 – Датчик Bosch Sensortec BME280

Оптический датчик OPT3001 является предпочтительной заменой для фотодиодов, фоторезисторов и других приборов, показания которых существенно зависят от инфракрасных составляющих спектра. Но растения состоят преимущественно из воды, поэтому под действием инфракрасного излучения они быстро нагреваются. Вода очень хорошо поглощает инфракрасные лучи. Следовательно, отражение инфракрасных лучей до того, как они попадут в теплицу, оказывает большое воздействие на температуру растений. Таким образом, данный датчик будет не полностью выполнять свои функции, а именно не будет показывать в какой момент растение будет нагреваться. Данную проблему можно решить двумя способами. Первый способ, подобрать другой датчик для фиксирования инфракрасного излучения. Второй способ, контролировать инфракрасного излучения за счет мониторинга температуры около растений.

Теперь давайте сравним выбранные датчики с другими датчика похожего функционала. Рассмотрим уже перечисленный датчики от Unwired Range.



а) в корпусе

б) без корпуса

Рисунок 8 – Датчик OPT3001

слева–в корпусе справа–без корпуса

LM75 представляет собой температурный датчик. Не дорогой датчик температуры с шиной I2C, датчик применяется в системах термозащиты. Хост может в любой момент послать LM75 запрос на измерение температуры. Хост может программировать как критический порог температуры (TOS), так и температуру, при которой критическая ситуация ликвидируется (THYST). Помимо этого, хост может считывать значения TOS и THYST регистров LM75. [22] На рисунке 9 изображен датчик LM75.

Питание от источника с напряжением от 3,0В до 5,5В, низкий потребляемый ток и I^2C интерфейс - всё это существенно расширяет область применения LM75. А именно: возможность использования LM75 для терморегулирования и как средства защиты в персональных компьютерах, электронном испытательном оборудовании и в офисной технике. Таким образом, данный датчик не подходит для работы в сфере сельского хозяйства. [22]Характеристики датчика:

- Диапазон температур: минус 55 °C … плюс (125 ±2) °C;
- Напряжение питания: 2,8V … 5,5V
- Связь по I2C, Поддерживает до восьми датчиков LM75 на шине I2C.

– Имеет «встроенный термостат» функцию управления цифровым выходом в зависимости от температуры.[22]

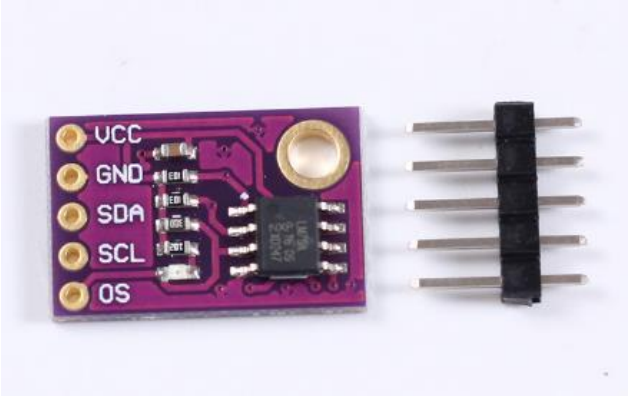


Рисунок 9 – Датчик LM75

Датчик температуры и влажности — SHT21 от производителя Sensirion. Компания собиралась установить новые стандарты с точки зрения размера и интеллекта, поэтому датчик один из самых маленьких в мире, 3x3 мм и Высота 1,1 мм. Сенсор имеет чип CMOSens®, датчик влажности емкостного типа и датчик температуры запрещенной зоны производительности был поднят даже за выдающийся уровень предыдущее поколение датчиков (SHT1x и SHT7x). Для Например, были приняты меры для стабилизации поведение при высоких уровнях влажности.[23]

Каждый датчик индивидуально откалиброван и протестирован. Много идентификация напечатана на датчике и электронном идентификационный код хранится на чипе - который может быть зачитал по команде. Датчик соотношение производительности и цены было улучшено и в конце концов, любое устройство должно извлечь выгоду из современный режим энергосбережения. [23] Данный датчик показан на рисунке 10.

Сам сенсор обладает следующими отличительными чертами:

– малые размеры. 3×3×1,1мм —один из маленький датчик влажности в мире. калиброванный, корпус DFN;

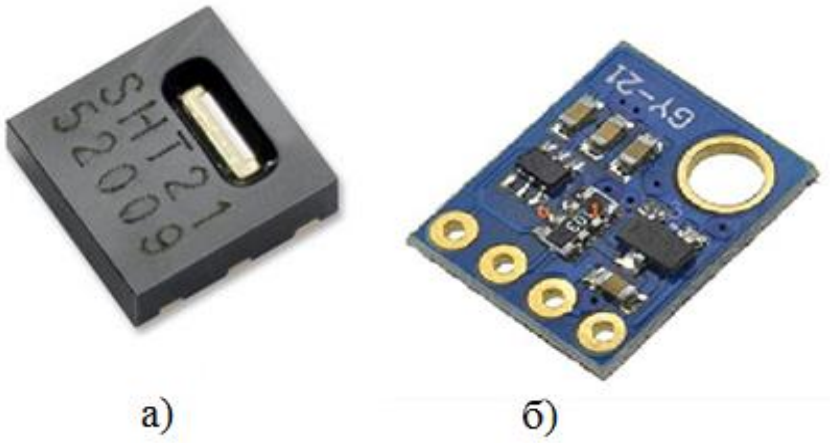
– датчик температуры с диапазоном минус 40 °C … плюс 125 °C, Точность измерения температуры: ±0,3 %;

– датчик влажности с диапазоном 0%RH — 100%RH, Точность измерения влажности: ±2 %;

– интерфейс: I2C, ШИМ, SDM;

– потребление в режиме измерения — 300uA, в спящем режиме — 0.15uA;

– напряжение питания: 2,1…3,6 В.[23]



а) в корпусе

б) без корпуса

Рисунок 10 – Датчик SHT21

Датчик LPS331 барометр, позволяет определить давление от 260 миллибар до 1260 миллибар (26 кПа до 126 кПа) с абсолютно точностью до ±2 миллибар (0,2 кПа) и среднеквадратичным значением шума в 0,02 миллибар (0,002 кПа) в высокоточном режиме. Это давление можно легко перевести в высоту. Плата поддерживает I²C и SPI интерфейсы. На плате установлен стабилизатор напряжения 3,3 В и схема смещения логического уровня от 2,5 до 5,5 В. Расстояние между выводами 2,54 мм делает плату простой в использовании со стандартными макетными и монтажными платами. Эта компактная (10,2 мм × 22,9 мм) плата, созданная на основе микросхемы ST LPS331 по МЭМС-технологии является датчиком абсолютного давления или барометром.LPS331 - это великолепный датчик , но маленький корпус делает её

использование затруднительным. Она также работает от напряжения ниже 3,6 В, что затрудняет её подключение к микроконтроллерам работающим от 5 В. Эти проблемы решены путём добавления дополнительных электронных компонентов, в том числе 3,3 В стабилизатора напряжения и схемы смещения логического уровня, сохраняя при этом компактный размер устройства. [24] На рисунке 11 показан сенсор LPS33 . Итак, основные параметры:

- рабочее напряжение: от 2,5 до 5,5 В;
- потребляемый ток: 2 мА;
- выходной формат (I^2C/SPI): 24-битное значение давления (4096 LSB/мбар);
- диапазон чувствительности: 260-1260 мбар (от 26 до 126 кПа);
- размеры (без штырьевых разъёмов): 10×23×3 мм;
- вес (без штырьевых разъёмов): 0,6 гр.[24]

Рисунок 11 – Датчик LPS331 в корпусе

Компания Texas Instruments имеет цифровой датчик температуры LMT01 с разрешающей способностью выше 0.1°C и работающий по двухпроводной линии. Температура выдается в виде количества импульсов, которое прямо пропорционально измеряемой температуре. Импульсы результата идут по тем же линиям что и питание датчика. Такой метод не требует формирования точных задержек и существенно упрощает программу микроконтроллера — достаточно подать питание на LMT01 и затем подсчитать число поступивших импульсов. В зависимости от температуры LMT01 выдает от 26 (-50°C) до 3218 импульсов (+150°C). Значение каждого импульса 0,0625°C. Импульсы следуют с частотой 88 кГц и могут быть подсчитаны

разными способами: программно, в прерывании от изменения сигнала на порту или помощью таймера в режиме счетчика. [25] Данный датчик изображен на рисунке 12.

Технические характеристики LMT01:

– высокая точность в широком диапазоне температур: минус 50°C … плюс 150°C: минус 20°C … (плюс 90 ±0,5°)C (макс.); 90°C …(150±0,62)°C; (макс.); минус 50°C … минус (20 ±0,7)°C (макс.);

– количество выходных импульсов пропорционально температуре с разрешением 0,0625°C;

– прецизионное цифровое измерение температуры Высокая помехоустойчивость, кабель до 2 м;

– тактовая частота: 88 кГц;

– непрерывное измерение с периодом 100 мс;

– потребляемый ток: 34 мкА «Плавающее» рабочее напряжение 2…5,5 В с защитой от ЭМИ;

– корпус: TO-92/LPG - размера обычного TO-92.[25]

Рисунок 12 – Датчик LMT01

После изучения рынка датчиков, делаем вывод, что очень много разных сенсоров, которые имеют похожие параметрами. Поэтому предлагается воспользоваться сенсорами уже знакомыми из курсов IoT Академин Samsung.

Теперь стоит выбрать какие из приведенных датчиков подойдут для данного проекта. Предлагается воспользоваться конечным устройством

bme280 ,который измеряет температуру, влажность и давления воздуха. Достоинства датчика, во–первых, он небольшого размера и он имеет жесткий корпус. Во–вторых, bme280 измеряет сразу несколько параметров.

Вторым датчиком будет датчик внешней освещённости opt3001. Как было ранее сказано, он измеряет освещенность в пределах 0,01…83886 люкс, что примерно соответствует диапазону от лунной ночи до солнечной освещённости при чистом небе. Этот сенсор позволит контролировать свет, который попадает на растении, а так же контроль за хранилищем семян.

Но датчик это не целое конечное устройство. Датчик — всего лишь деталь. Очень часто люди называют «датчиком» весь предмет, который мы подключаем к интернету вещей. Это ошибка: предмет собирают из разных компонентов, в том числе датчика, выбирая элементы под конкретного заказчика и задачу.[12] Рассмотрим рисунок 13, на котором показаны элементы конечного устройства.

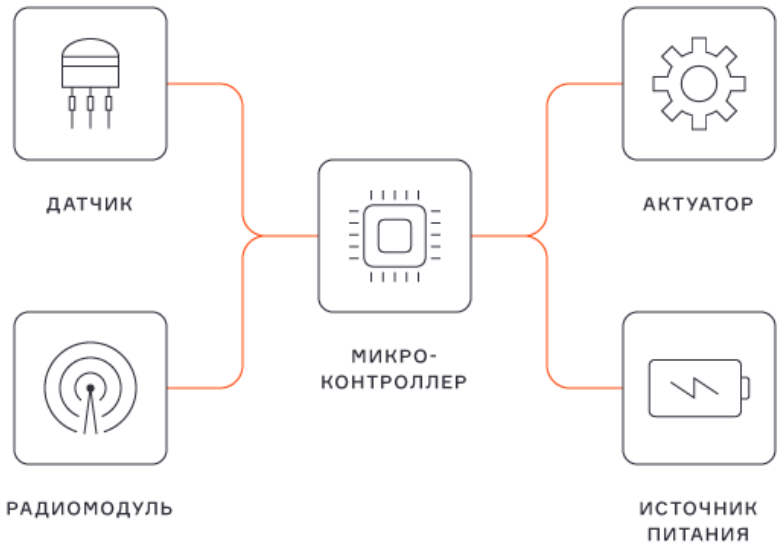


Рисунок 13 – Элементы готового конечного устройства

Из рисунка видно что главной компонентом является микроконтроллер. Это простой бортовой компьютер, который получает сигнал от датчика, обрабатывает его и решает что делать дальше. Так же он реализует логику работы всего устройства. Например, датчик может отдавать показания раз в

секунду, а микроконтроллер будет принимать решение, передавать ли данные человеку. Если мы программируем его реагировать только на температуру выше +30 С, он отправит нам уведомление лишь когда она превысит эту норму.[13]

Как уже было сказано, датчики Это чувствительные элементы, которые контактируют с окружающей средой: измеряет показания, реагирует на объекты и создает сигнал об этом. [12]

Для связи конечного модуля и базовой станции технологии передачи данных устанавливают радиомодуль. Например, когда требуется открыть дверь с умным замком, на его радиомодуль поступает входящий запрос от телефона или пропуска.[8]

Актуатор (исполнительное устройство) — это реле или транзистор, который мы добавляем к устройству, чтобы оно могло что-то переключать — по сигналу от микроконтроллера или по дистанционной команде, которую примет радиомодуль. Например, если температура выйдет за допустимый предел, актуатор может включить вентиляцию или кондиционер, а вы получите уведомление об этом. Как говорилось ранее, данный компонент не потребуются при разработке системы мониторинга.[8]

Еще один важный компонент это источник питания. Тут все просто — электронике нужно электричество. В зависимости от энергопотребления и задачи мы можем питать устройство от батарейки или от сети по проводу.[12]

На рынке огромное количества разных компонентов для устройств интернета вещей. Для проекта подойдут не больших размеров устройства, которые не требуется мощное логическое устройство, но требуется быстрая передача информации. Предлагается воспользоваться знакомым из курсов IoT Академии Samsung учебная плата от фирмы Unwired Devices. Она делится на две части: Unwired Range и UMDK-RF. Unwired Range(UNWR) содержит в себе микроконтроллер STM32 и приемопередатчик LoRa, таким образом, является минимальным конечным устройством “Интернета вещей” Плата UMDK-RF это

адаптер-переходник. К ней можно подключать UNWR, различные датчики и исполнительные устройства. [19]Для этой системы будет требоваться небольшой аккумулятор, например по типу Li Ion (Литий-ионный аккумулятор) На рисунке 14 изображена учебная плата.

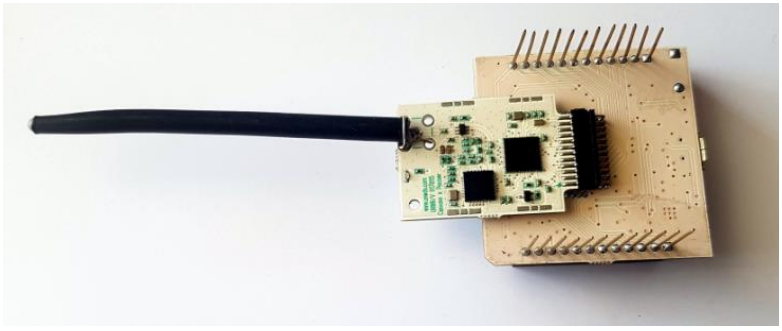


Рисунок 14 – Учебные платы Unwired Range и UMDK-RF

По рисунку видно, что данное оборудование мало подходить для сферы сельского хозяйства, так как антенна и линейки штырьков могут погнуться. При серийном производстве предлагается использовать либо металлический корпус собственного производства, либо выбрать более подходящие компоненты. Например, радиомодуль LRTX-868-PCB-CAAT от компании Лартех Телеком. Это приемопередатчик предназначенный для встраивания в различные устройства и организации обмена данными между устройствами и базовыми станциями радиосети. Он имеет собственный микроконтроллер ATSAMD20E18A-U, который может обеспечить логику для сенсоров. Интегрированная PCB-антенна обеспечивает дальность связи до 3,5км в плотной городской застройке и до 15км в зоне прямой видимости.[26] Данные модуль можно спрятать в небольшом корпусе и , который будет пропускать радиочастоты и иметь отверстья для датчиков. На рисунке 15 показана плата радиомодуля LRTX-868-PCB-CAAT.

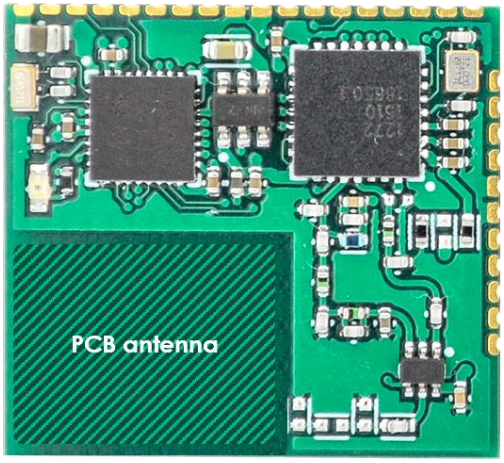


Рисунок 15 – Радиомодуль LRTX-868-PCB-CAAT

Таким образом, в данной проекте будут применяться два типа датчиков bme280 и opt3001, которые будут передавать показания температуры, влажности, давления и освещенности. Эти данные позволят фермерам делать выводы в каком состоянии находится почва для растений и какая окружающая среда в хранилище семян. Другие компоненты предлагается взять из курсов IoT Академии Samsung. Проектирование и программирование конечных устройств работа специалиста.

3.3Передача данных

После выбора датчиков надо установить с ними связь, по которой будет передаваться нужная информация. Как было ранее сказано, для передаче данных предлагается использовать технологию LORAWAN, которая имеет особенности при приеме и передаче информации. В этом разделе разберем особенности передачи данных, выберем подходящею топологию, проанализируем состояния рынка и опишем построение беспроводной сети передачи данных.

О том что следует строить сеть говорит тот факт, что проект имеет как большое количество датчиков и других элементов. Сеть передачи данных — совокупность трёх и более конечных устройств (терминалов) связи, объединённых каналами передачи данных и коммутирующими устройствами

(узлами сети), обеспечивающими обмен сообщениями между всеми конечными устройствами.

Передача данных — физический перенос данных в виде сигналов от точки к точке или от точки к нескольким точкам средствами электросвязи по каналу передачи данных, как правило, для последующей обработки средствами вычислительной техники. Примерами подобных каналов могут служить медные провода, ВОЛС, беспроводные каналы передачи данных или запоминающее устройство.[14]

На сегодняшний день большое развитие в области передачи данных получили беспроводные сети — сети радиосвязи. Это объясняется удобством их использования, дешевизной и приемлемой пропускной способностью.[10] Исходя из текущей динамики развития, можно сделать вывод о том, что беспроводные сети в данном проекте превосходят проводные сети. Данном проекте применяются беспроводные каналы связи.

Для построения сети передачи данных на основе технологии LoRaWAN, следует определится с топологией данной сети. Топология сети – геометрическая форма и физическое расположение устройств по отношению к друг другу. Топология сети позволяет сравнивать и классифицировать различные сети.[18] Обычно беспроводные технологии делят на две топологии «Точка-точка» и «Точка-многоточка», но можно рассмотреть и три основных вида топологии:

- шина;
- кольцо;
- звезда.

При построении сети по шинной схеме каждое устройство присоединяется к общему кабелю, на концах которого устанавливаются терминаторы.[27]

Топология кольцо представляет собой последовательное соединение, когда последний соединён с первым. Сигнал проходит по кольцу от устройства к устройству в одном направлении. Каждый сенсор работает как повторитель, усиливая сигнал и передавая его дальше. Поскольку сигнал проходит через каждый компьютер, сбой одного из них приводит к нарушению работы всей сети.[27]

При быстром анализе можно понять что первые две топологии не подходят для данного проекта, так как большое количество устройств не позволит большое количество соединений. Так же можно сделать вывод что данные топологии подходят больше для проводных систем. Поэтому рассмотрим подробнее топологию звезда.

Топология «Звезда» – схема соединения, при которой каждый устройство подсоединяется к сети при помощи отдельного канала связи. Эта топология с явно выделенным центром, к которому подключаются все другие абоненты. Весь обмен информацией идет исключительно через так называемый центральный компьютер, на который таким способом ложится очень большая нагрузка.[27] В технологии LoRaWAN таким компьютером выступает базовая станция, которая собирает информацию и передает ее на клиенту, а клиент может управлять через БС датчиками. Таким образом, никакие конфликты в сети с топологией «звезда» в принципе невозможные, потому что управление полностью централизовано. Устанавливать сеть топологии «Звезда» легко и недорого. Число узлов, которые можно подключить к базовой станции определяется возможностью самой станции. Рабочая группа, созданная по данной схеме может функционировать независимо или может быть связана с другими рабочими группами, что делает возможным использование других топологий при связи базовой станции, сервера и клиентской части. На рисунке 16 приведена схема соединения датчиков и базовой станции по типу звезда.

Теперь оценим достоинства и недостатки данного рода топологии сети.

Достоинства данного рода топологии сети.

1. Подключение новых рабочих устройств не вызывает особых затруднений.

2. Возможность мониторинга сети и централизованного управления сетью.

3. При использовании централизованного управления сетью локализация дефектов соединений максимально упрощается.

4.Хорошая расширяемость и модернизация.[27]

Недостатки данного рода топологии сети.

1. Отказ работы базовой станции приводит к отключению от сети всех рабочих устройств, подключенных к ней.

2. Достаточно большое количество каналов связи, что может усложнить реализации сети передачи данных.[27]

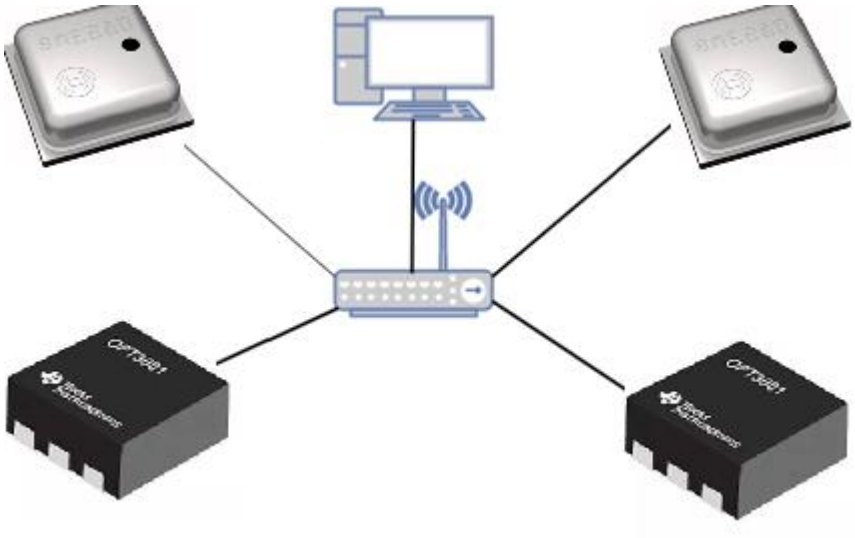


Рисунок 16 – Топология беспроводной сети типа Звезда

LoRa является низшим физическим уровнем (PHY), поэтому может быть использована во всех сетевых топологиях. Ячеистая сеть расширяет диапазон сети, но это происходит за счет ее пониженной пропускной способности, она требует дополнительных ресурсов на синхронизацию, а также влечет снижение срока службы батареи из-за синхронизации и постоянных прыжков по частотам. С наличием увеличенного энергетического баланса линии связи и

более широких возможностей системы LoRa, для расширения диапазона покрытия нет необходимости в использовании сети ячеистой архитектуры (Mesh), поэтому для LoRaWAN была выбрана радиальная архитектура типа звезда. Она позволила оптимизировать пропускную способность сети, увеличить срок службы батареи и упростить ее развертывание.[28]

Надо помнить, что каналы связи между конечными устройствами с базовой станцией и базовой станцией с сервером должны различаться. Это обусловлено пропускной способностью канала между БС и сервером, так как собранная информация со всех датчиков передается на сервер через единый канал связи, а так же следует этому каналу более высокую надежность. Если рассмотреть архитектуру проекта можно заметить, что данная архитектура требует комбинированную топологию передачи данных. Например для описания такой топологии подойдет древовидная структура. Проанализировав две выбранные топологии делаем вывод, что конечном итоге, сеть имеет топологию звезда из звёзд, имеет конечные устройства, которые через базовые станции, общаются с центральным сервером сети. На рисунках 17 и 18 показано сходство архитектуры системы мониторинга и древовидную топологию.

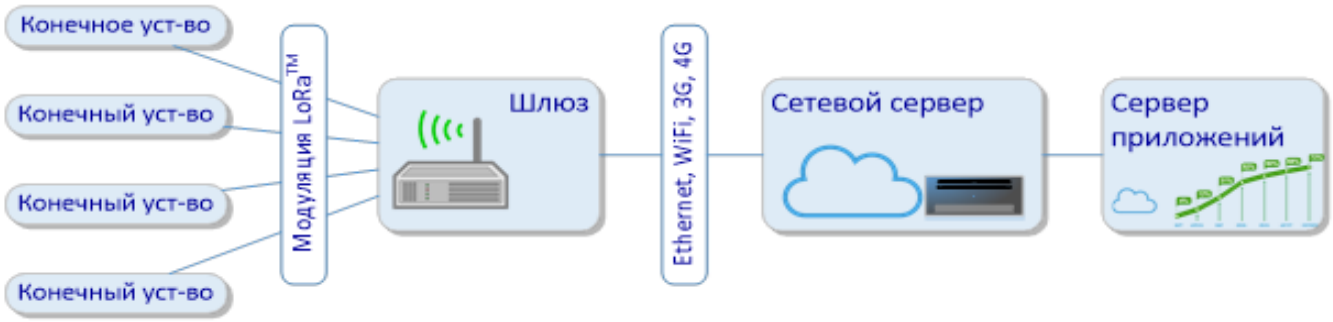


Рисунок 17 – Архитектура системы мониторинга

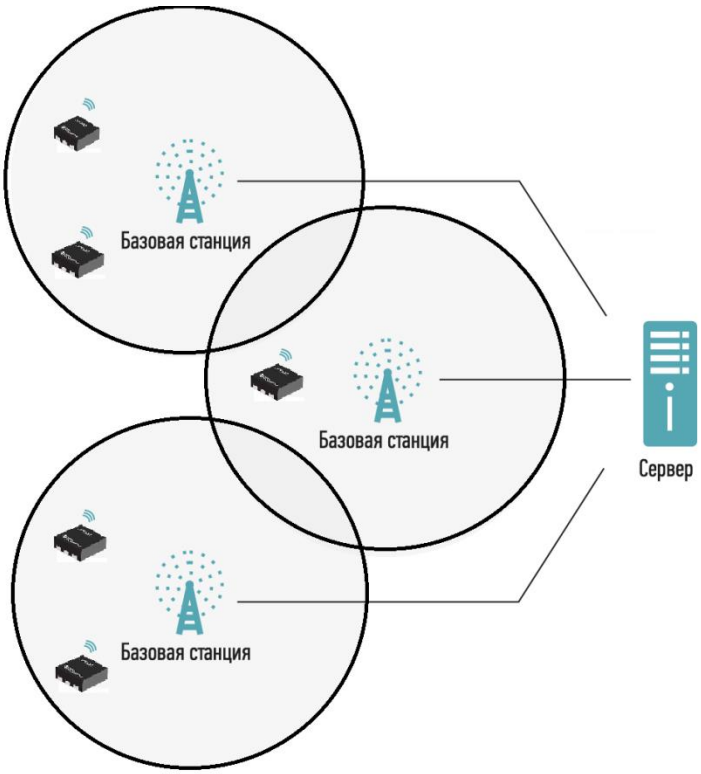


Рисунок 18 – Топология беспроводной передачи данных в системе мониторинга

После анализа топологии системы передачи данных следует разобраться в особенностях технологии LoRaWAN. Данная технология собирает информацию от конечных устройств и передает на сервер, а именно работает как шлюз между двумя элементами. Шлюз – устройство или технология, принимающее данные от конечных устройств с помощью радиоканала и передающее их в транзитную сеть. В качестве транзитной сети могут выступать сеть Ethernet, WiFi или сети подвижной радиотелефонной связи. Обычно данное устройство содержит многоканальные приёмопередатчики для обработки сигналов в нескольких каналах одновременно или даже, нескольких сигналов в одном канале.[6] Соответственно, в проекте несколько таких устройств, как базовая станция технологии LoRaWAN, должно обеспечивать зону радиопокрытия сети и прозрачную двунаправленную передачу данных между конечными устройствами и сервером.

Технология LoRaWAN сетью физического уровня, поэтому стоит описать немного ее модуляцию. Компания Semtech, разработавшая метод модуляции

LoRa создала физический радиоинтерфейс, который основан на использовании широкополосных радиосигналов, который основан на технике расширения спектра. Данный вид радиосигналов имеет две главные особенности.[28]

– ширина спектра радиосигнала *BW* значительно больше скорости передачи данных R_b ($BW >> R_b$);

– корреляционная функция модуляции LoRa является более узкой по сравнению с корреляционной функции узкополосного радиосигнала с базой $B=FT\sim1$, т.е. полоса частот сигналов с базой $B \sim 1$ и полоса частот передаваемого сообщения примерно одинаковые (*F*–частота радиосигнала, *T*–длительность сообщения)[29];

Частотная избыточность широкополосного радиосигнала обуславливает его высокую помехоустойчивость, а узкая корреляционная функция – высокую точность временной синхронизации.

Повышенная помехоустойчивость широкополосных сигналов объясняется следующим образом. Если на вход приёмника поступает сумма широкополосного сигнала и широкополосной помехи в одной и той же полосе частот, то после умножения этой суммы на расширяющую последовательность в приёмнике спектр сигнала становится узкополосным, а ширина спектра помехи либо не меняется, либо ещё более расширяется.

Широкополосный радиосигнал LoRa представляет собой сигнал с линейной частотной модуляцией (ЛЧМ) или CSS (Chirp Spread Spectrum). CSS это расширенный спектр ЛЧМ. Спектр Chirp с расширенным спектром был первоначально разработан, чтобы конкурировать со сверхширокополосной сетью для высокоточного определения диапазона и низкоскоростных беспроводных сетей в диапазоне 2,45 ГГц. [29] Математически ЛЧМ сигнал представляется в виде выражения:

$$x(t) = A_0 \cdot cos\left(\omega_0 \cdot t + \frac{\mu}{2} \cdot t^2\right), \text{где } \frac{T_{sym}}{2} \leq t < \frac{T_{sym}}{2}; \quad (1)$$

где, *BW* – ширина спектра радиосигнала;

SF – коэффициент расширения спектра;

$f_0; \omega_0 (= 2\pi f_0)$ – центральная (несущая) частота радиосигнала;

$f_н (= f_0 - BW/2); \omega_н (= 2\pi f_н)$ – нижняя частота радиосигнала;

$f_в (= f_0 + BW/2); \omega_в (= 2\pi f_в)$ – верхняя частота радиосигнала;

$T_{sym} = 2^{SF}/BW$ – длительность радиосигнала;

$\mu = BW/T_{sym}$ – скорость изменения частоты радиосигнала;

$B = BW \cdot T_{sym} = 2^{SF}$ – база радиосигнала.

Chirp–спектр с расширенным спектром идеален для приложений, требующих низкого энергопотребления и требующих относительно низких скоростей передачи данных (1 Мбит / с или менее). В частности, IEEE 802.15.4a определяет CSS как метод для использования в низкоскоростных беспроводных сетях.[28] Делаем вывод, что данный тип модуляции подходит для осуществления передачи информации для системы мониторинга. На рисунке 19 представлен вид радиосигнала с модуляцией ЛЧМ, а на рисунке 20 показан спектр данного сигнала.

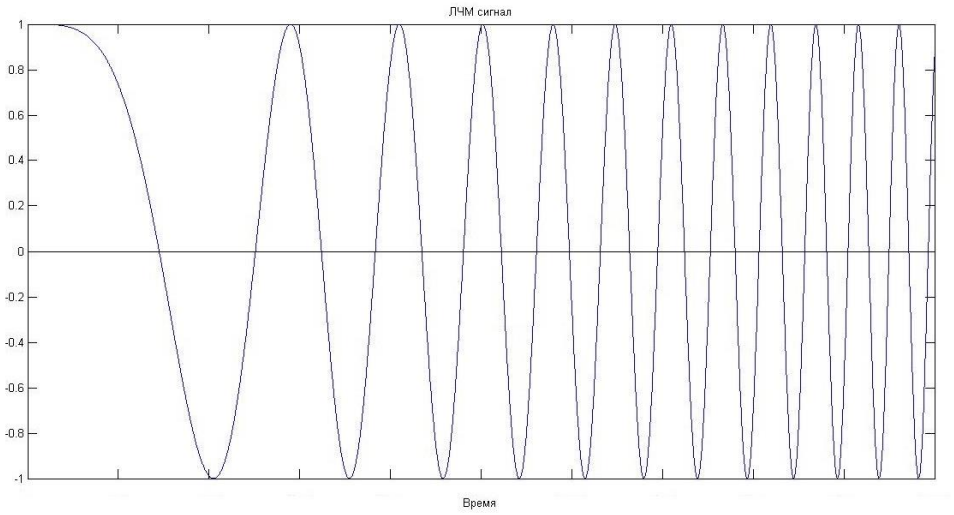


Рисунок 19 – Радиосигнал ЛЧМ

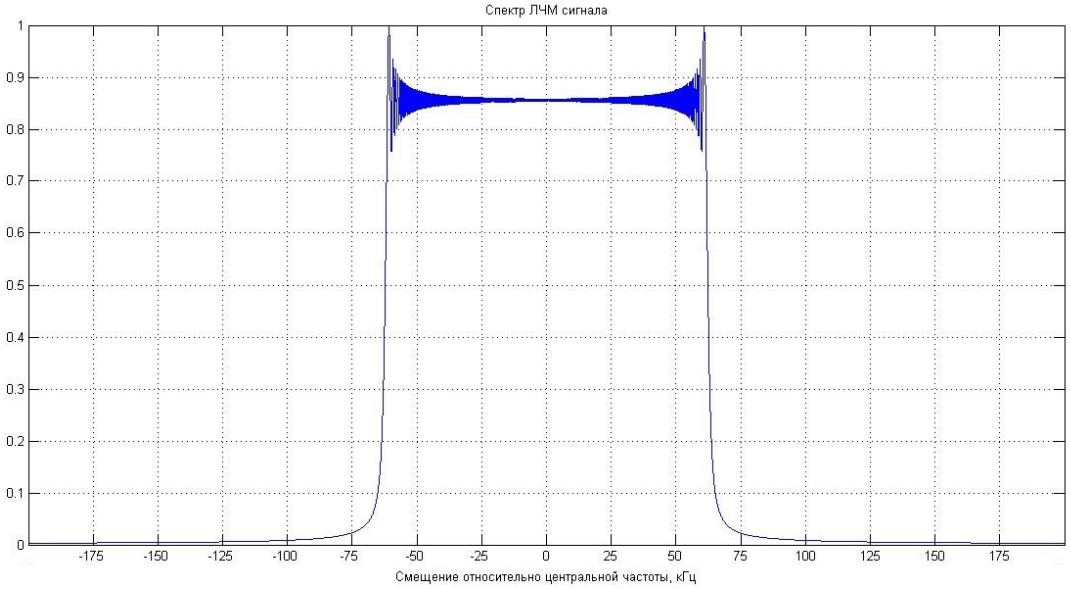


Рисунок 20 – Спектр сигнала ЛЧМ

Данная модуляция помогает бороться с многолучевостью, это острая проблема для коротковолнового диапазона. Решение применять не узкополосные , а сложные широкополосные сигналы с B = FT>>1. Широкополосными (сложными, шумоподобными) сигналами (ШПС) называют такие сигналы, у которых произведения активной ширины спектра F на длительность T много больше единицы. Это произведение называется базой сигнала B, данный параметр упоминался ранее. Повышение базы в ШПС достигается путем дополнительной модуляции (или манипуляции) по частоте или фазе на времени длительности сигнала.[9]

Для успешного функционирования любой системы обмена информацией необходима взаимная синхронизация приемника и передатчика, позволяющая определить временные границы приема-передачи как целого блока данных так и единичных символов. Технология LoRa использует асинхронный режим приема-передачи при котором передатчик может начать генерацию радиосигнала в любой момент времени. В этом случае требуется механизм, обеспечивающий синхронизацию приемника по сигналу от передатчика. В качестве такого механизма используется преамбула, предшествующая каждому сеансу связи. Преамбула включает в себя последовательность символов, позволяющих приемнику обнаружить активность передатчика, определить

используемый передатчиком коэффициент расширения спектра (SF) и выполнить символьную синхронизацию. По завершении преамбулы следует слово синхронизации (Sync Word) и блок данных физического уровня. Длина слова синхронизации настраивается в диапазоне от 1 до 8 байт. На рисунке 21 приведена общая структура кадра, обеспечивающего передачу одного блока данных.[29]

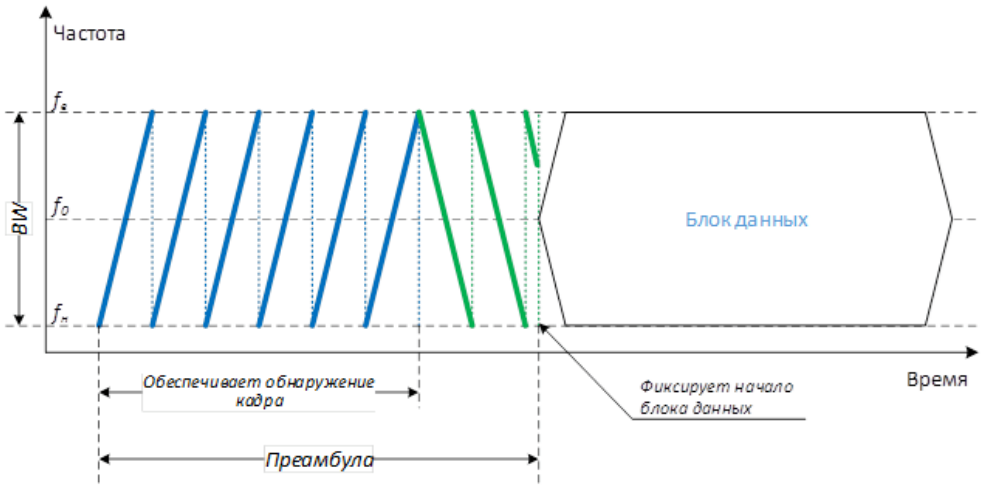


Рисунок 21 – Общая структура кадра

Зная как выглядит физический уровень сигнала, опишем сетевой уровень данной системы. Шлюзы подключаются к сетевому серверу через стандартные IP-соединения, а конечные устройства используют односкачковую беспроводную связь к одному или нескольким шлюзам, модуляция которой была описана ранее. Все конечные точки связи, как правило, являются двунаправленными, но они также поддерживают функционирование в режиме, обеспечивающем возможность осуществления группового обновления программного обеспечения по радиоканалу или передачу иных массовых сообщений, что позволяет сократить активное время на их передачу. [14]

Скорость передачи данных от конечных устройств до базовой станции по протоколу LoRaWAN в системе лежит в диапазоне 0,3–50 кбит/с. Для Европы доступен один GFSK-канал для передачи информации с потоком данных в 50 кбит/с. В Северной Америке из-за ограничений, накладываемых FCC (Федеральная комиссия по электросвязи США), минимальная скорость

передачи данных составляет 0,9 кбит/с. Чтобы продлить срок службы батареи/аккумулятора в конечном устройстве и общую пропускную способность сети, сетевой сервер LoRaWAN управляет скоростью передачи данных и радиочастотным выходом каждого конечного устройства по отдельности. Управление осуществляется с помощью алгоритма адаптивной скорости передачи данных (Adaptive Data Rate, ADR).[29] Это имеет решающее значение для высокой производительности сети и позволяет осуществлять ее необходимую масштабируемость. Если развернуто много шлюзов, то технология ADR будет смещать скорость передачи данных в сторону повышения, что обеспечит масштабирование емкости сети в несколько раз. Таким образом, скорость можно передачи возможно настроить при установки системы.

Одним из главным параметром при установке системы LoRa является мощность сигнала. Для большего покрытия местности с сенсорами требуется и большая мощность. Если выходная мощность непосредственно на выходе датчика равна +20 дБм, а на антенне, после согласования и фильтрации, в результате неизбежных потерь она составляет уже +19 дБм ±0,5 дБ.[28] Различные государства и даже их регионы имеют разные правила для максимально допустимой мощности. Чтобы достичь разрешенного максимума и, соответственно, максимальной зоны покрытия, протокол LoRaWAN позволяет установить различные значения выходной мощности, приемлемые для различных мест использования системы. Если уровень шума будет намного больше мощности передатчика, модем LoRa на имеет возможность подавления помех до 19,5 дБ (за счет Гаусовой фильтрации). Говоря иными словами, он может принимать и демодулировать сигналы на 19,5 дБ ниже уровня помех или шумов, что для правильной демодуляции большинству систем с частотной манипуляцией нужна мощность сигнала как минимум на 8-10 дБ выше уровня шума.[30] Технология LoRa значительно повышает чувствительность приемника и, аналогично другим методам модуляции с расширенным спектром,

использует всю ширину полосы пропускания канала для передачи сигнала, что делает его устойчивым к канальным шумам и нечувствительным к смещениям, вызванным неточностями в настройке частот при использовании недорогих опорных кварцевых резонаторов. Чувствительность составляет примерно до -137 дБм.[29]

Рассмотрим рисунок 22, на котором показана зависимость скорости передачи данных, потребление энергии и радиуса покрытия сети.

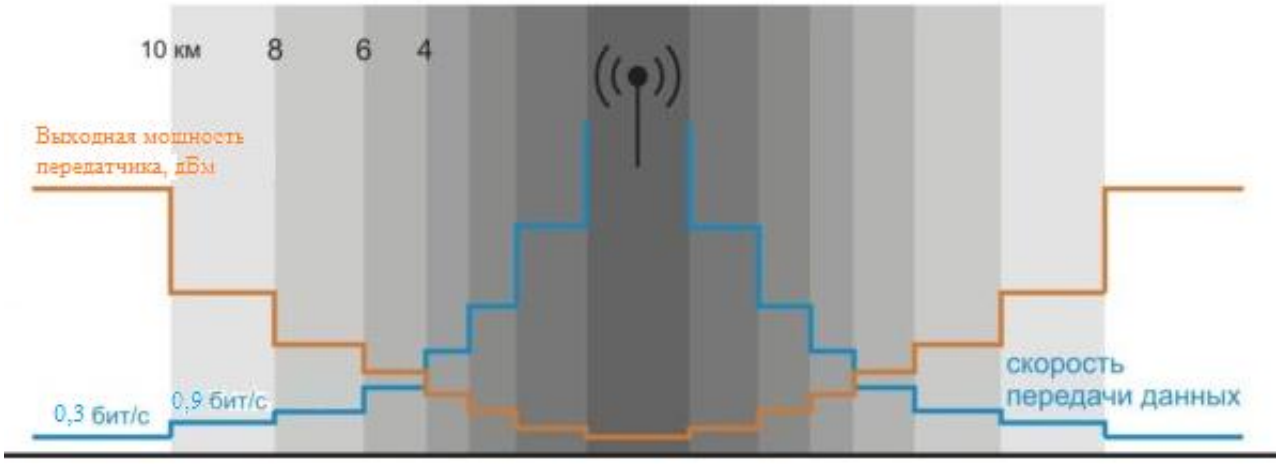


Рисунок 22 – Зависимость скорости передачи данных, потребление энергии и радиуса покрытия сети

На рисунке показано, что более близкие к шлюзу узлы будут использовать и более высокую скорость передачи данных следовательно, более короткое время активной передачи по радиоканалу, и меньшую выходную мощность. Наоборот, самые удаленные устройства будут иметь низкую скорость передачи данных и высокую выходную мощность передатчика. Надо помнить чем больше мощность датчика и дольше время в сети, тем больше потребляется энергии на одни выход в эфир. Но использование данной технологии помогает внести необходимые изменения в сетевую инфраструктуру и, таким образом, компенсировать различные потери на трассе передачи сигнала. Если развернуто много шлюзов, то технология ADR будет смещать скорость передачи данных в сторону повышения, что обеспечит масштабирование емкости сети в пределах от шести до восьми раз.[30]

После того как все данный пришли от разных датчиков, базовая станция должна сформировать общие пакеты данных и отправить на центральный сервер. Снова обратимся к архитекторе системы мониторинга, которая показана на рисунке 23.

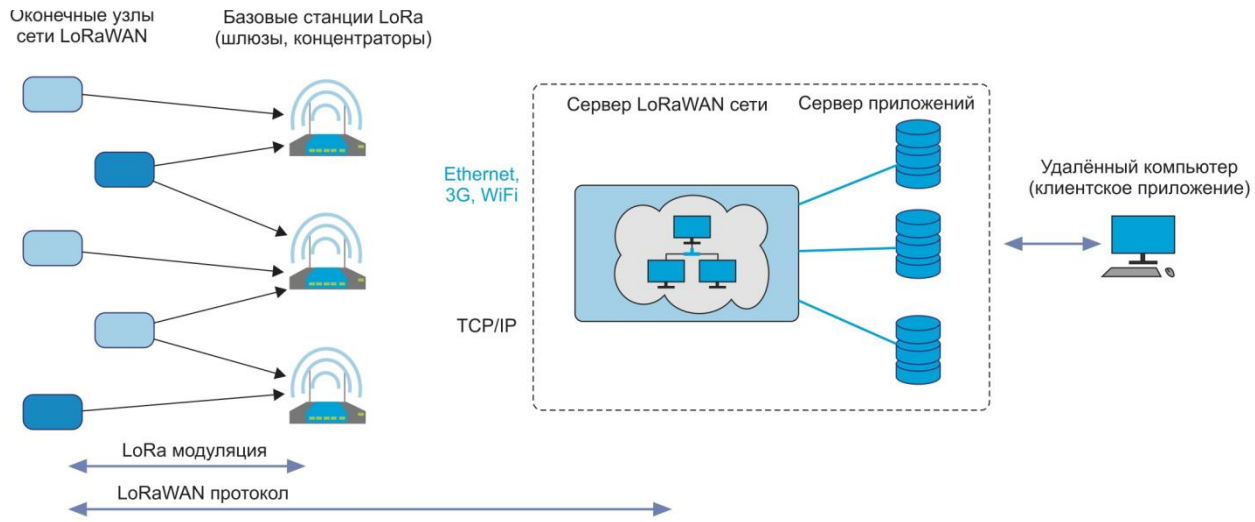


Рисунок 23 – Архитектура системы мониторинга

На рисунке видно, что после базовых станций информацию следует собрать в едином центральном сервере LoRaWAN сети. Центральный LoRaWAN-сервер осуществляет общее управление сетью, в частности принимает решение о необходимости адаптации скорости передачи данных, изменения мощности передатчика, выборе канала передачи, ее начале и продолжительности по времени, измеряет заряд батарей конечных узлов, т. е. полностью контролирует каждое абонентское устройство в отдельности.[6]

Базовые станции (шлюзы, концентраторы) LoRaWAN, предназначенные для использования в сетевой топологии типа звезда большого радиуса действия, формируют прозрачный мост ретрансляции сообщений между конечными устройствами и центральным сервером сети с помощью Ethernet, Wi Fi, GSM или других телекоммуникационных каналов связи путем организации стандартного IP-соединения.[17] Если проводить аналогии, то LoRa-шлюз функционально соответствует базовым станциям сотовых сетей и даже может располагаться вместе с ними. В этом случае LoRaWAN может дополнять сотовую связь и расширять ее возможности. В зависимости от

желаемой канальной емкости и мест установки доступны разные версии шлюзов, они могут монтироваться как внутри помещений, так и на вышках или зданиях.

Итак, сделаем вывод по данной главе. Шлюзы системы LoRaWAN предназначены для использования в радиальных звездообразных сетевых архитектурах большого радиуса действия. Из-за свойств технологии LoRa эти шлюзы могут представлять собой многоканальные мультимодемные передатчики и приемники, которые способны выполнять демодуляцию на нескольких каналах одновременно за счет особенностей модуляции и технологий. Эти шлюзы используют иные радиочастотные компоненты, чем те, которые применяются в конечной точке для обеспечения высокой мощности излучения непосредственно радиосигнала. Шлюзы служат в качестве интерфейса в виде прозрачного моста для передачи сообщений между конечными устройствами и центральным сервером сети.

3.4 Выбор и описание платформы IoT

Платформа IoT - это главный компонент IoT-системы мониторинга. Платформа в более широком смысле - это набор инструментов и сервисов, которые позволяют разработчикам разрабатывать и запускать приложения. Для поставщиков оборудования встроенные платы – это платформа IoT, где вы можете написать свои приложения IoT. Для поставщиков облачных услуг их инфраструктура представляет собой платформу IoT, где разработчик будет использовать данные с датчика и создавать приложения.[7] Для ясности, в проекте называем платформу IoT уровнем промежуточного программного обеспечения, ответственным за передачу данных от датчиков и устройств к серверу и получение значимых результатов и действий. Поэтому IoT платформа – программное обеспечение, предназначенное для подключения интернет вещей (датчиков, контроллеров и других устройств) к облаку и удаленного доступа к ним.

Создать масштабируемую и надежную платформу IoT сложно, поэтому сегодня рынок наводнен сотнями поставщиков IoT PaaS (платформа как услуга). Выбор лучшей платформы IoT для вашего решения сегодня намного сложнее, чем когда-либо прежде. Например Samsung SmartThings, система для умного дома. Данная система позволяет объединить устройства разных брендов в одну экосистему и управлять всеми конечными устройствами со смартфона. Как было сказано ранее подобные системы на рынке тоже называют платформами. SmartThings соединяется с конечными устройствами из комплекса Smart Things и с множеством сторонних устройств партнеров, имеет собственный концентраторы с поддержкой различных протоколов, создает объединение устройств, так называемые комнаты, ситуации(«сон», «отдых», «работа» устройств) и сценариев.[19] Так же свое приложение система имеет на Google Play, App Store и других магазин приложениях. Samsung SmartThings это пример того, как должна работать наша система мониторинга сельского хозяйства. На рисунке 24 показана архитектура и схема работы системы Samsung SmartThings, которая схожа с архитектурой системы мониторинга.

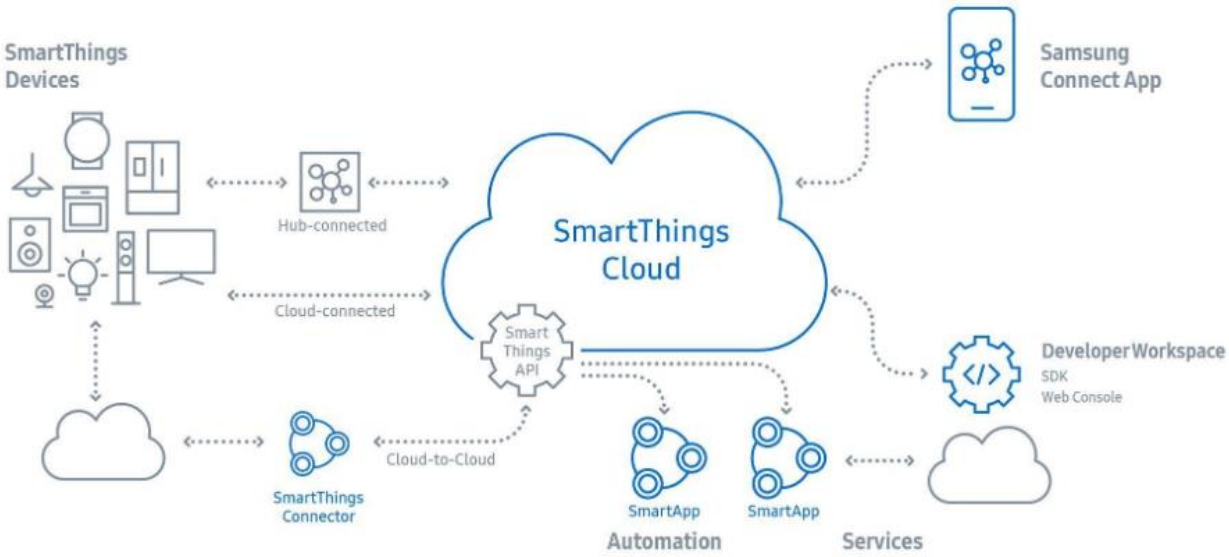


Рисунок 24 –Архитектура Схема работы Samsung SmartThings

В нашем случаи предлагается создать собственную экосистему, которая не будет зависеть от сторонних платформ и выполнять только задачи

мониторинга. Для того чтобы выбрать платформу, есть несколько важных технических критериев платформы, которые помогут при разработке проекта.

Один из главных критерием является масштабируемость. Потому что проектирование системы для обработки 100 устройств совершенно отличается от проектирования для миллиона. По мере роста числа подключенных устройств сложность их обработки увеличивается в геометрической прогрессии. Побочным эффектом этого подхода является увеличенная задержка между узлами связи. [7]

Стоит помнить по мере того, как все больше и больше устройств подключаются к платформе, увеличивается среднее время обработки и анализа каждого события. Простой критерий производительности для устройства IoT состоит в том, чтобы вызвать сигнал тревоги и измерить среднее время, которое требуется платформе для принятия мер в отношении этого события. Связь между облачной платформой и устройствами часто ограничена провайдером или возможностью терминации базовых провайдеров облачной системы. Обычно платформы IoT работают с общедоступными облачными системами, такими как AWS или Google Cloud, поэтому базовая пропускная способность является скорее ограничением инфраструктуры. Пропускная способность и производительность системы в данном проекте ограничена только ресурсами оборудования. [14]

Теперь поговорим о безопасности платформы. Один из примеров, как безопасность Интернета вещей может быть нарушена. Бот-сеть Mirai запустила одну из самых страшных DDoS-атак, похитив устаревшую IP-камеру и другие потребительские устройства в сети общего пользования осенью 2016 года.[16] Крайне важно, чтобы платформа предлагала комплексные инструменты для общей безопасности, то есть методы аутентификации, шифрования и ведение электронной базы. Как минимум, платформа IoT должна предоставить уникальные методы аутентификации на основе идентификаторов для ваших устройств. Регистрация сертификатов для

устройств на местах имеет свои проблемы, поэтому должны быть средства автоматизации для их обработки. Шлюзы платформы должны предлагать шифрование, например SSL или DTLS. [7]

Решение для платформы – это всего лишь промежуточное ПО. В конечном итоге полученные данные должны будут использоваться несколькими приложениями, которые могут быть недоступны на самой платформе. Кроме того, благодаря инновациям в области открытого исходного кода, все больше и больше предприятий сегодня используют открытый исходный код. Поэтому важно, чтобы платформа поддерживала интеграцию экосистемы с открытым исходным кодом, что позволит проекту достигать более высокой производительности и снижения совокупной стоимости владения.

Главный критерий при выборы платформы для системы мониторинга это протоколы, который поддерживаются платформой. Существуют различные подходы к классификации протоколов Интернета вещей. Один из них предполагает объединение протоколов по месту применения в клиент-серверной архитектуре сети:

- D2D (Device to Device) – протоколы взаимодействия конечных устройств между собой (самый распространенный протокол DDS);
- D2S (Device to Server) – для передачи данных, собранных устройствами на серверы для обработки сенсорный узел – сервер (CoAP, MQTT, XMPP, STOMP);
- S2S (Server to Server) – протоколы взаимодействия серверов друг с другом (AMQP, JMS).[6]

В данной главе стоит задача выбрать платформу для передачи данных от датчиков к серверу, поэтому в первых очередь выберем протокол группы D2S. Выбор пал на протокол передачи данный MQTT.

MQTT (Message Queue Telemetry Transport) – лёгкий сетевой протокол работающий поверх TCP/IP. Используется для обмена сообщения между

устройствами по принципу издатель-подписчик (publish–subscribe). В конце девяностых, когда разрабатывался MQTT, каналы связи имели низкую пропускную способность. В результате упор был сделан на минимальную вес сообщения и высокую надежность доставки.[19] Протокол имеет следующие достоинства:

– прост в использования. Протокол представляет собой программный блок без лишней функциональности, который может быть легко встроен в любую сложную систему;

– паттерн проектирования издатель-подписчик удобен для большинства решений с датчиками. Дает возможность устройствам выходить на связь и публиковать сообщения, которые не были бы заранее известны или предопределены;

– легок в администрировании;

– снижена нагрузку на канал связи. Сообщения, насколько это возможно, несут в себе только полезную нагрузку;

– работа в условиях постоянной потери связи или других проблем на линии;

– нет ограничений на формат передаваемого контента.[31]

Итак, рассмотрим как работает данный протокол. На рисунке 25 представлена схема работы протокола.

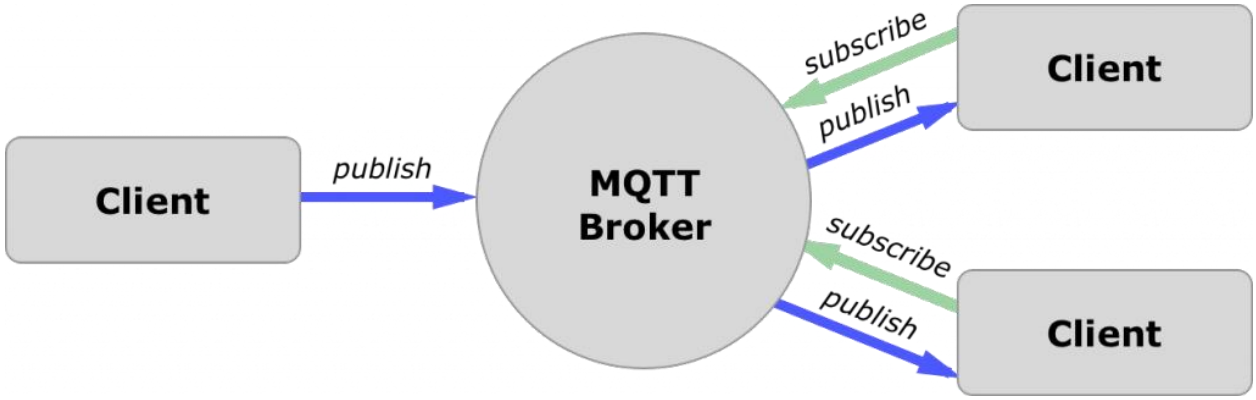


Рисунок 25 – Схема работы протокола MQTT

Рассмотрев рисунок, мы увидим, что протокол реализован следующим образом: устройства, которые хотят передать сообщение (называющиеся издатели в терминах MQTT), отправляют его платформе (брокеру), устройства или серверы которые хотят получить сообщения (подписчики), также подключаются к брокеру и получают сообщения от него. Всю обработку сообщений берёт на себя MQTT-брокер. Данный подход показал себя очень хорошо и эффективно в таких задачах, как, например, сбор данных, телеметрия и управление простыми устройствами. [31]

MQTT предоставляет способ создания иерархии каналов связи — своего рода ветвь с листьями. Всякий раз, когда у издателя есть новые данные для распространения среди клиентов, сообщение сопровождается примечанием контроля доставки. Клиенты более высокого уровня могут получать каждое сообщение, в то время как клиенты более низкого уровня могут получать сообщения, относящиеся только к одному или двум базовым каналам, «ответвляющимся» в нижней части иерархии. Это облегчает обмен информацией размером от двух байт до 256 мегабайт. [31]

В данной системе роль издателя принимают конечные устройства, а подписчиками являются серверы. Теперь остается выбрать брокера. Когда речь идет о протоколе MQTT, практически всегда на практике применяют в роли брокера Eclipse Mosquitto.

Mosquitto – платформа, реализующая возможность обмена сообщениями между компонентами системы на базе стандартов протокола MQTT. это брокер сообщений с открытым исходным кодом, который легок и подходит для использования на всех устройствах от одноплатных компьютеров с низким энергопотреблением до полных серверов. [19]

Данная платформа известна из курсов IoT Академия Samsung, на которых проектирование подобных систем мониторинга проходела на основе протокола MQTT и платформы Mosquitto. Поэтому, предлагается остановить свой выбор

на данной платформе, а в качестве главного протокола передачи данных взять MQTT.

3.5 Разработка программного обеспечения

Разработка программного обеспечения– это деятельность направленная на создание программ, использованием для управления системой мониторинга. Стоит определить, какие функции должно выполнять ПО.[18] В данном проекте требуется универсальное диспетчерское программное обеспечение для автоматизации процесса мониторинга и управления, агрегации и хранения данных о показаниях температуры, влажности и других показаниях. Следственно, первым шагом стоит разработать базу данных.

База данных –информационная модель, позволяющая упорядоченно хранить данные об объекте или группе объектов, обладающих набором свойств, которые можно категоризировать. При разработке базы данных обязательно учитывается возможность ее вертикального и горизонтального масштабирования. При не грамотном проектировании БД, вносить изменения в уже существующий программный продукт достаточно трудно. Одно из главных правил это контроль объема обрабатываемых данных. При больших объемах данных и сложных логических операциях, выполнение транзакций (группа последовательных операций с базой данных) может занимать неприемлемо большое время, например, при обработке входных данных полученных с использованием автоматизированных устройств таких как, сканер штрих кодов, электронная карта. В зависимости от качества спроектированной базы данных (таблицы, ключи, уникальные поля, связи, хранимые процедуры, функции и т.д.) напрямую зависит будущее программного приложения. [7]

Базы данных функционируют под управлением так называемых систем управления базами данных. Система управления базами данных (СУБД) — это комплекс программных средств, необходимых для создания структуры новой базы, ее наполнения, редактирования содержимого и отображения информации. В свою очередь, для удобства работы с СУБД используются специальные веб-

приложения, которые позволяют посредством графического интерфейса выполнять администрирование сервера баз данных и другие функции. Самыми популярными СУБД являются MySQL, MS SQL Server, PostgreSQL, Oracle.[32]

Современные базы данные (MS SQL, ORACLE, DB2) предназначены не только для хранения информации и выполнения примитивных операций, но и способны задавать логику обработки информации. В результате, программное обеспечения клиента (АРМ – автоматизированное рабочее место) несет только функциональную нагрузку по вводу значений (параметров) и отображению данных. Этот подход снижает требования к аппаратной части клиента, а все «тяжелые» операции выполняются на сервере баз данных. Поэтому целесообразна разработка сервера приложений. Сервер приложений является промежуточным звеном между конечным клиентом и базой данных (БД). На нём сосредоточена большая часть логики. Вне его остаются только фрагменты, экспортируемые на клиента (терминалы), а также элементы логики, погруженные в базу данных. Реализация данного компонента обеспечивается связующим программным обеспечением, которое обеспечивает взаимодействия между различными приложениями, системами, компонентами клиента и сервера.[32]

Перенос этой функциональности со стороны клиента на сервер приложений дает ряд преимуществ:

– внутри программная логика работы заложена централизовано, а значит, при изменении логики не требуется обновления клиентских приложений;

– клиент только вводит параметры и получает отображения данных.

– повышается безопасность использования автоматизированного рабочего места (АРМ) использующего приложение (систему, комплекс задач), за счет осуществления работы с назначением определенного порта обмена данными.

– повышается безопасность хранения данных на сервере БД, за счет возможности размещения сервера приложений в демилитаризованной зоне информационной инфраструктуры предприятия.

Теперь опишем как работает вычислительная архитектура и уточним некоторый термины. В данном случае применяется не просто архитектура клиент сервер, а трехуровневая схема клиент, сервер приложений и сервер. Данная архитектура показана на рисунке 26.

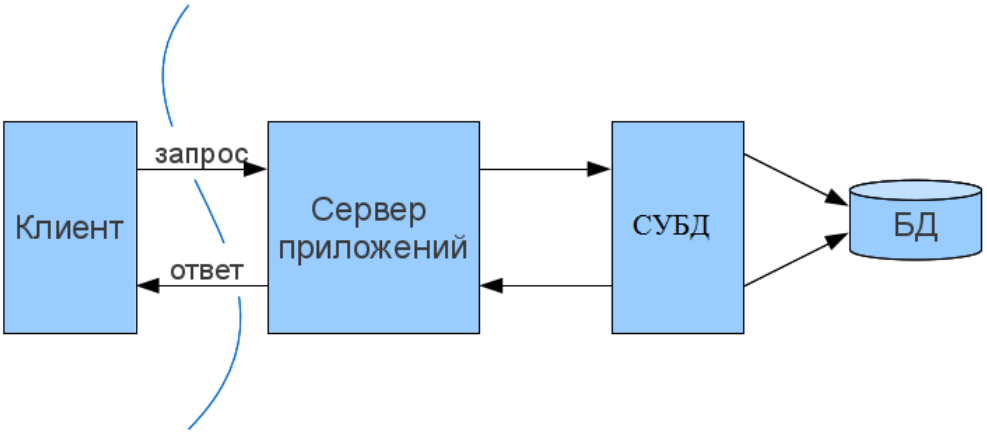


Рисунок 26 – Трехуровневая архитектура

При исследовании вопроса связи сервера приложений и клиента, появляется некоторые термины, которые могу сбить с толка. Например веб–сервер и веб–приложение.

Веб-сервер — сервер или специальная программа, принимающий HTTP-запросы от клиентов и выдающий им HTTP-ответы, как правило, вместе с HTML-страницей, изображением, файлом, медиа-потоком или другими данными. HTML– протокол прикладного уровня передачи данных, о нем будет рассказано в следующих разделах. Простыми словами разницу веб–сервера и сервера приложений можно объяснить так. Веб-сервер довольно узкое понятие, обозначающее программу, отдающую данные клиентам по протоколу HTTP.[11] Сервер приложений - это, программа, позволяющая централизованно на стороне сервера хранить и выполнять приложения, управляя при этом распределением нагрузки, обеспечивая работу с БД и т.д. Если сервер приложений работает с клиентами по протоколу HTTP, то он одновременно

является и веб–сервером. Рассмотрим рисунок 27, на котором показана схема передачи данных в данной архитектуре.

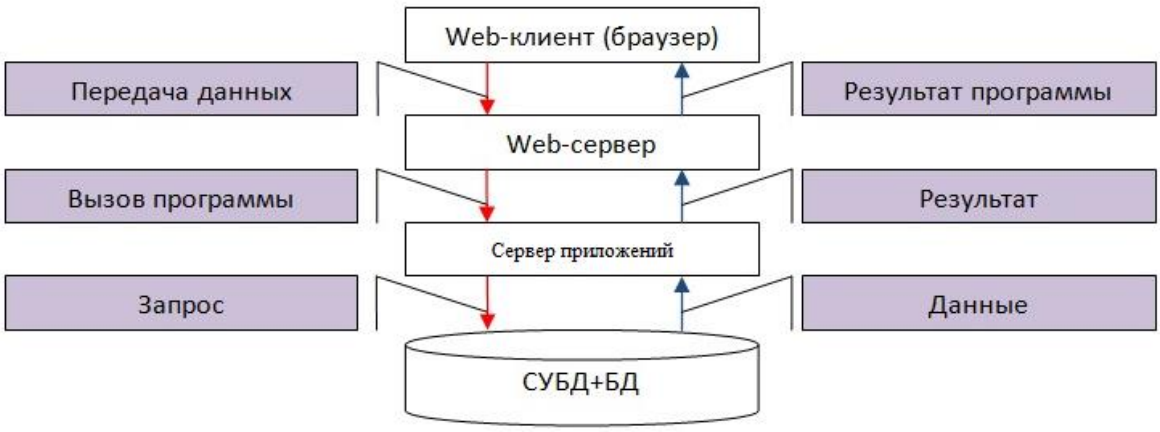


Рисунок 27 – Схема передачи данных от сервера до клиента

Веб–приложение — это решение, в основе которого лежит взаимодействие браузера и веб–сервера. Такие приложения являются кроссплатформенными сервисами, доступными с любого современного устройства, и не привязаны к архитектуре сети: вы можете получить доступ к ним с локального компьютера или со смартфона на другом конце света по удобному для вас протоколу, например, наиболее быстрому или зашифрованному.[11] Описание Веб–приложения будет в следующем параграфе.

Для того чтобы разрабатывать ПО сервера требуется определить, где аппаратно будет находится сервер. Обычно аппаратная часть сервера специализированный компьютер для выполнения серверных задач.

Серверы размещаются в специально оборудованных помещениях, называемых дата-центром, а небольшие модели серверов могут размещаться в обычных офисных помещениях, и от простых десктопных компьютеров их зачастую отличает лишь автономная работа и подключение к блоку бесперебойного питания повышенной ёмкости. Делаем вывод, что серверу нужно какое то место, а желательно отдельное помещение. [18]

Чтобы не задействовать дополнительное оборудование при создании системы мониторинга предлагается воспользоваться облачными технологиями.

Облачный сервер — услуга на базе облачных технологий, предоставляющая доступ к серверным ресурсам компании-провайдера. По своим возможностям облачный сервер идентичен его физическому аналогу, но у пользователей облачных серверов есть возможность корректировать их мощность по своему усмотрению. Все данные, профили и отдельные файлы хранятся на сервере стороннего провайдера.[10] Для работы с программными решениями и информацией на облачном сервере необходимо только наличие интернет-подключения. Сферы применения облачных технологий: от создания рабочих мест пользователей до организации персонального хранилища данных. Ключевые преимущества облачного сервера – быстрый доступ к информации, защищённость данных, высокая степень надёжности сервера.

При создании подобных систем в интернет–индустрии участвуют множество разработчиков. Все специалисты используют различных языков программирования, чтобы писать, обновлять, исправлять и совершенствовать алгоритмы для всевозможных десктопных и мобильных приложений, веб-сайтов. Обычно разработчики делятся на два направления.

Front-end developers или разработчик внешнего интерфейса. Такой сотрудник концентрируется на визуальной части проекта, включающей в себя внешний вид ресурса, его интерфейс, всевозможные приложения. Он делает максимально комфортным пользование каждой страницей сайта, то есть ориентируется на клиентскую сторону ресурса. Его основные инструменты – CSS, HTML, JavaScript. Подробнее про задачи и создание интерфейса рассказано в следующем параграфе.[11]

Back-end developers или разработчик программно–аппаратной части сервера. Этот специалист занимается серверными технологиями. Он получает пользовательский запрос от фронт-энда, обрабатывает его и передает обратно в доступной для клиента форме. То, что происходит на стороне сервера, пользователю недоступно, он видит лишь конечный результат и не может вмешаться извне в работу приложения. Разработчик по бек-энду использует

следующий инструментарий: разные языки программирования (PHP, Perl, Java, Python, Ruby), фреймворки (Kohana, Codeigniter, Yii), а также MySQL для сбережения данных и администрированием сервера . Именно про эту часть работы было описано в данном параграфе.[32]

Для оптимальной работы системы мониторинга требуется создание базы данных с помощью СУБД, сервер приложений для создания логики обработки данных и связующее ПО между сервером и клиентом. Стоит отметить, что разработка сервера в целом является сложной работой, которая может занять много времени и требует сторонней помощи. Из за данных трудностей проект не ставит задачи по созданию полноценной системы мониторинга, а только тестовый запуск отдельных подсистем. Поэтому предлагается при разработке системы мониторинга упростить создание базы данных.

3.6 Разработка приложения и интерфейса пользователя

При разработке клиентского приложения предшествует проектирование визуального интерфейса приложения. Так же необходима разработка серверного приложений, при необходимости должен быть определен протокол обмена данными между клиентским приложением и сервером приложений. На основании технического задания для какого либо проекта закладывается логика работы клиентской части, интерфейс и дизайн приложения. В основу проектирования визуальной части специалисты обычно закладывают два принципа:

- визуальный интерфейс должен быть понятен пользователю;
- должен быть осуществлен принцип преемственности.[11]

Итак, под интерфейсом понимается любой экранный информационный или интерактивный интерфейс. Таковыми являются:

- сайты;
- мобильные приложения;
- приложения для стационарных компьютеров;

– презентационные панели (включая touch);

– информационные стационарные экраны.

Полный цикл разработки интерфейса показан на рисунке 28.

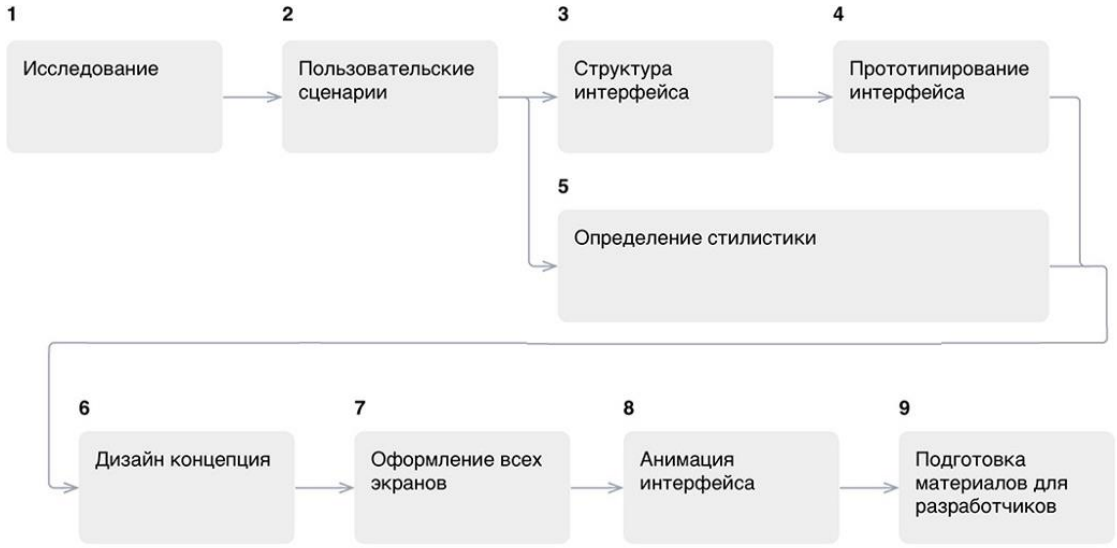


Рисунок 28 – Цикл разработки интерфейса

На этапе исследования проводится анализ о конечном продукте, потребителе продукта, конкурентах системы или близких аналогах. Этот этап помогает понять для кого разрабатывается интерфейс, с какими ограничениями следует его делать (размеры экранов, интерактивность), как не стоит делать (например, быть непохожими на конкурентов).[33]

На основе предоставленного описания работы интерфейса создается список задач. Другими словами начинается этап пользовательского сценария, которые может выполнять пользователь в рамках интерфейса. Все задачи расписываются по шагам, которые необходимо предпринять для решения задачи. [33] Например для системы мониторинга:

– зайти на сайт;

– авторизоваться;

– перейти в профиль;

– зайти в раздел мониторинга;

– выбрать интересующий параметр;

– получить данные от конечных устройств.

Составленные списки шагов для каждой задачи помогают понять где путь для решения слишком долог относительно остальных задач. Этап пользовательских сценариев больше всего подходит для сокращения пути решения задач пользователей в рамках интерфейса.[33]

Полученный список шагов на предыдущем этапе, ложится в основу структуры интерфейса. Становится известно количество экранов, их краткое содержание и положение в общей структуре.

Следующий этап включает в себя разработку нескольких вариантов интерфейса. Черновые прототипы помогают более наглядно понять на сколько объемным будет сайт, как много информации будет на каждом экране, как много нужно кликать, чтобы добраться до нужной страницы. В итоге выбирается финальный прототип в котором схемы страниц все еще остаются связанными между друг другом, но на страницах уже видны все кнопки, тексты , формы и прочие элементы.[33]

После этапа исследования и параллельно с этапами проектирования идет определение будущей стилистики интерфейса. Для выбора стилистики готовятся несколько наборов изображений. Эти наборы представлены страничками сайтов, иллюстрациями, кнопками, шрифтовыми композициями, связанными между собой стилистически.[33]

Дизайн концепция — это предположение как может выглядеть весь интерфейс. Дизайн концепция призвана показать оформление сайта и дать понять будущий вид всего сайта. Если предыдущий этап определения стилистики только дал направление, то дизайн концепция призвана скрестить выбранное направление с имеющимся содержанием интерфейса.

После утверждения дизайн концепции настает время оформления всех остальных экранов интерфейса. Когда же очередь доходит до оформления всех экранов, тогда и происходит финализация внешнего вида: становится ясно правильно ли подобран кегль или интерлиньяж, хорошо ли сочетается толщина

линий иконок с текстом, не конфликтует ли оформление форм (кнопок, полей ввода) с другими элементами экрана и многие другие случаи.[33]

Анимация интерфейса показывает как будет работать данный сайт и что будет видеть пользователь. Часто этот этап начинается еще с момента дизайн концепции и продолжается на протяжении всего этапа оформления всех экранов.

Последний этап предназначен, чтобы помочь разработчикам в реализации интерфейса и веб–приложения, готовятся все необходимые для этого материалы.

После создания сайта требуется связать интерфейс и показания датчиков, которые хранятся на сервере. Для это и существует веб–сервер, который может находится или клиентской части или на серверной части. Основная задача веб-сервера - принимать HTTP-запросы, обрабатывать их и выдавать HTTP-ответы. Таким образом, пользователь для запроса сайта вводит адрес сайта в строку браузера уже выполняться запрос к веб-серверу. Ответ, который отправляет веб-сервер, он содержит всю необходимую информацию, для того, чтобы браузер смог отобразить веб-страницу. В том виде, в котором это задумывал разработчик веб-сайта, в нашем случае график мониторинга температуры, влажности и других показателей. Таким образом, веб-сервер является связующим звеном между серверным языком программирования. Например, веб-сервер получил запрос, передал его какому-то серверному языку программирования и получил сразу какой-то определенный ответ, который обрабатывался на сервере. [34]

К данной технологии относится и использование WEB доступа. Неоспоримым плюсом использования WEB-доступа является возможность подключения с любого устройства имеющего выход в общедоступную сеть (Internet) и Web-браузер. Однако, эта возможность является и минусом: достаточно знать имя пользователя и пароль что бы воспользоваться доступом к данным. Для исключения этой ситуации необходимо применение

дополнительных средств защиты от несанкционированного доступа (таких как использование электронных ключей, внедрение центра сертификации и др.).[34]

Веб-сервер это программа, поэтому эту программу могут выпускать различные разработчики. Т.е. разные компании могут по своим алгоритмам разрабатывать веб-сервера, которые тем или иным образом работают. Самыми популярными веб-серверами, которые имеются на текущий день являются такие веб-сервера как: Apache, IIS, Nginx. Как говорилось ранее, сервер приложений может выполнять функции веб–сервера, это значит, что веб–сервер находится в серверной части.

Теперь когда связь между сервером и клиентской частью установлена, надо понять какие запросы будет отправлять пользователь и как взаимодействовать со веб–страницей. Если пользователю требуется тесное взаимодействие с сервером, например авторизация, запросов по изменению контента, совершение покупок, тогда требуется разработка веб–приложения.

Веб-приложение — это любой сайт с элементами интерактива. Это значит, что посетитель может взаимодействовать с материалом и функциями сервера. Таким образом, веб– приложение работает как простое приложение на компьютере и имеет динамическое взаимодействие с сервером. Простыми словами разница веб–приложения и веб–сайта в взаимодействии с сервером. Если сайт отправляет запрос серверу только при переходе на другую страницу, то веб–приложение работает с Back-end частью в интерактивном режиме. Приложения могут быть встроены в web-страницы, либо сами web-страницы могут быть приложениями. Например, Facebook, Gmail, YouTube, Ebay, Twitter, Amazon.[11]

Веб-приложение является более ресурсоемким, так как может взаимодействовать с пользователем и выполнять различные действия. По этой причине разработка web-сайта является более легким, недорогим проектом. Для

разработки web-приложения требуются большая вычислительная мощность, а также различные платформы и инструменты.

Таким образом, возникаем проблема в сложности разработки и клиентской части. Система должна производить непрерывный мониторинг показателей, поэтому требуется динамическое взаимодействие с сервером. Следственно, предлагается для облегчения задачи предлагается разработать приложение только на одной веб–странице, которая будет работать интерактивном режиме.

3.7 Выводы по главе

В данной главе были описаны конкретные компоненты и способы реализации проекта. Были подобраны конечные устройства, проанализировали технологию передачи данных LoRaWAN, выбрали основной сетевой протокол передачи данных, а также платформу IoT. Описали архитектуру серверной и клиентской части и определили какие языки программирования понадобится при разработки. При написании главы воспользовался наработками курсах IoT Академия Samsung.

Для реализации системы мониторинга на основе технологии интернета вещей следует придерживаться следующих выводов.

В данной проекте будут применяться два типа датчиков bme280 и opt3001, которые будут передавать показания температуры, влажности, давления и освещенности. Эти данные позволят фермерам делать выводы в каком состоянии находится почва для растений и какая окружающая среда в хранилище семян. Другие компоненты предлагается взять из курсов IoT Академии Samsung.

Шлюзы системы LoRaWAN предназначены для использования в радиальных звездообразных сетевых архитектурах большого радиуса действия. Из-за свойств технологии LoRa эти шлюзы могут представлять собой многоканальные мультимодемные передатчики и приемники, которые

способны выполнять демодуляцию на нескольких каналах одновременно за счет особенностей модуляции и технологий. Эти шлюзы используют из за высокой мощности излучения непосредственно радиосигнала. Шлюзы служат в качестве интерфейса в виде прозрачного моста для передачи сообщений между конечными устройствами и центральным сервером сети.

Платформа Mosquitto и протокол MQTT легки в применении и известны из курсов IoT Академия Samsung, поэтому предлагается разматывать систему мониторинга на данной платформе, а в качестве главного протокола передачи данных взять MQTT.

Для оптимальной работы системы мониторинга требуется создание базы данных с помощью СУБД, сервер приложений для создания логики обработки данных и связующее ПО между сервером и клиентом. Стоит отметить, что разработка сервера в целом является сложной работой, которая может занять много времени и требует сторонней помощи. Поэтому предлагается при разработке системы мониторинга упростить создание базы данных.

Система должна производить непрерывный мониторинг показателей, поэтому требуется динамическое взаимодействие с сервером. Следственно, предлагается для облегчения задачи предлагается разработать приложение только на одной веб–странице, которая будет работать интерактивном режиме.

4 Реализация проекта мониторинга SMART-АГРО

4.1 Обзор существующий систем мониторинга сельского хозяйства

Прежде чем разрабатывать систему мониторинга следует провести обзор подобных систем и проанализировать их достоинства и недостатки. На мировом рынке под понятием IoT сельского хозяйства подразумевается включение передовых технологий в существующие методы ведения сельского хозяйства для повышения эффективности производства и качества сельскохозпродукции. Уже внедряемые и грядущие сельскохозяйственные технологии можно разделить на три основные категории, которые и станут основой "умного" фермерства - это автономные роботы, беспилотные летательные аппараты (БПЛА) или дроны, и различные IoT-датчики.[10] Данный обзор сконцентрирован на третьей категории. Стоит отметить, что в основном системы на основе интернета вещей имеют классические технологии передачи данных, например Wi–Fi,LTE,GPS и другие.

Одна из российских фирм, которая работает в данной сфере, это тверская компания Tibbo Systems. Фирма работает в области аппаратных и программных решений для Интернета вещей, управления ИТ-инфраструктурами, автоматизации производственных и технологических процессов, удалённого мониторинга и обслуживания, контроля физического доступа и комплексной автоматизации центров обработки данных. Tibbo Systems входит в холдинг компаний, где вторая основная компания Tibbo Technology (Тайвань). Фирма стоит свои системы на основе интеграционная платформа интернета вещей AggreGate. Продукты на основе единой IoT платформы AggreGate позволяют автоматизировать множество аспектов деятельности сельхозпредприятий, повышая эффективность и приводя, в конечном итоге, к улучшению финансовых показателей деятельности. Системы выполняют многие функции, например мониторинг условий хранения, управление сортировкой, хранением и переработкой сырья,

управление активной вентиляцией и умные теплицы. Данной платформой пользуются многие сельскохозяйственная и продовольственная компания, такие как Русагро ,тепличный комбинат «Майский» и другие.[35]

Организации в различных отраслях ежедневно пользуются продуктами на основе AggreGate, чтобы упростить ИТ-задачи, сократить расходы и автоматизировать бизнес. Платформа AggreGate и другие продукты на её основе предлагают комплексные решения для различных сельское хозяйство, медицина, энергетика, нефтегазовая отрасль, производство, транспорт и многие другие. Универсальность платформы AggreGate обычно является преимуществом, часто необходимо использовать готовое решение, созданное для конкретной области применения.[35] Для таких случаев предлагается различные продукты, построенные на IoT-платформе AggreGate. Одновременно это является минусом, так как платформа это не отдельная система с решенными проблемами сельского хозяйства, а универсальное решение для производства. Получается системы мониторинга имеют разные технологии передачи данных, сервер приложений и интерфейс пользователя. Так же данная платформа предлагается для крупных фирм, где обычно существуют IT–отделы, которые способны внедрить платформу AggreGate в производство. Для частного фермера сложно будет разобраться в подключении данной платформы.

Теперь рассмотрим фирму Growlink, которая работает в соединенных штатах. Компания специализируется на получении данных в теплицах или хранилище продуктов.[36] Еще Growlink устанавливает системы дозированного полива, одна из таких систем показа на рисунке 29.

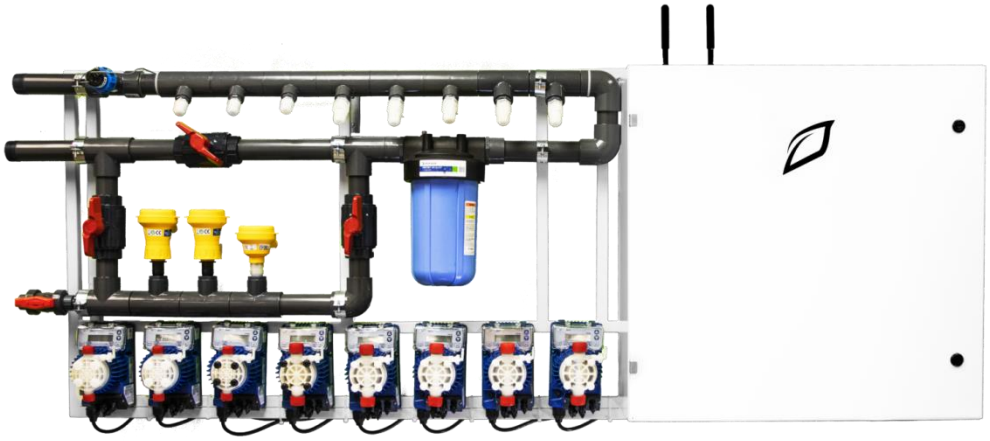


Рисунок 29 – Цикл разработки интерфейса

Данная фирма создает системы для конкретных клиентов. Это позволяет фермерам только заплатить за систему и не разбираться в ее архитекторе. Но как можно понять из рисунка, данные системы являются громоздкими и могут не подойти фермерам, которые работают на полях.[36] После изучения данной фирмы выяснилось, что специалисты устанавливают сервера в помещениях, где и организован мониторинг. Этот недостаток заставляет обслуживать большое количество оборудования. Могу предположить, что фирма использует технологию для передачи данных не LoRaWAN. Данный вывод сделан из за размеров теплиц и хранилищ.

Рассмотрим еще одну фирму Arable , которая расположилась в Сан–Франциско. Фирма специализируется на устройствах для точного земледелием и управлением растениеводством. Конечные устройства размещаются на местах для сбора данных, относящихся к земледелию; от температуры и осадков до потенциала листовой воды и общего состояния здоровья сельскохозяйственных культур. Таким образом, вы можете отслеживать рост вашего урожая и любые аномалии, чтобы эффективно предотвращать любые заболевания или заражения, которые могут повредить вашему урожаю. Сбор и обработка информации осуществляется через собственную платформу Arable, а датчики собраны в единую мини метеостанцию, которая показа на рисунке 30.[37]

Рисунок 30 – Датчики от фирмы Arable

Данная концепция походит на предложенную в данного проекте системы мониторинга сельского хозяйства. Единственный недостаток данного подхода, это мини метеостанции большого размера. В некоторый случаях требуется компактное конечное устройство.

Делая вывод по обзору существующий систем мониторинга можно сказать, что данные системы в России не распространены, а заграничные аналоги имеют некоторые недостатки по сравнению с нашей системой, такие как большое количество оборудования или неподходящие датчики для сбора информации.

4.2 Настройка конечных устройство

Конечные устройства будут устанавливаться в корпус, который будет находится на грядках или в земле на уровне нескольких сантиметров. Строение модулей позволит расположить в корпусах для крепления на DIN-рейку. Это позволит разместить устройства в хранилище семян.

Устройство будет иметь два чипсета(микросхема), которые имеют микроконтроллер STM32L151CC, приёмопередатчик SX1276 и место для крепления датчиков. Датчики или микропроцессор соединяются между собой шинами связи.[19] Датчики и микроконтроллер имеют одинаковую интерфейсную шину I^2C. Данные передаются по двум проводам — провод данных и провод тактов. Есть ведущий(master) и ведомый (slave), такты

генерирует master, ведомый лишь помогает при приеме байта. Всего на одной двупроводной шине может быть до 127 устройств.[13]

Любая микросхема имеет входы/выходы, которые имеют свои значения. Отметим, что микросхема имеет 2 шины I^2C и находятся на общих входах/выходах PB8 и PB9.[19]

Для начала предлагается рассмотреть характеристики модулей Unwired Range, которые показаны в таблице 4.

Таблица 4 –Параметры модулей Unwired Range

Параметр	Значение
Процессорное ядро	32-битное ARM Cortex-M3, 32 МГц
Разъёмы	PLLD-1,27-24 (все доступные интерфейсы и питание)
Антенна	Разъём SMA для внешней антенны; Несъёмная штыревая антенна
Радио	863-869 МГц, мощность до +14 дБм
Дальность связи	До 30 км в прямой видимости и 1…5 км в городской застройке
Скорость передачи данных	Минимальная: 50 бит/с Максимальная: 37,5 кбит/с
Операционная система	ОС реального времени RIOT OS
Размеры	25×36×2,0 мм (без учёта разъёмов и антенны)
Напряжение питания	Номинальное: 3,3 В; Допустимое: 2,0…3,6 В

Важно установить систему Ubuntu Linux, потому что программировать мы будем просто в текстовом редакторе, и будем активно пользоваться командной строкой. ля сборки программ нам понадобится набор инструментов — компилятор GCC для ARM и небольшая операционная система RIOT, ориентированная на беспроводные устройства Интернета вещей с низким энергопотреблением.[19]

Установим все нужное ПО и подключим устройство к компьютеру, используя MicroUSB-кабель. Теперь установим параметры соединения: COM-порт /dev/ttyACM0 и Скорость 115200. Данное действие изображено на

рисунке 31. После этого микроконтроллер напишет версию прошивки, платы и микроконтроллера.

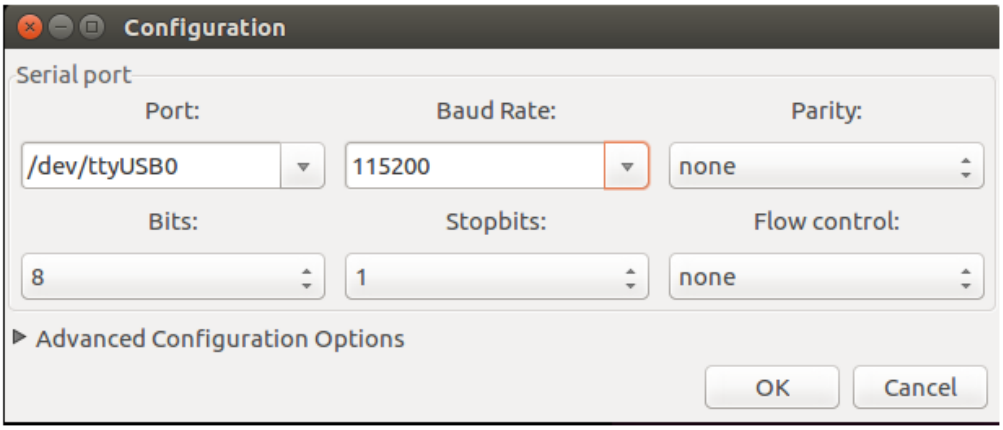


Рисунок 31 – Установка параметров

Может оказаться что данная прошивка устарела и не подходит для микроконтроллера. Для этого обновим программное обеспечение микроконтроллера. Для этого можно зайти на сайт Github, и там же узнать о наличии новой версии. Для конечного устройства нам понадобиться прошивка loralan-public.hex.[19]

В пределах одной сети каждый радиомодуль должен иметь уникальный адрес; использование EUI64 из адресного пространства, официально выделенного организацией IEEE, гарантирует это, так как исключает выпуск двух устройств с одинаковыми адресами EUI64, независимо от их модели и производителя. APPID64 — уникальный идентификатор приложения, работающего на радиомодуле. APPID64 может использоваться на шлюзе или сервере сети LoRaWAN для определения, что именно за данные поступают с этого радиомодуля; при этом у множества радиомодулей, выполняющих одну и ту же задачу, может быть один и тот же APPID64. [19]

Запустим функцию set joinkey. JOINKEY — ключ шифрования, используемый в данной сети. JOINKEY должен быть абсолютно одинаковым для всех устройств сети, а также для базовой станции. Ключ имеет разрядность 128 бит и задаётся в виде шестнадцатиричного числа (всего 32 символа), например set joinkey 918771929bbab19898c8a787f70a8810.[19]

После перейдем в настройку радиомодуля конечного устройства. Надо задать два параметра:

- maxretr — максимальное количество попыток передать данные (по умолчанию модули Unwired Range работают с подтверждением доставки пакетов);
- class — тип устройства. Устройства класса A включают приёмник только на короткий промежуток времени после завершения передачи, устройства класса B могут включать приёмник по расписанию, устройства класса C включены всегда.[13]

Устройства класса A между событиями передачи данных уходят в глубокий сон для экономии энергии, поэтому сменить их параметры из консоли невозможно. Поэтому предлагается выбрать для всех устройств класс C. На рисунке 32 показаны параметры радиомодуля.

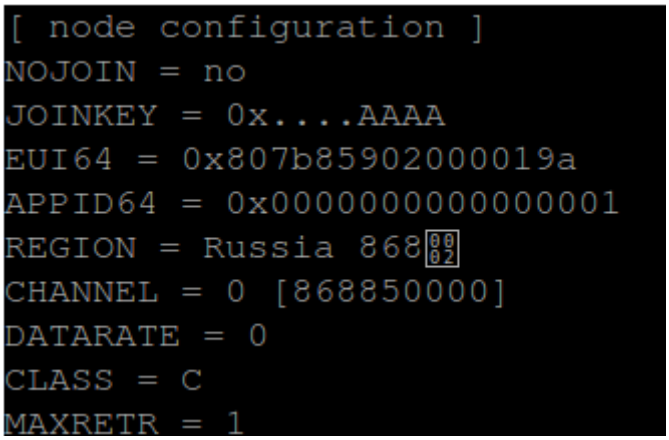


Рисунок 32 – Параметры радиомодуля

Конечное устройство должно только собирать информацию и передавать ее. Это упрощает задачу, так как нам требуется только запустить драйвера выбранных датчиков. Для включения того или иного драйвера периферийного устройства необходимо сначала командой lsmod вывести список доступных драйверов. Для микроконтроллера предоставлено около 17 драйверов для датчиков. Рассмотрим только два, а именно для выбранных датчиков. В таблице 5 показаны параметры драйверов для выбранных датчиков.

Таблица 5 – Стандартные драйвера для выбранных датчиков

ID	Драйвер	Описание	Интерфейс	Класс
15	opt3001	Датчик внешней освещённости (0…65535 лк)	I2C	A, C
17	bme280	Датчик температуры, влажности и давления воздуха	I2C	A, C

Для включения драйвера используется команда mod с названием или номером драйвера в качестве первого параметра и командой на включение (1 или enable) или выключение (0 или disable) в качестве второго.[19] После этих действий можем увидеть ответ микроконтроллера, который показан на рисунке 33.

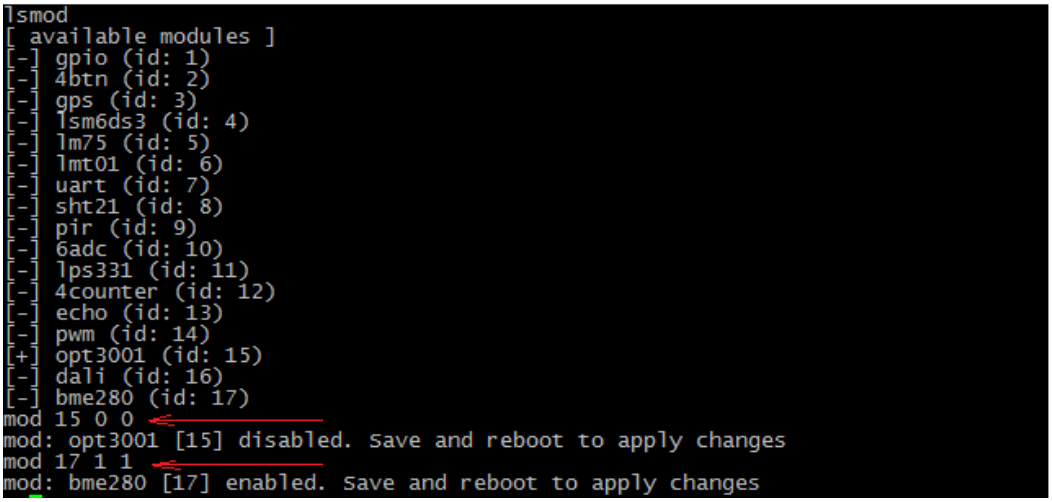


Рисунок 33 – Установка необходимых драйверов

После подключения модуля, сохранения настроек командой save и перезагрузки в системе команд модуля появятся дополнительные команды, одноимённые включённым драйверам. Если была произведена неправильная настройка конечного устройства, всегда модно сбросить настройки командами:

- clear all сбрасывает всю конфигурацию модуля, включая ключи шифрования и EUI64;
- clear keys сбрасывает только ключ шифрования;
- clear modules сбрасывает настройки драйверов (как включённых в данный момент, так и выключенных).[19]

После того как определили все нужные параметры, обновили ПО микроконтроллера, запустили драйвера и соединили все компоненты конечного устройства, надо запустить работу датчиков. Командой get мы запустим работу одного из датчиков. Данные действия показаны на рисунке 34.

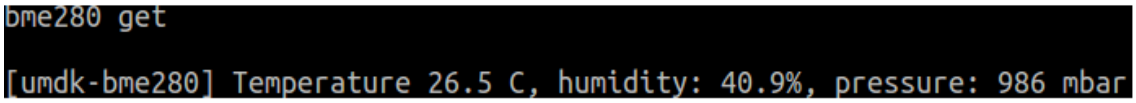


Рисунок 34 – Работа датчика bme280

Как видно из рисунку датчик собирает информацию об окружающей среде. Далее стоит установить с каким промежутком будут приходить сообщения от каждого датчика, для того чтобы не было засорение в эфире и затрат запаса аккумулятора.[13] Таким образом, была выполнена первичная настройка конечного устройства.

В данном параграфе было кратко описаны действия по настройки конечного устройства, установки датчиков и программировании микроконтроллера. При желании можно установить на устройства актуаторы, например светодиодную ленту, приводы для открытия и закрытья и другие. При создании более сложных устройств стоит помнить, что может произойти несовместимость выбранных датчиков. Далее требуется создать сеть передачи данных, которые собирают конечные устройства.

4.3 Настройка связи между компонентами системы

Прежде чем перейти к описанию проделанной работы, стоит сказать о проблеме нехватки оборудования. Ранее планировалось воспользоваться базовой станцией UNWD-BASE R1, но из–за самоизоляции не было возможность поработать с данным оборудованием. Поэтому в роли базовой станции будет выступать уже известный радиомодуль Semtech SX1276. Здесь имеется большая разница в производительности устройства. Задача микрокомьютера базовой станции — поддерживать работу радиочастотной сети, включая механизмы аутентификации, авторизации и шифрования, а также

транслировать компактный бинарный протокол радиообмена в сообщения. Базовая станция имеет веб-интерфейс для настройки сетевых интерфейсов, например для подключения к существующим сетям Wi-Fi. Данная разница резко сокращает возможности в разработки проекта.

Вся информация будет собираться одной базовой станцией. Для того нам понадобится еще один микроконтроллер с радиомодулем. На данный микроконтроллер требуется обновить ПО, но только теперь для БС. Для этого обратимся за помощью к сайту Github. Там можно найти прошивку для базовой станции по названием loralan-gateway.hex, которую потом загрузим в микроконтроллер.

Теперь еще раз взглянем на параметры конечного устройства. На рисунке 35 показан дублированный рисунок 32.

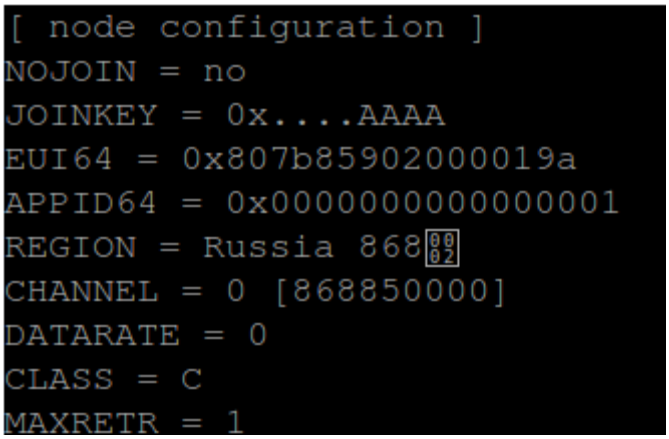


Рисунок 35 – Параметры радиомодуля

На рисунке видны нужные параметры для передачи данных. Из этих параметров самые важные, некоторые будут повторятся:

1. JOINKEY - ключ шифрования в сети, он исключает подсоединение посторонних. В нашем случае ключ очень простой – это AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA.

2.EUI64 - 64-битный идентификатор устройства.

3.REGION и CHANNEL - определяют частоту, на которой работает устройство

4.DATARATE - скорость передачи данных; для выполнения учебных задач оптимально будет поставить скорость 7 (наивысшую), тогда как для задач реальной жизни больше подойдет наименьшая скорость (0), поскольку в этом случае дальность будет максимальной. В таблице показано с какой скоростью может передаваться информация.

Таблица 6 – Стандарты скорости передачи данных радиомодулем

Режим работы	Скорость передачи
DR0	292 бит/с
DR1	537 бит/с
DR2	976 бит/с
DR3	1757 бит/с
DR4	3125 бит/с
DR5	5468 бит/с
DR6	10937 бит/с

5. MAXRETR – количество попыток передачи данных. Имеет смысл поставить 5, если много помех и устройства постоянно отсоединяются (вы увидите сообщения об этом в консоли). Тогда уменьшится количество повторных подключений к сети, и в целом система будет работать быстрее. По умолчанию стоит одна попытка - это означает, что устройство отправляет сообщение, ждёт 3 секунды, и если ответ не получен, то оно считает, что связь потеряна и пробует подключиться заново. Соответственно, если MAXRETR = 5, то будет 15 секунд на получение ответа.

Как мы видим для связи базовой станции требуется знать эти параметры, чтобы получать информацию от конечного устройства. Чтобы соединить устройства мы задаем параметры единый ключ шифрования в сети, идентификатор устройства и частота передачи. После этого базовая станция сама автоматически подключится к устройству. Подключим базовую станцию к компьютеру и посмотрим что будет получать прибор. На экране начнут раз в минуту появляться показатели датчика, который подключен к БС. Что бы

показать как изменяется показания датчиков, просто подышим на конечное устройство. На картинке 36 показана работа передачи сообщений.

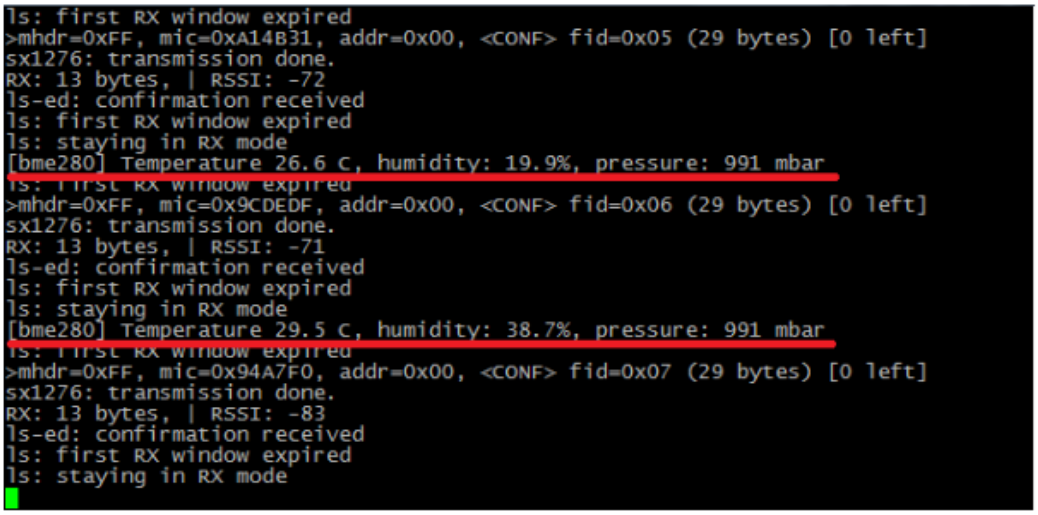


Рисунок 35 – Передача сообщений базовой станции

Таким образом, примерно так можно организовать связь между конечными устройствами и базовой станцией. Как и было ранее задумано, связь организуется с помощью технологии LoRaWAN, но теперь требуется передать все собранные сообщения на сервер. Для это предлагается соединить компоненты системы общим протоколом MQTT. Данный протокол работает поверх TCP/IP, поэтому он не имеет отношения к выборы технологии передачи данных.

При работе с данным протоколом требуется чтоб каждый компонент имел его MQTT-брокера mosquitto. Mosquitto существует в виде пакетов для разных дистрибутивов. Его можно поставить и в embedded-компьютере наподобие Raspberry Pi - это важно, если компьютер функционирует как шлюз между локальной сетью Интернета вещей (LoRa или ZigBee), и собственно Интернетом (WWW).[7] Получается данный MQTT-брокер должен иметься и на микрокомьютере базовой станции. Для простаты работы с MQTT и конечными устройствами радиосети необходим доступ к консоли по ssh. Это сетевой протокол прикладного уровня, позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений (например, для передачи файлов). SSH позволяет безопасно передавать в

незащищённой среде практически любой другой сетевой протокол. Это позволит установить связь базы данных и конечными устройствами.

Итак, роль интерфейса между радиосетью и MQTT выполняет программа MQTT-брокер — mosquitto. Чтобы посмотреть как работает данная платформа, рассмотрим простой пример как организуется подобная связь с помощью протокола MQTT. Для этого установим на свой компьютер mosquitto. Запустим сервер через терминал, но перед этим выберем порт Вообще обычно по умолчанию сервер стартует на 1883 порту и запускается в режиме демона, то есть без вашего участия. И порт может быть занят просто потому, что mosquitto уже стартовал в вашей системе (можете проверить это, выведя список всех процессов). Но нам сейчас важно потренироваться, поэтому запустим на другом порту для наглядности.[19] Данное действие показано на рисунке 36.

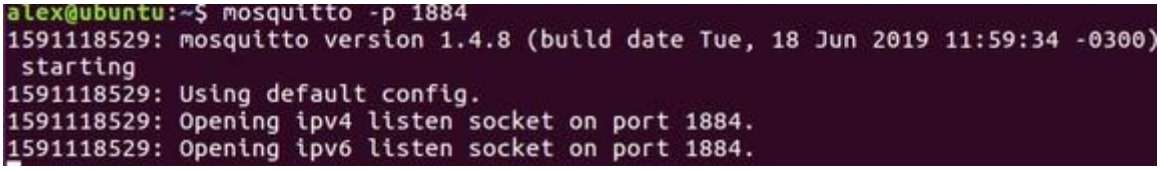


Рисунок 36 – Запуск MQTT–брокера

В другом окне терминала введем команду подписки на заданный нам темой. Команда имеет следующие параметры:

– -h — адрес сервера;

– -t — название топика, в котором публикуется сообщение;

– -m — текст сообщения;

– -q — качество обслуживания.[31]

.В качестве хоста указан localhost (то есть сам компьютер), а в качестве топик придумываем, например mytopic. *Топики* – это простая иерархическая система классификации ресурсов или устройств, описанная в стандарте MQTT. На рисунке 37 показано это действие.

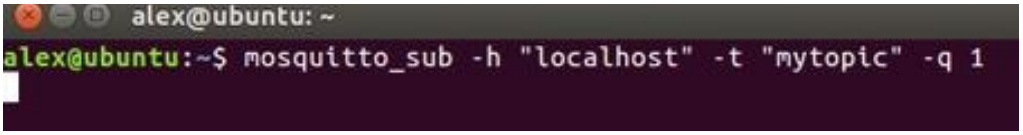


Рисунок 37 – Выполнение подписки на сервер(MQTT-брокер)

Наконец, в третьем окне терминала введите команду, отправляющую сообщение — mosquitto_pub**.** Данным сообщением мы посылаем на сервер (MQTT-брокер) наше сообщение и все кто подписан на данную тему. Рассмотрим все три открытых терминала на рисунке 38.

Рисунок 38 – Работа MQTT протокола

На данном примере понятно как просто работает данный протокол. Для того чтобы подписаться базе данный на все датчики, следует выполнить следующею команду: mosquitto_sub -h "106.109.9.67" -t "devices/lora/#" -q 1.

Параметры мы записываем следующие:

– -h это адрес базовой станции;

– -t название топика, на который происходит подписка. Все модули связи LoRa в нашем случае находятся в субтопике devices/lora/. Как несложно догадаться по названию, субтопик – это нижестоящий в иерархии топик. А что бы получать сообщения из всех топиков вводим #;

– -q качество обслуживания (Quality of Service, QoS). Единица в этом параметре означает, что будет как минимум одна попытка доставить сообщение, и отправитель будет ждать подтверждения. Предлагается

использовать именно такой вариант. Мы рассмотрим разные виды QoS в следующем кейсе;[19]

На данный момент работа с MQTT-сервером производилась просто в консоли при помощи команд mosquitto_sub и mosquitto_pub — это удобно для одноразовых задач вроде считывания данных или отправки тестовых команд. Для взаимодействия с MQTT-сервером существуют различные библиотеки. Например, на курсах IoT Академия Samsung были рассмотрины библиотеки из проекта Eclipse Paho.

Библиотеки Paho помогают работать серверу приложений с протокол MQTT. Paho дается доступ реализации к открытому протоколу MQTT в виде библиотек для почти всех популярных языков программирования: существуют реализации для C, C++, Java, Go, C#, Python, Javascript.

Таким образом, можно организовать передачи данных от конечный устройств до сервера с базой данных. А данной схеме работы издателем является устройства сбора информации из внешней среды, а сервер нашей системы мониторинга выступает в роли подписчика. В данном эксперимент сама платформа mosquitto предлагается установить на базовых станциях. В дальнейшем ее планируется установить в серверной части.

4.4 Создание пользовательской части

Для того чтобы работник фермы мог в любое время ознакомится с показаниями на своей хозяйстве, предлагается создать веб-сайт, в который пользователь мог зайти, если есть выход в интернет. Поэтому для начала стоит описать какими языками программирования будем пользоваться при создании клиентской части.

Самые популярные языки реализации веб–сайта HTML и CSS. Это два основных языка, который создают отображение внешнего вида страницы. Языки HTML и CSS предназначены для верстки сайтов. Верстка сайта — это один из важнейших этапов разработки онлайн-ресурса, в результате которого

нарисованный дизайнером макет превращается в HTML и CSS-код и отображалось в браузере.[11]

HTML – это язык разметки гипертекста. Он отображается в виде документа в удобной для человека форме и интерпретируется обрабатывается браузером. Именно HTML позволяет нам наполнить страницы определенным смыслом, а реализуется это с помощью так называемых тэгов. Тэги – это специальные маркеры, которые определенным образом интерпретируются браузером. Суть тэгов в том, что содержимое страницы, заключенное в разные тэги, по-разному обрабатывается браузером. Например, можем заключить какой либо контент в тэг параграфа, и данное содержимое будет считаться браузером параграфом страницы, или поместить содержимое в тэг списка, и тогда информация внутри него будет интерпретироваться как список. Текстовые документы, содержащие разметку на языке HTML (такие документы традиционно имеют расширение .html или .htm), обрабатываются специальными приложениями(браузерами), которые отображают документ в его форматированном виде.[38]

CSS – это язык описания внешнего вида документа, написанного с использованием языка разметки. Проще говоря, язык CSS предназначен для того, чтобы придавать необходимый внешний вид HTML-документам. Целью создания CSS было отделение описания логической структуры веб-страницы от ее внешнего вида**.** Разумеется, для того, чтобы использовать CSS для придания внешнего вида HTML-документу, нужно этот документ как-то связать со стилями, т.е. «сообщить» HTML-документу, что он будет оформлен с помощью CSS. Для этого существуют различные способы подключения CSS к документу, которые дают браузеру знать, что к странице в целом, либо к каким-то отдельным ее элементам должно быть применено стилевое оформление. Таблицы стилей могут располагаться как непосредственно внутри того, документа, к которым они будут применяться, так и находиться в отдельном файле, имеющем расширение .css.[38]

Главное для пользователя в системы мониторинга, чтобы было удобным отслеживание изменение показателей датчика, поэтому предлагается отображать основную информацию графиками. Для создания подобных сайтов существуют специальные люди веб–разрабочики, которые обучаются этому многие годы. Поэтому для выпускной работы было создано две страницы, а именно главная страница системы мониторинга и вывод показаний с датчиков. Для наглядности эти веб–страницы показаны на рисунках 39 и 40.

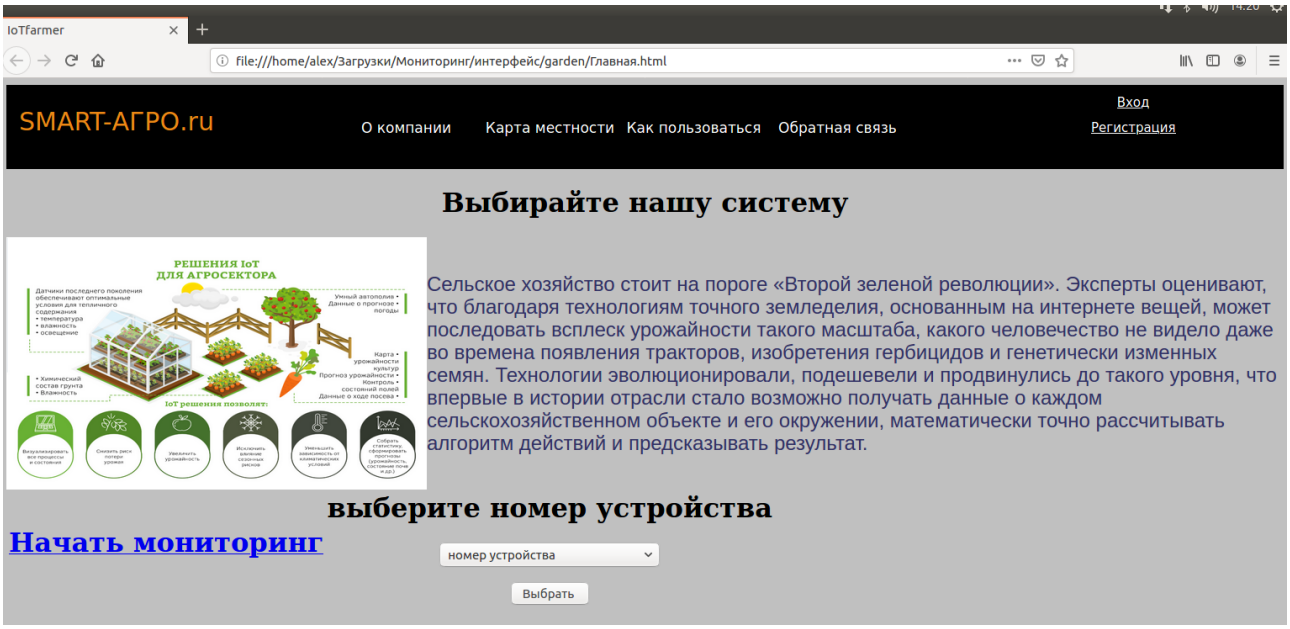


Рисунок 39 – Главная страница веб–сайта

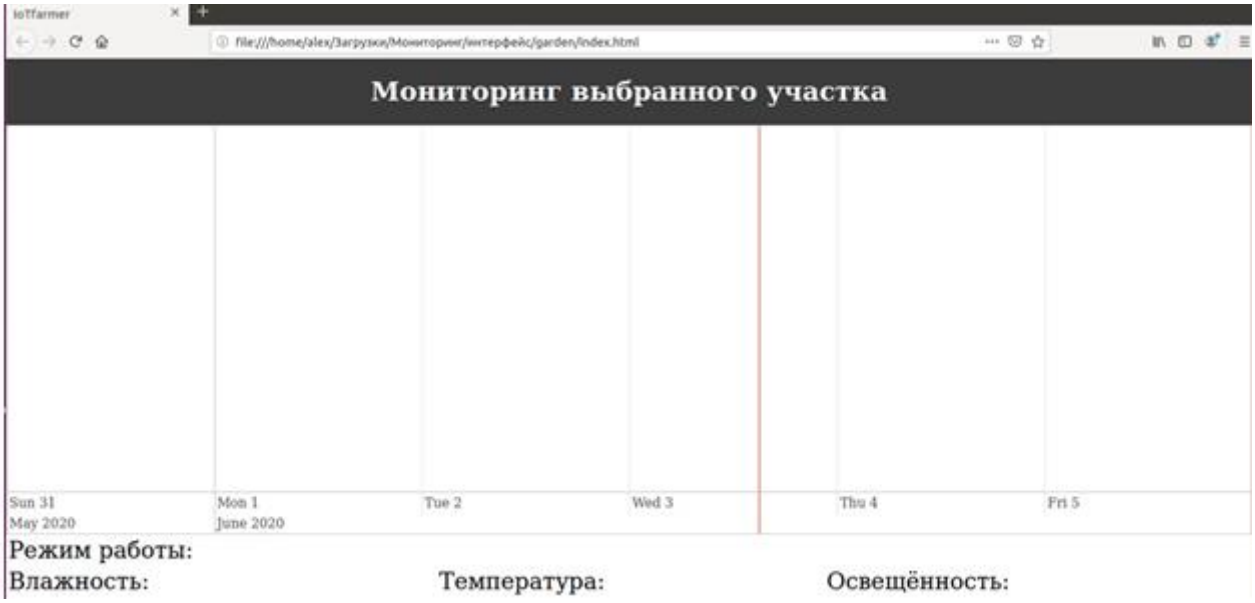


Рисунок 40 – Станица мониторинга датчиков

Как видно на рисунке 40, должен быть график. Но так как HTML и CSS отвечают только за отображение внешнего вида сайта, поэтому требуется еще один язык, который будет отображать взаимодействие с базой данных. Предлагается воспользоваться языком программирования JavaScript. Язык JavaScript поможет "оживить" сайт и будет динамично взаимодействовать с сервером.

JavaScript – это язык программирования, нашедший наиболее широкое применение в браузерах для придания интерактивности веб-страницам. Важно понимать, что это далеко не единственная область применения JavaScript, однако в рамках нашей темы наиболее уместно рассматривать именно такой вариант использования этого языка. Основной задачей JavaScript в рассматриваемом нами контексте является манипулирование элементами DOM-модели web-страницы. DOM предоставляет структурированное представление документа и определяет то, как эта структура может быть доступна из программ, которые могут изменять содержимое, стиль и структуру документа. Представление DOM состоит из структурированной группы узлов и объектов, которые имеют свойства и методы. DOM соединяет веб-страницу с языками описания сценариев либо языками программирования. Согласно DOM, документ (например, веб-страница) может быть представлен в виде дерева объектов, обладающих рядом свойств, которые позволяют производить с ним различные манипуляции. Именно эти манипуляции и позволяет нам осуществлять над элементами страницы язык JavaScript.[38]

Итак, можно сказать, что JavaScript – это язык, который позволяет активно управлять структурой нашей страницы, манипулировать ее элементами**.** На практике это находит свое применение при создании различных анимационных эффектов, эффектов перемещения, растворения, увеличения и уменьшения объектов. Программы, написанные на языке JavaScript, называются скриптами. Они могут встраиваться в HTML и выполняться автоматически при загрузке веб-страницы. Добавим в текстовую

программу скрипты языка JavaScript, а также библиотеку графиков и библиотеку jQuery, которая позволяет манипулировать документами HTML, совершать обработку событий, анимацию и построение пользовательских приложений для веб–приложений. Таким образом, создали минимальную клиентскую часть.

Из за того, что проект находился в разработке и в конце учебы не было доступа к оборудованию, возникает проблема, как проверить работу веб–сайта. Воспользуемся готовыми программами из курсов IoT Академия Samsung и запустим вспомогательную программу-эмулятор, которая начнет генерировать данные. В качестве серверная части воспользуемся программой написанной на серверном языке NodeJS, при запуске которых будем взаимодействовать с нашей клиентской частью. На рисунке 41 показано как будет выглядеть сайт после запуска вспомогательных программ. Таким образом, можно предположить, если к клиентской части добавить библиотеки протокола MQTT и полностью наладить взаимодействия сервера и вею–приложения, то система мониторинга была сделана для первых испытаний.

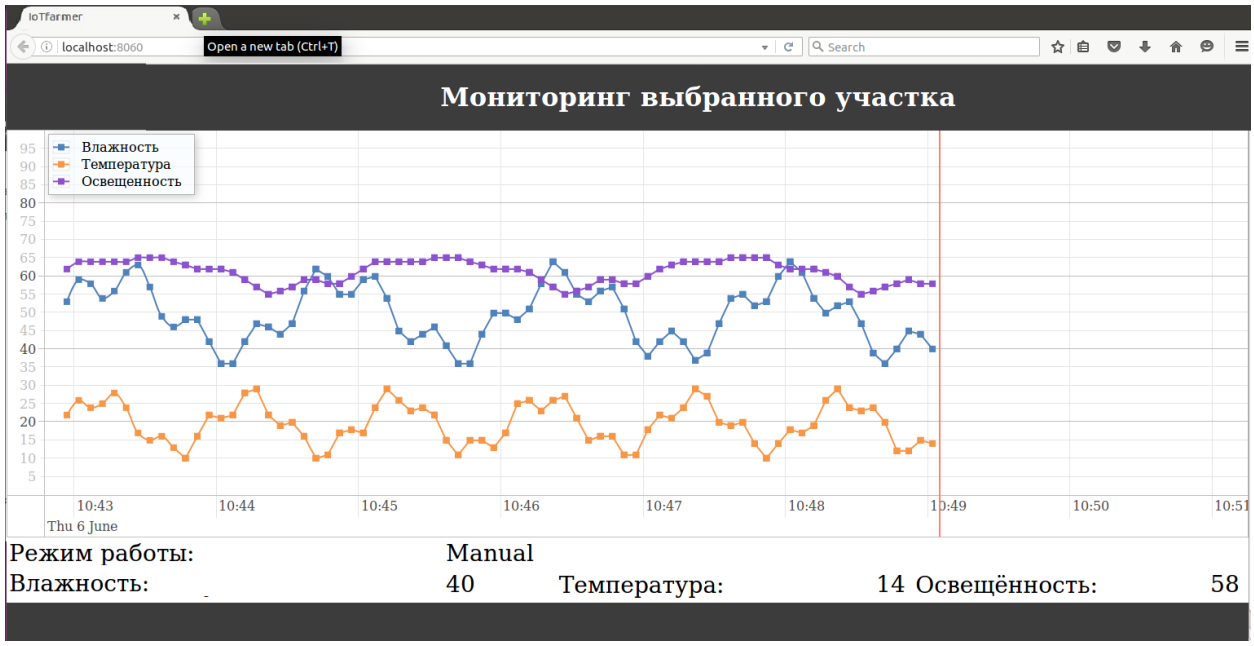


Рисунок 41 – Станица мониторинга датчиков

4.5 Описание дальнейшей работы и выводы по главе

В данном главе была проведен обзор аналогичный систем мониторинга для сельского хозяйства, выявлены недостатки систем, которые уже существуют на рынке, и сделаны следующие выводы. Данные системы в России не распространены, а заграничные аналоги имеют небольшие недостатки по сравнению с нашей системой, например большое количество оборудования или неподходящие датчики для сбора информации.

В главе кратко описаны эксперименты с имеющемся оборудованием. Проведены настройки конечного устройства, установки датчиков и программировании микроконтроллера. Показано как можно организовать передачи данных от конечный устройств до сервера с базой данных на примере протокола MQTT. Приведен пример реализации клиентской части и как предлагается отображать показания датчиков для понимания пользователя.

В дальнейшем предполагается создать макет для первичного тестирования. Но для этого нужно изучить серверные языки программирования для создания базы данных, сервера приложений с веб–сервером. Продолжить изучения клиентских языков программирования, которые будут способствовать созданию оригинального и простого веб-приложения. Изучить оборудование базовых станций с технологией LoRaaWAN, а так же проанализировать возможность использования облачных технологий для сервера.

Если продолжить разработку могут встретится проблемы с арендой доменного имени для сайта, так как это требует денег. Выбор датчиков в основном ограничивался имеющимся оборудованием на курсах IoT Академия Samsung. И главной проблемой является сложность создания системой мониторинга сельского хозяйства и внедрение ее в обиход одним человеком.

ЗАКЛЮЧЕНИЕ

В рамках дипломной работы мною был предложен проект системы мониторинга сельского хозяйства SMART-АГРО. На данный момент продовольственная проблема одна из главных в мире. На мой взгляд одним из наиболее эффективных инструментов решения данной проблемы это цифровизация сельского хозяйства. Создания «Умного сельского хозяйства» может максимально автоматизировать сельскохозяйственную деятельность и даст подробный мониторинг нужных параметров, что повысить урожайность и качество продукции. Поэтому мной был проведен анализ в данной области.

При обзоре аналогичный систем мониторинга для сельского хозяйства, выявлены недостатки систем, которые уже существуют на рынке. Например, данные системы в России не распространены, а заграничные аналоги имеют небольшие недостатки по сравнению с нашей системой, например большое количество оборудования или неподходящие датчики для сбора информации.

В работе содержится описание концепции технологии интернета вещей, развитие данной технологии, описание архитектуры технологии, а так же была предложена архитектура системы мониторинга на основе интернета вещей. Были описаны и проанализированы основные компоненты архитектуры технологии интернета вещей, а именно конечные устройства, технологии беспроводной связи, программное обеспечение и пользовательский интерфейс. В итоге этого была выбрана архитектура проекта:

- объекты и сенсоры, которые собирают информацию;
- базовые станции какой либо технологии беспроводной связи;
- iot платформа для обработки и анализа информации;
- устройства и приложения для мониторинга информации.

Далее определили конкретные компоненты для данного проекта и предложили способы реализации проекта. Были подобраны конечные устройства, проанализировали технологию передачи данных LoRaWAN, выбрали основной сетевой протокол передачи данных, а также платформу IoT.

Описали архитектуру серверной и клиентской части и определили какие языки программирования понадобится при разработки.

На практике были проведены эксперименты с имеющемся оборудованием. Проведены настройки конечного устройства, установки датчиков и программировании микроконтроллера. Показано как можно организовать передачи данных от конечный устройств до сервера с базой данных на примере протокола MQTT. Приведен пример реализации клиентской части и как предлагается отображать показания датчиков для понимания пользователя.

При написании выпускной работы воспользовался наработками курсах IoT Академия Samsung. Ранее проект системы мониторинга параметров окружающей среды для сельского хозяйства разрабатывался в рамках программы поддержки коммерчески ориентированных научно-технических проектов молодых ученых «УМНИК». Стоит отметить, что проект имеет некоторые проблемы, например нехватка нужного оборудования или приобретение доменного имени требуется денежные вложения.

Надо помнить, что интернет вещей становится реальностью. Постоянный и увеличивающийся обмен данными требует развития новых сервисов, которые должны соединить нас с физическим миром вокруг. Эти сервисы также должны быть построены на полностью новых бизнес-моделях и обеспечить новые финансовые потоки. С помощью Интернета вещей взаимодействие объектов, среды и людей будет во многом переплетено, что обещает сделать мир «умным» – более благоустроенным для человека. Поэтому развитие подобных систем будет способствовать развитию многих сфер нашей страны.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Ковалев Е. Мировой продовольственный кризис: эскалация проблем.– Мировая экономика и международные отношения.– 2010, № 4, С. 15-23

2. Кучерявый А. Е., Прокопьев А. В., Кучерявый Е. А. Самоорганизующиеся сети.– СПб.: Любавич. 2011. 312 с.

3. Кранц Мачей, Интернет вещей. Новая технологическая революция.– Издательство: Бомбора, 2018, 336с.

4. Власова О.И., Плодородие черноземных почв и приемы его воспроизводства в условиях Центрального Предкавказья.–Ставрополь: АГРУС Ставропольского гос. аграрного ун– та, 2014., 308с.

5. Кучерявый А. Е., Интернет Вещей. Электросвязь.– 2013. № 1. С. 21–24.

6. Ли Перри, Архитектура интернета вещей.– Издательство: ДМК-Пресс, 2019.454 с.

7. Тихвинский В.О., Бабин А.И. и Бочечка Г.С Сети IoT/M2M: технологии, архитектура и приложения.–М.: Медиа паблишер, 2017, 319с.

8. Пьявченко О.Н., Панич А.Е., Мокров Е.А., Принципы построения и архитектура перспективных информационно-измерительных систем мониторинга, диагностики и управления на базе интеллектуальных датчиков.– Нано- и микросистемная техника, 2002, № 10, с. 30–33.

9. Кашкаров А. П., Электронные устройства для глушения беспроводных сигналов (GSM, Wi-Fi, GPS и некоторых радиотелефонов).– Издательство: ДМК Пресс, 2016. 210с.

10. Интернет вещей» (IoT) в России: технология будущего, доступная уже сейчас. – https://www.pwc.ru/ru/publications/iot/iot-in-russia-research-rus.pdf.,64с.

11. Д. Д. Гуинар, В. М. Трифа., Создание Сети вещей.–Издательство: MANNING,2016,344с.

12. В.М. Шарапова, Е.С. Полищука, Датчики: Справочное пособие.– М.: Техносфера, 2012. 624 с.

13. Ревич Ю.В., Практическое программирования микроконтроллеров Atmel AVR на языке ассемблера.–3-е изд., испр.–СПб.: БХВ–Петербург, 2014,368с.

14. Пол Беделл, Сети. Беспроводные технологии.–Издательство: НТ Пресс,2008,395с.

15.Федоров В. В., Модули Bluetooth в промышленных приложениях и системах сбора информации.– Беспроводные технологии 2006. № 2.

16.Каулио В. В. Управление встраиваемым Wi-Fi-модулем WIZ610wi компании WIZnet .– Беспроводные технологии. 2010. № 2.

17. Интернет-журнал IoT ИННОВАТОР. – https://controleng.ru/wp-content/uploads/Innovator2.pdf. – 2017,10с.

18. Леонтьев В.П., Новейшая самоучитель. Компьютер и интернет., – М.: Издательство "ОЛМА Медиа Груп",2017,688с.

19. IT Академия Samsung. – https://myitacademy.ru/edu/.

20. Спецификация Bosch BME280 Humidity and Pressure Sensor. – https://ru.mouser.com/datasheet/2/783/BST-BME280-DS002-1509607.pdf., 2018,55с.

21. Спецификация OPT3001 Ambient Light Sensor. – http://www.ti.com/lit/ds/sbos681c/sbos681c.pdf?ts=1591049668764.,2017,47с.

22. Спецификация LM75 Digital Temperature Sensor and Thermal Watchdog with Two-Wire Interface. – http://www.gaw.ru/pdf/NS/Thermal_Management/LM75.pdf. – 2002,18с.

23. Спецификация Datasheet SHT21. – https://ru.mouser.com/datasheet/2/682/Sensirion_Humidity_Sensors_SHT21_Datasheet-1511537.pdf. – 2014, 15с.

24. Спецификация LPS331AP. – https://www.pololu.com/file/download/LPS331AP.pdf?file_id=0J622. – 2012,36с.

25. Спецификация LMT01LPG, ДАТЧИК ТЕМПЕРАТУРЫ ЦИФРОВОЙ. – https://www.elruselement.ru/catalog/datchiki/ datchiki_temperatury/ lmt01lpg/.

26. Спецификация РАДИОМОДУЛЬ LoRaWAN CLASS C ДИАПАЗОНА 868МГц С ИНТЕГРИРОВАННОЙ PCB-АНТЕННОЙ. – https://www.compel.ru/item-pdf/978f3eeb83a83a7e3c2fcf1a9ac2e650/pn/_larteh~lrtx-868-pcb-caat.pdf. –2017,5с.

27. Сергеев А.Н., Основы локальных компьютерных сетей. Учебное пособие.,–Издательство Лань,2016,184с.

28. Статья от компании Mikro tik Интернет вещей: LoRa устройства. – https://lanmarket.ua/stats/internet-veshchey-lora-ustroystva-ot-mikrotik/.– 2019.

29. Спецификация LoRa Specification. – https://www.rs-online.com/designspark/rel-assets/ds-assets/uploads/knowledge-items/application-notes-for-the-internet-of-things/LoRaWAN%20Specification%201R0.pdf. – 2015, 82с.

30. iTech Технологии связи сборник статей. Технология LoRaWAN. – https://itechinfo.ru/content/%D1%82%D0%B5%D1%85%D0 %BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%8F-lora. – 2020.

31.IOT Интернет вещей. Протокол MQTT. – http://i-o-t.ru/protokol-mqtt/,2020.

32.Черняк Л. Платформа Интернета-вещей., Открытые системы. СУБД. – 2012.- №7. – С.44-45.

33. Тихонов А., Этапы разработки интерфейса. – https://designpub.ru/%D1%8D%D1%82%D0%B0%D0%BF%D1%8B%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81%D0%B0-baf666d6ad8f. – 2017.

34. Гребешков А., Самсонов М. Интернет вещей. – Самара: ПГУТИ, ООО «Издательство Ас Гард», 2014. - С. 15–162.

35. Компания Tibbo Systems. – https://aggregate. tibbo.com/ru/.

36.Компания Growlink. – https://growlink.com/ controllers/.

37.Компания Arable. – https://www.arable.com/.

38.Крис Аквино, Тодд Ганди., Front-end. Клиентская разработка для профессионалов.–СПб.: Издательство: Питер., 2018,512с.

Приложение А

(index.js)

```
var PROTOCOL = 'lora';
var SRV_PORT = 8060;
//var DB_NAME = ':memory:';
var DB_NAME = 'smartgarden.db';
var MQTT_ADDRESS = 'localhost';
var MQTT_PORT = 1884;
var MQTT_CLIENT = 'smartgarden_server';
var APP_MODULES = [
	'devices/'+PROTOCOL+'/807B85902000021E',
	'devices/'+PROTOCOL+'/807B85902000032D',
];
var GPIO_MODULE = 'devices/'+PROTOCOL+'/807B85902000021C';
var GPIO_MAP = {
	light: {port: 17, state: -1},
	pump: {port: 16, state: -1},
	auto: {port: -1, state: 0}
};
var express = require('express');
var path = require('path');
var bodyParser = require('body-parser');
var jsonParser = bodyParser.json();
var sql = require('sqlite3').verbose();
var db = new sql.Database(DB_NAME);
db.serialize(function() {
	db.run("CREATE TABLE IF NOT EXISTS temperature (time INT, value INT)");
	db.run("CREATE TABLE IF NOT EXISTS humidity (time INT, value INT)");
	db.run("CREATE TABLE IF NOT EXISTS luminosity (time INT, value INT)");
});
var mqtt = require('mqtt');
var client = mqtt.connect('mqtt://'+MQTT_ADDRESS+':'+MQTT_PORT, MQTT_CLIENT);
client.on('connect', onConnected);
client.on('message', onMessageReceived);
var app = express();
app.use(express.static('garden'));
app.get('/', express.static('garden'));
app.get('/get/:value', get);
app.get('/get/:value/:mode', get);
app.get('/get/:value/:mode/:days', get);
app.get('/cmd/:command/:value', cmd);
app.listen(SRV_PORT);
function subscribeModule(topic) {
	client.subscribe(topic + '/miso/#');
}

function onMessageReceived(topic, message) {
	//console.log('> onMessageReceived: ' + message);
```

```
	try {
		var m = message.toString().replace(/([\d]),([\s]*\])/g, '$1$2');
		m = JSON.parse(m);
		if (m.data !== undefined) {
			var val = undefined;
			var t = undefined;
			if (m.data.adc2 !== undefined) {
				t = 'humidity';
				val = m.data.adc2.toString();
				//val = Math.floor(100-(parseInt(m.data.adc2.toString())-
590)/910*100);
				if (val > 100) val = 100;
				if (val < 0) val = 0;
				db.run("INSERT INTO " + t + " VALUES ("+now()+", "+val+")");
			}
			if (m.data.adc3 !== undefined) {
				t = 'temperature';
				val = m.data.adc3.toString();
				//val = Math.floor((parseInt(m.data.adc3.toString())-750)/10+21);
				db.run("INSERT INTO " + t + " VALUES ("+now()+", "+val+")");
			}
			if (m.data.luminocity !== undefined) {
				t = 'luminosity';
				//val = m.data.luminocity.toString();
				val =
Math.floor(Math.log(parseInt(m.data.luminocity.toString())+1)*10);
				db.run("INSERT INTO " + t + " VALUES ("+now()+", "+val+")");
			}
			if (m.data.gpios !== undefined) {
				for(var cmd in GPIO_MAP) {
					if (GPIO_MAP[cmd].port > -1) {
						GPIO_MAP[cmd].state =
parseInt(m.data.gpios[GPIO_MAP[cmd].port]);
					}
				};
			}
		} else {
			console.log('No data found: ' + m.toString());
		}
	} catch (e) {
		console.log('Invalid JSON: ' + message + e);
	}
}
function onConnected() {
	subscribeModule(GPIO_MODULE);
	APP_MODULES.forEach(subscribeModule);

	var topic = GPIO_MODULE + '/mosi/gpio';
	var msg = 'get all';
	client.publish(topic, msg.toString());
}
```

```
function now(dayshift) {
	var days = Math.floor(Date.now()/1000);
	if (typeof dayshift === "number" || typeof dayshift === 'string') {
		days += parseInt(dayshift)*86400;
	}
	return days;
}

function index(req, res) {
	res.send('INDEX');
}

function get(req, res) {
	var p = req.params;
	try {
		switch (p.value) {
			case "latest":
/**/
				db.get("SELECT humidity.value as Hv, humidity.time as Ht, \
					temperature.value as Tv, temperature.time as Tt, \
					luminosity.value as Lv, luminosity.time as Lt FROM humidity
\
					LEFT JOIN temperature ON temperature.time = (SELECT
max(time) FROM temperature) \
					LEFT JOIN luminosity ON luminosity.time = (SELECT
max(time) FROM luminosity) \
					WHERE humidity.time=(SELECT max(time) FROM
humidity)", function(err, row) {
					var data = {
						history : {
							temperature : null,
							humidity : null,
							luminosity : null
						},
						status : GPIO_MAP
					};
					if (row !== undefined) {
						console.log(JSON.stringify(row));
						data.history.temperature = {time : row.Tt, value:
row.Tv};
						data.history.humidity = {time : row.Ht, value: row.Hv};
						data.history.luminosity = {time : row.Lt, value:
row.Lv};
						res.json(data);
					} else {
						console.error('Error:',err);
						res.status(200).json(data);
						return;
					}
				});
				break;
```

```
                        case "data":
                                var mode = '';
                                switch (p.mode) {
                                        case 'temperature': case 'humidity': case 'luminosity': mode =
p.mode; break;
                                        default:
                                                res.status(404).json({error:404});
                                                return;
                                }
                                if (typeof p.days !== 'undefined') {
                                        var limit = parseInt(p.days.toString());
                                }

                                db.all("SELECT * FROM " + mode + " ORDER BY time DESC" +
(limit ? " LIMIT "+limit : ''), function(err, rows) {
                                        res.json(rows);
                                });
                                break;
                        default:
                                res.status(404).json({error:404});
                                return;
                }
        } catch (e) {
                console.log('Invalid parameter');
        }
}
function cmd(req, res) {
        var p = req.params;
        try {
                switch (p.command) {

                }
        } catch (e) {
                console.log('Wrong command');
        }
}
```

Приложение Б

(sensor_emul.js)

```
var PROTOCOL = 'lora';
var MQTT_ADDRESS = 'localhost';
var MQTT_PORT = 1884;
var MQTT_CLIENT = 'smartgarden_emul';
var EUI_ADC = "807B85902000021E";
var EUI_LIT = "807B85902000032D";
var EUI_me = "807B85902000021C";
var TIMER_INTERVAL = 5000;

var mqtt = require('mqtt');
var client = mqtt.connect('mqtt://'+MQTT_ADDRESS+':'+MQTT_PORT, MQTT_CLIENT);
client.on('connect', onConnected);
client.on('message', onMessageReceived);

var GPIO = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0];

function onMessageReceived(topic, message) {
	var t = topic.toString();
	var m = message.toString();
	console.log(' > ', t, m);

	if (t != "devices/"+PROTOCOL+"/"+EUI_me+"/mosi/gpio") return;

	if (m == "get all") {
		var data_GPIO = {
			data : {
				gpios: GPIO
			},
			status : {
				devEUI : EUI_me,
```

```
                        rssi: -16,
                        temperature : 28,
                        battery : 3300
                    }
                };
                client.publish("devices/"+PROTOCOL+"/"+EUI_me+"/miso",
JSON.stringify(data_GPIO));
        }

}
function onConnected() {
        setInterval(sendData, TIMER_INTERVAL);
        client.subscribe("devices/"+PROTOCOL+"/"+EUI_me+"/#");
}

function sendData() {
        var data_ADC = {
                data : {
                        adc2                                                    :
Math.floor(Math.sin(Date.now()/15000)*10+Math.sin(Date.now()/4000)*5+50),
                        adc3                                                    :
Math.floor(Math.sin(Date.now()/10000)*7+Math.sin(Date.now()/3000)*3+20)
                },
                status : {
                        devEUI : EUI_ADC,
                        rssi: -16,
                        temperature : 30,
                        battery : 3300
                }
        };
        var data_LIT = {
                data : {
```

```
                luminocity:
Math.floor(Math.sin(Date.now()/20000)*200+Math.sin(Date.now()/5000)*50+Math.sin(Date.now()
/1000)*5+500)
            },
            status : {
                devEUI : EUI_LIT,
                rssi: -16,
                temperature : 28,
                battery : 3300
            }
        };
        var mADC = JSON.stringify(data_ADC);
        var mLIT = JSON.stringify(data_LIT);
        console.log('Sending: ' + mADC + '; ' + mLIT)
        client.publish("devices/"+PROTOCOL+"/"+EUI_ADC+"/miso/adc", mADC);
        client.publish("devices/"+PROTOCOL+"/"+EUI_LIT+"/miso/opt3001", mLIT);
}
```

Приложение В

(index.html)

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8" />
   <meta name="description" content="IoT" />

   <title>IoTfarmer</title>

   <link rel="stylesheet" type="text/css" href="css/vis.min.css" />
   <link rel="stylesheet" type="text/css" href="css/style.css" />
        <script src="js/jquery-3.3.1.min.js"></script>
        <script src="js/vis.min.js"></script>
        <script src="js/mqttws31.min.js"></script>
        <script src="js/main.js"></script>
</head >

   <body>
<h1 align="center" ; font-size:30px"> Мониторинг выбранного участка </h1> <body/>
                </tr>
        </table>

        <div  style="background-color:white" id="visualization"></div>

        <table bgcolor=white width=100%>
        <tr>
        <td>  <div class="information">Режим работы:</div></td>
        <td><div class="information" id="str-auto"></div></td>
        </tr>
        <tr>
        <td><div class="information">Влажность: </div></td>
        <td><div class="information" id="str-humidity"></div></td>
        <td><div class="information">Температура:</div></td>
        <td><div class="information" id="str-temperature"></div></td>
        <td><div class="information">Освещённость:</div></td>
        <td><div class="information" id="str-luminosity"></div></td>
        </tr>
        </table>
</body>
</html>
```

Приложение Г

(main.js)

```
var dataset;
var MANUAL=0;
var AUTO=1;

var mode=0; //0 is manual, 1 is automatic
var Status = undefined;

var HISTORY_LIMIT = 10;
var TIMER_INTERVAL = 1000*2 // 2 sec

var timerRequest = undefined;
var dataGroups = {
	humidity: 0,
	temperature: 1,
	luminosity: 2
};

function init_graph() {
	var groups = new vis.DataSet([
	{id: 0, content: 'Влажность'},
	{id: 1, content: 'Температура'},
	{id: 2, content: 'Освещенность'}
	]);

	var container = document.getElementById('visualization');
	dataset = new vis.DataSet();
	var options = {
		legend: true,
		height: 500,
		dataAxis: {
			left: {
				range: {
					min:0, max: 100
				}
			}
		}
	};
	var graph2d = new vis.Graph2d(container, dataset, groups, options);
}

function showStatus(type, value) {
	switch (type) {
		case "auto":
			var val = ["Manual", "Auto"];
			$("#str-"+type).text(val[value]);
			break;
```

```
		default:
			$("#str-"+type).text(value);
	}

}

function setButtonState(mode, state)
{
	var b = $("#btn-"+mode);
	switch (state) {
		case 0:
			b.removeClass("btn-disabled");
			b.removeClass("btn-on");
			break;
		case 1:
			b.removeClass("btn-disabled");
			b.addClass("btn-on");
			break;
		default:
			b.addClass("btn-disabled");
	}
}

function toggleCmd(cmd) {

	if (typeof Status !== 'undefined') {
		var val = Status[cmd].state;
		if (val != -1) {
			val = val == 0 ? 1 : 0;
			var url = '/cmd/'+cmd+'/'+val;

			var data = {
				url : url,
				method: "GET",
				error: onError,
				timeout: 5000,
				cache: false
			};

			data.success = function(d) {
				if (d && d.result == 'OK') {
					Status[cmd].state = this.success.val;
					setButtonState(cmd, this.success.val);
					showStatus(cmd, this.success.val);
				}
			};
			data.success.val = val;
			$.ajax(data);


		}
```

```
	}
}

function toggleLight() {
	toggleCmd('light');
}

function togglePump() {
	toggleCmd('pump');
}

function toggleMode() {
	toggleCmd('auto');
}


function onError() {

}

function onGetLatest(data, s, x) {

	if (data != undefined) {
		console.log(data);
		if (data.history != undefined) for (var type in dataGroups) {

			if (data.history[type] != null) {
				dataset.add({
					x: data.history[type].time * 1000,
					y: data.history[type].value,
					group: dataGroups[type]
				});
			}

			showStatus(type, data.history[type].value);
		}
		if (data.status != undefined) {
			Status = data.status;
			for (var mode in data.status) {
				setButtonState(mode, data.status[mode].state);
				showStatus(mode, data.status[mode].state);
			}
		}
	}
}

function onGetHistory(data, type) {
	if (data.length != undefined) {
		data.forEach(function(p) {
			dataset.add({
				x: p.time * 1000,
```

```
                                y: p.value,
                                group: dataGroups[type]
                        });
                });
        }
}

function requestLatest() {
        $.ajax({
                method: "GET",
                url: '/get/latest',
                success: onGetLatest,
                error: onError,
                timeout: 5000,
                cache: false
        });
}

function requestHistory() {
        var data = {
                method: "GET",
                error: onError,
                timeout: 5000,
                cache: false
        };

        for (var group in dataGroups) {
                data.url = '/get/data/' + group + '/' + HISTORY_LIMIT;
                data.success = function(d) { onGetHistory(d, this.success.group);}
                data.success.group = group;
                $.ajax(data);
        }

}

$(document).ready( function() {
        init_graph();

        requestHistory();
        timerRequest = setInterval(requestLatest, TIMER_INTERVAL);
});

function control_humidity(humidity) {

}
```

№	Формат	Обозначение	Наименование	Кол-во
			Тестовые документы	
1	А4	ЮУрГУ – М.11.04.02.2020.577.00 ПЗ	Пояснительная записка	117
			Графические документы	
1	А1	ЮУрГУ – М.11.04.02.2020.577.00 Д2	Архитектура системы	
			мониторинга	1
2	А1	ЮУрГУ – М.11.04.02.2020.577.00 ТБ	Сравнительные таблицы	1
3	А1	ЮУрГУ – М.11.04.02.2020.577.00 Д6	Компоненты конечного устройства	1
4	А1	ЮУрГУ – М.11.04.02.2020.577.00 Д7	Беспроводные технологии	1
5	А1	ЮУрГУ – М.11.04.02.2020.577.00 Д8	Алгоритм работы системы	1
6	А1	ЮУрГУ – М.11.04.02.2020.577.00 Д9	Клиентский интерфейс	1

					ЮУрГУ – М.11.04.02.2020.577.00 Д4			
Из	Лист	№ докум.	Подп	Дата				
Разраб.		Горшков А.В.			Разработка системы мониторинга в сфере сельского хозяйства на основе технологии интернета вещей Ведомость дипломной работы	Лит.	Лист	Листов
Пров.		Николаев				М	1	1
Рецензен						ЮУрГУ Кафедра ИКТ		
Н.контр.		Спицына						
Утв.		Даровских С.Н.						