

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

УТВЕРЖДАЮ
Заведующий кафедрой
_____ О.В. Непомнящий
«__» _____ 2024 г.

БАКАЛАВРСКАЯ РАБОТА

090301 Информатика и вычислительная техника

Разработка системы мониторинга и управления абонентскими терминалами в сети спутниковой связи

Руководитель	_____	_____	ст. преподаватель	А.Г. Хантимиров
	<i>подпись</i>	<i>дата</i>	<i>должность, ученая степень</i>	
Выпускник	_____	_____		Д.А. Анциферов
	<i>подпись</i>	<i>дата</i>		
Нормоконтролёр	_____	_____	ст. преподаватель	А.Г. Хантимиров
	<i>подпись</i>	<i>дата</i>	<i>должность, ученая степень</i>	

Красноярск 2024

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

УТВЕРЖДАЮ

Заведующий кафедрой

_____ О.В. Непомнящий

« ____ » _____ 2023 г.

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы**

Красноярск 2023

Студенту Анциферову Денису Александровичу

фамилия, имя, отчество

Группа КИ20-06Б Направление (специальность) 090301

номер

код

Информатика и вычислительная техника

наименование

Тема выпускной квалификационной работы: «Разработка системы мониторинга и управления абонентскими терминалами в сети спутниковой связи»

Утверждена приказом по университету № _____ от _____

Руководитель ВКР: А.Г. Хантимиров, ст. преподаватель кафедры ВТ,

инициалы, фамилия, учёная степень, должность, место работы

ИКИТ СФУ

Исходные данные для ВКР:

1. SNMP-протокол

2. Осуществить анализ существующих систем

3. Разработать и протестировать систему

Перечень разделов ВКР:

1. Анализ предметной области

2. Проектирование системы управления терминалами

3. Разработка и тестирование системы.

Перечень графического материала: демонстрационный видеоматериал,
презентация со слайдами

Руководитель ВКР

подпись

А.Г. Хантимиров

инициалы, фамилия

Задание принял к исполнению

подпись

Д.А. Анциферов

инициалы, фамилия

дата

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка системы мониторинга и управления абонентскими терминалами в сети спутниковой связи» содержит 45 страниц текстового документа, 28 рисунков, 11 использованных источников.

NMS, SNMP, MIB, ЦЕНТРАЛЬНАЯ СТАНЦИЯ, АБОНЕНТСКИЙ ТЕРМИНАЛ, СИСТЕМА УПРАВЛЕНИЯ И МОНИТОРИНГА СТАНЦИЯМИ, PYTHON, C++, FLASK, NET-SNMP

Цель работы – проектирование и разработка системы управления абонентскими терминалами и центральными станциями средствами протокола SNMP.

Задачи, решенные в процессе разработки:

- проведен анализ существующих аналогов;
- обзор предметной области;
- сформированы требования к разрабатываемой системе;
- выбраны инструменты для разработки системы;
- предложена своя реализацию;
- выполнено тестирование;
- написаны инструкции по сборке, тестированию и использованию разработанных программ.

В ходе разработки системы были спроектированы и разработаны SNMP агент, который управляет устройством, а также NMS, которая позволяет оператору составлять расписания, по которому управляются устройство.

СОДЕРЖАНИЕ

Введение.....	4
1 Анализ предметной области	5
1.1 Обзор аналогов	5
1.1.1 OpenWISP.....	5
1.1.2 NetXMS	6
1.1.3 OpenNMS	6
1.2 Постановка задачи.....	7
1.2.1 SNMP протокол.....	8
1.3 Выбор инструментов	10
1.3.1 Выбор языка программирования для сервера.....	10
1.3.2 Выбор языка программирования для агента	11
1.3.3 Выбор базы данных	12
1.4 Общая структура разрабатываемой системы	12
1.5 Выводы по главе.....	14
2 Проектирование системы управления терминалами.....	15
2.1 Разработка таблицы MIB.....	15
2.2 Разработка SNMP-агента.....	19
2.3 Разработка NMS	22
2.3.1 Диаграмма прецедентов	22
2.3.2 Разработка диаграмм последовательности.....	23
2.3.3 Модель базы данных.....	26
2.4 Макеты интерфейсов	28
2.4.1 Страница «Контроль исполнения расписания»	28
2.4.2 Страница «Состояние устройства»	29
2.4.3 Страница «Сценарии и расписание».....	30
2.4.4 Страница «Добавление станции»	31
2.4.5 Страница «Таблица станций и их параметров»	32
2.5 Вывод по главе	33

3 Разработка и тестирование системы	34
3.1 Инструкции к программам	34
3.1.1 Инструкция по сборке и запуску агента	34
3.1.2 Инструкция по сборке и запуску сервера	35
3.2 Инструменты разработчика	36
3.2.1 Разработка SNMP-агента	36
3.2.2 Разработка NMS	36
3.3 Тестирование	37
3.3.1 Тестирование SNMP-агента	37
3.3.2 Тестирование NMS	39
3.4 Выводы по главе	41
Заключение	42
Список сокращений	43
Список использованных источников	44
ПРИЛОЖЕНИЕ А Исходный код Dockerfile	45

ВВЕДЕНИЕ

В наше время, с развитием Интернета количество пользователей, которые пользуются этой технологией, постоянно растет. Вместе с этим возрастают и требования к стабильности и надежности соединения устройств в сети.

Существует несколько видов подключений к интернету: Ethernet, модемное подключение, спутниковое соединение и другие. Одной из особенностей спутникового подключения являются временные задержки в передаче данных от одного устройства к другому. Это связано с тем, что сигнал должен пройти долгий путь до спутника и обратно, что занимает определенное время.

Для обеспечения стабильности работы такой системы необходимо внимательно контролировать каждое устройство, подключенное к сети, учитывая особенности передачи данных через спутниковое соединение. Также необходимо учитывать задержки в передаче информации и принимать меры для минимизации возможных проблем, связанных с этим. Детальный мониторинг сети и оптимизация процесса передачи данных помогут обеспечить стабильную работу системы при использовании спутникового интернета.

Совместно с компанией ООО «ПК Дельта» ведётся разработка системы управления и мониторинга абонентскими терминалами в сети спутниковой связи. Управление устройствами, а также просмотр их состояния, осуществляется через веб-интерфейс.

1 Анализ предметной области

1.1 Обзор аналогов

Система мониторинга и управления систем, она же Network Management System (NMS) существует уже давно. Рассмотрим проекты с открытым исходным кодом (open-source проекты).

1.1.1 OpenWISP

OpenWISP [1] – это система управления сетью с открытым исходным кодом, предназначенная для недорогих сетей: устройства IoT (Internet of Things, интернет-вещей) и коммутаторы (Рисунок 1.1). Используемые языки программирования: Python.

Преимущества: эргономичный дизайн, поддержка обновления прошивки устройств, разделение на пользователей на роли

Недостатки: поддерживаются устройства, на которых установлена операционная система OpenWrt, что не подходит к модемам, которыми планируется управлять



Рисунок 1.1 – Панель управления в OpenWISP

1.1.2 NetXMS

NetXMS [2] – это open-source система мониторинга и управления сетями и инфраструктурой (Рисунок 1.2). Используемые языки программирования: Java, C++ и C.

Преимущества: кроссплатформенность, гибкий интерфейс

Недостатки: сложность настройки, неподдерживаемая документация, ограниченная поддержка относительно в сравнении с другими NMS.

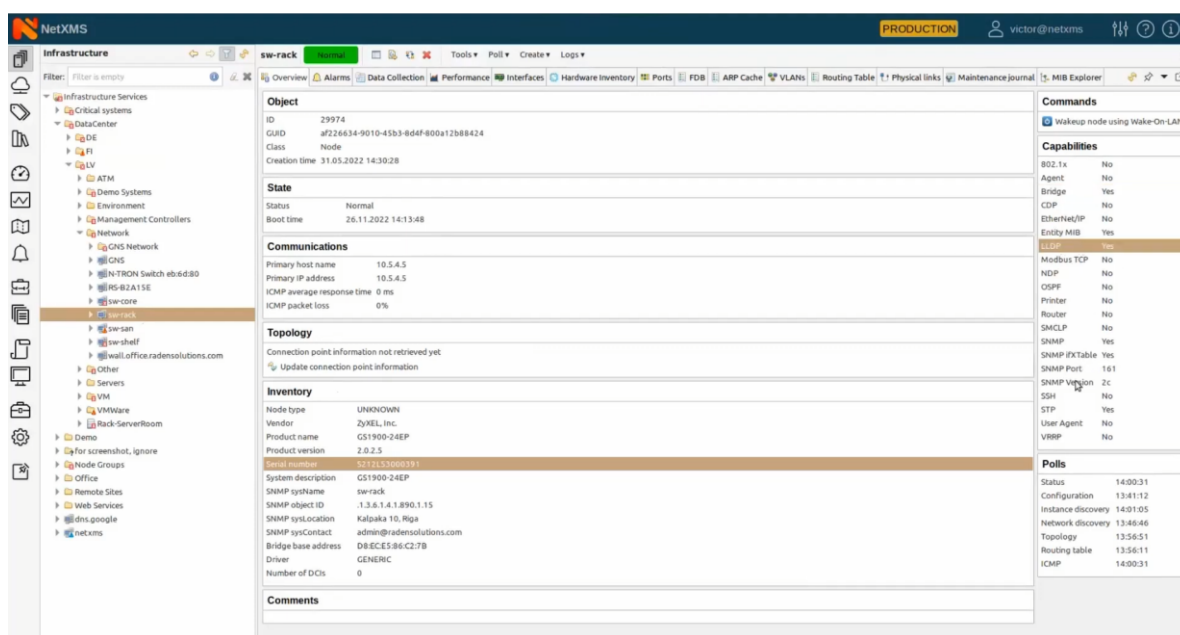


Рисунок 1.2 – Панель управления в NetXMS

1.1.3 OpenNMS

OpenNMS [3] – это бесплатная платформа с открытым исходным кодом для мониторинга и управления сетью корпоративного уровня (Рисунок 1.3). Он разработан и поддерживается сообществом пользователей и разработчиков, а также группой OpenNMS, предлагая коммерческие услуги, обучение и поддержку. Используемые языки программирования: Java.

Достоинства: гибкость настройки, автоматизация задач и активное сообщество.

Недостатки: сложность настройки, высокое потребление вычислительных ресурсов сервера, сложность обновлений.

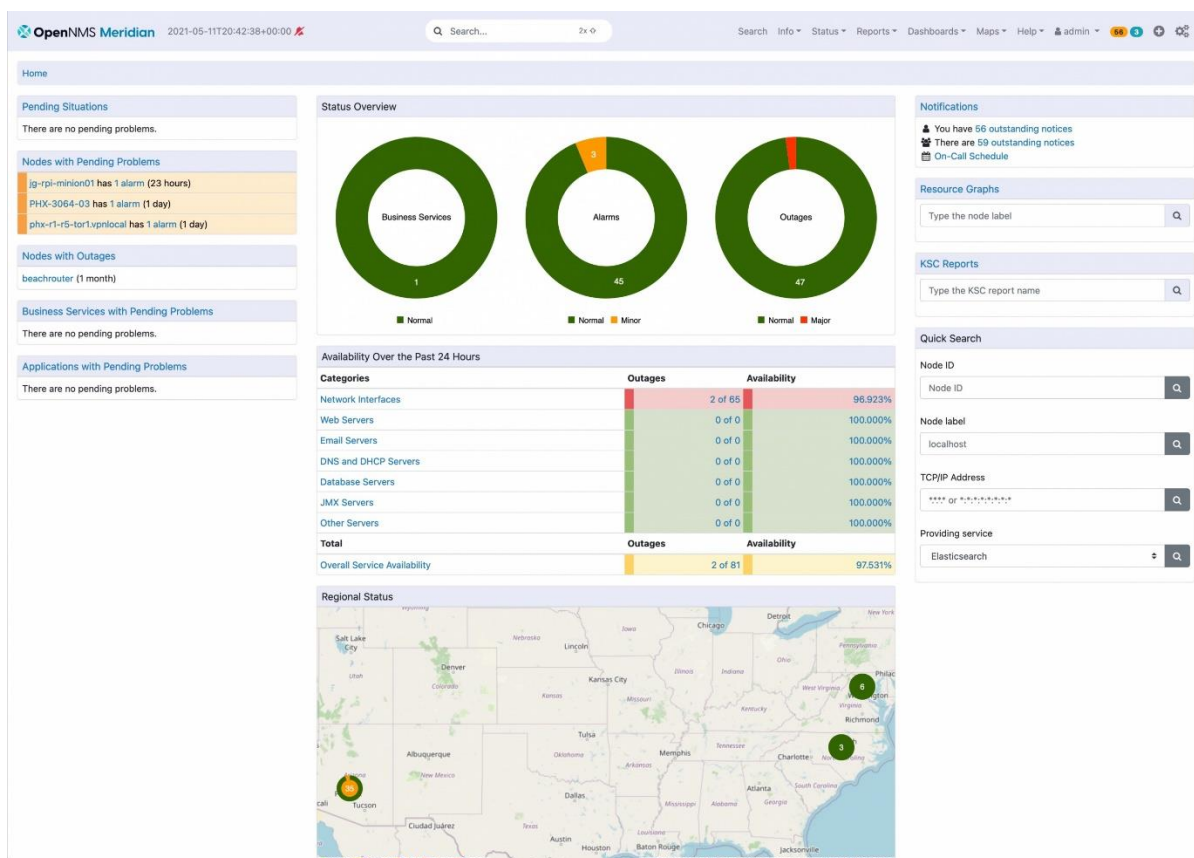


Рисунок 1.3 – Панель управления в OpenNMS

1.2 Постановка задачи

Исходя из исследуемых NMS были выявлены следующие функциональные требования для разрабатываемой системы:

- управление конфигурациями центральной станции (ЦС) и абонентским терминалом (АТ), подключенных к сети;
- в случаях ошибок в работе системы, отображение на экране оператора текущего состояния АТ;
- разработка сайта, в котором будет выполняться отображение и управление устройствами через графическое представление;

– создание базы данных, в которой будут храниться подключенные устройства: параметры и конфигурация.

1.2.1 SNMP протокол

Для взаимодействия сервера между ЦС и АТ был выбран интернет-протокол для управления устройствами Simple Network Management Protocol (SNMP). Данный протокол используется в системах сетевого управления для контроля подключённых к сети устройств на предмет условий, которые требуют внимания администратора. Он описан в Request for Comments (RFC) 1157, который является документом Internet Engineering Task Force (IETF), устанавливающим стандарты для SNMP.

SNMP состоит из следующих компонентов:

- сетевая станция управления, включающая в себя сетевого менеджера;
- агенты – программа на устройстве, которая принимает и отправляет пакеты по SNMP протоколу;
- устройство, у которого опрашиваются и управляются компоненты.

Взаимосвязь компонентов представлена на рисунке 1.4.

Management Information Base (MIB) – база управляющей информации – это иерархическая база данных со сведениями об устройстве. У каждого устройства своя MIB-таблица: например, у принтера в ней содержится информация о состоянии картриджей, а у коммутатора – данные о трафике. Благодаря MIB менеджер знает, какую информацию он может запросить у агента устройства.

Также следует рассказать о версиях протокола.

1. SNMP v1: первая версия протокола SNMP, которая была определена в RFC 1157. SNMPv1 предоставляет базовые функции управления сетью, такие как чтение и запись значений переменных управления. Однако у SNMPv1 есть ограничения в области безопасности, так как он передает сообщения в открытом виде.

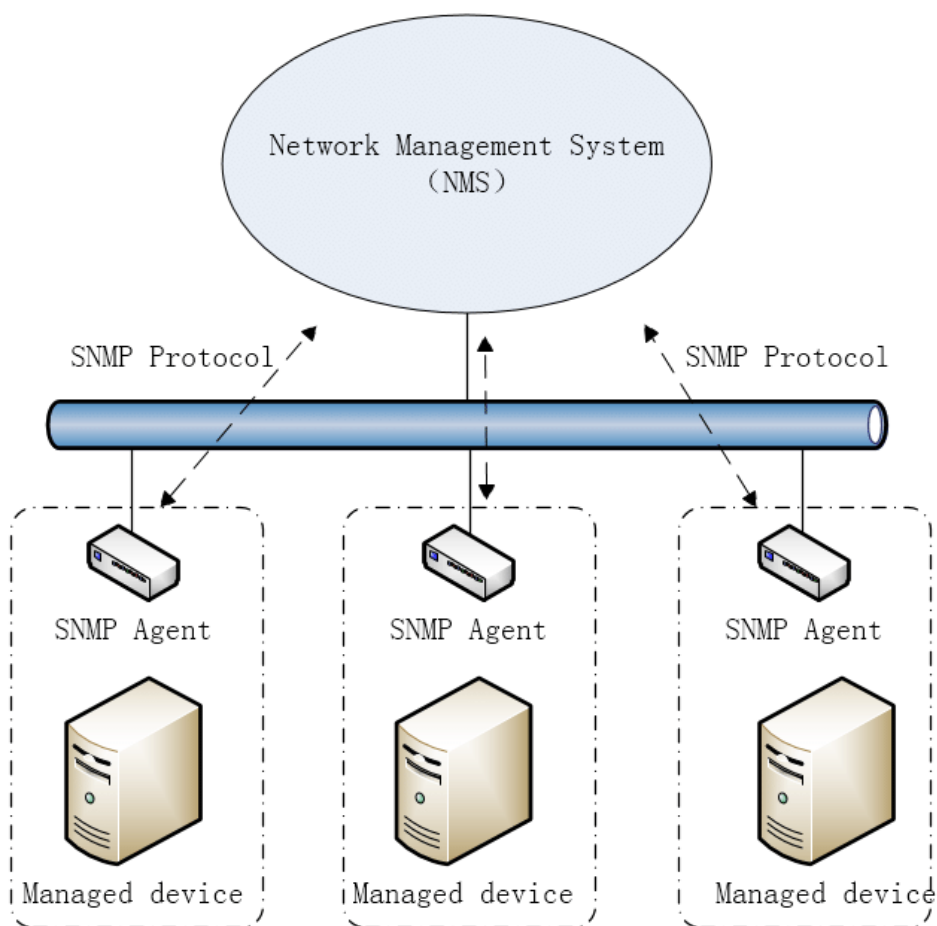


Рисунок 1.4 – Схема SNMP

2. SNMPv2c: вторая версия протокола SNMP, называемая «SNMPv2c» (где «с» означает «community»), была разработана для улучшения некоторых аспектов SNMPv1. Она добавляет новые типы сообщений и улучшает некоторые функции, но все еще имеет недостатки в области безопасности. Хотя она и популярнее, чем третья версия, её проблемы с безопасностью являются ключевой проблемой данной версии.

3. SNMPv3: на момент написания работы, последняя и безопасная версия протокола SNMP. Данная версия была разработана для решения проблем безопасности, с которыми сталкивались предыдущие версии. Она предлагает возможности аутентификации, шифрования и контроля доступа, что делает ее более безопасной для использования в сетевом окружении.

Кроме того, существуют дополнительные расширения и улучшения протокола SNMP, такие как SNMPv2u (User-based Security Model) и SNMPv2t

(Transport Mappings), которые добавляют дополнительные функции и возможности к протоколу SNMP.

Значительным недостатком данного протокола является его проблемы с безопасностью. Данный протокол имеет множество способов [4], способных злоумышленнику заполучить данные об устройстве.

1.3 Выбор инструментов

1.3.1 Выбор языка программирования для сервера

В качестве языка программирования для реализации сервера был выбран Python по нескольким причинам:

Во-первых, наличие обширного набора библиотек в данном языке программирования значительно ускоряет процесс разработки системы по сравнению с большинством других популярных языков программирования. Это обусловлено не только широким набором библиотек, но и наличием эффективных инструментов для отладки кода.

Во-вторых, поддержка многопоточности, позволяющая осуществлять опрос устройств, а также выполнение задач по времени параллельно действиям, выполняемым в веб-интерфейсе.

Веб-интерфейс будет разрабатываться на основе Flask [5] – это фреймворк для создания веб-приложений на языке программирования Python. Использует набор инструментов Werkzeug, а также шаблонизатор Jinja2.

Альтернативой Flask является Django, однако выбор был сделан в сторону первого. Flask, в отличие от Django, относится к микрофреймворку – минимальное количество зависимостей и предоставление лишь базовых возможностей. Это является ключевой причиной выбора так, как большая часть возможностей Django использоваться не будет, в частности административная панель.

1.3.2 Выбор языка программирования для агента

Клиентская часть, связанная с передачей информации об устройстве менеджеру, будет написана на языке C++ так как данный язык программирования:

1. Является высокопроизводительным языком программирования, что особенно важно для разработки программного обеспечения, которое должно обрабатывать большие объемы данных и оперировать сетевыми пакетами. Эффективное использование ресурсов и оптимизация производительности могут быть критически важными для SNMP агента.

2. Позволяет более тесно работать с аппаратным обеспечением и операционной системой, что полезно при разработке сетевого программного обеспечения, такого как SNMP агент. Благодаря возможности работы с указателями и управлению памятью, можно более гибко управлять ресурсами.

Для ускорения процесса разработки SNMP агента, а также более удобной его поддержки, была выбрана библиотека Net-SNMP [6]. Преимуществами данной библиотеки являются:

- поддержка и активное сообщество: Net-SNMP является широко используемым и поддерживаемым инструментом с активным сообществом разработчиков;
- простота использования: Net-SNMP обладает простым и понятным интерфейсом;
- большой набор функций: Net-SNMP предлагает широкий набор функций и возможностей для разработки агентов SNMP, что позволяет легко настраивать и мониторить сетевые устройства.

Также Net-SNMP предоставляет удобные средства для работы с протоколом SNMP, включая автоматическую генерацию MIB, программы для взаимодействия по SNMP протоколу такие как *snmpwalk* и *snmpset*, и другие функциональности.

1.3.3 Выбор базы данных

В качестве базы данных для параметров устройства, а также сбора статистики об устройстве, будет использоваться документно-ориентированная система управления базами данных MongoDB [7]. Выбор был сделан по следующим причинам:

1. Гибкость и масштабируемость, достигаемая за счет самой структуры хранения данных: хранятся они в формате документов, которые схожи с JSON. Это полезно для NMS систем, так как данные о сети могут быть разнообразными и изменяться со временем.

2. Высокая производительность за счет использования индексов кэширования и параллельной обработки запросов. Это особенно важно в подобных системах, где требуется быстрый доступ к данным о сети для анализа, мониторинга и управления устройствами.

3. Мощный язык запросов в MongoDB позволяет выполнять разнообразные операции с данными, включая сортировку, фильтрацию и другие операции. Это упрощает разработку сложных запросов для анализа и отображения данных в NMS.

1.4 Общая структура разрабатываемой системы

Структура системы в контексте одной сети представлена на рисунке 1.5. Описание компонентов рисунка:

1. Абонентский терминал (АТ) – представляет собой устройство, которое необходимо регулировать. В частности, настройка центральной частоты и символической скорости на модуляторе и демодуляторе, включение передачи, а также сбор полезной информации такой как уровень шума, частоты смещения и другие.

2. Центральная станция (ЦС) – многоканальное устройство, состоящее из модулятора и нескольких демодуляторов. В ходе работы используется ЦС с 8-ю

демодуляторами, однако их количество в будущем может возрасти до 32. Этот факт влияет на разработку интерфейса оператора, в котором удобно следить за всеми показателями с минимальным количеством переходов страниц.

3. Сервер – устройство, на котором запущена NMS и база данных.

4. Компьютер оператора – устройство, с помощью которого оператор управляет сетью, подключенное к серверу.

5. Спутниковая тарелка – антенна, используемая для передачи и приёма сигнала между земной станцией спутниковой связи и искусственным спутником Земли.

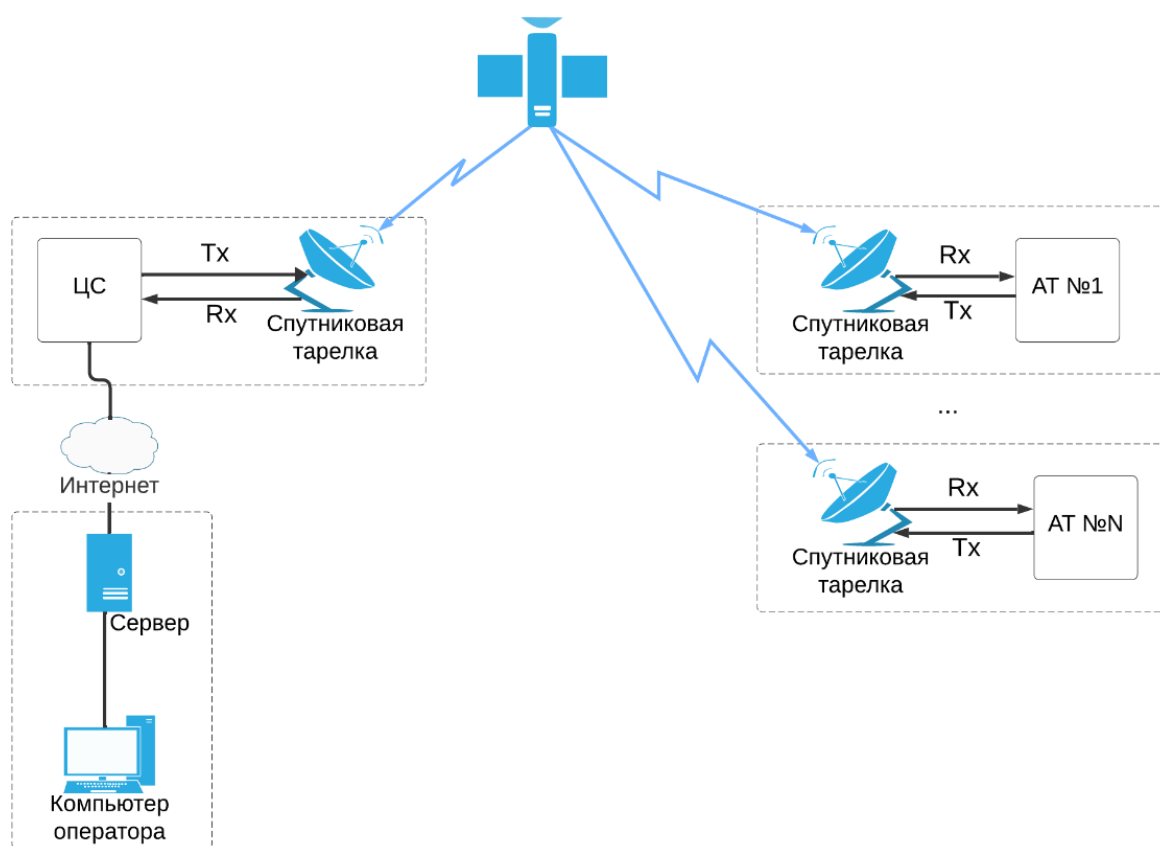


Рисунок 1.5 – Архитектура сети

Взаимодействие между пользователем и сервером основано на клиент-серверной архитектуре, которая является моделью разработки программного обеспечения. В этой модели приложение разделяется на две части: клиентскую и серверную.

1. Клиент представляет собой программу, используемую на стороне пользователя, например, браузер.

2. Сервер, в свою очередь, является программой, выполняющейся на стороне сервера и обрабатывающей данные, полученные от клиента.

В данном случае это веб-интерфейс – отдельный модуль, отвечающий за формирование и отображение данных в удобном для пользователя виде.

1.5 Выводы по главе

В результате анализа поставленной задачи, были сформулированы четкие требования к разрабатываемой системе, а также были выявлены преимущества и недостатки существующих решений. Также был сделан выбор инструментов, с помощью которых и будут выполнены цели и задачи системы.

2 Проектирование системы управления терминалами

2.1 Разработка таблицы MIB

Согласно пункту 1.2.1, в MIB хранятся объекты управления, в которой описан путь к ним и значение, которое опрашиваемый менеджер получит. Данные объекты можно представить в виде дерева, которые строятся с использованием уникальных идентификаторов объектов, называемые как Object Identifiers (OIDs). OID представляет собой уникальный путь к конкретному объекту в дереве MIB.

В стандарте SNMP, дерево MIB начинается с корня, который имеет OID 1.3.6.1. Последующие узлы в дереве MIB определяются путем добавления дополнительных числовых идентификаторов к корню. Следом строится путь до уникального номера компании: это обычно соответствует поддереву MIB, зарезервированному для использования конкретной компанией или организацией.

Номер компании обычно начинается с 1.3.6.1.4.1, где:

- «1» – корень;
- «3» – «org»;
- «6» – «dod»;
- «1» – «internet»;
- «4» – «private» (частные);
- «1» – «enterprises» (предприятия).

Следующей цифрой добавляется номер компании. Каждая компания, использующая SNMP для управления своим оборудованием, должна получить уникальный номер компании, чтобы избежать конфликтов OID. В случае компании, для которой проектируется MIB, это «61503».

Как видно из рассмотренного примера выше, деревья в MIB строятся иерархически от корня к конкретным объектам, а номера компаний располагаются в отдельной ветви поддерева MIB для частных организаций.

Также хорошей практикой является создание универсальности к разрабатываемым МІВ, поскольку они редко обновляются использующими их. Потому для начала необходимо разработать структуру, в которой можно свободно добавлять элементы, при этом чтобы при работе со старыми версиями МІВ, обработка параметров была корректной и не было коллизий. В такой системе предварительно создают верхний уровень, в котором все продукты компании разбиваются на характерные отличия.

На рисунке 2.1 представлен верхний уровень компании.

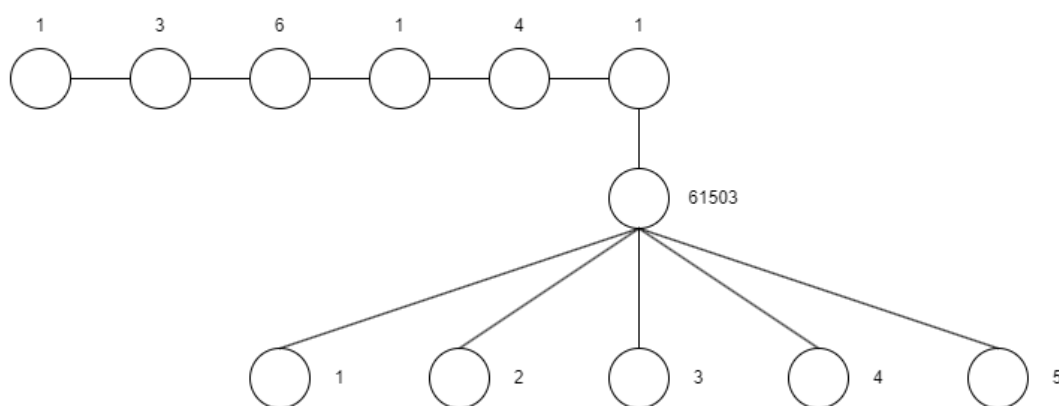


Рисунок 2.1 – Верхний уровень разрабатываемой МІВ

Как видно на рисунке 2.1, от корневого дерева компании идут пять объектов. Эти объекты разбиты на следующую логику, где номер от корня соответствует названию объекта:

1. Modules. В нем описываются все последующие разрабатываемые МІВ'ы. Данный модуль особенно важен в случаях большого количества файлов, хранящих описание объектов. В нем можно будет кратко узнать информацию о существующих МІВ и какой номер следующим использовать от существующего.

2. Products. В данном модуле расписываются разрабатываемые компанией устройства. В случае рисунка 2.1 от продукта идет один объект и это «sateliteModems» – спутниковые модемы. В будущем, если компания решит разрабатывать другие устройства, требующие мониторинга и управления, но не

являющимися спутниковыми модемами, будет создана новая категория, например, наземные модемы.

3. Reg. В него записывается название устройства при запуске агента (для рисунка 2.2 это значение равно Basis20Reg).

4. Traps. Данный модуль необходим для мониторинга состояния устройства, показания которых не должны выходить за пределы значений. В нем хранятся параметры устройств, которые отслеживаются в реальном времени и в случае, если полученное значение выходит за пределы допустимых значений, отправляется SNMP-пакет указанному адресу об этом значении параметра.

5. Experimental. Данный модуль используется для предварительной отладки работы агента при новом добавляемым параметров в основную ветвь. Одна из причин создания данной ветви – защита от использования некорректных запросов при ошибке ввода устройства.

6. Textual conventions. Данный модуль не отображен в верхнем уровне, однако используется для описания управляемых или отслеживающими параметрами. В нём описывается формат получаемого значения, а также приписки типа значения к нему. Например, уровень шума является дробным числом, но в RFC данный тип не описан. Потому, получаемое значение отправляется агентом в виде целого числа, а следом, по правилам в данном модуле, на стороне менеджера, делится на 100. К полученному числу приписывается система счисления dB (децибел).

На рисунке 2.2 представлено продолжение дерева от модуля «Products». В нем изображено разбиение параметров устройства Basis20.

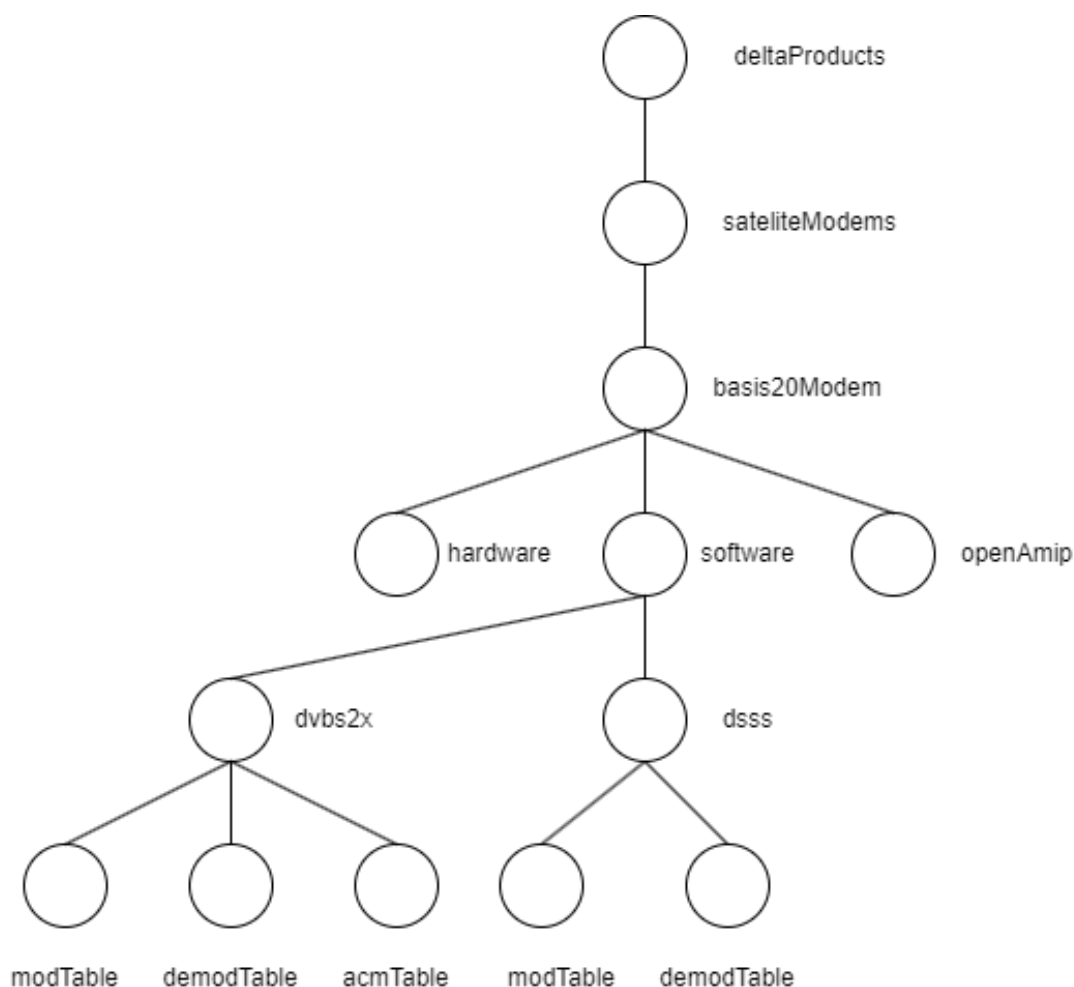


Рисунок 2.2 – Дерево параметров Basis20

Пример опроса устройства с преобразованием данных с использованием разработанных MIB изображен на рисунке 2.3.

Отображение параметров устройства (кроме параметров OpenAMIP) изображены в виде таблицы. Причиной тому является количество самих модуляторов и демодуляторов на устройстве. Как видно на рисунке 2.3, параметры DELTA-BASIS20-DVBS2X идут с припиской «1» – данное число обозначает номер модулятора или демодулятора в зависимости от параметра. В случае ЦС параметры с припиской «demod» будут заканчиваться на число 1-8, где значение обозначает номер демодулятора.

```

DELTA-BASIS20-DVBS2X-MIB::modBucState.1 = INTEGER: off(1)
DELTA-BASIS20-DVBS2X-MIB::modBucVMeas.1 = Gauge32: 0 V
DELTA-BASIS20-DVBS2X-MIB::modBucAmeas.1 = Gauge32: 0 A
DELTA-BASIS20-DVBS2X-MIB::modOutputLevel.1 = INTEGER: -25.00 dB
DELTA-BASIS20-DVBS2X-MIB::modCenterFreq.1 = Gauge32: 1600000.000 KHz
DELTA-BASIS20-DVBS2X-MIB::modSymbolRate.1 = Gauge32: 5000000 sym/s
DELTA-BASIS20-DVBS2X-MIB::modBitrate.1 = Gauge32: 20991 kb/s
DELTA-BASIS20-DVBS2X-MIB::modPackets.1 = Gauge32: 212
DELTA-BASIS20-DVBS2X-MIB::modBytes.1 = Gauge32: 1523008 bytes
DELTA-BASIS20-DVBS2X-MIB::modTxMode.1 = INTEGER: off(1)
DELTA-BASIS20-DVBS2X-MIB::modRef.1 = INTEGER: on(2)
DELTA-BASIS20-DVBS2X-MIB::modInvSpecter.1 = INTEGER: on(2)
DELTA-BASIS20-DVBS2X-MIB::modDvbs2xModcod.1 = INTEGER: a32psk89(27)
DELTA-BASIS20-DVBS2X-MIB::modDvbs2xFramelength.1 = INTEGER: normal(1)
DELTA-BASIS20-DVBS2X-MIB::modDvbs2xRolloff.1 = INTEGER: r25(5)
DELTA-BASIS20-DVBS2X-MIB::demodTableLnbVmeas.1 = INTEGER: 0 V
DELTA-BASIS20-DVBS2X-MIB::demodLnbAmeas.1 = Gauge32: 11 mA
DELTA-BASIS20-DVBS2X-MIB::demodEsno.1 = INTEGER: -90.0 dB
DELTA-BASIS20-DVBS2X-MIB::demodFreqOffset.1 = INTEGER: 0 Hz
DELTA-BASIS20-DVBS2X-MIB::demodCenterFreq.1 = Gauge32: 1600000.000 KHz
DELTA-BASIS20-DVBS2X-MIB::demodSymbolRate.1 = Gauge32: 5000000 sym/s
DELTA-BASIS20-DVBS2X-MIB::demodLnbPresetVoltage.1 = INTEGER: 1
DELTA-BASIS20-DVBS2X-MIB::demodRef.1 = INTEGER: on(2)
DELTA-BASIS20-DVBS2X-MIB::acmState.1 = INTEGER: on(2)
DELTA-BASIS20-DVBS2X-MIB::acmLocalMargin.1 = INTEGER: 5.0
DELTA-BASIS20-DVBS2X-MIB::acmRemoteMargin.1 = INTEGER: 5.0
DELTA-BASIS20-DVBS2X-MIB::acmMinModcod.1 = INTEGER: qpsk12(4)
DELTA-BASIS20-DVBS2X-MIB::acmMaxModcod.1 = INTEGER: a64psk56(31)
DELTA-BASIS20-OPENAMIP-MIB::openAMIPConnAddr.0 = IPAddress: 1.0.0.9
DELTA-BASIS20-OPENAMIP-MIB::openAMIPConnManufactures.0 = STRING: "OpenAMIP"
DELTA-BASIS20-OPENAMIP-MIB::openAMIPConnModel.0 = STRING: "OpenAMIP"
DELTA-BASIS20-OPENAMIP-MIB::openAMIPConnPort.0 = Gauge32: 5005
DELTA-BASIS20-OPENAMIP-MIB::openAMIPConnTimeout.0 = Gauge32: 60 seconds
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingLongitude.0 = INTEGER: 90.1 degrees C
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingHuntBandwidth.0 = INTEGER: .512 MHz
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingHuntFreq.0 = INTEGER: 950.1 MHz
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingRxLclOsc.0 = INTEGER: 10000.0 MHz
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingTxLclOsc.0 = INTEGER: 10000.0 MHz
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingRxPolarization.0 = INTEGER: horizontal(3)
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingTxPolarization.0 = INTEGER: vertical(4)
DELTA-BASIS20-OPENAMIP-MIB::openAMIPIntervalsLatlong.0 = INTEGER: 60 seconds
DELTA-BASIS20-OPENAMIP-MIB::openAMIPIntervalsKeepalive.0 = INTEGER: 10 seconds

```

Рисунок 2.3 – Отображение параметров с устройства Basis20

2.2 Разработка SNMP-агента

В разработке агента будет использовать вторая версия по нескольким причинам:

1. Простота использования: SNMP v2с обычно считается более простым в настройке и использовании по сравнению с SNMP v3. Он имеет менее сложные механизмы безопасности.

2. Совместимость: Некоторые устройства и системы могут поддерживать только SNMP v2c, поэтому использование этой версии может быть обусловлено совместимостью с имеющимися устройствами.

3. Меньшие требования к ресурсам: SNMP v2c обычно требует меньше вычислительных ресурсов (например, процессорного времени) и сетевой пропускной способности, чем SNMP v3, что важно для устройств с ограниченными ресурсами.

Так как в сети отсутствуют посторонние люди было принято решение разрабатывать агента на данной версии. В будущем планируется перейти на третью версию, поскольку безопасность является ключевой особенностью при работе с модемами, от которых зависят многие процессы.

Общая логика работы с библиотекой Net-SNMP запечатлена на рисунке 2.4. Под клиентом подразумевается внутренняя библиотека `grps-client`, которая является прослойкой между устройством и программой, и позволяет получать значения с модема или их устанавливать.

Daemon Net-SNMP процесс является демоном – программой, работающей в фоновом режиме, ожидая событий и предлагая какие-то службы для их выполнения. Перед началом работы с ним, необходимо приложение инициализировать: зарегистрировать программу в демоне, а также подготовить функции для обработки запросов по указанным OID в соответствии с разработанными MIB. После инициализации, происходит процесс постоянного ожидания пакета на зарегистрированные OID программой. Когда приходит соответствующий пакет, программа перенаправляет запрос в нужную функцию, которая обрабатывает весь запрос.

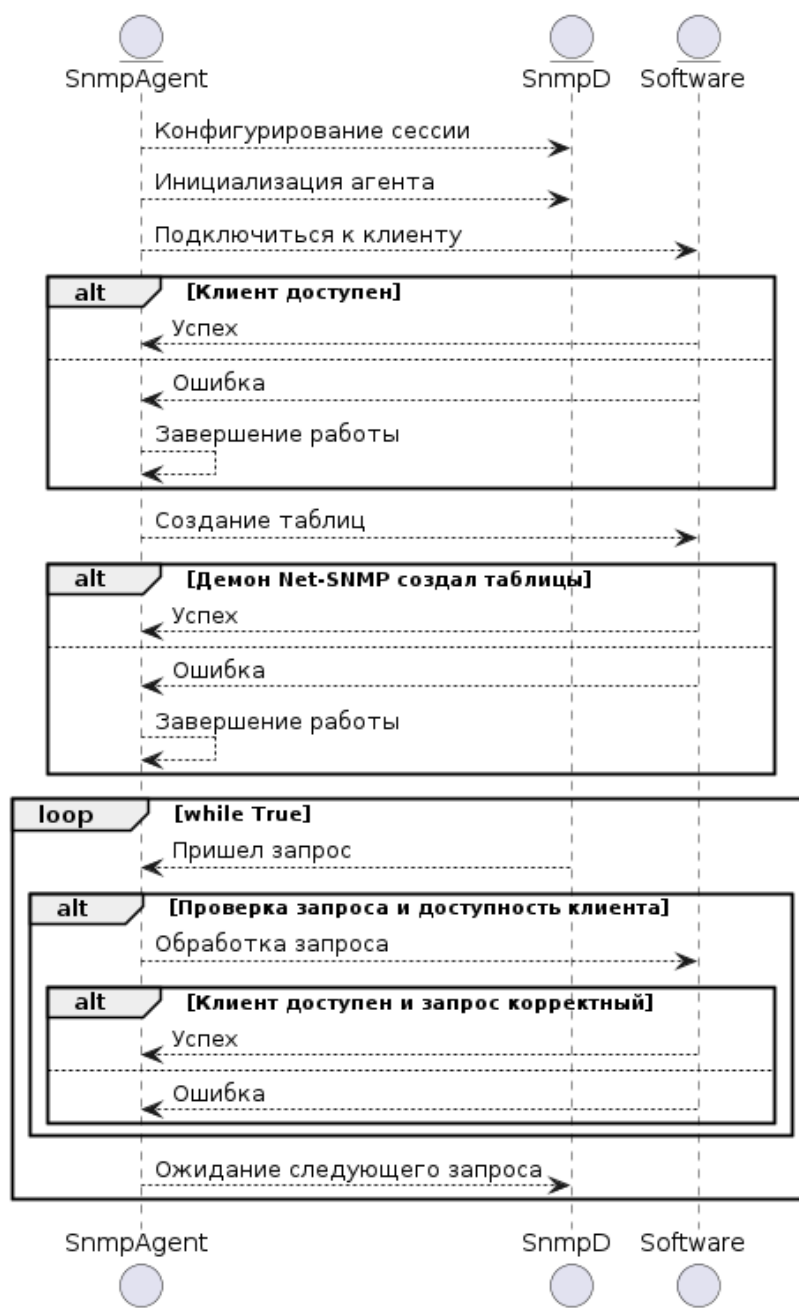


Рисунок 2.4 – Диаграмма последовательности работы SNMP агента

Также в программе предусмотрены случаи, когда модем не отвечает или устанавливаемые тип и значения не соответствует ожидаемым. Таким образом происходит процесс отсеивания пакетов, которые потенциально могут нарушить работу устройства.

2.3 Разработка NMS

2.3.1 Диаграмма прецедентов

В разрабатываемой системе используется две группы пользователей: оператор и администратор. Диаграмма прецедентов представлена на рисунке 2.5.

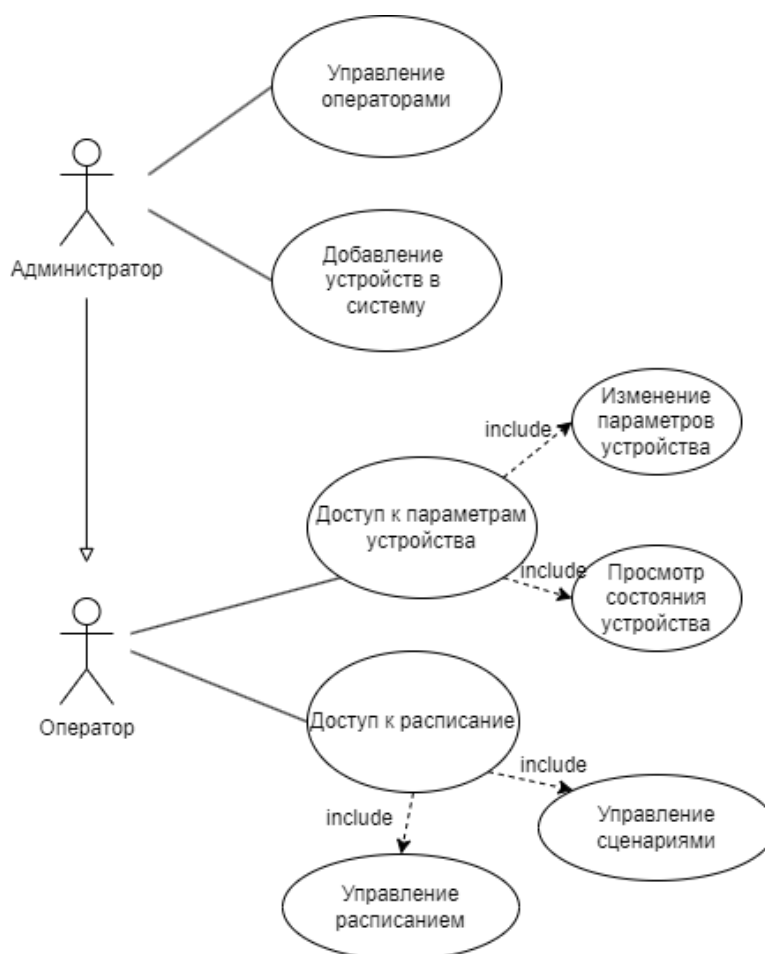


Рисунок 2.5 – Диаграмма прецедентов

В разрабатываемой системе мониторинг и управление устройствами осуществляется оператором. В ней он может посмотреть историю параметров устройства и изменить их значение. Во вкладке расписание оператор может составить частотный план для центральной станции, а также составить расписание выполнения сценариев в системе. Для экстренных ситуаций оператор мо-

жет составить альтернативное расписание для устройств и переключиться на него в необходимый момент. Сценарии – это набор параметров для выбранных станций, которые должны примениться при исполнении. Перед исполнением сценария происходит отправка всем незадействованным АТ команды на отключение передачи, для предотвращения коллизии между сценариями.

Администратор создает операторов, указывая им подсеть, в которой они работают, а также администратор добавляет новые устройства в систему. К каждой добавляемой в систему устройству добавляется подсеть для обеспечения разграничения доступа к устройствам между операторами.

2.3.2 Разработка диаграмм последовательности

На рисунке 2.6 представлена диаграмма последовательности «Добавить станцию». Администратор в меню выбирает пункт создания станции и заполняет форму. Перед тем, как отправить данные, сайт проверяет данные на корректность ввода. После успешного создания станции, администратор будет переадресован на страницу с отображением параметров абонентским терминалов.

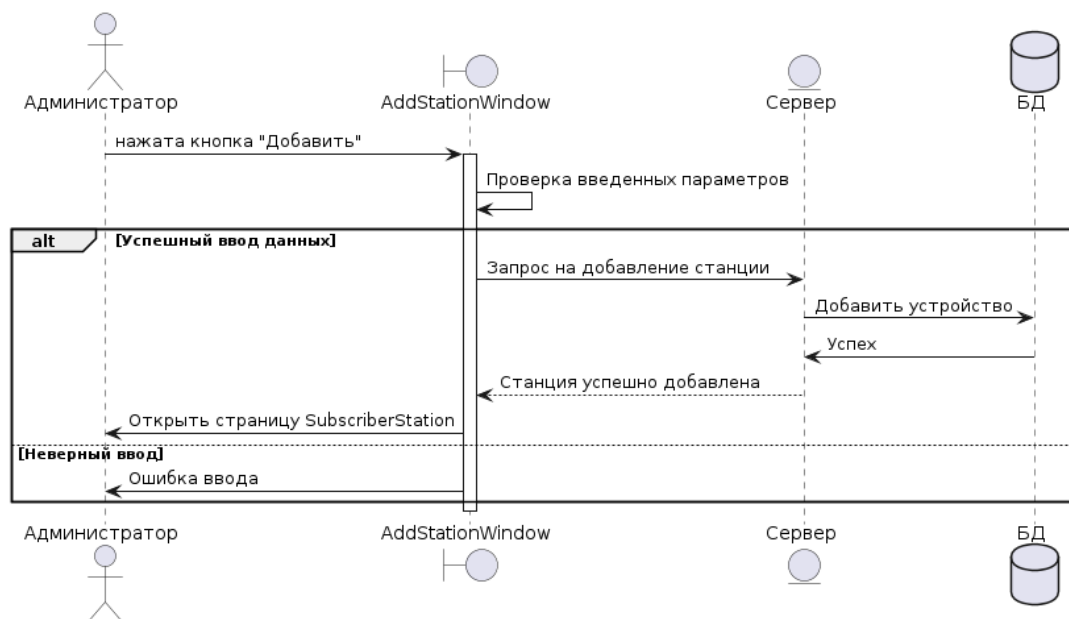


Рисунок 2.6 – Диаграмма последовательности «Добавить станцию»

На рисунке 2.7 представлена диаграмма последовательности «Добавление оператора». На нём администратор заполняет поля логин и пароль, а также выбирает из выпадающего списка сеть, которой он будет управлять.

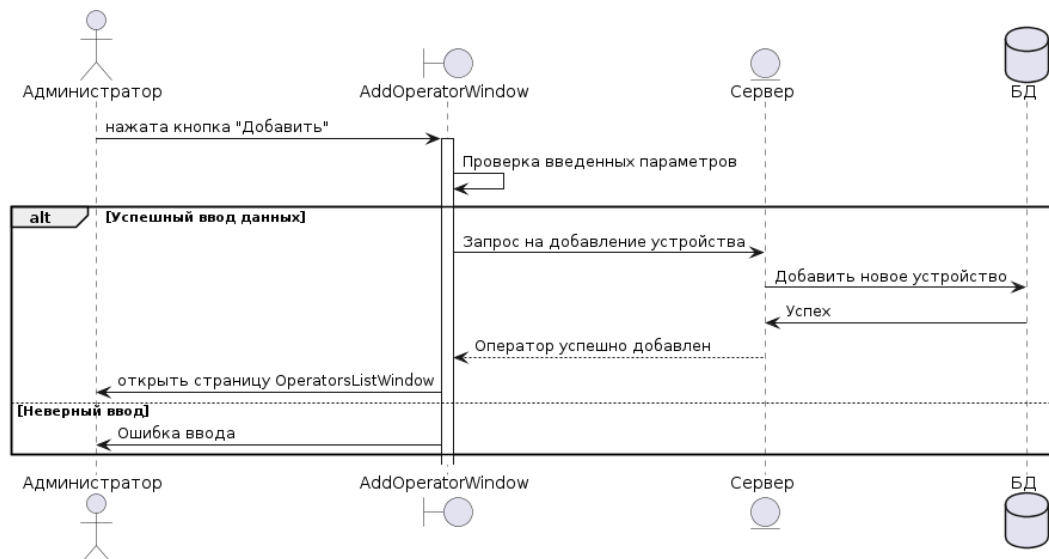


Рисунок 2.7 – Диаграмма последовательности «Добавление оператора»

Диаграмма последовательности «Запрос данных о станциях» представлен на рисунке 2.8.

В окне «Абонентские терминалы» оператору выводится список всех доступных ему станций. В нем он может запросить вывод конкретного параметра для всех параметром, например, уровень шума на модуляторе. Происходит опрос сервера на получение данных. В приоритете получить данные из базы данных, чтобы лишней раз не нагружать канал связи. Если данных нет или они устаревшие (параметр, настраиваемый в конфигурационном файле), то происходит опрос устройства.

Устройство может быть недоступно по нескольким причинам: физически, на устройстве отключен модулятор или другие причины. В таком случае сервер отправляет сигнал оператору, что устройство не отвечает.

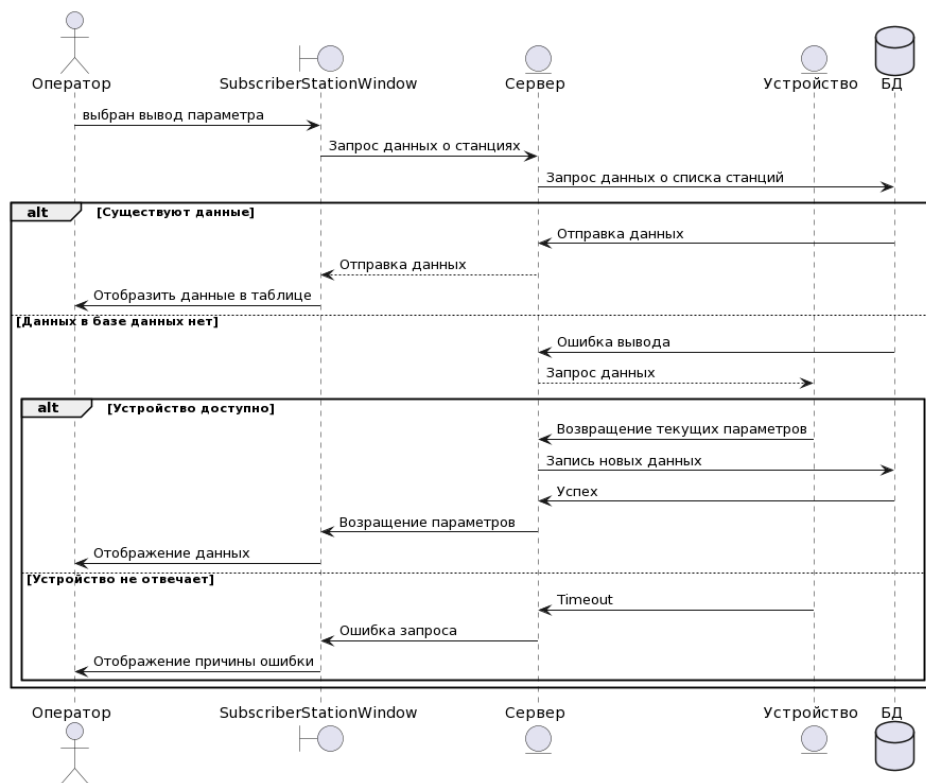


Рисунок 2.8 – Диаграмма последовательности «Запрос данных о станциях»

На рисунке 2.9 представлена диаграмма последовательности «Изменение параметров устройства».

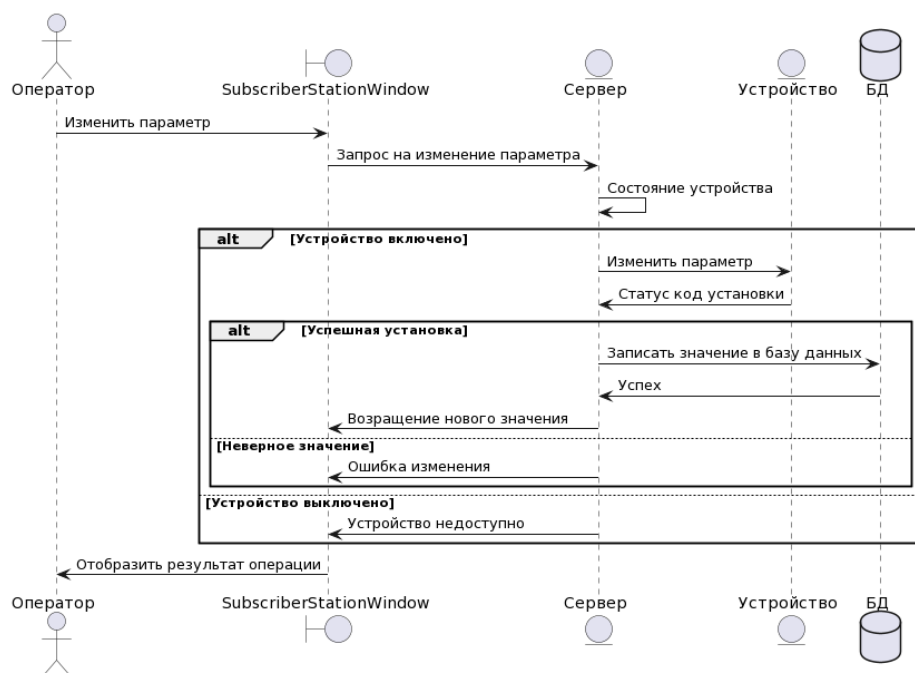


Рисунок 2.9 – Диаграмма последовательности «Изменение параметров устройства»

При изменении параметров устройства проверяется состояние устройства: параметр состояния модулятора записано в БД. Если параметр отображается как «Включен», то происходит процесс проверки доступности устройства путём использования программы *ping*. При успешной проверке, происходит процесс изменения параметров. Причина проверки – неопределенное состояние устройства в случае, если устройство недоступно.

Например, если отправить запрос на изменение коэффициента скругления во время отключенного модулятора, то параметр может и установится, но вернется все равно как ошибка установки по причине отключенного режима передачи модулятора.

2.3.3 Модель базы данных

Как написано в первой главе, в качестве базы данных была выбрана MongoDB. На рисунке 2.10 представлена модель базы данных.

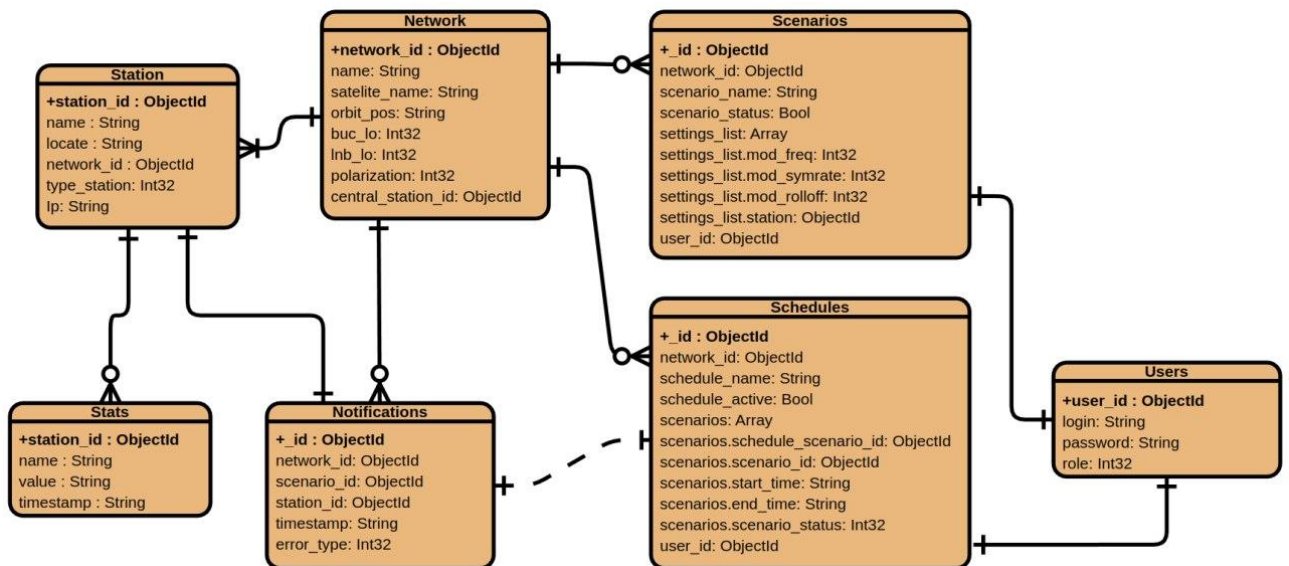


Рисунок 2.10 – Модель базы данных

Согласно приведенной диаграмме, реализовано семь коллекций базы данных:

Таблица «Сеть» является ключевой таблицей в сети. Каждая такая таблица обозначает новую сеть, и все остальные элементы работают между собой только внутри одной сети. В данную таблицу записывается название сети, а также название спутника, его орбитальная позиция, значения ВUC и LNB – устройств-конвертеров, находящихся в составе антенной системы, а также поляризация, в которой работает данная спутниковая система. Данные никак не влияют на работу системы, являясь подсказкой оператору при составлении частотного плана.

Таблица «Станции». В ней перечислены все станции системы, где тип указывает АТ или ЦС и обозначается числом 0 или 1 соответственно. Также в эту таблицу записывается её название и её расположение – на данный момент это строковое представление для удобства работы с ним. В будущем название будет устанавливаться на устройство, а расположение будет представлять собой координаты для отображения устройства на карте.

Таблица «Расписание». Для каждой сети существует своё расписание. Строка таблицы состоит из идентификатора сценария, время его выполнения, идентификатором оператора, добавившего сценарий в расписание, и статуса – результат работы сценария. Сценарии из расписания не удаляются, а помечаются в статусе особым номером, обозначающим удаление сценария из расписания.

Таблица «Сценарий». Данная таблица содержит в себе: идентификатор, который используется таблицей «Расписание», а также из настроек, представляющих собой массив до восьми элементов, в котором указываются: станция, частота, символьная скорость, а также коэффициент скругления, если абонентский терминал использует стандарт DVBS2X.

Таблица «Статистика», представляющий собой большую таблицу из данных об устройствах системы. Данная таблица должна заполняться всякий раз, когда произошёл опрос параметров устройства, обеспечивая актуальность данных, а также минимизация использования канала связи.

Таблица «Пользователи». Данная таблица содержит все данные об пользователях NMS, а также идентификатор пользователя используется в формировании таблицы и сценария для использования логирования действий операторов при работе с системой.

Таблица «Уведомления». Данная таблица содержит в себе записи, заполняемые при работе со станциями. Уведомления отображаются оператору на главной странице сети и сообщают ему о корректной или некорректной работе системы с указанием ошибки.

2.4 Макеты интерфейсов

2.4.1 Страница «Контроль исполнения расписания»

На рисунке 2.11 представлен макет окна «Контроль исполнения расписания». В нём оператор может следить за тем, как исполняются сценарии (какие частоты установлены и для каких станций, а также задействованные демодуляторы). В этом окне оператор также может перейти в настройки демодулятора для просмотра параметров на устройстве, а также посмотреть другие параметры (Рисунок 2.12-2.13) выбрав нужный ему демодулятор – обозначение дем.1, дем.2, ..., дем.8 на рисунке 2.11.



Рисунок 2.11 – Окно мониторинга состояния сети

Также слева от графика отображены уведомления об статусе выполнения расписания. Зеленым цветом обозначаются успешно выполненные сценарии, красным – ошибка выполнения. Обычно, такие уведомления сообщают об ошибке ввода частот оператором или отсутствие ответа от одного из модемов.

В правом блоке находится таблица расписания. Данная таблица состоит из сценариев, которые планируются или уже выполнены, статуса их выполнения и времени.

2.4.2 Страница «Состояние устройства»

Окно просмотра параметров абонентского терминала, а также их установка, состоит из таблицы параметров, отсортированных по категориям. Изменять их может как оператор, так и администратор. Их установка предварительно проверяется на стороне веб-сайта. В случае, если оператору все же удастся установить некорректное значение или устройство недоступно, новое значение параметра не будет установлено и будет выведено окно с ошибкой установки.

На рисунке 2.12 представлена вкладка состояние. В ней изображены неизменяемые параметры устройства.

Терминал basis-20-000

Состояние					
Состояние	Основные	Модулятор	Демодулятор	Интерфейсы	OpenAMIP
Тип CCC					
dvbs2x					
Состояние модулятора	Скорость передачи	Напряжение ВУС	Ток ВУС		
normal	25329000 Мбит/с	0 В	0 А		
Модуляция	Тип кадра				
invalid	invalid				
Состояние демодулятора	Уровень шума (Esno)	Част. расстр.	Напряжение МШУ	Ток МШУ	
up	-99 дБ	0	1.4 В	0.064 А	
Сост. SAT0	Сост. GE0	Сост. GE1			
up	up	down			

Рисунок 2.12 – Вкладка «Состояние» окна «Состояние устройства»

На рисунке 2.13 изображены параметры управления модулятором. В данной вкладке изображены последние записанные параметры устройства. Пользователь может поменять значения параметров, но как сказано выше, данные параметры принимают значения в пределах.

Управление станцией

Состояние | Основные | **Модулятор** | Демодулятор | OpenAMIP

Центральная частота: 1600037700 Гц

Символьная скорость: 5056100 сим/с

Режим передачи: Норм.

Опорная частота: Отключена

Инверсия спектра: Включена

Питание ВУС: Выключен

Выходной сигнал: -30.5 дБ

Модуляция: APSK32 4/5

Длина пакета: Нормальный

Коэффициент скругления: 0.05

Сохранить

Рисунок 2.13 – Вкладка «Модулятор» окна «Состояние устройства»

2.4.3 Страница «Сценарии и расписание»

На рисунке 2.14 представлен макет «Настройка расписания сети», в котором оператор и администратор могут добавлять, удалять или редактировать сценарии (Рисунок 2.15), а также составлять расписание для сети на сутки. В расписании задействованы сценарии, составляющие в левой части страницы.

Расписание сети Ангара-С

Сценарии +

#	Название	
1	Сценарий 1	
2	Сценарий 2	
3	Сценарий 3	
4	Тест 1	
5	Тест 2	
6	Тест 3	

Расписание +

#	Название сценария	Начало	Конец	
1	Сценарий 3	17:00	17:01	
2	Тест 1	17:15	17:16	
3	Тест 2	17:23	17:24	
4	Тест 3	17:27	17:28	

Рисунок 2.14 – Интерфейс «Настройка расписания сети»

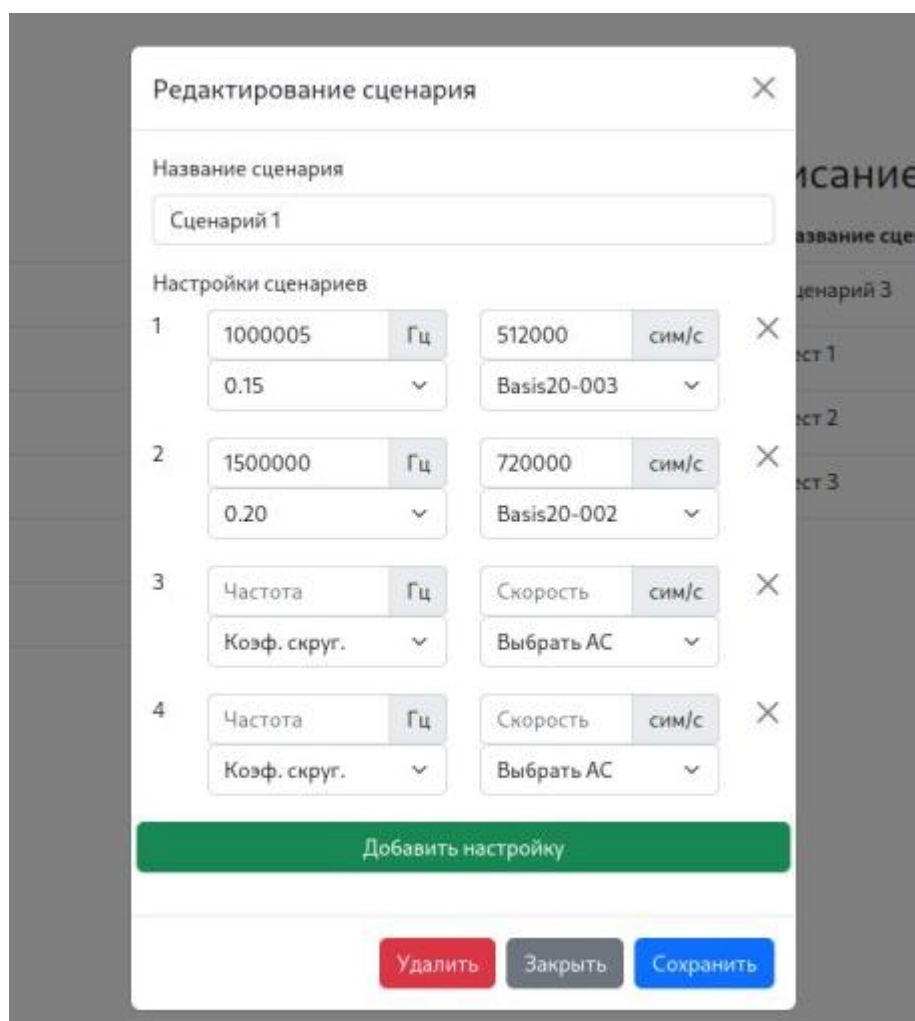


Рисунок 2.15 – Создание сценария для сети

Количество настроек сценариев ограничено количеством демодуляторов ЦС. Также, создаваемые сценарии имеют привязку к сети. Таким образом, чтобы оператору настроить АТ другой сети на такие же параметры, ему необходимо вручную повторить данную операцию по причине привязки параметров к выбранному устройству.

2.4.4 Страница «Добавление станции»

На рисунке 2.16 изображена форма добавления станций в сеть. Под периодичностью полного опроса является опрос всех доступных компонентов сети, под частичным опросом – опрос уровня шума и частоты смещения. На данный момент эти два параметра нужны чаще для анализа сети. В будущем планиру-

ется установка времени для каждого параметра поля частоты опроса. В случае, если устройство недоступно, то опрос игнорируется.

Тип станции
Абонентская станция

IP-адрес

Название станции

Тип CCC
DVB-S2X

Местоположение

Заказчик

Сеть

Подсеть

Периодичность полного опроса
120 с

Периодичность частичного опроса
60 с

Добавить

Рисунок 2.16 – Форма добавления АТ в сеть

2.4.5 Страница «Таблица станций и их параметров»

На рисунке 2.17 представлено окно просмотра состояний абонентских терминалов. В нем представлены станции, которые работают внутри просматриваемой сети. Обновление состояний происходит с периодичностью 15 секунд.

В случае, если устройство доступно, то новый параметр будет отображен, а последняя активность будет обновлена. В случае, если устройство недоступно, то об этом будет сообщено оператору в соответствующей ячейке.

☰

Список абонентских станций

Статус	Название	Место положения	Последняя активность	Параметр №1	Параметр №2	Параметр №3
Все	Станция	Место положения	Дата	Скорость соединения	Центральная частота (демодулятор)	Есно
●	Basis20-001	Красноярск	2024-06-07 15:10:26	23338500	1500027300	-99
●	Basis20-002	Зеленоград	2024-06-07 15:10:26	4207000	1499956100	-91
●	Basis20-003	Назарово	2024-06-07 15:10:26	24824000	1500018600	-95
●	Basis20-004	Хабаровск	2024-06-07 15:10:26	37712000	1500024800	-86
●	Basis20-005	Красноярск	2024-06-07 17:28:01	37226500	1500000000	-93

Рисунок 2.17 – Окно «Список абонентских терминалов»

2.5 Вывод по главе

В данной главе была рассмотрена общая схема разрабатываемой системы, описана MIB-таблица, разработка SNMP-агента, диаграммы прецедентов, последовательности и описана модель база данных NMS, а также интерфейсы веб-версии системы.

3 Разработка и тестирование системы

3.1 Инструкции к программам

3.1.1 Инструкция по сборке и запуску агента

Для сборки программы под устройство необходимо воспользоваться средствами кросс-компиляции. Для сборки программы под модем использовался образ Gentoo, в которой установлены средства для компиляции программы с использованием инструкций архитектуры ARM64 из-под системы, использующая архитектуру x86-64.

Если окружение предварительно настроено, для сборки агента необходимо запустить программу CMake в директории программы с указанием путей библиотеки grpc-client и libbackproto исходных файлов и скомпилированных библиотек:

```
1) cmake -S . -B build
-D CMAKE_TOOLCHAIN_FILE=/path/to/cross-comp.cmake
-D LIBBACKENDPROTO_INCLUDE_PATH=/path/to/libbackendproto/build
-D LIBBACKENDPROTO_LIBRARY_PATH=/path/to/libbackendproto/build
-D LIBGRPCCLIENT_INCLUDE_PATH=/path/to/grpc-client/
-D LIBGRP-CCLIENT_LIBRARY_PATH=/path/to/grpc-client/
```

```
2) cmake --build build
```

После компиляции программы, в папке build появится программа с названием snmp_agent.

Для запуска программы необходима предварительно запустить демон Net-SNMP, а также переместить собранные библиотеки используемых агентом в папку /usr/lib. Запуск программы происходит с правами суперпользователя.

Полученную программу необходимо передать на устройство, на котором уже установлены упомянутые ранее библиотеки и настроены конфигурационные файлы (согласно пункту 3.3.1).

В случае успешного запуска программы, в терминале будет написано успешное подключение к демону, а также какие таблицы были созданы (модулятор или демодулятор DVBS2X или DSSS и OpenAMIP). Пример успешного запуска программы продемонстрирован на рисунке 3.1.

```
GetPrivilege result = 1
Software: Тип CCC = 1
Таблица 'Модулятор DVBS2X' создана
Таблица 'Модулятор DVBS2X' заполнена
Таблица 'Демодулятор DVBS2X' создана
Таблица 'Демодулятор DVBS2X' заполнена
Таблица 'АКМ' создана
Таблица 'АКМ' заполнена
Software: OpenAMIP найден
NET-SNMP version 5.9.4.pre2 AgentX subagent connected
```

Рисунок 3.1 – Результат запуска агента на устройстве

3.1.2 Инструкция по сборке и запуску сервера

Для запуска сервера на операционной системе Ubuntu 22.04 (и других дистрибутивов на основе ядра Linux) необходимо:

- 1) установить Python версии 3.10 и выше;
- 2) скачать репозиторий сервера из Gitlab;
- 3) создать локальное для директивы окружения Python следующей командой: `python -m venv venv` и активировать данное окружение командой `source venv/bin/activate`;
- 4) установить библиотеки, написанные в файле `requirements.txt`:
`pip install -r requirements.txt`;
- 5) запуск происходит командой: `flask run`;
- 6) в случае успешного запуска, в терминал будет выведено сообщение `Debug mode: off`.

После этого сайт будет запущен на порту 5000. Открыть сайт можно по адресу `127.0.0.1:5000`.

3.2 Инструменты разработчика

3.2.1 Разработка SNMP-агента

Настройка окружения для разработки программы SNMP агента необходимо предварительно установить пакет Net-SNMP. Далее, для управления параметрами спутникового модема необходимо установить библиотеку gRPC-client и libbackproto.

В качестве среды разработки для агента использовалась Visual Studio Code 1.86 [8]. Необходимо установить следующие расширения: clang для подсветки синтаксиса, а также установить пути к заголовочным файлам используемых библиотек (Net-SNMP, gRPC-client и libbackproto).

Для сборки проекта использовать CMake [9] версии минимум 3.12.

В качестве системы контроля версии использовался Git [10]. Оформлять новые коммиты необходимо использовать со стандартом предприятия.

3.2.2 Разработка NMS

Для разработки NMS предварительно должно быть настроено окружение (см. пункт 3.1.2).

В качестве среды разработки использовалась программа PyCharm 2023.2.6 [11].

В качестве базы данных используется MongoDB 7.0.

При разработке сервера необходимо придерживаться правил стандарта PEP 8.

В качестве системы контроля версии использовался Git. Оформлять новые коммиты необходимо в соответствии стандарта предприятия.

3.3 Тестирование

3.3.1 Тестирование SNMP-агента

Перед тем, как начинать тестирование программы, необходимо переместить MIB из папки mibs в директорию хранения MIB Net-SNMP. Для Ubuntu 22.04 это папка по следующему пути: /usr/share/snmp/mibs.

Также необходимо настроить snmpd.conf и указать использование agentx по адресу 127.0.0.1. Пример готового snmpd.conf представлен на рисунке 3.2.

```
sysServices 72
master agentx
agentAddress 127.0.0.1

rwcommunity public 127.0.0.1 .1
```

Рисунок 3.2 – Минимальная настройка snmpd.conf

Также необходимо отредактировать файл snmp.conf и добавить использование MIB: либо все (mibs: all), либо их перечислить через двоеточие как на рисунке 3.3.

```
mibs :DELTA-GENERAL-MIB:DELTA-PRODUCTS-MIB:DELTA-REG-MIB:DELTA-TC-
MIB:DELTA-BASIS20-MODEM-MIB:DELTA-BASIS20-DVBS2X-MIB:DELTA-BASIS20-DSSS-
MIB:DELTA-BASIS20-OPENAMIP-MIB
```

Рисунок 3.3 – Пример заполненного snmp.conf с перечислением

После изменения этих двух файлов перезапустить демон Net-SNMP (для Ubuntu 22.04 это sudo systemctl restart snmpd).

Для проверки работы SNMP-агента необходимо собрать программу на локальном устройстве с теми же флагами, как и в пункте 3.1.1, добавив еще один: -DDUMMY_SNMP=ON. Данный флаг отключает подключение к клиенту и выводит заранее подготовленные значения. Для проверки работы можно вос-

пользоваться программой *snmpwalk* из пакета Net-SNMP. Запустить программу *snmpwalk* со следующими флагами: `-v 2c -c public localhost 1.3.6.1.4.1.61503`

В качестве результата работы будет выведены следующие параметры (Рисунок 3.4).

```
DELTA-BASIS20-DVBS2X-MIB::modBucState.1 = INTEGER: off(1)
DELTA-BASIS20-DVBS2X-MIB::modBucVMeas.1 = Gauge32: 1 V
DELTA-BASIS20-DVBS2X-MIB::modBucVMeas.1 = Gauge32: 0 A
DELTA-BASIS20-DVBS2X-MIB::modOutputLevel.1 = INTEGER: -30.00 dB
DELTA-BASIS20-DVBS2X-MIB::modCenterFreq.1 = Gauge32: 1600000.000 KHz
DELTA-BASIS20-DVBS2X-MIB::modSymbolRate.1 = Gauge32: 5000000 sym/s
DELTA-BASIS20-DVBS2X-MIB::modBitrate.1 = Gauge32: 474 kb/s
DELTA-BASIS20-DVBS2X-MIB::modPackets.1 = Gauge32: 21
DELTA-BASIS20-DVBS2X-MIB::modBytes.1 = Gauge32: 21 bytes
DELTA-BASIS20-DVBS2X-MIB::modTxMode.1 = INTEGER: up(2)
DELTA-BASIS20-DVBS2X-MIB::modRef.1 = INTEGER: off(1)
DELTA-BASIS20-DVBS2X-MIB::modInvSpecter.1 = INTEGER: on(2)
DELTA-BASIS20-DVBS2X-MIB::modDvbs2xModcod.1 = INTEGER: a32psk45(25)
DELTA-BASIS20-DVBS2X-MIB::modDvbs2xFrameLength.1 = INTEGER: normal(1)
DELTA-BASIS20-DVBS2X-MIB::modDvbs2xRolloff.1 = INTEGER: r10(2)
DELTA-BASIS20-DVBS2X-MIB::demodLnbVMeas.1 = INTEGER: 0 V
DELTA-BASIS20-DVBS2X-MIB::demodLnbVMeas.1 = Gauge32: 11 mA
DELTA-BASIS20-DVBS2X-MIB::demodEsno.1 = INTEGER: -90.0 dB
DELTA-BASIS20-DVBS2X-MIB::demodFreqOffset.1 = INTEGER: 0 Hz
DELTA-BASIS20-DVBS2X-MIB::demodCenterFreq.1 = Gauge32: 1500000.000 KHz
DELTA-BASIS20-DVBS2X-MIB::demodSymbolRate.1 = Gauge32: 5000000 sym/s
DELTA-BASIS20-DVBS2X-MIB::demodLnbPresetVoltage.1 = INTEGER: 1
DELTA-BASIS20-DVBS2X-MIB::demodRef.1 = INTEGER: on(2)
DELTA-BASIS20-DVBS2X-MIB::acmState.1 = INTEGER: on(2)
DELTA-BASIS20-DVBS2X-MIB::acmLocalMargin.1 = INTEGER: 4,5
DELTA-BASIS20-DVBS2X-MIB::acmRemoteMargin.1 = INTEGER: 4,5
DELTA-BASIS20-DVBS2X-MIB::acmMinModcod.1 = INTEGER: qpsk12(4)
DELTA-BASIS20-DVBS2X-MIB::acmMaxModcod.1 = INTEGER: a64psk56(31)
DELTA-BASIS20-OPENAMIP-MIB::openAMIPConnAddr.0 = IpAddress: 1.0.0.91
DELTA-BASIS20-OPENAMIP-MIB::openAMIPConnManufactures.0 = STRING: "OpenAMIP"
DELTA-BASIS20-OPENAMIP-MIB::openAMIPConnModel.0 = STRING: "OpenAMIP"
DELTA-BASIS20-OPENAMIP-MIB::openAMIPConnPort.0 = Gauge32: 5005
DELTA-BASIS20-OPENAMIP-MIB::openAMIPConnTimeout.0 = Gauge32: 60 seconds
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingLongitude.0 = INTEGER: 91.1 degrees C
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingHuntBandwidth.0 = INTEGER: .512 MHz
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingHuntFreq.0 = INTEGER: 950.1 MHz
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingRxLclOsc.0 = INTEGER: 10000.0 MHz
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingTxLclOsc.0 = INTEGER: 10000.0 MHz
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingRxPolarization.0 = INTEGER: horizontal(3)
DELTA-BASIS20-OPENAMIP-MIB::openAMIPPointingTxPolarization.0 = INTEGER: vertical(4)
DELTA-BASIS20-OPENAMIP-MIB::openAMIPIntervalsLatlong.0 = INTEGER: 60 seconds
DELTA-BASIS20-OPENAMIP-MIB::openAMIPIntervalsKeepalive.0 = INTEGER: 10 seconds
```

Рисунок 3.4 – Вывод *snmpwalk* при компиляции с флагом отладки

Если вместо DELTA-BASIS20-DVBS2X выводятся 1.3.6.1.4.1.61503.2.1.1.2.1..., то некорректно были установлены MIB или некорректный запуск *snmpd* (проверить `sudo systemctl status snmpd`) и перепроверить файл *snmp.conf*.

В случае корректного вывода, скомпилировать программу уже без использования флага `DUMMY_SNMP` и запустить агента на устройстве. Запустить программу `snmpwalk` с такими же флагами, указав вместо `localhost` IP-адрес устройства. В результате работы будут получены уже реальные значения с устройства. Проверить сходства значений с модема можно другими средствами отладки.

3.3.2 Тестирование NMS

Тестирование работы NMS происходит в ручном режиме. Для тестирования сервера необязательно использовать модем, достаточно запустить агента в Docker-контейнере. Для этого необходимо в директории агента создать папку `modules`, переместить исходные файлы `libbackproto` и `grpc-client`. После создания образа, его необходимо запустить и вывести его IP-адрес. Выполнять команду `docker run` ровно столько раз, сколько планируете использовать для отладки системы (достаточно 4-6 образов). Исходный код Docker представлен в приложении А.

На рисунке 3.5 представлен код для создания и запуска Docker-контейнера.

```
# Создать образ
docker build -t snmp-agent-new .

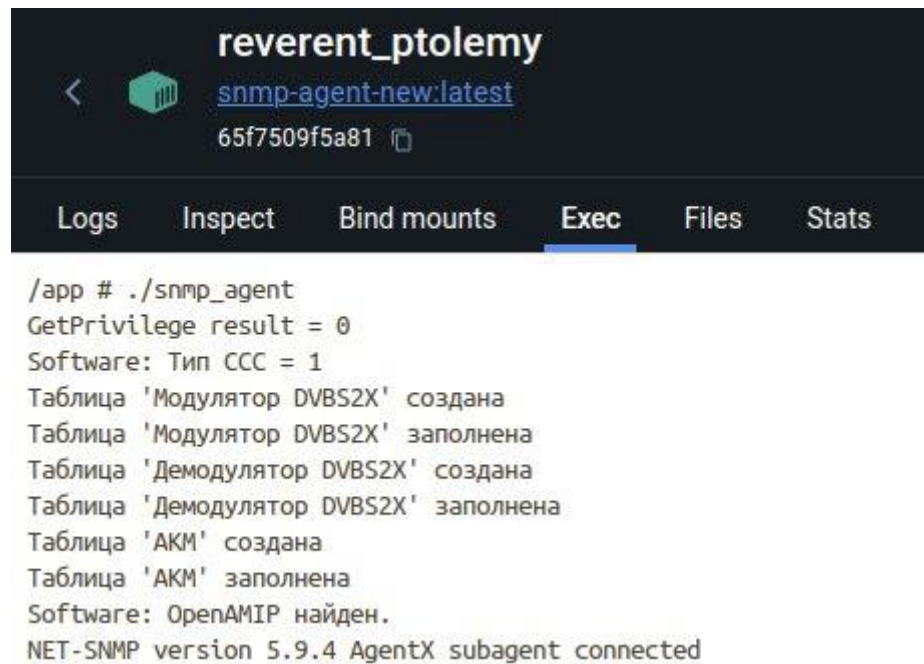
# Запустить образ
docker run -d snmp-agent-new:latest



# Получить список запущенных контейнеров
docker ps -a

# Вывести IP адрес устройства
docker inspect --format='{{.NetworkSettings.IPAddress}}' CONTAINER_ID
```

Рисунок 3.5 – Создание и запуск Docker-контейнера

Результат запуска программы в контейнере изображен на рисунке 3.6.



```
reverent_ptolemy
<  snmp-agent-new:latest
65f7509f5a81 

Logs Inspect Bind mounts Exec Files Stats

/app # ./snmp_agent
GetPrivilege result = 0
Software: Тип CCC = 1
Таблица 'Модулятор DVBS2X' создана
Таблица 'Модулятор DVBS2X' заполнена
Таблица 'Демодулятор DVBS2X' создана
Таблица 'Демодулятор DVBS2X' заполнена
Таблица 'АКМ' создана
Таблица 'АКМ' заполнена
Software: OpenAMIP найден.
NET-SNMP version 5.9.4 AgentX subagent connected
```

Рисунок 3.6 – Результат запуска программы в контейнере

После этого запустить NMS как в пункте 3.1.2, создать сеть, добавить абонентские терминалы с адресами, полученными в предыдущем абзаце. После этого создать пару сценариев и добавить их в расписание. В случае корректной работы, в логах запущенных устройств будет сообщено о настройке новых параметров (центральные частоты и символьные скорости модулятора и демодулятора, а также включение передачи модулятора), а в окне «Контроль выполнения расписания», будут указаны созданные абонентские терминалы, их частоты и зеленый цвет статуса выполнения сценария в окне «Уведомления».

Если перейти в параметры демодуляторов центральной станции, то будут указаны частоты, указанные при создании сценария.

В окне «Абонентские терминалы», задействованные станции будут выдавать текущие параметры станций (меняться они не будут, так как имитация уровня шума отсутствует), а статус их работы будет помечен зеленым кругом.

При просмотре параметров абонентского терминала значения будут такими же, как на рисунке 3.2, кроме значений центральные частоты и символьные скорости модулятора и демодулятора, а также включение передачи модулятора.

3.4 Выводы по главе

В процессе программной реализации были сделаны следующие задачи:

- разработана архитектура системы;
- разработано программное обеспечение, для сбора и управления параметрами модема;
- разработано программное обеспечение, управляющая компонентами устройства по расписанию;
- разработана МПВ, в которой отражены компоненты системы, их описание и тип возвращаемых параметров;
- разработан интерфейс для взаимодействия оператора с системой.

Планируется добавить:

- создание и удаление учетных записей операторов;
- реализация графика контроля управления сценариями, а также визуализация параметров в виде графика.

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной квалификационной работы была спроектирована и реализована система мониторинга и управления абонентскими терминалами в сети спутниковой связи.

На первом этапе были проведены анализ существующих аналогов, их преимущества и недостатки, на основе которых были поставлены задачи, решаемые в ходе работы. Также были выбраны инструменты, с помощью которых будут реализованы функционал сервера и клиента.

Следом была разработана общая архитектура сети, таблицы MIB, макеты интерфейсов и диаграммы логики работы NMS. Это позволило перейти к разработке системы.

На последнем этапе были продемонстрированы инструкции по сборке, разработке и тестированию разработанного программного обеспечения.

Поставленные цели и задачи были выполнены в полном объеме. В дальнейшем планируется модернизировать разработанную систему в некоторых аспектах, таких как:

- контроль опрашиваемых параметров и установка частоты их опроса;
- создание более удобного интерфейса для мониторинга системы оператором.

Для разработки системы были спроектированы и написаны SNMP агент, который управляет устройством, а также NMS, которая позволяет оператору составлять расписания, по которому управляются устройство.

СПИСОК СОКРАЩЕНИЙ

- АТ – абонентский терминал;
- ЦС – центральная станция;
- NMS (Network Management System) – система управления сетью;
- SNMP (Simple Network Management Protocol) – простой протокол сетевого управления;
- MIB (Management Information Base) – информационная база управления;
- OID (Object Identifier) – идентификатор объекта;
- БД – база данных;
- DVBS2X (Digital Video Broadcasting Satellite 2 Extension) – стандарт спутникового телевидения;
- DSSS (ШПС) – широкополосная система;
- BUC (Block Upconverter) – устройство преобразования частоты и усиления передачи данных на спутник;
- LNB (Low-Noise Block downconverter) – устройство, понижающее частоту сигнала, принимаемое со спутника;
- OpenAMIP (Open Antenna Modem Interface Protocol) – открытый протокол интерфейса модема.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Особенности OpenWISP // Open Source Network Management System. – URL: <https://openwisp.org/whatis.html> (дата обращения: 17.03.2024).
2. Особенности NetXMS // NetXMS – enabling networks. – URL: <https://netxms.com/features> (дата обращения: 17.03.2024).
3. Особенности OpenNMS // Network Monitoring Software – OpenNMS. – URL: <https://www.opennms.com/> (дата обращения: 17.03.2024).
4. Проблемы безопасности SNMP на практике // Habr : сайт. – URL: <https://www.habr.com/ru/companies/selectel/articles/719402/> (дата обращения: 17.03.2024).
5. Микрофреймворк веб-приложений // Flask: сайт. – URL: <https://flask.palletsprojects.com/en/3.0.x/> (дата обращения: 17.03.2024).
6. Библиотека Net-SNMP // Net-SNMP.org : сайт. – URL: <http://www.net-snmp.org/> (дата обращения: 17.03.2024).
7. База данных MongoDB // MongoDB.com : сайт. – URL: <https://www.mongodb.com/> (дата обращения: 17.03.2024).
8. Редактор кода Visual Studio Code // Страница загрузки Visual Studio Code : сайт. – URL: <https://www.code.visualstudio.com/> (дата обращения: 17.03.2024).
9. Система сборки CMake // CMake.org : сайт. – URL: <https://www.cmake.org/> (дата обращения: 17.03.2024).
10. Система управления версиями Git // Git-scm.com : сайт. – URL: <https://git-scm.com/> (дата обращения: 17.03.2024).
11. Редактор кода JetBrains PyCharm Community // Страница загрузки JetBrains : сайт. – URL: <https://www.jetbrains.com/pycharm/download/> (дата обращения: 17.03.2024).

ПРИЛОЖЕНИЕ А

Исходный код Dockerfile

```
FROM alpine:3.19.1 AS build
RUN apk update && \
    apk add --no-cache \
        build-base \
        cmake \
        grpc \
        grpc-dev \
        protobuf \
        protobuf-dev \
        net-snmp \
        net-snmp-dev
ADD ./modules /modules
WORKDIR /modules/libbackendproto/
RUN cmake -S . -B build && \
    cmake --build build && \
    cp /modules/libbackendproto/build/libbackendproto.so /usr/local/lib/
WORKDIR /modules/grpc-client/
RUN cmake -S . -B build \
    -DLIBBACKENDPROTO_INCLUDE_PATH=/modules/libbackendproto/build \
    -DLIBBACKENDPROTO_LIBRARY_PATH=/modules/libbackendproto/build \
    -DLIBHELPERS_INCLUDE_PATH=/modules/libhelper && cmake --build build && \
    cp /modules/grpc-client/build/libgrpc-client.so /usr/local/lib/
ADD ./src /modules/snmp-agent/src
ADD ./cmake /modules/snmp-agent/cmake
ADD ./CMakeLists.txt /modules/snmp-agent/CMakeLists.txt
WORKDIR /modules/snmp-agent/
RUN cmake -S . -B build \
    -DLIBBACKENDPROTO_INCLUDE_PATH=/modules/libbackendproto/build \
    -DLIBBACKENDPROTO_LIBRARY_PATH=/modules/libbackendproto/build \
    -DLIBGRPC-CLIENT_INCLUDE_PATH=/modules/grpc-client/ \
    -DLIBGRPC-CLIENT_LIBRARY_PATH=/modules/grpc-client/build \
    -DDUMMY_SNMP=ON && \
    cmake --build build

FROM alpine:3.19.1 AS runtime
RUN apk update && \
    apk add --no-cache \
        build-base \
        libstdc++ \
        libgcc \
        grpc-cpp \
        protobuf \
        net-snmp
WORKDIR /app
COPY --from=build /usr/local/lib/libbackendproto.so /usr/lib/
COPY --from=build /usr/local/lib/libgrpc-client.so /usr/lib/
COPY --from=build /modules/snmp-agent/build/snmp_agent .
RUN echo "
    sysServices 72
    master agentx
    agentAddress udp:161
    rwcommunity public default .1" >> /etc/snmp/snmpd.conf
RUN snmpd -Lo
ENTRYPOINT [ "./snmp_agent" ]
```

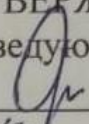

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

УТВЕРЖДАЮ

Заведующий кафедрой

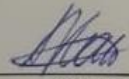
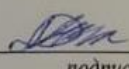
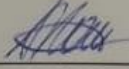
 О.В. Непомнящий

« 17 » 06 2024 г.

БАКАЛАВРСКАЯ РАБОТА

090301 Информатика и вычислительная техника

Разработка системы мониторинга и управления абонентскими терминалами в сети спутниковой связи

Руководитель	 подпись	17.06.24 дата	ст. преподаватель должность, ученая степень	А.Г. Хантимиров
Выпускник	 подпись	17.06.24 дата		Д.А. Анциферов
Нормоконтролёр	 подпись	17.06.24 дата	ст. преподаватель должность, ученая степень	А.Г. Хантимиров