

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ  
федеральное государственное бюджетное образовательное учреждение  
высшего профессионального образования  
«Московский государственный индустриальный университет»  
(ФГБОУ ВПО «МГИУ»)**

**Кафедра информационных систем и технологий**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА**

**по направлению «Информационные системы и технологии»**

**на тему «Система управления и контроля сетевых устройств на основе  
простого протокола управления сетями»**

Студент-дипломник \_\_\_\_\_ Д.Г. Грачев

Руководитель работы \_\_\_\_\_ нач. БТО УИТ «МГИУ» Ю.В. Курасов

**ДОПУСКАЕТСЯ К ЗАЩИТЕ**

Заведующий кафедрой \_\_\_\_\_ доцент, к.ф.-м.н. Е.А. Роганов

**МОСКВА 2015**

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ  
федеральное государственное бюджетное образовательное  
учреждение высшего профессионального образования  
«Московский государственный индустриальный университет»  
(ФГБОУ ВПО «МГИУ»)

Кафедра информационных систем и технологий

Зав. Кафедрой  
\_\_\_\_\_/Роганов Е.А. /  
« \_\_\_\_ » \_\_\_\_\_ 2015г.

**Задание**  
на выпускную работу по направлению «Информационные системы и  
технологии»

230100 «Информатика и вычислительная техника»

студента Д.Г.Грачев группы 111131

1. Тема работы «Система управления и контроля сетевых устройств на основе простого протокола управления сетями»
2. Сроки начала работы
3. Руководитель дипломной работы: Курасов Юрий Викторович
4. Задание дипломной работы:
  1. Цель работы — разработка системы мониторинга локальной вычислительной сети.
  2. Содержание дипломной работы:
    1. Введение.
    2. Литературный обзор.
    3. Структура системы.
    4. Пользовательский интерфейс.
  3. Используемые технологии: C++, SQLite3, SNMP, JavaScript, JQuery, HTML5, CSS3, Ruby, Ruby on Rails.
  4. Практическая реализация: данная работа может найти применение в БТО ИВЦ «МГИУ».
  5. Графическая часть:
    1. Диаграмма использования.
    2. Диаграмма классов.
    3. Реализация интерфейса.

Студент-дипломник \_\_\_\_\_

Д.Г. Грачев

Руководитель работы \_\_\_\_\_ нач.БТО УИТ «МГИУ» Ю.В. Курасов

## Аннотация

Работа посвящена созданию системы для мониторинга устройств в локальной вычислительной сети ФГБОУ ВПО «МГИУ» по протоколу SNMP v3 на языке программирования. Включающую фоновое приложение на C++, а также WEB интерфейс для интерактивного взаимодействия с пользователем реализованного на фреймворке Ruby on Rails.

Требуется реализовать следующие подзадачи:

- Основная программа для сбора статистики должна работать в фоновом режиме;
- Опрос устройств в асинхронном режиме;
- Работу с базой данных;
- С фоновым приложением можно взаимодействовать через CLI;
- Необходима авторизация как через приложение, так и на WEB сервере используя данные о пользователях из одной базы данных.

WEB приложение должно иметь возможность предоставления отчета как в виде графика, так и в виде списка, создания/редактирования/удаления пользователей, вспомогательных словарей, устройств из базы данных.

Работа содержит 40 страниц, 13 иллюстраций, 1 приложение.

Ключевые слова: Система мониторинга, SNMP, Ruby, C++, Ruby on Rails, HAML, UML, локальная вычислительная сеть.

## Оглавление

Аннотация.....	3
Введение.....	5
1. Литературный обзор.....	7
1.1 Постановка задачи.....	7
1.2 Обзор существующих решений.....	8
2 Проектирование системы.....	13
2.2 Требования к системе.....	15
2.3 Обзор используемых технологий.....	16
2.4 Реализация.....	20
3. Описание пользовательских интерфейсов.....	26
4. Заключение.....	36
5. Список используемой литературы.....	37
6. Приложение.....	38
6.1 Полная схема базы данных Web приложения.....	38

## Введение

В современном мире информационные системы усложняются с каждым днем. Возрастают требования к вычислительным сетям, благодаря которым происходит передача и обработка данных. В нынешнее время для успешной работы организаций, компаний, необходима стабильная информационная система. От нее может зависеть огромное количество факторов от которых зависит функциональность организации. К примеру это может быть внутренний почтовый сервис или система документооборота.

Ежедневно по вычислительной информационной системе может передавать огромное количество информации. Потому возникает необходимость в административной поддержки.

Для успешного администрирования вычислительной системы, далее ЛС, возрастает необходимость в инструментах позволяющих эффективно отслеживать состояние ЛС. Так же для успешного администрирования сети необходимо знать состояние каждого ее элемента с возможностью изменять параметры его функционирования. Обычно сеть состоит из устройств различных производителей и управлять ею было бы нелегкой задачей если бы каждое из сетевых устройств понимало только свою систему команд. Поэтому возникла необходимость в создании единого языка управления сетевыми ресурсами, который бы понимали все устройства, и который, в силу этого, использовался бы всеми пакетами управления сетью для взаимодействия с конкретными устройствами.

Подобным языком стал SNMP - Simple Network Management Protocol. Разработанный для систем, ориентированных под операционную систему UNIX, он стал фактически общепринятым стандартом сетевых систем управления и поддерживается подавляющим большинством производителей сетевого оборудования в своих продуктах. В силу своего названия - Простой

Протокол Сетевого Управления - основной задачей при его разработке было добиться максимальной простоты его реализации. В результате возник протокол, включающий минимальный набор команд, однако позволяющий выполнять практически весь спектр задач управления сетевыми устройствами - от получения информации о местонахождении конкретного устройства, до возможности производить его тестирование.

Но так как SNMP требует определенных навыков в использовании UNIX систем, возникает необходимость в квалифицированных работниках. Часто важно иметь сотрудника который способен быстро устранить неисправность в ЛС, отследить состояние, а так же перенастроить оборудование. Для обучения молодого персонала требуется время и средства. Соответственно возникает потребность в сокращении времени на обучение сотрудника. Либо средства позволяющие на интуитивном уровне администрировать ЛС.

В ФГБОУ ВПО «МГИУ» информационная сеть содержит сотни узлов от состояний которых может зависеть учебно — производственный процесс. Требуется отслеживать состояние отдельных узлов, их характеристик таких как свободное место на жестком диске, занятой оперативной памяти, нагрузке центрального процессора, время работы, сообщений событий и многое другое.

Цель работы – разработка эффективной, простой в использовании системы для мониторинга локальной вычислительной сети в рамках ИВЦ ФГБОУ ВПО «МГИУ».

Для этого необходимо решить следующие задачи:

1. Изучение протокола SNMP
2. Разработка алгоритма сбора информации
3. Проектирование хранилища базы данных
4. Реализовать фоновое приложение для сбора статистики
5. Реализовать Web интерфейс

# 1. Литературный обзор

## 1.1 Постановка задачи

Необходимо реализовать систему мониторинга, далее СМ, узлов входящих в локальную вычислительную сеть ФГБОУ ВПО «МГИУ», по протоколу SNMPv3. Под узлом подразумеваются устройства поддерживающие данный протокол. Это могут быть кондиционеры, маршрутизаторы, электронные замки, персональные компьютеры. Передача данных должна передаваться в зашифрованном виде. Системе необходимо производить асинхронный опрос устройств на определенные параметры, такие как показания датчиком температуры, время работы узла, занятая оперативная память, сообщения событий. Результаты необходимо хранить в базе данных. Система должна содержать фоновое приложение для работы с устройствами и иметь возможность работы через CLI. Инициализация данных происходит посредством чтения данных из базы данных. Так же необходимо реализовать WEB интерфейс предоставляющий возможность просмотра статистики, добавления устройств и настройки их параметров через браузер в интерактивном режиме. Более подробное описание приведено ниже.

**CLI** ( Command line interface) – в данном случае подразумевается взаимодействие пользователя с приложением посредством консоли после подключения через TCP/IP протокол и авторизацию.

**SNMP** (Simple Network Management Protocol ) - стандартный интернет-протокол для управления устройствами в IP-сетях на основе архитектур TCP/UDP. К поддерживающим SNMP устройствам относятся маршрутизаторы, коммутаторы, серверы, рабочие станции, принтеры, модемные стойки и другие.

## 1.2 Обзор существующих решений

Существует несколько решений, для мониторинга устройств в ЛВС по протоколу SNMP.

Один из самых известных и наиболее мощных является Zabbix. [1]

**Zabbix**—свободная система мониторинга и отслеживания статусов разнообразных сервисов компьютерной сети, сетевого оборудования.

Структура:

- **Zabbix сервер** — это ядро программного обеспечения Zabbix. В его возможности входит удаленная проверка сетевых сервисов, хранение конфигурационных файлов. Он является тем субъектом в программном обеспечении Zabbix, который оповестит администраторов в случае возникновения проблем с контролируемым оборудованием.
- **Zabbix прокси** — собирает данные о производительности и доступности от имени Zabbix сервера. Все собранные данные заносятся в буфер на локальном уровне и передаются Zabbix серверу, к которому принадлежит прокси-сервер. Zabbix прокси является отличным решением для централизованного удаленного мониторинга мест, филиалов, сетей, не имеющих локальных администраторов. Он способен быть также использован для распределения нагрузки одного Zabbix сервера. В этом случае, прокси только собирает информацию, тем самым сервер испытывает меньшую нагрузку.
- **Zabbix агент** — наблюдение за состоянием локальных ресурсов и приложений (таких как жесткие диски, память, статистика процессора и т. д.) на сетевых системах, они должны работать с запущенным Zabbix агентом.



- **Web-интерфейс** — интерфейс является частью Zabbix сервера, и, как правило, запущен на том же физическом сервере, что и Zabbix. Написан на PHP, требует веб сервер. Данной особенностью веб-интерфейса Zabbix является тот факт, что он не является фронтэндом в традиционном понимании этого слова: все операции чтения/записи веб-интерфейс осуществляет напрямую с базой данных, минуя собственно сервер zabbix. Таким образом, если не учитывать гипотетическую возможность записи пользователем в СУБД напрямую (что сильно осложняется отсутствием гарантий совместимости структуры базы данных от версии к версии), то во-первых сервер zabbix без web-интерфейса нефункционален, а во-вторых - сторонние разработчики на практике не могут написать "альтернативный" веб-интерфейс, поскольку тот должен будет привязываться к базе данных, спецификация которой может меняться без уведомления со стороны разработчиков Zabbix произвольным образом.

#### Возможности:

- Поддержка до 1000 узлов в сети. Конфигурация младших узлов полностью контролируется старшими узлами, находящимися на более высоком уровне иерархии;
- Планировка на основе полученных данных;
- Автоматическое обнаружение;
- Централизация лог-файлов;
- Возможность создания карты сетей.
- Web-интерфейс для администрирования и настройки;
- Отчетность и тенденции;
- SLA мониторинг;

- Поддержка высокопроизводительных агентов (zabbix-agent) практически для всех платформ;
- Поддержка IPMI;
- Комплексная реакция на события;
- Поддержка SNMP v1, 2, 3;
- Поддержка SNMP ловушек;
- Поддержка мониторинга JMX приложений из коробки;
- Выполнения запросов в различные базы данных без необходимости использования скриптовой обвязки;
- Выполнения внешних скриптов;
- Гибкая система групп и шаблонов;

И многое другое входит в возможности данного продукта. Данный продукт разрабатывается и поддерживается с 1998 года. Он включает мощный инструментарий для мониторинга устройств в локально вычислительной сети. Но его мощность и есть его недостаток — он достаточно сложен в освоении и требует определенного опыта как в программировании так и в понимании взаимодействия устройств. Так же можно к минусам отнести и то что приложение от сторонних разработчиков. SNMP v1 и v2 не поддерживают шифрования.

Так же многое из этого приложения просто не является востребованным для БТО ИВЦ «МГИУ».

**Zabbix** имеет поддержку, но она платная. И чем больше необходимо инструментов, расширенного функционала, тем дороже выйдет.

Так же подходящим решением может служить система мониторинга как **PowerSNMP Free Manager**.

**PowerSNMP Free Manager** - это бесплатный, полнофункциональный SNMP- менеджер, построенный с использованием PowerSNMP для .NET. Для работы с узлами сети, с возможностью просматривать MIB деревья и анализировать сетевые запросы. Отлично подходит для легких умеренных задач управления. [2]

**MIB** - виртуальная база данных, используемая для управления объектами в сети связи. Наиболее часто это понятие связывают с Simple Network Management Protocol (SNMP).

#### Особенности:

- Поддерживает SNMP версии 1, 2 и 3;
- Простой, легкий в использовании, доступный интерфейс;
- Работа по сети;
- Определяет SNMP ловушки;
- Построен с использованием надежной системой PowerSNMP для компонентов .NET Реализация задачи;
- Создает автоматические уведомления по электронной почте при переменных выпадающих из диапазона.

Так как в ИБЦ «МГИУ» используют только Unix подобные системы, а платформа .NET для семейства MS-DOS, **PowerSNMP Free Manager** уже не подходит для использования. Так же как и в **Zabbix** для расширения функционала, поддержки, необходима определенная плата.

Другие продукты либо малофункциональны, либо платные, либо устарели. В связи с этим необходимо реализовать свой собственный программный продукт удовлетворяющий требованиям мониторинга сети и имеющий простой интерфейс для работы, но имеющий достаточный функционал для анализа полученных данных.

## 2 Проектирование системы

### 2.1 Общие сведения

Оценив недостатки существующих систем, было принято решение о создании новой системы основными преимуществами которой должны стать:

- Использование легковесной базы данных Sqlite3;
- Простота в освоении и использовании;
- Количество поддерживаемый устройств не менее тысячи;
- Быстрый выбор и анализ необходимой информации;
- Шифрование на уровне сессии - SNMP v3;
- Удаленное отслеживание состояние компонентов ЛВС;
- Возможность сбора отдельных параметров с устройств;
- Минимизация человеческого ресурса для поддержки системы;
- Приложение для опроса устройств;
- Web приложение;

В системе предлагается выделить следующие функциональные подсистемы:

- Приложение работающее в фоновом режиме для сбора статистики и последующим сохранением в базу данных;
- Web приложение для быстрого анализа элементов локальной вычислительной сети;
- Генерация отчетов по отдельным запрашиваемым параметрам устройства и сохранения в файл;
- Возможность графического представления, если позволительно по типу данных, анализа данных;

Приложение для сбора статистики, далее менеджер, будет общаться с устройствами по SNMPv3, что позволит шифровать сессию для каждого соединения. Менеджер при запуске будет считывать данные об устройствах из базы данных и пытаться установить с ними соединение. В случае успеха, для каждого устройства будет произведена инициализация набора команд, которые позволят получать необходимые данные и сохранять их. Ответ от устройства приходит в виде строки, содержащая **oid** запроса и его значение.

**Oid (object identifier)** представляет собой иерархический запрос, где первый символ являясь вершиной дерева.

При запуске программа будет переходить в фоновый режим, в котором создаст два дополнительных потока в одном из которых поднимается асинхронный SNMP менеджер, а в другом сервер для обработки запросов от пользователя через CLI. Пользователь системы будет иметь возможность активировать/деактивировать любую из команд, устройство и многое другое.

Данные будут передаваться в зашифрованном виде благодаря поддержке SNMP v3. Полученные данные будут проверены на стороне приложения на валидность. После проверки валидации данные записываются в базу данных. Приложение предоставит пользовательский интерфейс после авторизации, который будет иметь возможность настройки приложения, просмотра отчетов, сведений об устройствах и т. п.

## 2.2 Требования к системе

К системе так же были предъявлены следующие требования:

1. Логирование;
2. Единая база данных для фонового приложения и web интерфейса;
3. Авторизация пользователей;
4. Зашифрованная передача данных между устройствами и приложением;
5. Валидация входных данных;
6. Кол-во поддерживаемых устройств — 1000;
7. Возможность создания отдельных запросов к устройству;
8. Необходимо иметь возможность подключение по TCP/IP к приложению для сбора статистики. (Реализовать сервер для работы с пользователем);
9. В Web интерфейсе должна быть возможность представления данных в графическом представлении;
10. Поиск по данным для быстрого анализа.

Использование данной системы будет возможно на любой операционной системе при помощи web-браузера. Web интерфейс позволит авторизованному пользователю просматривать статистику, создавать отдельные запросы к устройствам, генерировать отчеты по необходимым oid, управлять данными об устройствах и многое другое. А так же иметь возможность просмотра журнала работы фонового приложения (менеджера).

## 2.3 Обзор используемых технологий

Приведенные ниже технологии являются наиболее распространенные и оптимально подходящие для выполнения поставленной задачи.

### Ruby on Rails

**Ruby on Rails** - фреймворк, содержащий в себе достаточно различных пакетов, которые нужны каждому разработчику для создания нового проекта, позволяющий при меньших усилиях достигать решения поставленной задачи в короткие сроки. Написан на языке программирования Ruby. Ruby on Rails предоставляет архитектурный образец Model-View-Controller (модель-представление-контроллер) для веб-приложений, а также обеспечивает их интеграцию с веб-сервером и сервером базы данных. [7]

Некоторые принципы **Ruby on Rails**:

- **MVC** (Model-View-Controller);
- **DRY** – “Don’t Repeat Yourself” принцип разработки нацеленный на снижение повторения информации различного рода;
- **Convention Over Configuration** - используются соглашения по конфигурации, типичные для большинства приложений;
- **REST** - шаблон для веб приложений;



## JAVASCRIPT

**JavaScript** – это интерпретируемый язык программирования, с помощью которого веб-страницам придается интерактивность. С его помощью создаются приложения, которые включаются в **HTML**-код (например, анкеты или формы регистрации, которые заполняются пользователем).

Сценарии **JavaScript** выполняются на компьютере пользователя и поэтому представляют некоторую опасность. Возможность доступа к связанной с возможным конфиденциальной информации пользователя.

Например, при соответствующих настройках браузеры способны разрешать сценариям считывать файлы, в которых могут содержаться важные данные, например, пароли доступа. Поэтому в браузерах предусмотрена возможность отключения выполнения сценариев **JavaScript**. Это следует учитывать при разработке web-страницы с использованием **JavaScript**.

Если предполагается использовать один и тот же сценарий на многих web-страницах, лучше разместить его в отдельном файле и затем сослаться на этот файл. Это можно сделать даже в том случае, если код будет использован только на одной странице. Например, если сценарий слишком большой и громоздкий, то выделение его в отдельный файл облегчает восприятие и отладку кода web-страницы.

## C++

**C++** - компилируемый статически типизированный язык программирования общего назначения. Поддерживает такие парадигмы программирования как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование, обеспечивает модульность, отдельную компиляцию, обработку исключений, абстракцию данных, объявление типов (классов) объектов, виртуальные

функции и многое другое. Стандартная библиотека включает, в том числе, общепотребительные контейнеры и алгоритмы. C ++ сочетает свойства как высокоуровневых, так и низкоуровневых языков.

## **SNMP**

**SNMP** (англ *Simple Network Management Protocol*. - Простой протокол сетевого управления) - протокол для управления устройствами в сетях на основе архитектур TCP/UDP. К устройствам поддерживающим SNMP относятся серверы, маршрутизаторы, коммутаторы, рабочие станции, модемные стойки, принтеры и многое другое. Протокол используется в системах сетевого управления для отслеживания состояния подключенных к сети устройств на предмет условий, которые требуют внимания администратора. SNMP обозначен Инженерным советом интернета (IETF) как часть TCP / IP. В него входят стандарты для сетевого управления, включая протокол прикладного уровня, схему баз данных и набор объектов данных. [3]

## **Net-SNMP**

**Net- SNMP** - набор программного обеспечения для развёртывания и использования протокола SNMP различных версий . Он поддерживает различные версии интернет-протоколов , сокеты доменов Unix и некоторые другие. Содержит общие клиентские библиотеки , набор консольных приложений , расширяемый SNMP - агент , модули Perl и модули Python . [4]

## **Bcrypt**

**Bcrypt** - односторонняя криптографическая хеш - функция . Используется для защищенного хранения паролей.

## UML

Унифицированный язык моделирования (далее — UML) является языком графического описания для объектного моделирования в сфере разработки программного обеспечения. Будучи широкопрофильным языком, UML является открытым стандартом, использующим для создания абстрактной модели системы, именуемой UML-моделью, графические обозначения. Унифицированный язык моделирования был создан для определения, визуализации, проектирования и документирования программных систем. Несмотря на то, что сам UML не является языком программирования, разработчикам доступна генерация кода на основании UML-модели.

## 2.4 Реализация

Первоначально необходимо было разобраться с тем как будет выглядеть конечный продукт. Работа состоит из двух частей: реализации приложения для мониторинга устройств и WEB приложения, которое должно уметь только работать с данными из базы данных. Разработку необходимо начать с первого.

Прежде чем приступать к разработке самого приложения, необходимо определиться с тем, что будет храниться в базе данных. Опираясь на необходимый функционал и требования к системе за основу была принята следующая схема:

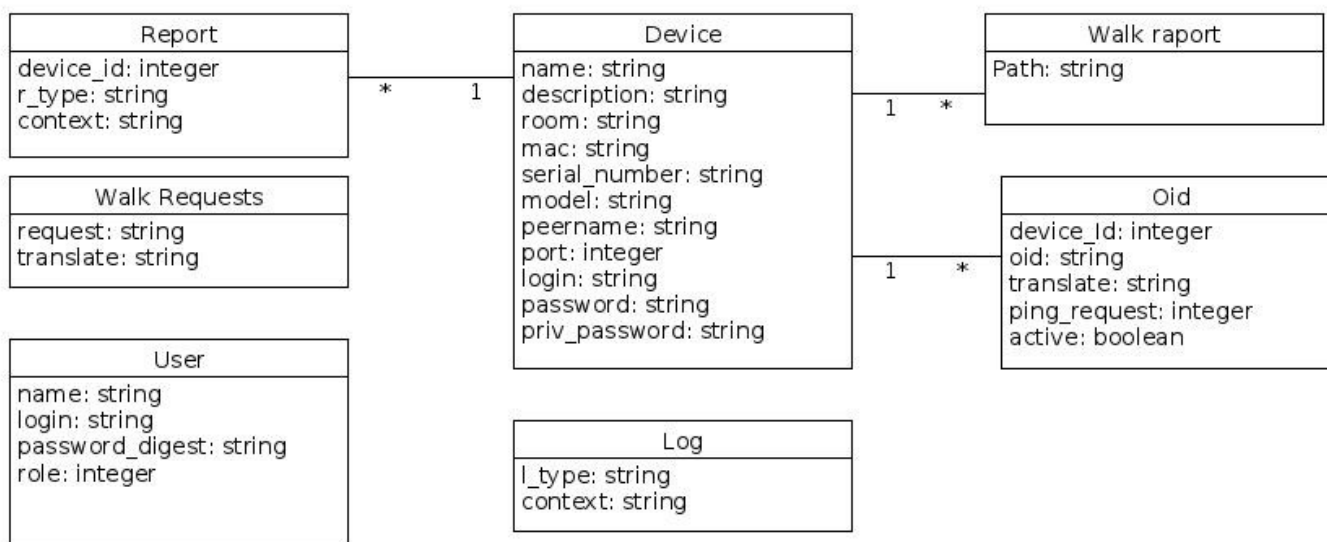


Рис. 1 Диаграмма классов

**User** — содержит логин и пароль в зашифрованном виде для авторизации в системе.

**Device** — описание устройства. Устройство может иметь имя, небольшое описание (до 1024 символов) для подсказки пользователю системы. Room — по возможности отображение нахождения данного устройства в определенном кабинете здания. Mac — уникальный идентификатор сетевой карты внутри сети. Serial Number — серийный номер устройства. Model — описание модели

устройства. Peername — адресс устройства. Может задаваться как IP так и строкой, при условии наличия соответствующего DNS соответствия. Port — порт по которому необходимо подключиться к устройстве. Так как менеджер будет работать по SNMPv3, то для авторизации необходимы login (имя пользователя), password и private password (для шифрованного соединения).

**Oid** — в данном случае это объект в котором храниться необходимый Oid для запроса определенного параметра с данного устройства, translate — описание, для облегчения работы пользователя. Оно хранит значение oid, чтобы не надо было обращаться к документации для данного устройства, за значением данной команды. Ping Request — это интервал в секундах между запросами данного параметра. Active — для удобства пользователя, команду можно временно включить или отключить. Вместо того, что бы ее удалять, а потом снова обращаться к документации, создавая ее, в случае необходимости.

Известно что приложение будет реализовано на языке программирования C++ и оно должно уметь работать с базой данных SQLite3. Соответственно применяем DAO шаблон проектирования и ORM технологию.

**Report** — таблица для хранения отчетов. В ней описывается для какого устройства какая команда была отправлена, и результат запроса.

**Walk Request** — содержит в себе набор oid общего назначения (такие как «system», которые стандартизированы). По ним можно генерировать целые отчеты, прямо из браузера пользователя для отдельных устройств, которые сохраняются в отдельный файл и доступны пользователю для просмотра. Соответственно в Walk Report, path — это путь к файлу отчета.

**Log** — это таблица хранящая сообщения об определенных событиях произошедших в менеджере. Так же доступен через web интерфейс.

У всех атрибутов так же имеются поля как created\_at и updated\_at, хранящие время создания и обновления.

Программе для сбора статистики необходимо не только уметь работать с

устройствами по SNMPv3, но и с базой данных Sqlite3. Причем эта база будет так же использоваться и Web приложением, соответственно необходимо учесть принцип ее взаимодействия. Для этого были применены ORM технология и шаблон проектирования DAO.

**DAO** — это объект, который предоставляет абстрактный интерфейс к какому-либо типу базы данных или механизму хранения. [5]

**ORM** — технология программирования, которая связывает базы данных с концепциями объектно-ориентированного программирования, создавая «виртуальную объектную базу данных». [6]

В последствии используя полученные данные была разработана конечная структура приложения. Для удобной разработки был использован UML.

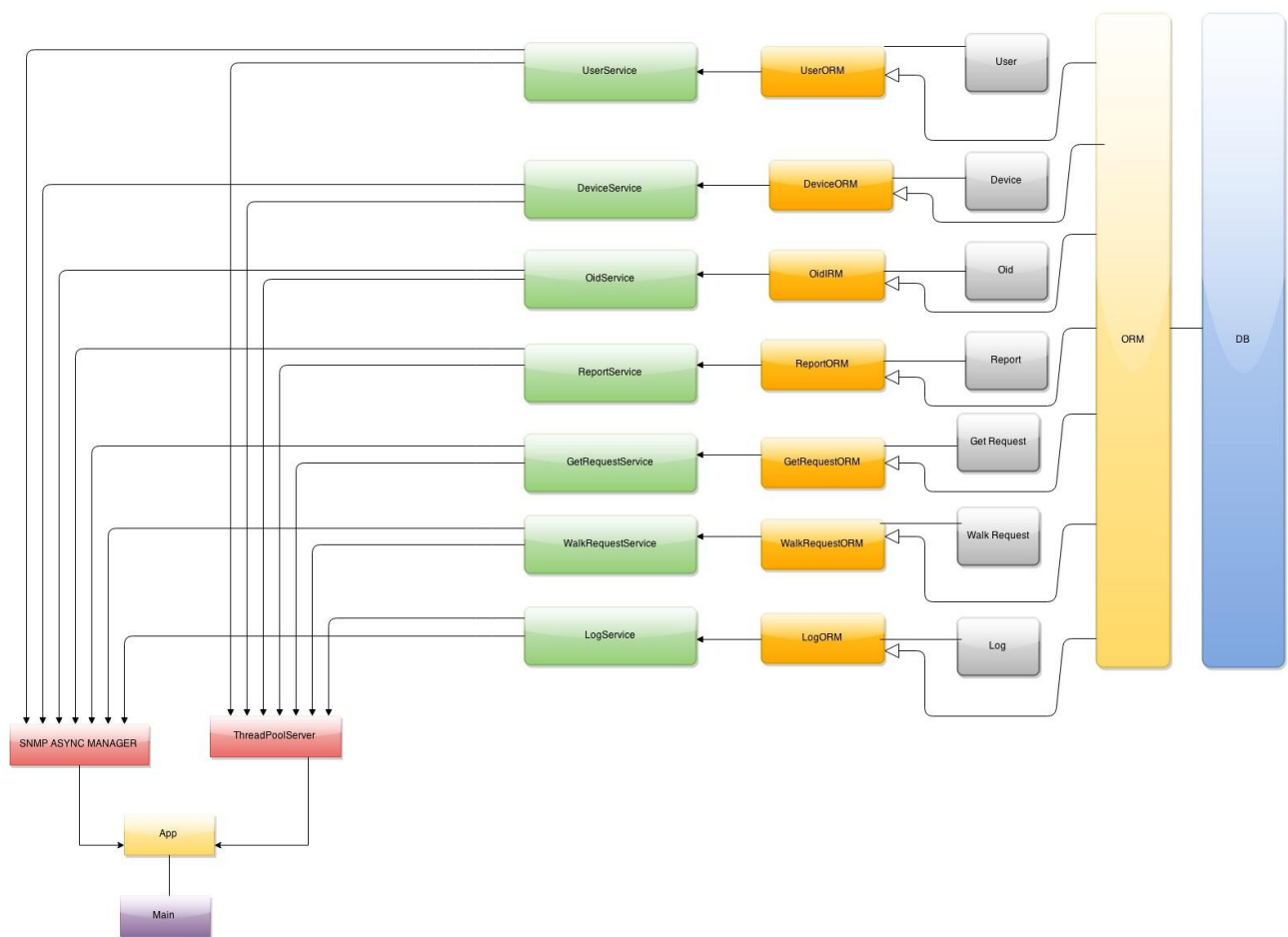


Рис. 2 Результат разработки модели менеджера

Данная диаграмма дает общее представление о связях между объектами внутри приложения. Особенность в свою очередь скрыты и остаются на совести программиста. Серым цветом выделены модели объектов, зеленым объекты позволяющие полученные данные из базы данных и превращать их в объекты определенного типа. Это и есть ORM технология. Зеленым отмечены сервисы. Они представляют собой обертки поверх основных классов для более удобной работы с объектами и базой данных. Именно их и будет использовать приложение.

Представим как пользователь будет взаимодействовать с конечным продуктом. Приложение будет постоянно «общаться» с устройствами внутри локальной вычислительной сети. Вести отчеты. У клиента есть два способа взаимодействовать с базой данных.

Первый это через приложение. Администратор будет подключаться к приложению по TCP/IP проходить авторизацию и получать доступ к сервису. Второй, самый простой, с помощью веб интерфейса. Который будет предоставлять удобный доступ к необходимым данным и дружелюбным для пользователя способом.

Это позволит человеку, неопытному в программировании и не знакомым с CLI, работать в интерактивном режиме. Интерфейс спроектирован так, что не требует более трех кликов из начального меню, чтобы получить доступ к необходимому функционалу. При разработке web интерфейса была заложена возможность быстрого обучения новых пользователей.

Примерно представить возможное взаимодействие можно следующим способом:



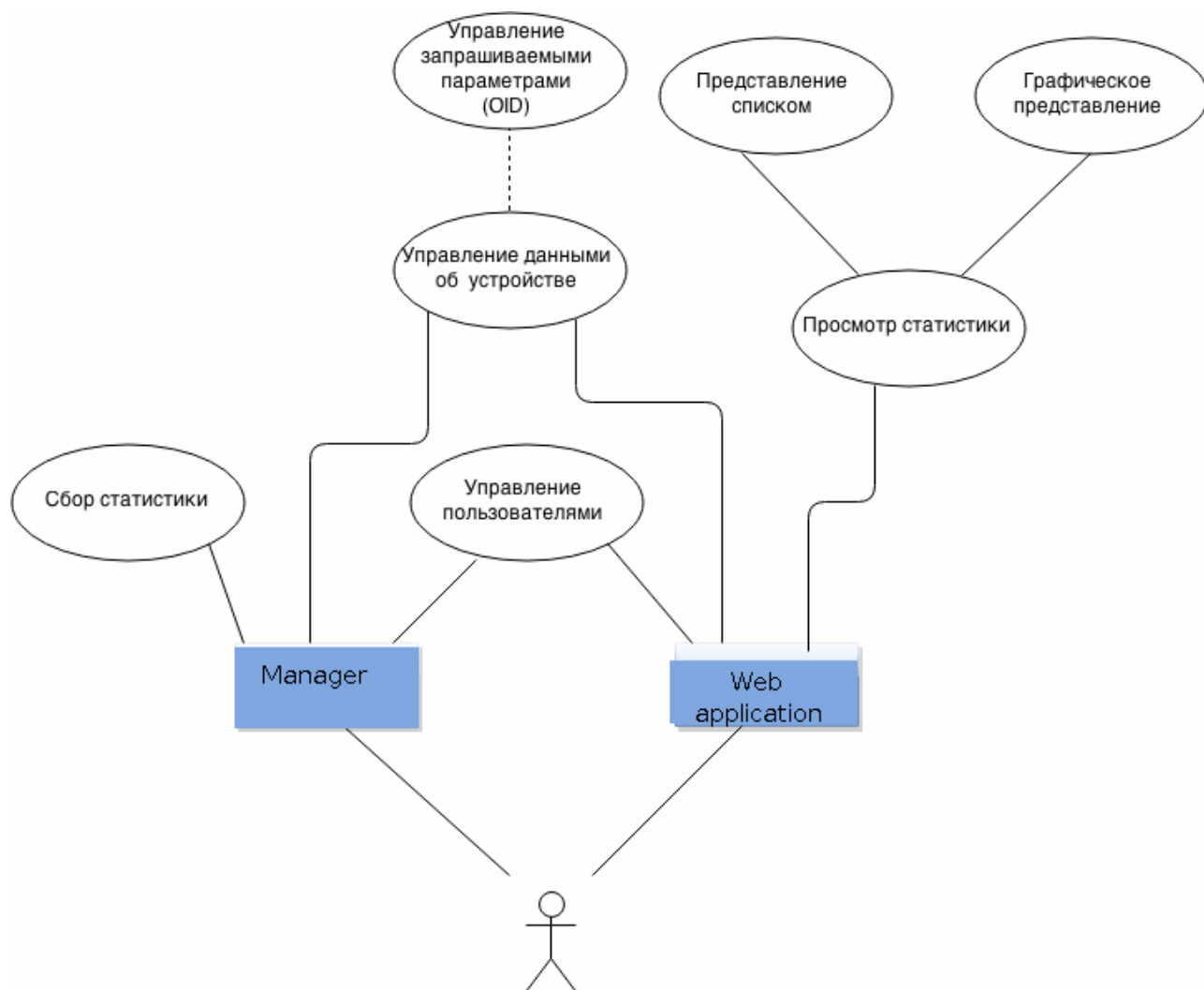


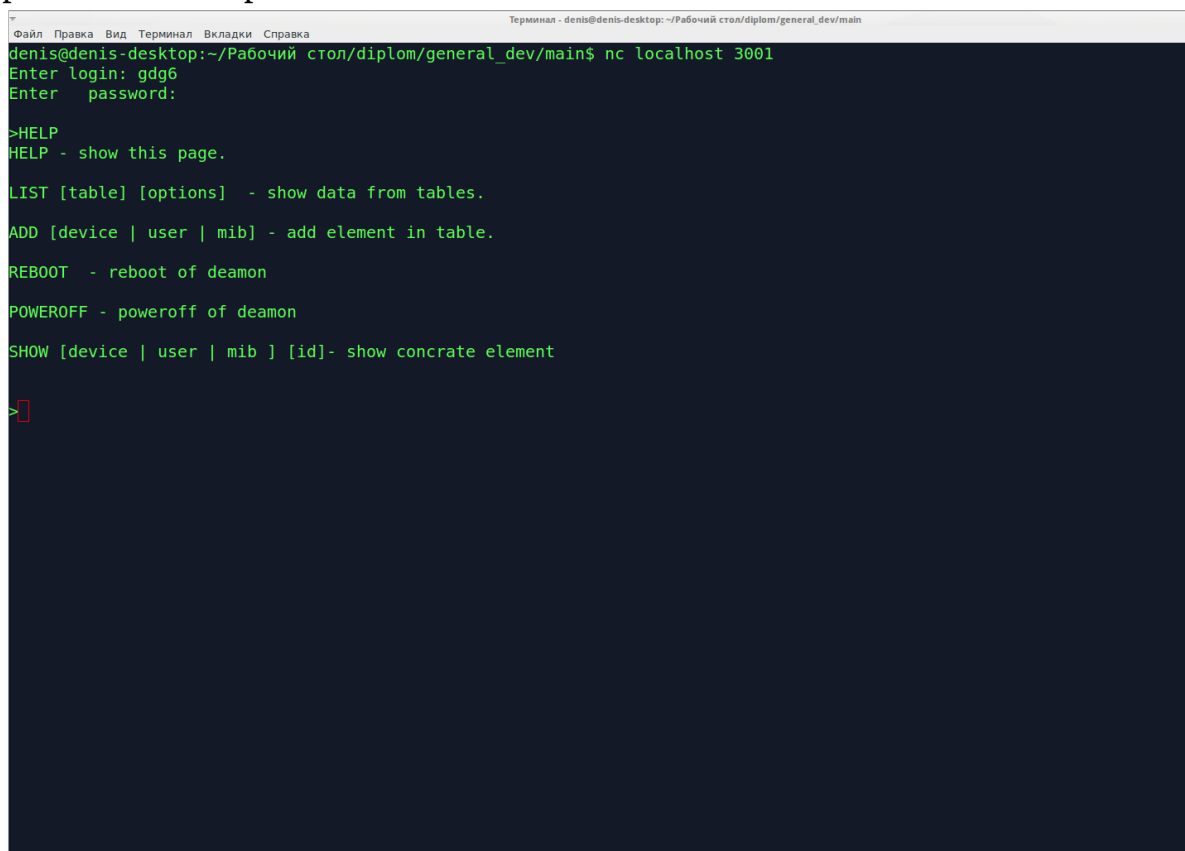
Рис. 3 Примерная диаграмма использования

### 3. Описание пользовательских интерфейсов

В результате проделанной работы был разработан менеджер для опроса устройств по SNMPv3, со встроенным сервером для принятия пользовательских соединений с авторизацией. С разделяемой базой данных между менеджером и Web приложением.

#### 3.1 Фоновое приложение

Так как приложение для сбора данных работает в фоновом режиме и все данные записывает в базу данных, то нет возможности визуально отобразить процесс работы, только если как в диспетчере задач. Конечно же процесс работы приложения можно видеть по результатам — записям в базе данных. Данные из которой можно отобразить и через браузер, используя WEB интерфейс. Так же есть возможность взаимодействия пользователя с приложением через CLI:



```
Файл Правка Вид Терминал Вкладки Справка
Терминал - denis@denis-desktop: ~/Рабочий стол/diplom/general_dev/main
denis@denis-desktop:~/Рабочий стол/diplom/general_dev/main$ nc localhost 3001
Enter login: gdg6
Enter password:

>HELP
HELP - show this page.

LIST [table] [options] - show data from tables.

ADD [device | user | mib] - add element in table.

REBOOT - reboot of daemon

POWEROFF - poweroff of daemon

SHOW [device | user | mib ] [id]- show concrete element

>
```

рис. 4 Взаимодействие с приложением через CLI

### 3.2 WEB приложение

Данное приложение реализовано с помощью фреймворка Ruby on Rails. Его архитектура основана на шаблоне проектирования MVC. Коротко описывая проделанную работу, необходимо сказать о том что были сделаны миграции, для взаимодействия с существующей базой данных. Валидации на уровне моделей. Для сессий использовались определенные дополнительные пакеты и плагины.

Для создания интерфейсов были использованы современные технологии разработки web-приложений такие как: HTML5, CSS3, jQuery. Так же был использован пакет net-snmp для возможности работы с устройствами через Web интерфейс. Перед работой в системе пользователю необходимо пройти авторизацию.

Авторизация

Login  
Denis

Password  
\*\*\*\*\*

Войти

Рис. 5 Авторизация пользователя при входе в систему.

Устройства

Тип статистики

Журнал событий

Пользователи

Выход

Назад

# Создание устройства

Название устройства

ice11

Краткое описание

2204

Комната

2204

Mac

bc:5f:f4:78:ba:0a

Serial number

00000000

Model

ASUS

Peername

192.168.5.11

Port

161

Login

snmpuser

Password

\*\*\*\*\*

Private password

\*\*\*\*\*

Сохранить

Back

Рис. 6 Добавление нового устройства в систему

При запуске приложения инициализируются устройства. Причем нужно подметить, что работа с устройствами производится при наличии двух условий:

1. Устройство доступно и соединение возможно
2. Есть набор команд по которым необходимо собирать статистику. Это логично, так как незачем работать с устройствами, если с ними ничего не надо делать.

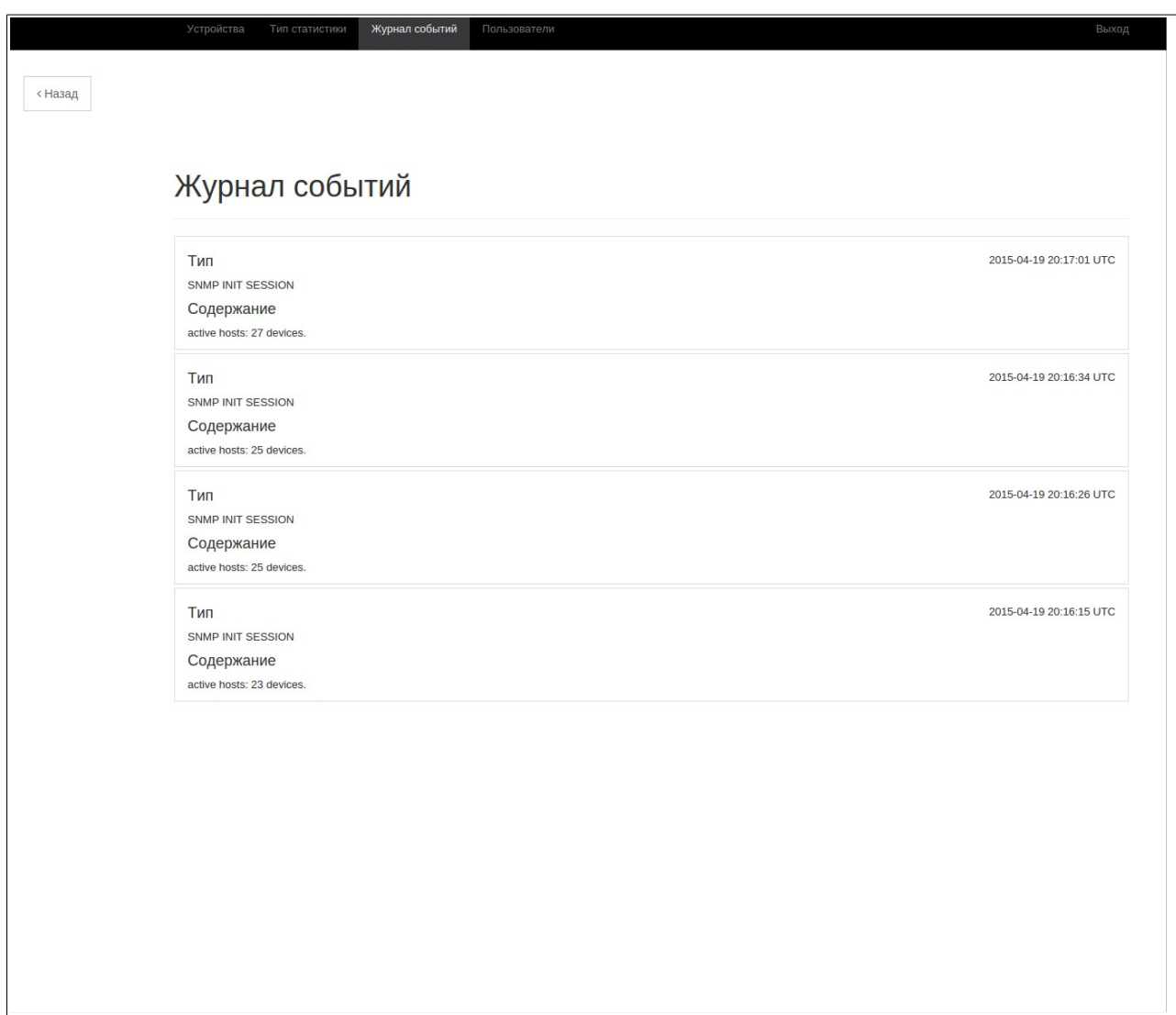


Рис.7 В журнале можно наблюдать успешный запуск менеджера

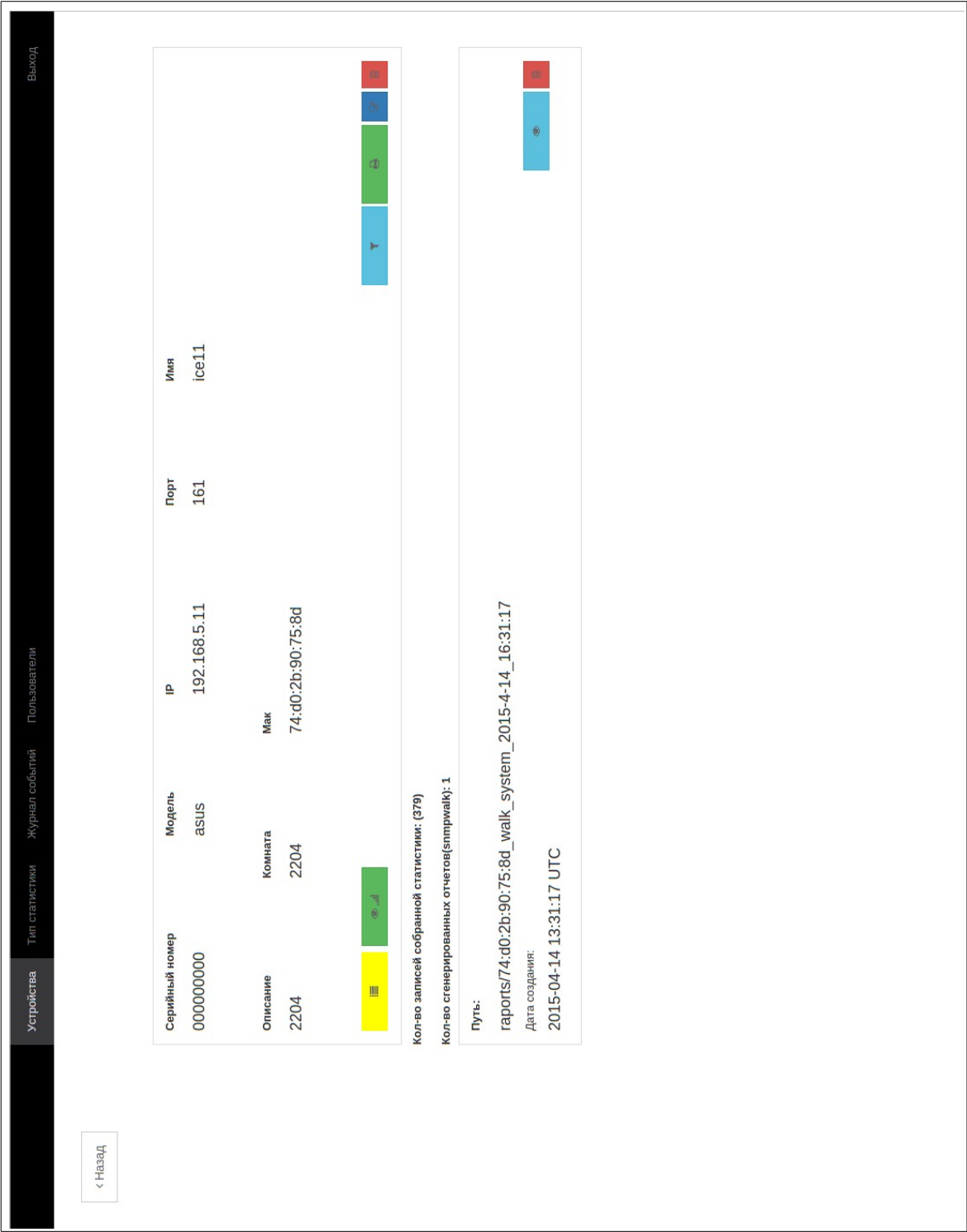


Рис. 8 Интерфейс устройствам

Устройство имеет интерфейс для просмотра и настройки набора Oid команд, просмотра статистики, и если есть необходимость узнать определенный атрибут без перезапуска менеджера, то имеется возможность разового запроса к устройству. Для генерации отчета по стандартизированным командам предусмотрено отдельное меню. А так же есть возможность редактирования и удаления устройства. На изображении можно видеть короткую информацию, говорящую о общем количестве собранной статистики, а так же список сгенерированных отчетов. Так как отчетов может быть большое количество, то интерфейс предоставляет постраничную навигацию.

The screenshot shows a web application interface for editing Oid commands. At the top, there is a dark navigation bar with links: 'Устройства' (Devices), 'Тип статистики' (Type of statistics), 'Журнал событий' (Event log), 'Пользователи' (Users), and 'Выход' (Logout). Below the navigation bar, on the left, is a button labeled '< Назад' (Back). The main content area is titled 'Редактирование oid' (Editing oid). It contains four input fields: 'Oid' with the value '.1.3.6.1.4.1.2021.4.11.0', 'Значение' (Value) with the text 'Свободная память' (Free memory), and 'Таймаут запроса (в секундах)' (Request timeout (in seconds)) with the value '15'. Below these fields is a checkbox labeled 'Активный' (Active) which is checked. At the bottom of the form are two buttons: 'Сохранить' (Save) in blue and 'Show' in a smaller font.

Рис. 9 Интерфейс для добавления/редактирования oid команды для устройства



Рис. 10 Набор oids для каждого устройства



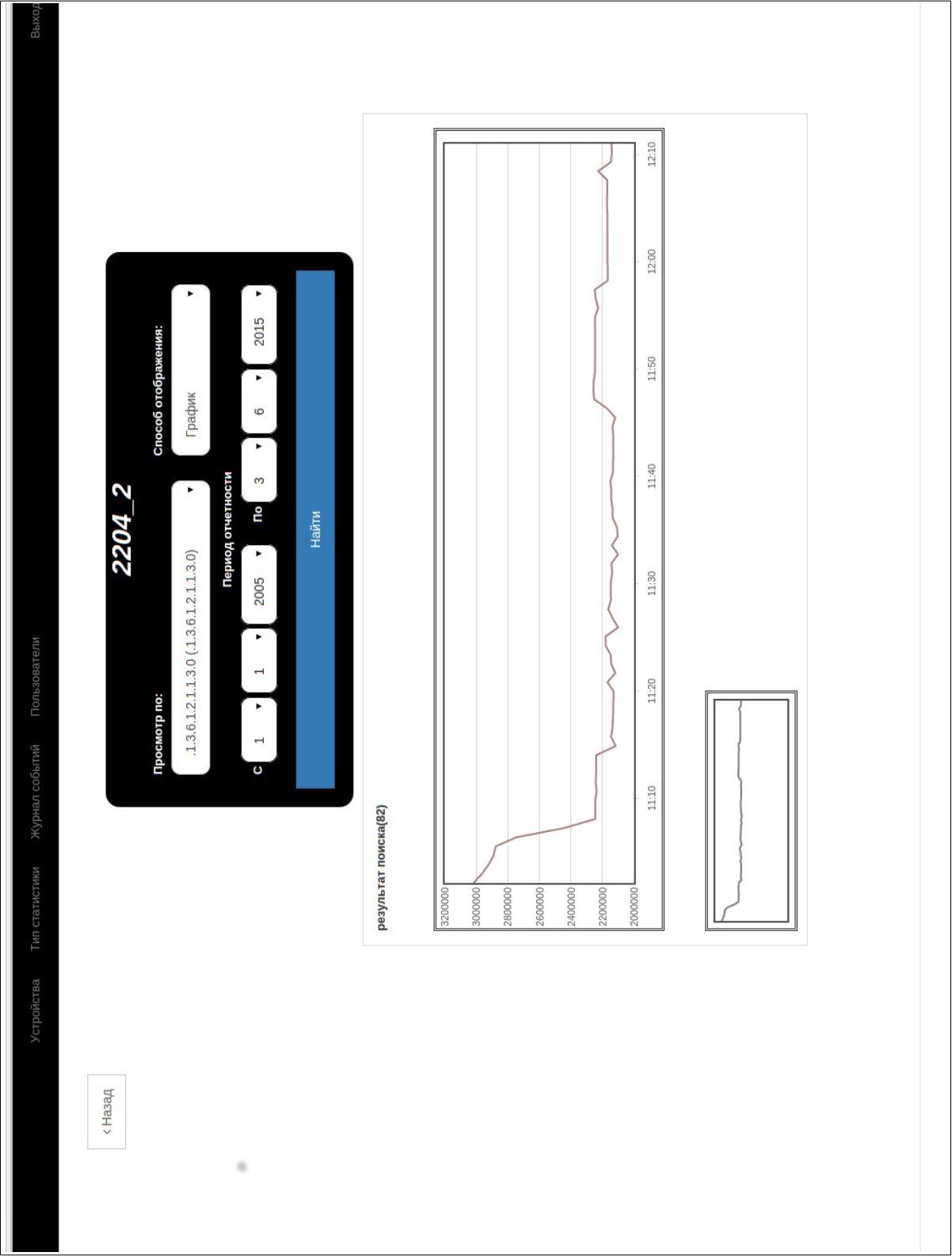


Рис. 11 Просмотр статистики

Данный интерфейс позволяет произвести быстрый анализ необходимых параметров каждого из устройств. Он предоставляет возможность навигации по типам запроса с выборкой за определенный интервал времени, а также возможность предоставления как в графическом, так и в виде списка.

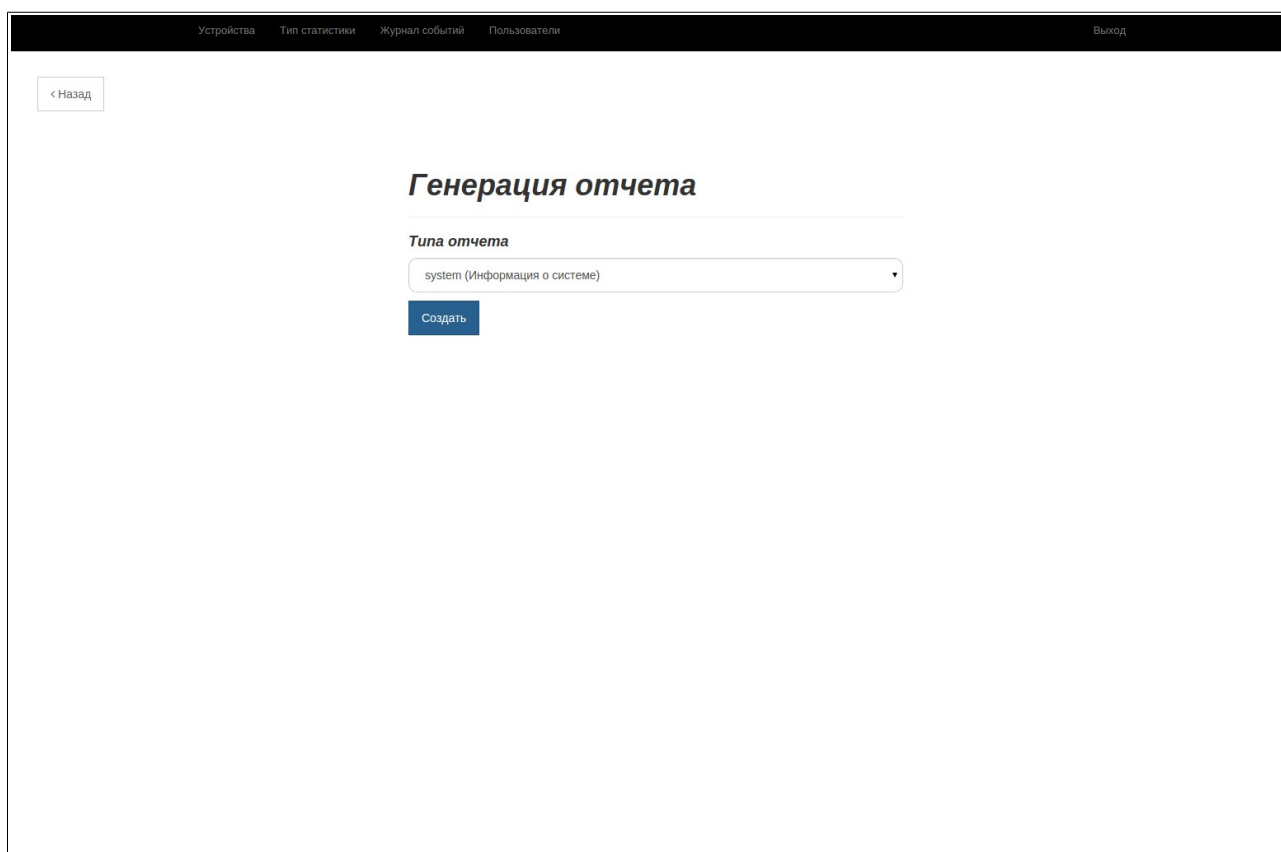


Рис. 12 Генерация отчета

На данном изображении показан интерфейс для генерации отчета, тип которого представляется в виде отдельного Oid. Данный набор команд является общепринятым, поддерживаемый net-snmp. Примером может служить команда system, позволяющая получить полный список атрибутов и их значений.

Устройства	Тип статистики	Журнал событий	Пользователи	Выход
####				
Create by gdbg 2015-04-14 16:31:17 +0300				
snmpwalk -v 3 -a md5 -A password -x des -X password -u snmpuser -I priv localhost161 system				
####				
===				
key: 1.3.6.1.2.1.1.1.0				
value: Linux denis-desktop 3.13.0-48-generic #80-Ubuntu SMP Thu Mar 12 11:16:15 UTC 2015 x86_64				
===				
key: 1.3.6.1.2.1.1.2.0				
value: 1.3.6.1.4.1.8072.3.2.10				
===				
key: 1.3.6.1.2.1.1.3.0				
value: 14296				
===				
key: 1.3.6.1.2.1.1.4.0				
value: @no.where				
===				
key: 1.3.6.1.2.1.1.5.0				
value: denis-desktop				
===				
key: 1.3.6.1.2.1.1.6.0				
value: Unknown				
===				
key: 1.3.6.1.2.1.1.8.0				

Рис. 13 Пример полученного отчета.

## 4. Заключение

В ходе реализации выпускной квалификационной работы была разработана система для сбора информации о состоянии узлов локальной вычислительной сети по SNMP v3. Это позволило просматривать удаленно такие характеристики устройств как температура, время работы, нагрузка центрального процессора.

Были решены следующие задачи:

1. Проанализирована существующая локально-вычислительная информационная сеть «МГИУ»;
2. Изучен SNMP;
3. Выбрана архитектура системы;
4. Определены модели системы, разработана структура базы данных;
5. Разработан интерфейс визуализации отчетов;
6. Разработан интерфейс управления устройствами.

Система включает в себя две основные программы:

1. Фоновый менеджер для сбора статистики с возможностью подключения через CLI по TCP/IP с авторизацией;
2. Web приложение, разработанное с помощью фреймворка Ruby on Rails.

Разработанная система имеет весь необходимый функционал для сбора и анализа данных с элементов локальной вычислительной сети «МГИУ». Наиболее приоритетным путем развития системы является – доработка существующих моделей, интерфейса пользователя и возможность просмотра состояния устройств в реальном времени. Это позволит получать актуальные данные без необходимости дополнительных операций со стороны пользователя.

## 5. Список используемой литературы

- [1] Ресурс посвященный Zabbix [электронный ресурс] -  
<http://www.zabbix.com/ru/>
- [2] Официальный сайт SNMP FREE MANAGER [электронный ресурс] -  
<http://www.dart.com/snmp-free-manager.aspx>
- [3] Ресурс посвященный SNMP [электронный ресурс] -  
<https://ru.wikipedia.org/wiki/SNMP>
- [4] Ресурс посвященный работе с NET-SNMP [электронный ресурс] -  
<http://www.net-snmp.org/>
- [5] Паттерн DAO [электронный ресурс] -  
[https://ru.wikipedia.org/wiki/Data\\_Access\\_Object](https://ru.wikipedia.org/wiki/Data_Access_Object)
- [6] Технология ORM [электронный ресурс] -  
<https://ru.wikipedia.org/wiki/ORM>
- [7] Ресурс посвященный Ruby on Rails [электронный ресурс] -  
<http://rusrails.ru/>

## 6. Приложение

### 6.1 Полная схема базы данных Web приложения

```
create_table "devices", force: true do |t|
  t.string "name"
  t.string "description", limit: 1024
  t.string "room", limit: 25
  t.string "mac"
  t.string "serial_number"
  t.string "model", limit: 1024
  t.string "peername"
  t.integer "port"
  t.string "login"
  t.string "password"
  t.string "priv_password", limit: 1024
  t.datetime "created_at"
  t.datetime "updated_at"
end
```

```
create_table "logs", force: true do |t|
  t.string "l_type"
  t.text "context"
  t.datetime "created_at"
  t.datetime "updated_at"
end
```

```
create_table "oids", force: true do |t|
  t.integer "device_id", null: false
  t.string "oid"
  t.string "translate"
  t.integer "ping_request", null: false
  t.boolean "active", null: false
  t.datetime "created_at"
  t.datetime "updated_at"
end
```

```
add_index "oids", ["device_id"], name: "index_oids_on_device_id"
```

```
create_table "reports", force: true do |t|
  t.integer "device_id"
  t.string "r_type"
  t.text "context"
  t.datetime "created_at"
  t.datetime "updated_at"
end
```

```
create_table "users", force: true do |t|
  t.string "login"
  t.string "password_digest"
  t.integer "role"
  t.datetime "created_at"
  t.datetime "updated_at"
end
```

```
create_table "walk_reports", force: true do |t|
  t.string "path"
  t.integer "device_id", null: false
  t.datetime "created_at"
  t.datetime "updated_at"
end
```

```
add_index "walk_reports", ["device_id"], name:
"index_walk_reports_on_device_id"
```

```
create_table "walk_requests", force: true do |t|
  t.string "request"
  t.string "description"
  t.datetime "created_at"
  t.datetime "updated_at"
end
```