

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**  
по направлению 230100 «Информационные системы и технологии»

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

Студент \_\_\_\_\_/Грачев Д.Г.  
Руководитель работы \_\_\_\_\_/Курасов Ю.В.

**ДОПУСКАЕТСЯ К ЗАЩИТЕ**

Зав. каф. 36, доцент, к.ф.-м.н. \_\_\_\_\_/Роганов Е.А./

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 технологий**

Зав. Кафедрой 36  
\_\_\_\_\_/Роганов Е.А. /  
« \_\_\_\_ » \_\_\_\_\_ 20\_\_ г.

**Задание**

на выпускную работу по направлению «Информационные системы и технологии»

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. Исходные данные.
  2. Цель работы — разработка системы мониторинга локальной вычислительной сети.
  3. Содержание дипломной работы:
    1. Введение.
    2. Литературный обзор.
    3. Структура системы.
    4. Пользовательский интерфейс.
  4. Используемые технологии: C++, Sqlite3, SNMP, JavaScript, JQuery, HTML5, CSS3, Ruby, Ruby on Rails.
  5. Практическая реализация: данная работа может найти применение в БТО ИВЦ «МГИУ».
  6. Графическая часть:
    1. Диаграмма использования.
    2. Диаграмма классов.
    3. Реализация интерфейса.

Руководитель \_\_\_\_\_ ( \_\_\_\_\_ )

Дипломник \_\_\_\_\_ ( \_\_\_\_\_ )

## 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 демоном 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.

## Содержание

1. Введение .....	5
2. Литературный обзор .....	7
1. Постановка задачи .....	7
2. Обзор существующих решений .....	8
3. Реализация задачи .....	13
3. Проектирование системы. ....	17
1. Описание пользовательских интерфейсов .....	25
1. Демон .....	25
2. Web интерфейс .....	26
4. Заключение .....	35
5. Список используемой литературы .....	36
6. Приложение .....	37
1. Полная схема базы данных Web приложения .....	38

# Введение

В сложившемся современном мире усложняются информационные системы. Вместе с ними возрастают требования к вычислительным сетям, благодаря которым происходит передача и обработка данных. В нынешнее время для успешной работы компаний, организаций необходима стабильная информационная система. Так как 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, интуитивно-понятной системы для мониторинга локальной вычислительной системы в рамках ИВЦ ФГБОУ ВПО «МГИУ».

# Литературный обзор

## 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.

## Обзор существующих решений

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

Один из самых известных и наиболее мощных является Zabbix.

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

*Структура:*

- **Zabbix сервер** — это ядро программного обеспечения Zabbix. Сервер может удаленно проверять сетевые сервисы, является хранилищем, в котором хранятся все конфигурационные, статистические и оперативные данные, и он является тем субъектом в программном обеспечении Zabbix, который оповестит администраторов в случае возникновения проблем с любым 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 идеальным решением для централизованного удаленного мониторинга мест, филиалов, сетей, не имеющих локальных администраторов. Он может быть также использован для распределения нагрузки одного Zabbix сервера. В этом случае, прокси только собирает данные, тем самым на сервер ложится меньшая нагрузка на ЦПУ и на ввод-вывод диска.
- **Zabbix агент** — контроль локальных ресурсов и приложений (таких как жесткие диски, память, статистика процессора и т. д.) на сетевых системах, эти системы должны работать с запущенным Zabbix агентом. Zabbix агенты являются чрезвычайно эффективными из-за использования родных системных вызовов для сбора информации о статистике.



- **Веб-интерфейс** — интерфейс является частью Zabbix сервера, и, как правило (но не обязательно), запущен на том же физическом сервере, что и Zabbix сервер. Работает на PHP, требует веб сервер (напр. Apache). Неочевидной 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 веб-интерфейса оказывается просто 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 совершенно произвольным образом.

#### *Обзор возможностей:*

- Распределенный мониторинг вплоть до 1000 узлов. Конфигурация младших узлов полностью контролируется старшими узлами, находящимися на более высоком уровне иерархии;
- Сценарии на основе мониторинга;
- Автоматическое обнаружение;
- Централизованный мониторинг лог-файлов;
- Веб-интерфейс для администрирования и настройки;
- Отчетность и тенденции;
- SLA мониторинг;

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

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 ИВЦ «МГИУ».

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

Еще подходящим решением может служить **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. Идеально подходит для легких умеренных задач управления.

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

### Особенности:

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

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 интуитивно-понятный интерфейс, но имеющий достаточный функционал для анализа полученных данных.

## Реализация задачи

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 является 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.

## Требования к системе

К приложению 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 — не менее 1000;
7. Возможность создания отдельных запросов к устройству;
8. Необходимо иметь возможность подключение по TCP/IP к демону. (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).

## Обзор используемых технологий

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

### Ruby on Rails

**Ruby on Rails** - фреймворк, написанный 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. Данный фреймворк содержит в себе достаточно различных пакетов, которые нужны каждому разработчику для создания нового проекта, что позволяет при меньших усилиях достигать решения поставленной задачи в короткие сроки.

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 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 несанкционированную опасность, доступом к связанную с возможным конфиденциальной информации.

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, удобно разместить его в отдельном файле и затем сослаться на этот файл. Это целесообразно 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. Стандартная библиотека включает, в том числе, общеупотребительные контейнеры и алгоритмы. C ++ сочетает свойства как высокоуровневых, так и низкоуровневых языков

## **SNMP**

**SNMP** (англ *Simple Network Management Protocol* . - Простой протокол сетевого управления) - 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. Протокол обычно используется в системах сетевого управления для контроля подключенных к сети устройств на предмет условий, которые требуют внимания администратора. SNMP определен Инженерным советом интернета (IETF) как компонент TCP / IP. Он состоит из набора стандартов для сетевого управления, включая протокол прикладного уровня, схему баз данных и набор объектов данных.

## **Net-snmp**

**Net- SNMP** представляет собой набор программного обеспечения для развёртывания и использования протокола SNMP (v1, v2c и v3 и протокол AgentX субагента ) . Он поддерживает IPv4 , IPv6 , IPX , AAL5 , сокеты доменов Unix и других протоколов . Он содержит общие клиентские библиотеки , набор консольных приложений , расширяемый SNMP - агент , модули Perl и модули Python .

## **Bcrypt**

**Bcrypt** - адаптивная криптографическая хеш - функция , используемая для защищенного хранения паролей .

## **UML**

**UML** ( англ Unified Modeling Language - Унифицированный язык моделирования ) - язык графического описания для объектного моделирования в области разработки программного обеспечения . UML является языком широкого профиля , это - открытый стандарт , использующий графические обозначения для создания абстрактной модели системы , называемой UML - моделью . UML был создан для определения , визуализации , проектирования и документирования , в основном , программных систем . UML не является языком программирования , но на основании UML - моделей возможна генерация кода .

## Проектирование системы

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:

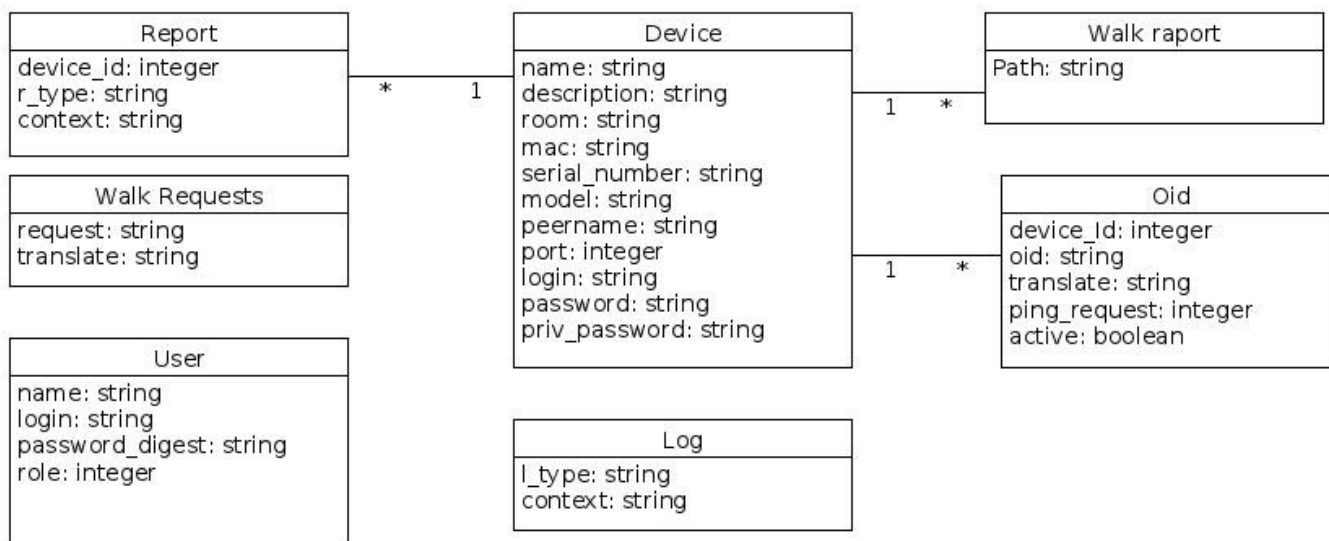


Рис. 1 Диаграмма классов

**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 snmp демону на устройстве. Так как менеджер будет работать по SNMPv3, то для авторизации необходимы login (имя пользователя), passsword 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 зарпоса.

**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 Raport, 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.

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

**DAO** — это объект, который предоставляет абстрактный интерфейс к какому-либо типу базы данных или механизму хранения. Осталось только изобразить конечный вид структуры приложения.



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 ТСР/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.

Примерно представить возможное взаимодействие можно следующим способом:

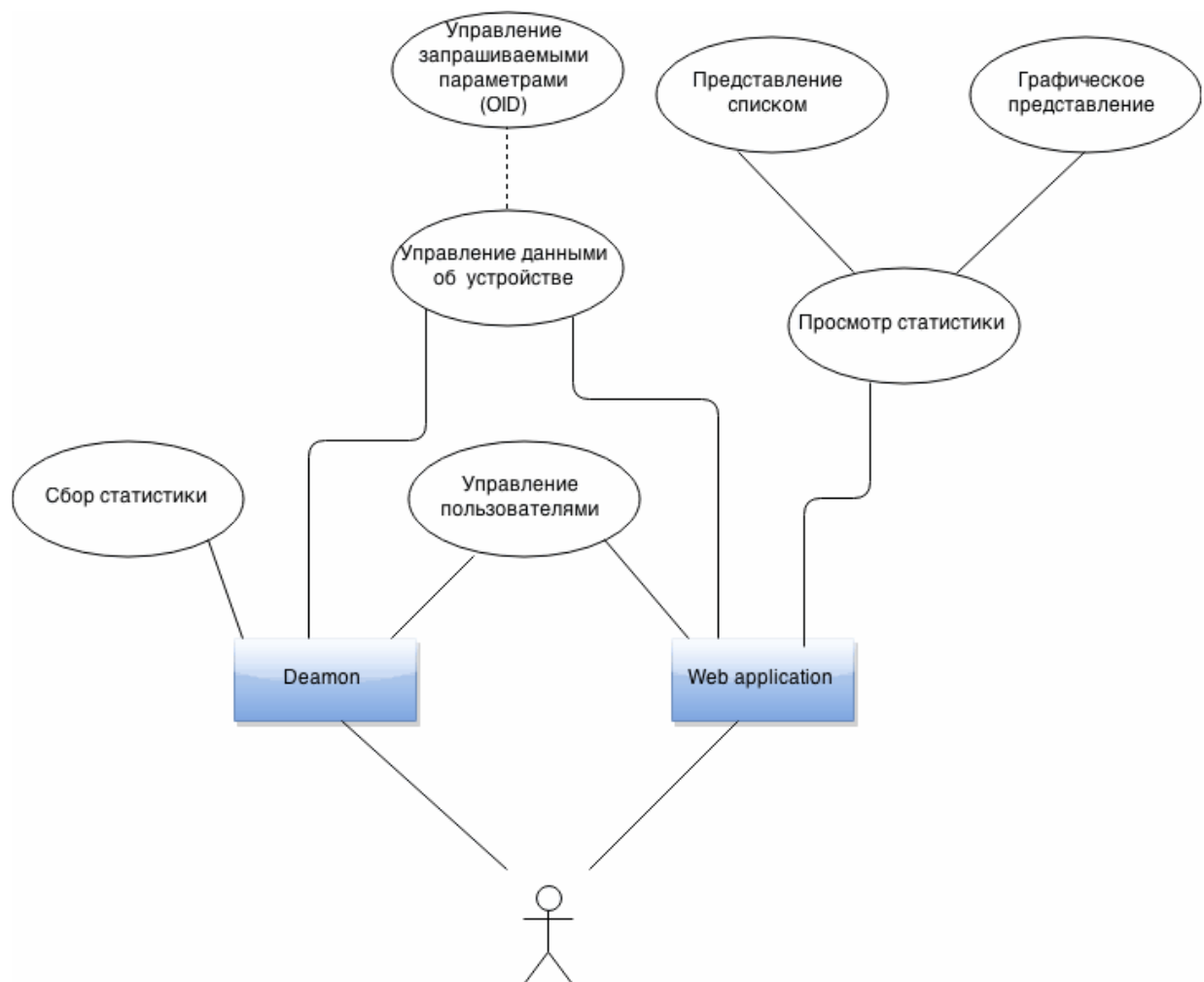


Рис. 3 Примерная диаграмма использования

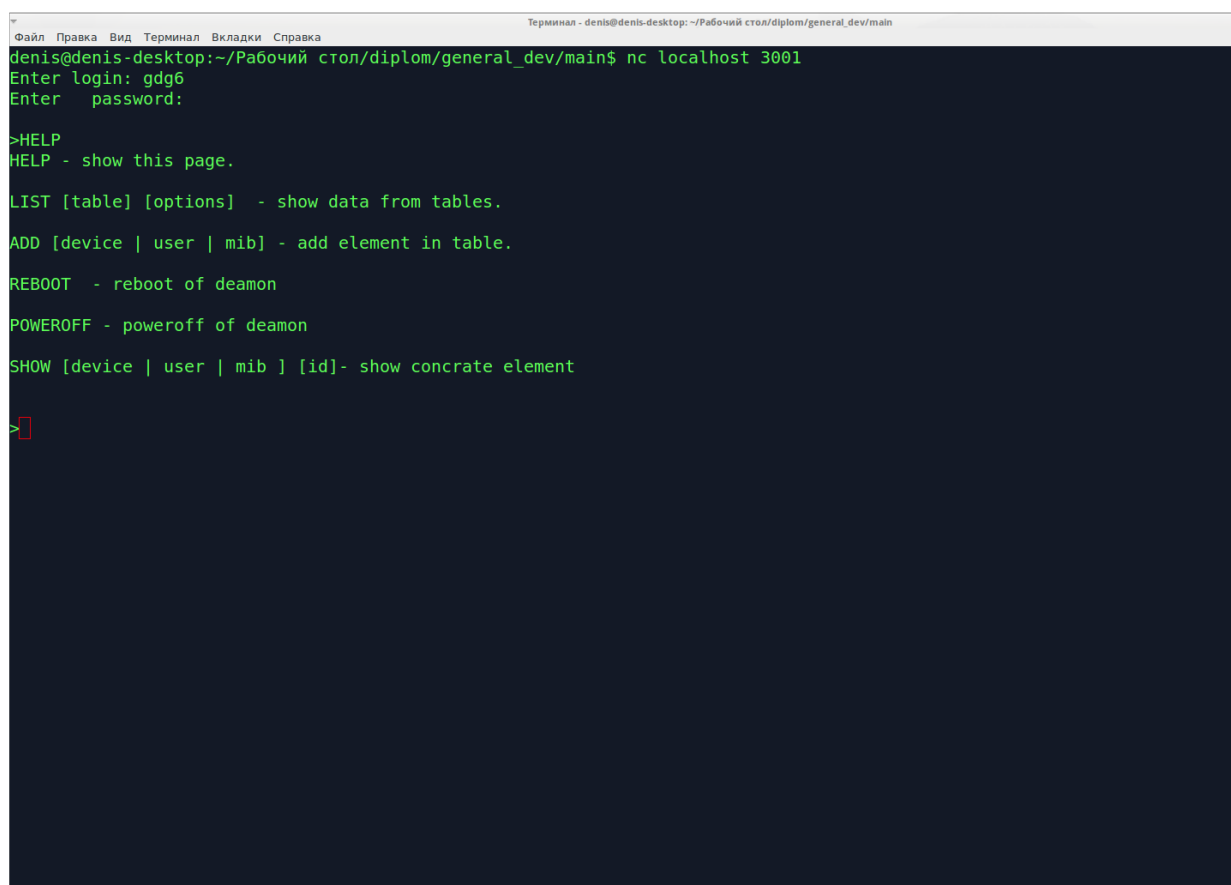


## Описание пользовательских интерфейсов

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.

### Демон

Так как приложение переходит в фоновый режим 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:

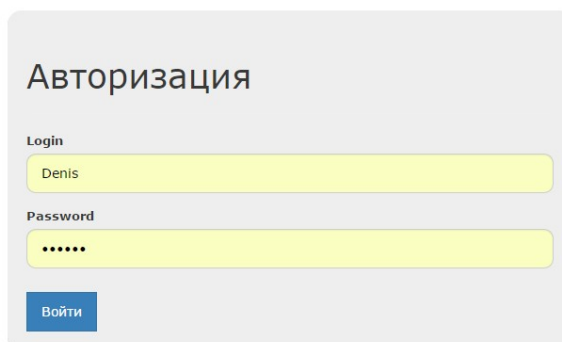


```
Терминал - denis@denis-desktop: ~/Рабочий стол/diplom/general_dev/main
denis@denis-desktop:~/Рабочий стол/diplom/general_dev/main$ nc localhost 3001
Enter login: gdg6
Enter password:
>HELP
HELP - show this page.
LIST [table] [options] - show data from tables.
ADD [device | user | mib] - add element in table.
REBOOT - reboot of daemon
POWEROFF - poweroff of daemon
SHOW [device | user | mib] [id] - show concrete element
>
```

рис. 4 Взаимодействие с демоном через CLI

## WEB приложение

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



The image shows a web form titled "Авторизация" (Authorization). It contains two input fields: "Login" with the text "Denis" and "Password" with masked characters "\*\*\*\*\*". Below the fields is a blue button labeled "Войти" (Login).

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:14:78:ba:0a

Serial number

00000000

Model

ASUS

Replname

192.168.5.11

Port

161

Login

snmpuser

Password

\*\*\*\*\*

Private password

\*\*\*\*\*

Сохранить

Back

Рис. 6 Добавление нового устройства в систему

При запуске демона 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 зачем работать с устройствами, если с ними нечего делать.

Устройства

Тип статистики

Журнал событий

Пользователи

Выход

Назад

### Журнал событий

Тип SNMP INIT SESSION Содержание active hosts: 27 devices.	2015-04-19 20:17:01 UTC
Тип SNMP INIT SESSION Содержание active hosts: 25 devices.	2015-04-19 20:16:34 UTC
Тип SNMP INIT SESSION Содержание active hosts: 25 devices.	2015-04-19 20:16:26 UTC
Тип SNMP INIT SESSION Содержание active hosts: 23 devices.	2015-04-19 20:16:15 UTC

Рис.7 В журнале можно наблюдать успешный запуск менеджера

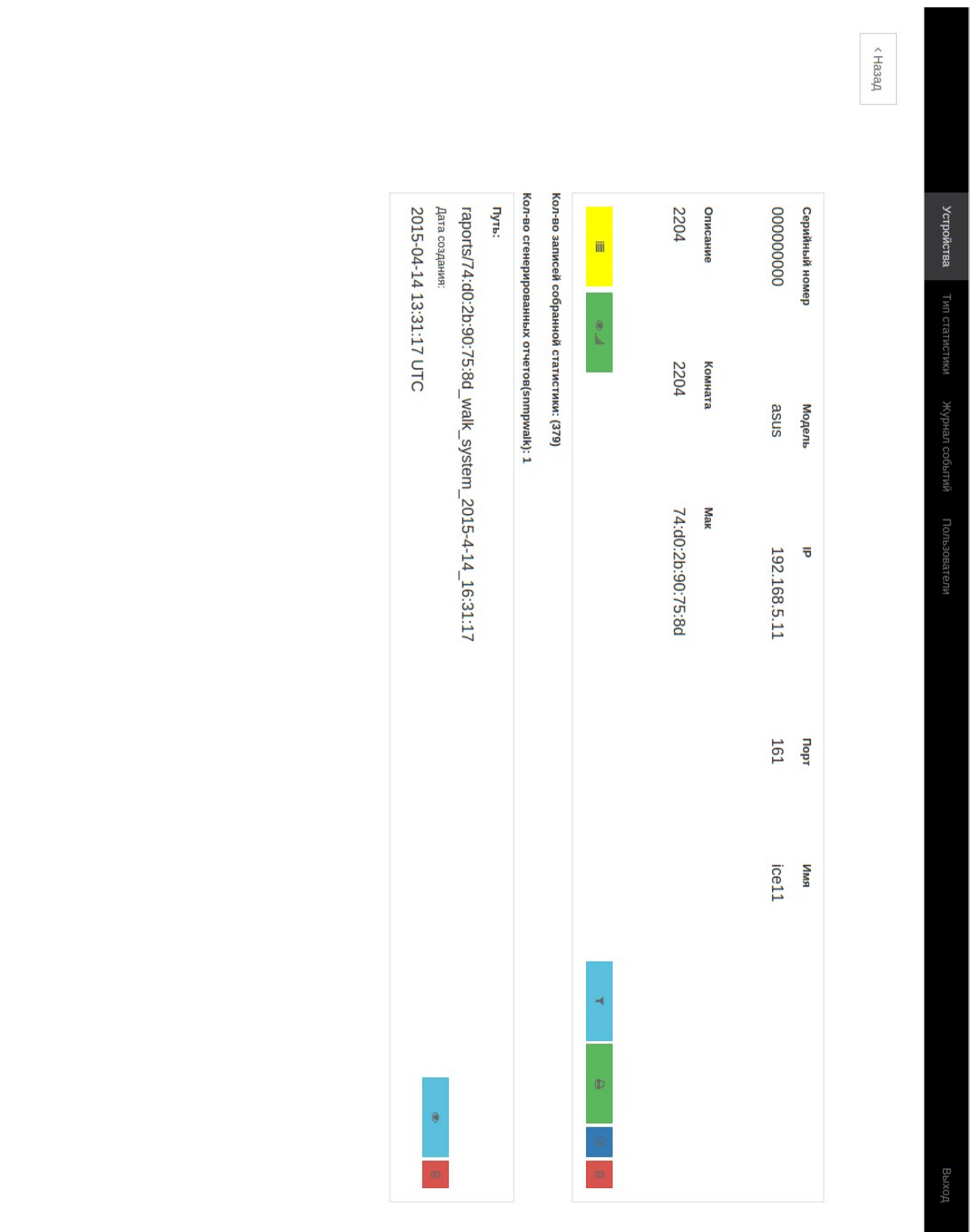


Рис. 8 Интерфейс устройства

Устройство имеет интерфейс для просмотра и настройки набора Oid команд, просмотра статистики, и если есть необходимость узнать определенный атрибут без перезапуска демона, 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 информации, говорящая о общем количество собранной статистики, а так же список сгенерированных отчетов. Так как отчетов может быть большое количество, то интерфейс предоставляет постраничную навигацию.

[Устройства](#) [Тип статистики](#) [Журнал событий](#) [Пользователи](#) [Выход](#)

[< Назад](#)

### Редактирование oid

Oid

Значение

Таймаут запроса (в секундах)

Активный






☒

Сохранить

Show

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

Список Oids

Описание: ".1.3.6.1.2.1.1.3.0"	Old: .1.3.6.1.2.1.1.3.0	Active: Неактивен	Отправлять каждые: 20 секунд	 
Описание: "PERCENTAGE_USER_CPU_TIME"	Old: .1.3.6.1.4.1.2021.11.9.0	Active: Активен	Отправлять каждые: 15 секунд	 
Описание: "TOTAL_SWAP_SIZE"	Old: .1.3.6.1.4.1.2021.4.3.0	Active: Активен	Отправлять каждые: 90 секунд	 
Описание: "Свободная память"	Old: .1.3.6.1.4.1.2021.4.11.0	Active: Активен	Отправлять каждые: 50 секунд	 

+ Добавить

Рис.10 Набор oid для каждого устройства

[← Назад](#)

2204\_2

Просмотр по:

Способ отображения:

13.6.12.113.0 (13.6.12.113.0)

График

Период отчетности

с 1 1 2005 по 3 6 2015

Найти

результат поиска(82)

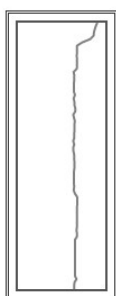
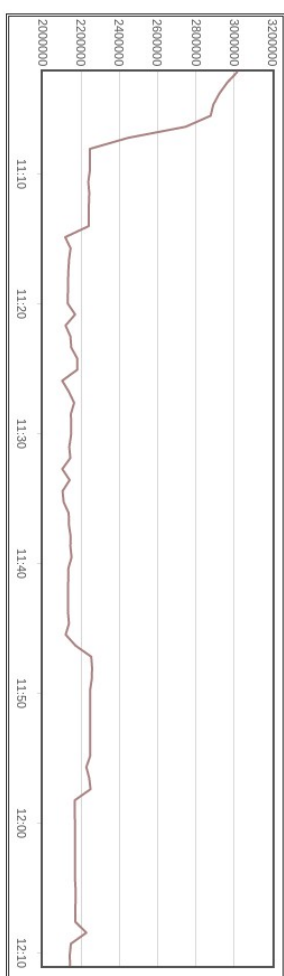


Рис. 11 Просмотр статистики



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.

Устройства Тип статистики Журнал событий Пользователи Выход

< Назад

### Генерация отчета

Тип отчета

system (Информация о системе)

Создать

Рис. 12 Генерация отчета

На данном изображении показан интерфейс для генерации отчета, тип которого представляется в виде отдельного Oid.

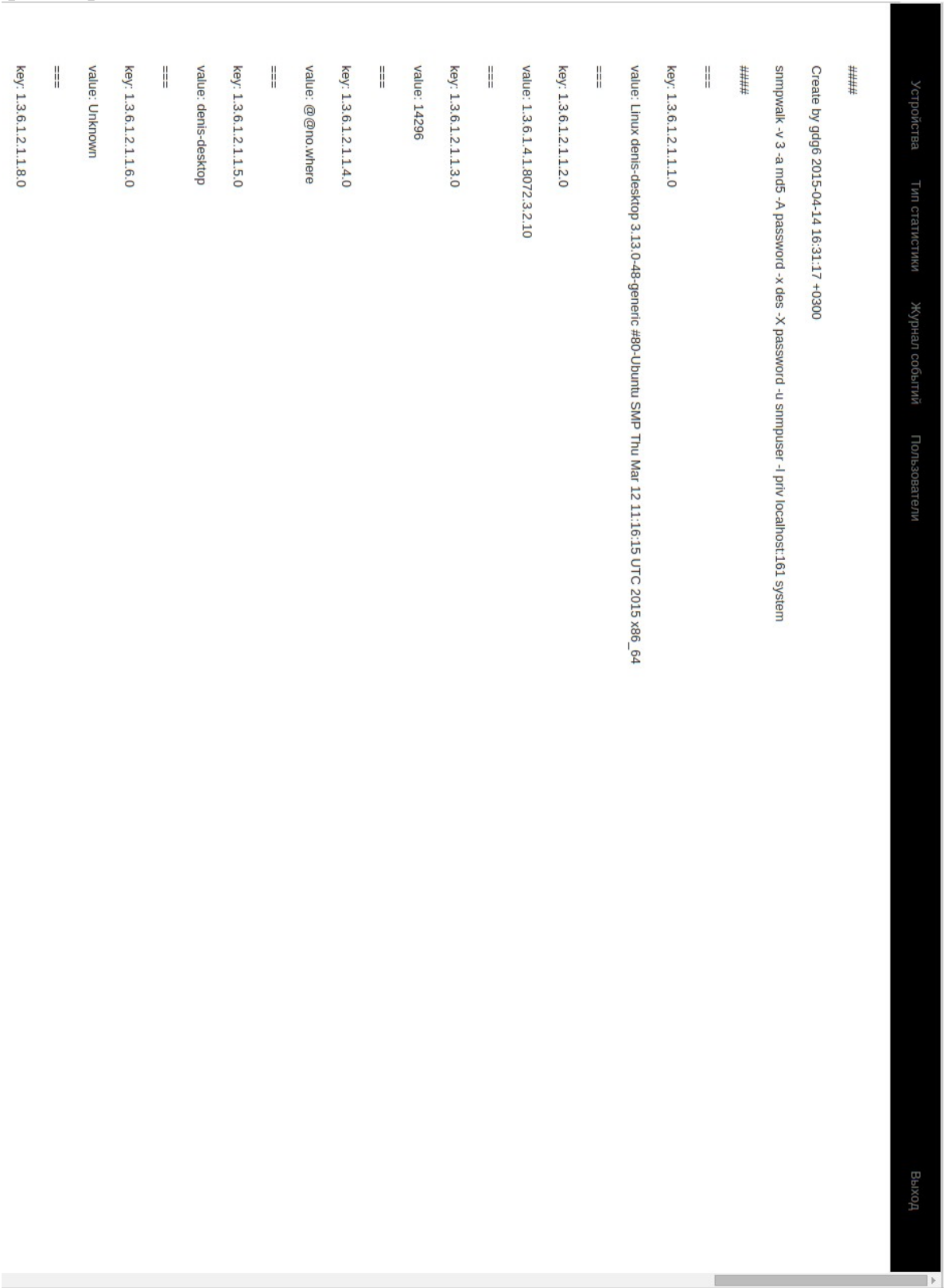


Рис. 13 Пример полученного отчета.

## 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 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.

## **Список используемой литературы**

1. [https://ru.wikipedia.org/wiki/Data\\_Access\\_Object](https://ru.wikipedia.org/wiki/Data_Access_Object)
2. <https://ru.wikipedia.org/wiki/ORM>
3. <https://ru.wikipedia.org/wiki/Zabbix>
4. <http://www.zabbix.com/ru/> - ресурс посвященный Zabbix
5. <http://www.net-snmp.org/> - ресурс посвященный библиотеке для работы с
6. SNMP.
7. <http://www.dart.com/snmp-free-manager.aspx> — ресурс посвященный SNMP  
FREE MANAGER.
8. <https://www.google.ru/> - поисковая система google.

## Приложение

### 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
```