|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство образования и науки Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_\_\_\_«Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_\_«Системы обработки информации и управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ***

***НА ТЕМУ:***

*Подсистема сбора и обработки данных технических параметров конечных интернет-пользователей \_*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Студент \_\_\_ИУ5-84\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_**И.А. Матвейчук **\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель ВКР **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_**В.А. Галкин **\_\_**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Нормоконтролер **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2018 г.*

Задание

Аннотация

Объектом разработки является подсистема сбора и обработки данных технических параметров конечных интернет пользователей.

Квалификационная работа на тему «Подсистема сбора и обработки данных технических параметров конечных интернет пользователей» посвящена разработке подсистемы, позволяющей значительно упростить процесс анализа данных об интернет-соединении конечных пользователей.

Цель разработки подсистемы сбора и обработки данных технических параметров конечных интернет пользователей — автоматизация процесса обработки данных и сохранение их в базе данных.

В процессе выполнения квалификационной работы бакалавра было проведено исследование предметной области, анализ аналогов и прототипов, разработка структуры и архитектуры системы, построена инфологическая и даталогическая модель, разработан интерфейс пользователя, спроектирована подсистема сбора и обработки данных технических параметров конечных интернет пользователей.

Исследовательская часть квалификационной работы бакалавра включает раздел по моделированию данной подсистемы на языке GPSS. В экономической части представлены расчеты себестоимости и цена решения.

Содержание

[Задание 2](#_Toc515895589)

[Аннотация 3](#_Toc515895590)

[Содержание 4](#_Toc515895591)

[Обозначения и сокращения 6](#_Toc515895592)

[Введение 7](#_Toc515895593)

[1. Конструкторско-технологическая часть 10](#_Toc515895594)

[1.1. Общетехническое обоснование разработки 10](#_Toc515895595)

[1.1.1. Постановка задачи проектирования 10](#_Toc515895596)

[1.1.2. Описание предметной области 10](#_Toc515895597)

[1.1.3. Перечень процессов подлежащих автоматизации 11](#_Toc515895598)

[1.1.4. Функциональные задачи системы 11](#_Toc515895599)

[1.1.5. Выбор и обоснование критериев качества 12](#_Toc515895600)

[1.1.6. Анализ аналогов и прототипов 13](#_Toc515895601)

[1.2. Алгоритм разработки программного обеспечения 17](#_Toc515895602)

[1.3. Разработка программного изделия 28](#_Toc515895603)

[1.3.1. Проектирование базы данных 28](#_Toc515895604)

[1.3.1.1. Инфологическая модель 28](#_Toc515895605)

[1.3.1.2. Выбор СУБД 38](#_Toc515895606)

[1.3.1.3. Даталогическая модель 40](#_Toc515895607)

[1.3.1.4. Расчет объема базы данных 44](#_Toc515895608)

[1.3.2. Выбор программных средств 47](#_Toc515895609)

[1.3.3. Выбор аппаратных средств 48](#_Toc515895610)

[1.3.4. Структура программного изделия 48](#_Toc515895611)

[1.3.5. Диаграмма классов 48](#_Toc515895612)

[1.3.5.1. Описание классов 55](#_Toc515895613)

[1.3.6. Программа и методика испытаний 59](#_Toc515895614)

[1.4. Разработка интерфейса взаимодействия с пользователем 62](#_Toc515895615)

[1.4.1. Граф диалога 63](#_Toc515895616)

[1.4.2. Экранные формы 64](#_Toc515895617)

[2. Научно-исследовательская часть 65](#_Toc515895618)

[2.1. Моделирование 65](#_Toc515895619)

[2.1.1. Параметризация системы 65](#_Toc515895620)

[2.1.2. Имитационная модель 66](#_Toc515895621)

[2.1.3. Результаты моделирования 67](#_Toc515895622)

[3. Экономическая часть 69](#_Toc515895623)

[3.1. Смета затрат на создание программного изделия 69](#_Toc515895624)

[3.2. Обоснование сметы затрат 69](#_Toc515895625)

[3.2.1. Расчет затрат на расходные материалы 69](#_Toc515895626)

[3.2.2. Расчёт затрат на программное обеспечение 70](#_Toc515895627)

[3.2.3. Расчёт затрат на оборудование 70](#_Toc515895628)

[3.2.4. Расчёт затрат на услуги сторонних организаций 71](#_Toc515895629)

[3.2.5. Расчёт заработной платы 71](#_Toc515895630)

[3.2.6. Расходы на дополнительную заработанную плату 72](#_Toc515895631)

[3.2.7. Расчет отчислений на социальные нужды 73](#_Toc515895632)

[3.2.8. Расчет налоговых выплат 73](#_Toc515895633)

[3.2.9. Расчет накладных расходов 74](#_Toc515895634)

[3.2.10. Расчет прочих расходов 74](#_Toc515895635)

[3.2.11. Расчет себестоимости 74](#_Toc515895636)

[3.2.12. Расчет прибыли 74](#_Toc515895637)

[3.2.13. Цена 75](#_Toc515895638)

[3.2.14. Договорная цена 75](#_Toc515895639)

[Заключение 76](#_Toc515895640)

[Список использованных источников 77](#_Toc515895641)

[Приложение 1. Техническое задание 79](#_Toc515895642)

[Приложение 2. Графическая часть 88](#_Toc515895643)

[Приложение 3. Текст программы 99](#_Toc515895644)

Обозначения и сокращения

ПО – программное обеспечение;

СУБД – средство управления базой данных;

ПО – программное обеспечение;

БД – база данных;

SQL – Structured Query Language;

GPSS – General Purpose Simulation System;

ICMP - Internet Control Message Protocol;

HTTP - HyperText Transfer Protocol.

QoE – Quality of Experience;

ПК – персональный компьютер4

QoS - Quality of Service;

SNMP - Simple Network Management Protocol;

CMIP - Common Management Information Protocol;

OID - Object Identifier;

MIB - Management Information Base;

TCP - Transmission Control Protocol;

UDP - User Datagram Protocol;

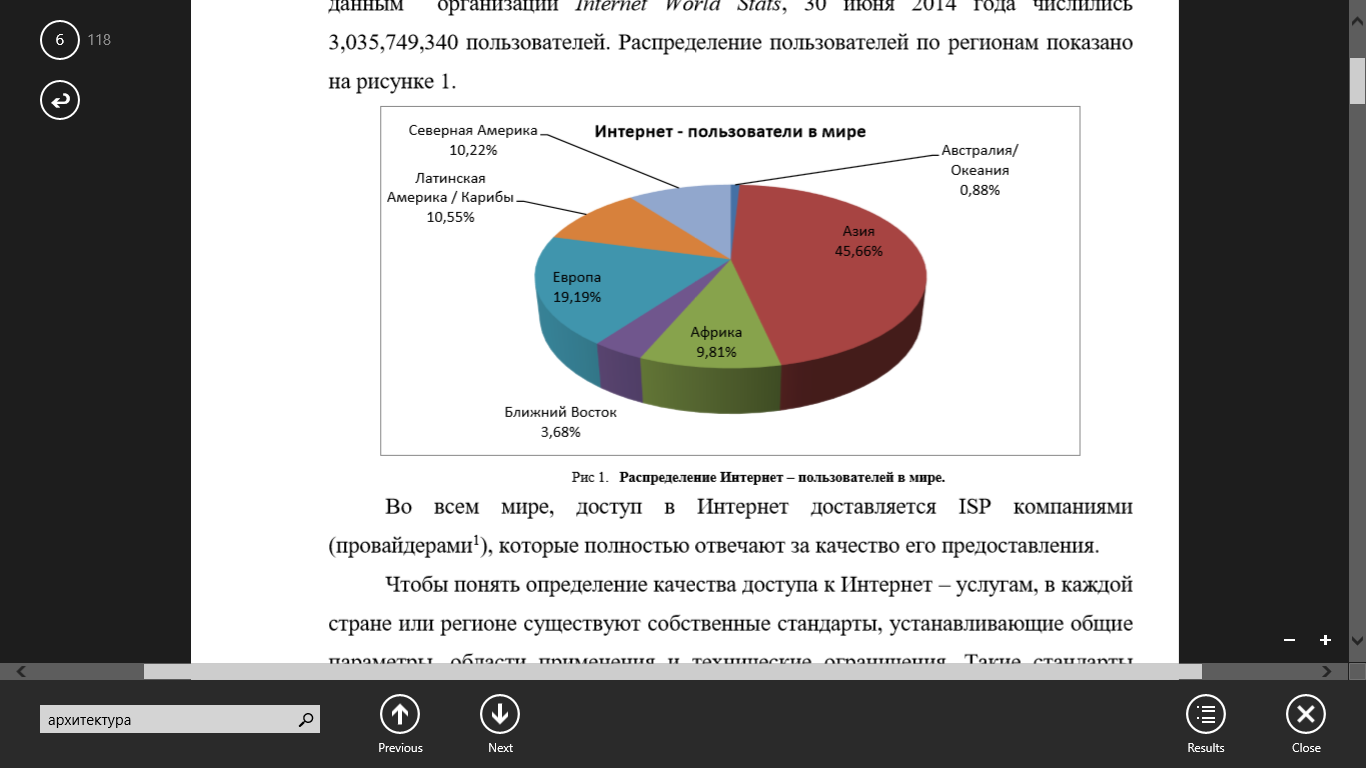
FTP - File Transfer Protocol;

СМО – система массового обслуживания;

IP - Internet Protocol.

Введение

Доступ в Интернет постоянно растет во всем мире. В данный момент, числятся более 3 миллиардов конечных Интернет – пользователей. Всего, по данным организации Internet World Stats, 30 июня 2014 года числились 3,035,749,340 пользователей. Распределение пользователей по регионам показано на рисунке 1.



*Рис 1. Распределение Интернет – пользователей в мире.*

Во всем мире, доступ в Интернет доставляется ISP компаниями (провайдерами1), которые полностью отвечают за качество его предоставления. Чтобы понять определение качества доступа к Интернет – услугам, в каждой стране или регионе существуют собственные стандарты, устанавливающие общие параметры, области применения и технические ограничения. Такие стандарты могут быть предназначены для использования представителями государственных органов, осуществляющих регулирование и контроль, научными организациями при разработке и проектировании сетей и служб, частными предприятиями оказания услуг доступа в Интернет, органами сертификации и пользователями.

Операторы связи (ISP), обязаны предоставлять услугу конечным пользователям в рамках технических показателей, определенных в существующих стандартах. Такие показатели (контролируемые параметры) имеют четкое определение в стандартах, и периодически проверяются контролирующими органами. Контроль происходит, в основном, в технических системах ISP операторов, где собираются и сравниваются измерения в определенных стандартах, и, в случае не достижения минимальных значений, оператору назначается административное наказание. В большинстве случаев, эти наказания представляются в виде административного штрафа, сумма которого зависит от действующего законодательства. Такая система, на первый взгляд, является достаточно справедливой, так как позволяет осуществлять непрерывный контроль качества услуги Интернет – доступа и требовать от операторов постоянных решений общих проблем. Но в действительности, именно понятие общих проблем делает систему не такой полезной, как казалось бы. Дело в том, что общие показатели определяют среднее техническое поведение сети или системы оператора, но не выделяют особые состояния системы, создающие конкретные или общие проблемы предоставления услуг группам пользователей. Например, имеется некий технический показатель определяющий коэффициент доступности к услуге (процентный показатель успешных попыток логического Интернет – доступа). Такой показатель, скажем, должен (по стандарту) достигать 98% по всей сети оператора и измеряется ежемесячно. Допустим, что показатель берется в системах, реализующих логическое IP – соединение, в виде «количество успешных попыток / общее количество попыток доступа». Такое измерение не сможет доказать специфическую проблему в определенном регионе страны, которая привела к неуспешной попытке доступа к услуге многих пользователей, на протяжении нескольких дней, в связи с технической проблемой конкретного конечного оборудования. Правила и нормы в отношении качества предоставления Интернет - доступа, имеют своей целью надлежащую доставку услуги до конечного потребителя, при определении параметров и целевых значений, которые должны быть выполнены поставщиками услуг (ISP). При этом, единственным источником обнаружения проблем по обслуживанию является получение жалоб от клиентов. При этой форме контроля, Интернет-провайдер (ISP) несет полную ответственность за обработку данных и доклад контролирующему органу. Такой подход соответствует практике "добросовестности", но не отражает фактические технические результаты, важные для сбора статистики и выявления существующих недостатков в сетях ISP. Данные, которые служат для анализа предоставления услуги, в настоящее время основаны на числе жалоб клиентов, а не соответствуют реальным техническим данным поведения службы [1]. Такое сочетание предоставления услуги и его регулирование/контролирование, приводит к задаче сбалансированного определения общих параметров услуги «доступ в Интернет» и специфических технических параметров, которые смогли бы быть четко получены и отражали бы настоящие проблемы конечных пользователей. Важным является необходимость понятия термина «качество услуги», и все перспективы, с которых эти понятия могут быть анализированы:

* требования качества от пользователя,
* спроектированное качество, предлагаемое оператором,
* достигнутое оператором качество,
* качество, воспринимаемое пользователем.

Таким образом, имея точное определение качества услуги, технические параметры, корректно отражающие это качество, и восприятие этих параметров со стороны конечного пользователя, можно оценить настоящее предоставления оператором, услуги Интернет – доступа.

1. Конструкторско-технологическая часть
   1. Общетехническое обоснование разработки
      1. Постановка задачи проектирования

Реализуемая подсистема должна самостоятельно отслуживать появление новых файлов с данными об интернет-соединении конечных пользователей, вычислять необходимые для оценки качества параметры и заносить все данные в базу данных.

Администратор должен иметь возможность добавления нового пользователя в систему и изменения параметров вычисления.

Таким образом, задача проектирования может быть сформулирована следующим образом:

- изучить предметную область;

- спроектировать структуру базы данных;

- оценить экономическую себестоимость разработки;

- реализовать алгоритм сбора и обработки данных;

- реализовать пользовательский интерфейс для администратора системы;

- спроектировать автоматизированную подсистему, отвечающую требованиям производительности и отказоустойчивости.

* + 1. Описание предметной области

После передачи данных на сервер, информация переходит в состояние ожидания обработки. Центральный сервер содержит программу обработки данных, которая поддерживает объем входящей информации от всех конечных пользователей. Обработка данных проходит автоматически на следующих этапах:

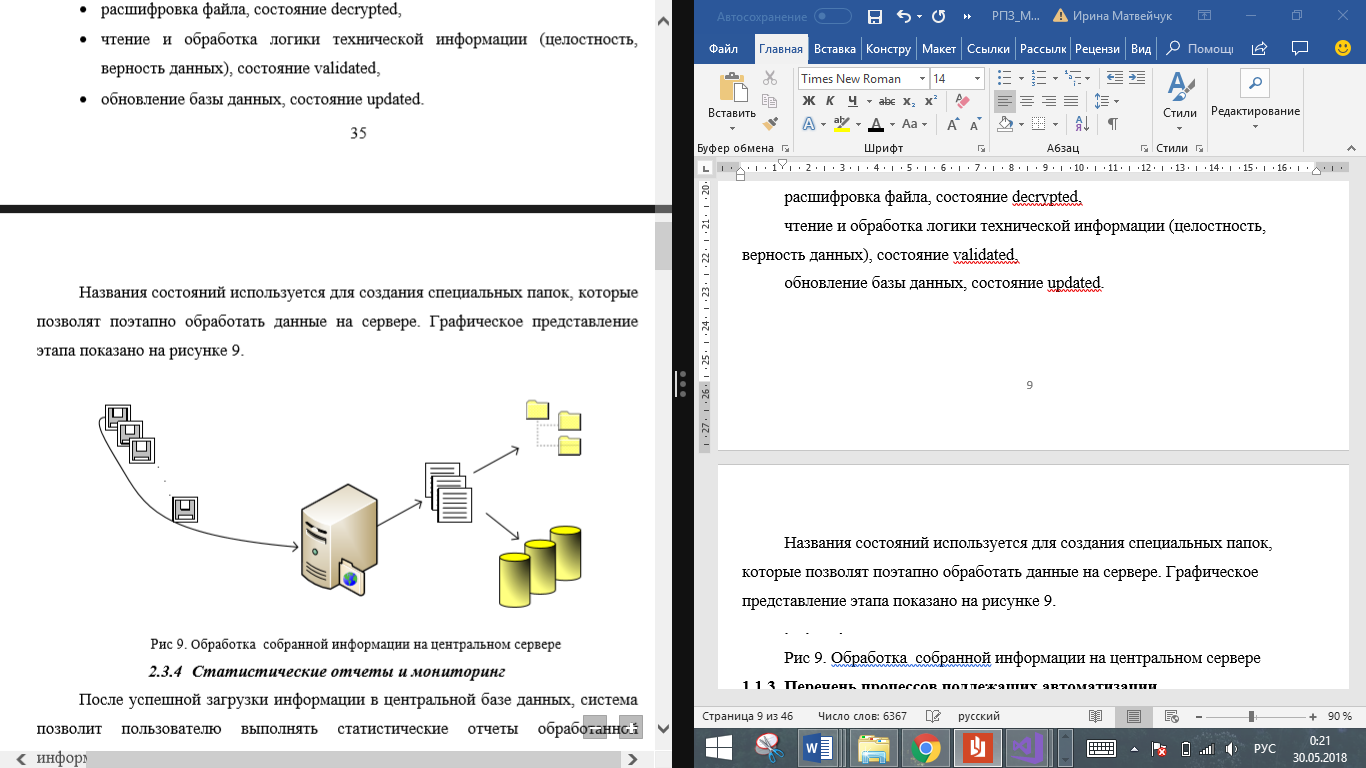
прием файла на диске, состояние receipt,

расшифровка файла, состояние decrypted,

чтение и обработка логики технической информации (целостность, верность данных), состояние validated,

обновление базы данных, состояние updated.

Названия состояний используется для создания специальных папок, которые позволят поэтапно обработать данные на сервере. Графическое представление этапа показано на рисунке 2.



*Рис 2. Обработка собранной информации на центральном сервере*

* + 1. Перечень процессов подлежащих автоматизации

При создании подсистемы сбора и обработки данных автоматизации подлежать следующие процессы:

* Обнаружение нового файла с данными;
* Вычисление необходимых нормированных показателей;
* Запись данных в базу данных;
* Добавление нового пользователя;
* Изменение пороговых значений для расчета нормированных показателей
* Изменение констант для расчета нормированных показателей.
  + 1. Функциональные задачи системы

В подсистеме сбора и обработки данных администратор должен иметь возможность и право:

* Добавить нового пользователя;
* Изменить пороговые значения для вычисления нормированных показателей;
* Изменить константы для расчета нормированных показателей;
  + 1. Выбор и обоснование критериев качества

Для проектируемого программного комплекса приоритетными являются следующие критерии качества:

* Скорость работы;
* Выявление ошибочной информации;
* Удобство работы;
* Отказоустойчивость системы;
* Документация;
* Возможность расширения функционала.

**Скорость работы** включает требования к временным характеристикам:

Время выполнения запросов к СУБД и выставления оценки не должно превышать 2 сек.

**Выявление ошибочной информации** включает обнаружение ошибок при вводе данных с несоответствующим необходимому типом, а также выявление зависимостей, препятствующих корректной обработке данных алгоритмом. Обеспечивает простоту эксплуатации.

**Удобство работы**. означает простоту и понятность работы с подсистемой. Обеспечение удобства интерфейса позволяет минимизировать время, затрачиваемое при работе с системой и получать максимальную эффективность работы.

**Документация** включает в себя обеспечение программной и эксплуатационной документацией. При оценке по данному критерию необходимо учитывать учебную направленность проектирования и создания данного программного изделия.

**Возможность расширения функционала** – так как данная подсистема является частью другой системы, при её проектировании предусматривалась способность к дальнейшему использованию её в качестве модуля другой программы. Также предусматривалась возможность расширения данной подсистемы путем добавления дополнительных программных модулей. Дополнительные модули для расширения функционала могут быть выполнены как в виде добавления изменений в существующие экранные формы, так и в виде разработки новых форм и модулей для обеспечения новых возможностей с точки зрения хранения, обработки и анализа информации, удобства пользования и эффективности работы.

* + 1. Анализ аналогов и прототипов

Далее, проводится анализ настоящих существующих средств, реализующие аналогичные задачи.

**Веб – тестеры**

Такие средства реализуют определенные тесты по желанию пользователя в реальном времени, через специальные Интернет – сайты. Зайдя на сайт, пользователь выбирает вариант выполнения теста на скорость передачи данных через Интернет.

Веб – приложение на севере и связанный браузер выполняют обычно следующие действия:

* определение общих технических характеристик ПК пользователя (используемый браузер – язык, IP адрес – страна, и др.),
* тесты ICMP – протокола на ближайшие сервера, установленной компанией предлагающей услугу, или сервером, присоединившимся к сети компании,
* вывод подходящего сервера для проведения теста – самого близкого к пользователю по наименьшей задержки ICMP протокола,
* тесты протокола ICMP, проверяя задержку сети,
* тесты протокола передачи данных (в основном HTTP), проверяя скорость передачи на выбранный сервер,
* вывод результатов пользователю. На следующем рисунке, показан результат тестирования, используя такой подход.

Некоторые сайты, так же позволяют выполнять регистрацию и визуально анализировать результаты ранее проведенных тестов.

**Интернет – логгеры**

Такие программные обеспечения, позволяют выполнять периодические тесты определенных параметров качества Интернет – услуги, и также сохраняют журналы результатов проведенных испытаний (logs). Примерами таких ПО являются Network Logger, Net Uptime Monitor, PingPlotter, Automatic Speed Test, JDast, Speed Test Logger, которые тестируют скорость передачи данных и хранят файлы с предыдущими результатами для статистического анализа. Такие испытания выполняются с помощью ICMP тестов на определенные веб – сервера, и передачи данных по протоколам FTP или HTTP. Журнал результатов может хранить недоступность услуги (если тест против всех серверов не выполняется), а также моментальные и средние значения скорости передачи данных и задержки.

**Протоколы сетевого управления**

Существуют стандартные Интернет – протоколы, которые выполняют удаленное или локальное управление над устройствами в IP-сетях на основе архитектур TCP/UDP. Управление выполняется передачей стандартизованных сообщений управляющим устройством на один или несколько серверов протокола.

Обычно, на управляющем устройстве установлен программный модуль сетевого управления, который реализует перевод требуемой информации в специфичную форму для передачи на сервер. Данные отражают конфигурацию и функционирование управляемых систем, а также решает активные задачи управления, такие как изменение и применение новой конфигурации устройств.

Управляемые устройства – оборудования или программное обеспечение, установленное на компьютере. Примерами таких элементов сети являются маршрутизаторы, персональные компьютеры или сервера, коммутаторы, мосты, IP-телефоны, IP-видеокамеры, принтеры и т.п.

К таким протоколам относятся SNMP (Simple Network Management Protocol – простой протокол сетевого управления) или CMIP (Common Management Information Protocol – протокол общей управляющей информации). Первый из них (SNMP) более распространен в настоящее время, хранит и передает информацию, используя базы управляющей информации (MIB). Базы MIB описывают структуру управляемых данных устройств. Используют иерархическое пространство имен, содержащее идентификаторы объектов (OID). Параметры, которые контролируются на каждом управляемом устройстве, имеют многие области технического применения: сетевые показатели (скорость передачи данных, задержки, состояние сети, ошибки, специфическое поведение сетевого протокола, и др.), системное поведение устройства (ОС, процессы, память, услуги, и др.), физическое состояние (температура, давление, и др.), а так же специфические показатели, ограниченные на устройства определенных производителей.

Существует большое количество SNMP – интерпретаторов и мониторов, открытые ПО (open source), и коммерческие платные реализации.

**Анализ существующих средств**

Технические средства, анализирующие в предыдущих пунктах, обладают многими свойствами для контроля технических параметров для определения качества Интернет – услуги. В основном, можно выделить следующие достоинства таких средств:

* являются приложениями готовыми к использованию,
* существуют разные варианты на выбор пользователя,
* многие из них являются открытыми, что существенно уменьшает стоимость проекта внедрения для такой системы,
* обладают графическими модулями, показывающие результаты измерений,
* для веб – тестеров, наличие множества серверов позволяет реализовать тесты в реальной Интернет – среде,
* пользователь может выполнить тест в любое желаемое время.

Несмотря на это, также можно выделить некоторые недостатки анализированных подходов:

* для веб – тестеров нет возможности статистического анализа, так как нет автоматического хранения данных предыдущих тестов,
* многие сайты, показывают плохие результаты скорости передачи данных, и приглашают пользователя скачать приложения для устранения этой проблемы, что явно заставляет задуматься над надежностью таких результатов,
* в большинстве случаев, пользователь не может выбрать сервера для реализации тестов,
* большинство систем не хранят данные в долговременный период (скажем, два месяца), а реализуют контроль параметров в реальном режиме времени,
* для систем использующие протоколы сетевого управления (SNMP например), хоть и существуют большие способности контроля параметров качества Интернет – услуги, нет возможности управлять параметрами контроля т.к. они являются «пакетными». Так, например, для некоторых ОС, возможность контроля определенных параметров зависит от встроенной SNMP услуги операционной системы, и добавление параметров невозможна или не простая в реализации,
* SNMP протокол предназначен для управления сетевых устройств, но нет возможности конфигурации специфических задач в удаленном элементе, такие как скачивание файла с определенного сервера в ежедневно определенное время.

Анализ приводит к выводу, что не существует одной системы, которая:

* реализует автоматизированный и настраиваемый контроль определенных параметров и обеспечивает надежную передачу их на централизованный сервер для дальнейшей обработки и конечного статистического анализа;
* обеспечивает хранение результатов для будущего анализа и сравнения предоставляемой услуги, с точки зрения конечного пользователя.
  1. Алгоритм разработки программного обеспечения

**Общие понятия**

Качество обслуживания телекоммуникационных сетей (QoS) определяет её способность к обеспечению необходимых ресурсов для предоставления предлагаемой клиентам услуги. Показатель QoS напрямую зависит от технических характеристик канала, которые в свою очередь могут быть легко измеряемы используя сетевые протоколы и конкретные оборудования связи.

Определение наиболее значимых технических параметров сети становится первой задачей для полноценной оценки качества. При измерении QoS необходимо учитывать наименьшее количество технических характеристик, которые в наибольшей форме влияют на конечное восприятие пользователя. Данные характеристики должны поддерживать процессы динамической оптимизации сетевых ресурсов и полностью определять услугу с самого начала и до конца его предоставления (*end-to-end*).

Задача определения характеризующих услугу «доступ в Интернет» технических параметров в наибольшей степени соответствует экспертной оценке существующих параметров, которые напрямую влияют на восприятие качества конечным пользователем (QoE). Более того, определение технических характеристик сети приводит к задаче выявления реальных потребностей клиентов к предоставленной услуге. Такие потребности должны отражать пользовательские предпочтения к предложенной услуге и соответствовать техническим возможностям оператора к обеспечению этих потребностей. В свою очередь, технические возможности оператора связи напрямую отражаются в выбранных для оценки QoS показателях сети.

Такой подход охватывает все составляющие для полноценного измерения качества с точки зрения конечного пользователя и придерживается рекомендациям международных стандартов и организаций по регулированию в области телекоммуникационных услуг.

С другой стороны, определение и измерение технических параметров сети не может применятся для оценки качества предлагаемой услуги всех пользователей, учитывая неоднородные требования. Таким образом, необходимо диверсифицировать предпочтение и восприятие разных клиентов.

**Субъективные и технические параметры для оценки услуги с точки зрения конечного пользователя**

Решение проблемы группировки пользователей учитывает определение наиболее значимых субъективных характеристик услуги «доступ в Интернет» и предпочтения клиентов с использованием многомерного метода статистического анализа данных. Данные субъективные характеристики напрямую влияют на степень удовлетворения клиента – общую оценку качества с точки зрения конечного пользователя (QoE).

Из вышесказанного следует, что для корректной оценки QoE необходимо определить конкретные и измеряемые технические параметры, которые в большей степени отражают данные характеристики. Таким образом, Таблица 1 представляет общие требования пользователей к услуге и их соответствующие технические характеристики.

1. Субъективные и технические характеристики оценки качества услуги «доступ в Интернет».

|  |  |  |  |
| --- | --- | --- | --- |
| **Субъективная характеристика** | **Описание** | **Измеряемый параметр** | **Обозначение** |
| Степень использования | Периодичность и уровень использования (как часто нужна клиенту услуга) | Уровень использования | Rg |
| Цель использования | Наиболее значимое назначение использования услуги (скачивание файлов, доступ к информации, социальные сети, просмотры видео контента) | Группировка пользователей | Pf |
| Предпочтения о качестве | Главное субъективное понятие качества пользователя, отражающиеся в таких понятиях как скорость или надежность | Скорость  Задержка  Ошибки сети  Временное окно оценок | Th  Dy  Er  Wd |

Измерение субъективных характеристик оценки качества с целью группировки пользователей и неординарное оценивание качества каждой из определённых групп возможен путём прямого опроса. В свою очередь, измерение технических показателей требует конкретный подход, каждый из которых использует собранные автоматизированной системой технические данные на ПК конечного пользователя.

После определения технических показателей, необходимо установить математическую формулу для преобразования каждого из этих параметров к конкретным индикаторам качества, которые в свою очередь будут далее использоваться для определения общей оценки качества (см. формулу 1). Также, необходимо провести нормализацию параметров, с целью их подготовки к обработке конечным алгоритмом анализа данных.

Далее, описана методология измерения и математические модели для каждого из предложенных в таблице выше технических показателей.

*Уровень использования (Rg)*

Показатель, отражающий уровень использования услуги «доступ в Интернет» позволит определить степень значимости измеряемых параметров (QoS) и их влияние на общую оценку качества с точки зрения конечного пользователя (QoE). Для полноценного измерения такого показателя необходимо непрерывное отслеживание сетевого адаптера и периодическая регистрация скорости передачи данных скачанных на ПК.

*Rg* представляет собой одномерный массив N элементов:

(2)

Где *p1*, *p2, …, pN* – процент соответствия передачи данных к определённым группам скоростного режима, заданных экспертной оценкой. Пример такого подхода предусматривает группировку передачи данных на ПК пользователя в четырёх скоростных диапазонах: *p1 –* 0 кбит/c - 256 кбит/c, *p2 –* 256 кбит/c - 512 кбит/c, *p3 –* 512 кбит/c - 1000 кбит/c, *p4 –* 1000 кбит/c - и выше.

Таким образом, массив Rg приобретает значение подобное [0.05, 0.035, 0.855, 0.06], представляющий процент времени за период измерения, в котором сетевой адаптер достигал скорости определённого диапазона. Значимость показателя на общую оценку качества обслуживания (QoS), в виде величины качества *QRg*, получает форму экспоненциальной [4] функции (3). Согласно данному параметру, длительное нахождение в более скоростном режиме принимает более высокое влияние на общий показатель качества предоставленной услуги.

(3)

Где *a1* >> *a2* >>… *>>aN* – константы, установленные экспертной оценкой, определяют вес каждого параметра в общей формуле показателя качества.

Нормализованный показатель качества уровня использования услуги *kRg*, учитывает минимальное и максимальное допустимые значения *QRg* (линейная нормализация):

(4)

*Скорость (Th)*

Данный показатель фиксирует максимальную скорость передачи данных за период измерения в двух направлениях: скачивание (*ThD* – *download*) и загрузка (*ThU* – *upload*).

Показатель измеряет скорость передачи данных на сетевом адаптере в кбит/с и отражает приближение к максимальной пропускной способности предложенной услуги, прописанной в пользовательском тарифе и договоре о предоставлении данного сервиса. Такой показатель имеет прямое влияние на качество с точки зрения конечного пользователя, так как большинство клиентских требований к услуге «доступ в Интернет» решаются повышением пропускной способности сетевого канала.

Для достижения наиболее точных показаний данного параметра, избегая измерения пиковых значений скорости, формирующиеся в начальном потоке передачи данных (известно, что сервер достигает максимальную скорость в начальных моментах передачи), данный показатель объединён со средней скоростью передачи данных конкретного файла с определённого тестового сервера. Таким образом, показатель скорости включает в себя максимальные значения передачи данных скачивания и загрузки с определённого (реального) Интернет-сервера, с которым взаимодействует пользователь, а также среднюю скорость скачивания данных с тестового сервера. Показатель приобретает форму:

(5)

Где *ThD* – максимальная скорость скачивания данных за период измерения (кбит/с), *ThU* – максимальная скорость загрузки данных за период измерения (кбит/с), – средняя скорость скачивания файла с тестового сервера (кбит/с).

Значимость показателя на общую оценку качества обслуживания (QoS), в виде величины *QTh*, получает форму экспоненциальной функции (6). Согласно данному параметру, достигнутые высокие скорости напрямую влияют на общий показатель качества предоставленной услуги.

(6)

Где

*b1* ≈ *b2* ≈ *b3* – константы, установленные экспертной оценкой, определяют вес каждого параметра в общей формуле показателя качества.

*Thrp* – максимальная установленная оператором связи скорость для пользовательского тарифа. Чем больше скоростные показатели приближаются (или повышают) данную максимальную скорость, тем большее значение достигает данный технический показатель качества.

Нормализованный показатель качества скорости передачи данных *kTh*, учитывает минимальное и максимальное допустимые значения *QTh* (линейная нормализация):

(7)

*Задержка (Dy)*

Задержка сетевых пакетов (время достижения конкретного сервера путём отправления сетевого пакета), позволяет оценить целостность и качество соединения, и определяет стабильность сетевого оборудования, обслуживающего конечного пользователя. Данная оценка реализуема используя протокол ICMP, при котором пакеты отправляются в виде запросов на удалённый узел и позволяет определить двусторонние задержки (RTT) по установленному сетевому маршруту.

Анализ задержки сети с точки зрения конечного пользователя позволит провести постоянную диагностику сетевого оборудования, что напрямую влияет на качество предоставленной услуги.

Показатель измеряет время доступа и возврата сетевого пакета к определённым серверам услуги «доступ в Интернет». Выбор серверов является одним из задач экспертного оценивания и настройки алгоритма анализа качества сервиса, в зависимости от наиболее влияющих на предоставления качественной услуги оборудования:

(8)

Где

*Dyi* – задержка сетевого пакета протокола ICMP к i-ому серверу.

*Dymax* – максимальная задержка сетевого пакета протокола ICMP за период измерения.

*ci* – константы, установленные экспертной оценкой, определяют вес каждого параметра в общей формуле показателя качества.

Нормализованный показатель качества задержки *kDy*, учитывает минимальное и максимальное допустимые значения *QDy* (линейная нормализация):

(9)

*Ошибки сети (Er)*

В контексте анализа качества конкретной телекоммуникационной услуги, наличие сетевых ошибок может существенно влиять на восприятие качества с точки зрения конечного пользователя. Таким образом, необходимо разделять критичность ошибок в период измерения технических показателей, что приводит данный параметр к форме:

(10)

Где:

*еr1* – наличие критичной ошибки услуги в период изменения технических показателей (при наличии – 1, в другом случае – 0). Ошибка является критичной если она способствует сбою подключения к предоставленной Интернет-услуге, что напрямую влияет на субъективную оценку качества конечного пользователя.

*еr2* – наличие не критичной ошибки услуги в период изменения технических показателей (при наличии – 1, в другом случае – 0). Такая ошибка не нарушает подключение к услуге, но может воздействовать ухудшению её качества.

Показатель качества представлен в виде величины *QEr*, которая получает форму экспоненциальной функции:

(11)

Где *d1* >> *d2* – константы, установленные экспертной оценкой, определяют вес каждого параметра в общей формуле показателя качества.

Показатель качества *QEr* снижает своё значение при наличии ошибок в период измерения, и в большей степени при наличии критической ошибки *er1*.

Нормализованный показатель качества задержки *kEr*, учитывает минимальное и максимальное допустимые значения *QEr* (линейная нормализация):

(12)

*Временное окно предыдущих оценок качества (Wd)*

Восприятие качества телекоммуникационной услуги имеет субъективную составляющую, которая медленно меняется во времени. Предыдущие оценки качества продолжают влиять на текущее впечатление о полученной услуге. Последовательность низких оценок, полученных за определённый промежуток времени, оставляет за собой плохое восприятие с точки зрения конечного пользователя и повышает вероятность низкого оценивания, в случае, когда технические показатели не испытывают резкое улучшение за период измерения.

Таким образом, необходимо определить связь между последней оценки качества услуги «доступ в Интернет» и его предыдущие значения. Для этого, вводится параметр временного окна оценки качества, который учитывает последние выявленные показатели качества:

*Wd =* [*Q-1,Q-2,…Q-*p] (13)

Где

*Q-1,Q-2,…Q-*p –значения оценки качества последних *p* измерений (*Q-1* – последнее измерение, *Q-p* – самое раннее последних *p* измерений).

Показатель качества принимает значение убывающей экспоненциальной функции, в которой наиболее «далёкие» во времени оценки качества в наименьшей степени влияют на общий показатель:

(14)

Нормализованный показатель качества временного окна оценок *kWd*, учитывает минимальное и максимальное допустимые значения *QWd* (линейная нормализация):

(15)

*Группировка пользователей на основе их субъективных требований к услуге (Pf)*

Группировка пользователей является важным шагом в процессе оценки качества телекоммуникационной услуги. Такой подход позволяет установить дифференцированный анализ данных для каждой группы и становится более реалистичным в рамках предоставления услуги, так как принимает во внимание разные требования клиентов и выделяет конкретные технические параметры.

Исследование показало, что использование таких субъективных характеристик как степень и цель использования, предпочтения о качестве и социальные (конкретные) предпочтения, позволят определить группы пользователей для дифференцированного анализа оценки качества предоставленной услуги.

Таким образом, для решения задачи параметризации оценки качества, вводится параметр пользовательского профиля, полученного после конкретного анализа субъективных характеристик к данной услуге:

*QPr =* [*gr1, gr2, …, grm*] (16)

Где *gr1, gr2, …, grM –* группы пользователей. Принадлежность к конкретной группе отмечается единицей в массиве *QPr*, в то время как его отсутствие отмечается нулевым значением (например, [0, 1, …,0], указывает что пользователь находится во второй группе).

Данный показатель не имеет количественное представление качества, а учитывает принадлежность пользователя к конкретной группе.

**Алгоритм измерения параметров качества для оценки услуги «доступ в Интернет»**

После определения технических параметров качества, необходимо установить алгоритм, позволяющий извлечь и подготовить показатели, которые будут направлены на вход алгоритма оценки качества услуги.

Указанные в предыдущих пунктах технические данные собираются автоматизированной системой на устройстве конечного пользователя и сохраняются в уникальном файле, переданный на сервер, где проходит предобработка, указанная данным алгоритмом.

1. Определение констант и лимитов (экспертная оценка)
   1. Уровень использования
      * Установить количество групп для показателя (*N* – целое число).
      * Установить нижние и верхние лимиты скорости для определения принадлежности к данной группе конкретного измерения (кбит-с).
      * Определить константы *a1* >> *a2* >>… *>>aN* (целые числа).
   2. Скорость
      * Определить максимальную скорость передачи данных для конкретного пользователя (*Thrp* – кбит/c).
      * Определить константы *b1* ≈ *b2* ≈ *b3* (целые числа).
   3. Задержка
      * Определить количество серверов для измерения *i* (целое число).
      * Определить константы *ci* для каждого сервера (целые числа).
   4. Ошибки сети
      * Определить константы *d1* >> *d2* (целые числа).
   5. Временное окно предыдущих оценок качества
      * Определить количество последних измерений *p* (целое число).
   6. Группировка пользователей
      * Определить количество пользовательских групп *m* (целое число).
      * Определить массив *QPr* – группу принадлежности конкретного пользователя.
2. Извлечение данных (из файла измерений или хранилище АСОИ)
   1. Извлечь данные одномерного массива *Rg* (Уровень использования).
   2. Извлечь показатели скорости – массив *Th* (Скорость).
   3. Извлечь показатели задержки на каждый сервер *Dyi* (Задержка).
   4. Извлечь данные одномерного массива *Er* о наличии ошибок сети (Ошибки).
   5. Извлечь данные одномерного массива *Wd* последних оценок качества (Временное окно оценок).
3. Расчёт показателей качества
   1. Рассчитать показатель качества уровня использования *QRg* по формуле 3.
   2. Рассчитать показатель качества скорости *QTh* по формуле 6.
   3. Определить максимальное измерение задержки за период измерения (*Dymax*).
   4. Рассчитать показатель качества задержки *QDy* по формуле 8.
   5. Рассчитать показатель качества ошибок сети *QEr* по формуле 10.
   6. Рассчитать показатель качества временного окна оценок *QWd* по формуле 14.
4. Расчёт нормализованных показателей качества
   1. Вычислить показатель *kRg* по формуле 4.
   2. Вычислить показатель *kTh* по формуле 7.
   3. Вычислить показатель *kDy* по формуле 9.
   4. Вычислить показатель *kEr* по формуле 12.
   5. Вычислить показатель *kWd* по формуле 15.

**Показатели качества**

Таким образом, после определения параметров, извлечения данных и показателей измерения, и изначального расчёта, следующие показатели поступают на вход алгоритма оценки качества телекоммуникационной услуги с точки зрения конечного пользователя:

1. *kRg* – Уровень использования.
2. *kTh* - Скорость.
3. *kDy* – Задержка сети.
4. *kEr* – Ошибки сети.
5. *kWd* – Временное окно оценок качества.
6. *QPr* – группа, пользовательский профиль.
   1. Разработка программного изделия
      1. Проектирование базы данных
         1. Инфологическая модель

**Теоретическая часть**

На этапе анализа предметной области были выделены основные объекты предметной области, то есть была проведена предварительная структуризация объектов предметной области. Следующим шагом является решение вопроса, какая информация об объектах должна храниться в базе данных и как ее представить с помощью данных. Сущностью концептуального этапа проектирования базы данных является установление соответствия между состоянием предметной области, его восприятием и представлением в базе данных.

Для описания инфологической схемы используются диаграммы «сущность – связь» (ER-диаграмма). Эта схема позволяет моделировать объекты предметной области и отношения между ними. Для ER-диаграмм существуют формальные механизмы преобразования их понятий в понятия реляционной модели данных, поддерживаемой большинством современных СУБД.

Для построения модели типа «сущность – связь» используются три основных конструктивных элемента – сущность, атрибут и связь.

*Сущность* – поименованная конструкция для моделирования объектов предметной области. Это собирательное понятие, некоторая абстракция реально существующего объекта, процесса или явления, о котором необходимо хранить информацию в базе данных. Каждый экземпляр сущности должен быть уникален. Для этого вводят первичные и вторичные ключи.

Первичный ключ (Primary Key) – это атрибут (или группа атрибутов), однозначно идентифицирующий экземпляр сущности. При этом значение идентифицирующего атрибута находится во взаимно-однозначном соответствии с экземплярами сущности. Первичный ключ должен удовлетворять ряду требований:

Уникальность. Два экземпляра сущности не должны иметь одинаковых значений ключа.

Минимум атрибутов в ключе. Составной первичный ключ (группа атрибутов) не должен содержать ни одного атрибута, удаление которого не приводило бы к утрате уникальности.

Вторичный (или внешний) ключ (Foreign Key) – он появляется в процессе установления связей между сущностями. Это описательный атрибут, который назначается в дополнение к идентифицирующему атрибуту и служит не для идентификации этого уникального экземпляра, а для выделения из набора тех объектов, которые обладают интересующими нас свойствами. Этот ключ часто называют "ключом поиска". Ограничения на состав вторичных ключей определяются логикой запроса

*Атрибут* – поименованная конструкция для моделирования свойств объекта предметной области (сущности). Основное назначение атрибута – описание свойства сущности, а также идентификация экземпляра сущностей.

*Связь* – поименованная конструкция для моделирования отношений между объектами (сущностями). При анализе связей между сущностями могут встречаться бинарные (между двумя сущностями), тернарные (между тремя сущностями) и, в общем случае n-арные связи. Может также встречаться унарные (рекурсивные) связи, когда экземпляр определенного типа сущности связан с другим экземпляром той же самой сущности. Наиболее часто встречаются бинарные связи. При проведении классификации видов связей обычно выделяют следующие виды связей:

отображение типа 1:1 (связь "один-к-одному");

отображение типа 1:М (связь "один-ко-многим");

отображение типа М:1 (связь "многие-к-одному");

отображение типа М:М (связь "многие-ко-многим").

Различают зависимые и независимые сущности. Тип сущности определяется связью между сущностями.

Независимая сущность – обычная сущность, которая не зависит при идентификации от других объектов в модели, или просто независимая, если каждый экземпляр сущности может быть однозначно идентифицирован без определения его отношений с другими сущностями.

Зависимая сущность – это дочерняя сущность, уникальность которой зависит от атрибута внешнего ключа, или просто сущность, у которой однозначная идентификация экземпляра сущности зависит от его отношения к другой сущности.

Зависимые сущности далее классифицируются на сущности, которые не могут существовать без родительской сущности, и сущности, которые не могут быть идентифицированы без использования ключа родителя (сущности, зависящие от идентификации).

Идентифицирующая связь – это связь, которая характеризуется глагольной формой со сказуемым "должен", где подлежащим является имя дочерней таблицы. Она используется, когда требуется, чтобы внешний ключ передавался в дочернюю сущность (и, в результате, создавал зависимую сущность). Идентифицирующая связь устанавливается между независимой (родительской – начало связи) и зависимой (дочерней – конец связи) сущностями.

Очень важно обратить внимание на то, что при установлении идентифицирующей связи атрибуты первичного ключа родительской сущности автоматически переносятся в состав первичного ключа дочерней сущности. Происходит миграция атрибутов. В дочерней сущности новые атрибуты помечаются как внешний ключ (foreign key).

Неидентифицирующая связь – это связь, которая характеризуется глагольной формой со сказуемым "может", где подлежащим является имя дочерней таблицы. Она используется для отображения другого типа передачи атрибутов внешних ключей – передача в область данных дочерней сущности. Очень важно обратить внимание на то, что при установлении неидентифицирующей связи происходит миграция атрибутов первичного ключа родительской сущности не в состав первичного ключа дочерней сущности, а в состав неключевых атрибутов дочерней сущности.

Существенно, что при идентифицирующей связи внешний ключ в дочерней сущности не может принимать значение NULL. Если внешний ключ должен существовать, то это означает, что запись в дочерней сущности может существовать только при наличии ассоциированной с ним родительской записи.

**Сущности и их атрибуты**

*Таблица 3. Сущность «Файл»*

|  |  |  |
| --- | --- | --- |
| **Логическое имя атрибута** | **Тип данных** | **Ключ** |
| ID файла | Числовой | Первичный (PK) |
| Имя файла | Текстовый |  |
| Начало сбора данных | Дата и время |  |
| Конец сбора данных | Дата и время |  |
| Оценка | Числовой |  |
| ID пользователя | Числовой | Внешний (FK) |

*Таблица 4. Сущность «Устройство»*

|  |  |  |
| --- | --- | --- |
| **Логическое имя атрибута** | **Тип данных** | **Ключ** |
| ID устройства | Числовой | Первичный (PK) |
| Производитель | Текстовый |  |
| Модель | Текстовый |  |
| Версия ОС | Текстовый |  |
| ID файла | Числовой | Внешний (FK) |

*Таблица 1. Сущность «Пользователь»*

|  |  |  |
| --- | --- | --- |
| **Логическое имя атрибута** | **Тип данных** | **Ключ** |
| ID пользователя | Числовой | Первичный (PK) |
| Идентификатор пользователя | Текстовый |  |
| Фамилия | Текстовый |  |
| Имя | Текстовый |  |
| Отчество | Текстовый |  |
| Пароль | Текстовый |  |
| Группа | Числовой |  |

*Таблица 6. Сущность «Сетевая ошибка»*

|  |  |  |
| --- | --- | --- |
| **Логическое имя атрибута** | **Тип данных** | **Ключ** |
| ID сетевой ошибки | Числовой | Первичный (PK) |
| Начало сбора данных | Дата и время |  |
| Ошибки пакетов на вход | Числовой |  |
| Процент ошибок на вход | Числовой |  |
| Ошибки пакетов на выход | Числовой |  |
| Процент ошибок на выход | Числовой |  |
| ID файла | Числовой | Внешний (FK) |

*Таблица 7. Сущность «Ошибка на уроне ОС»*

|  |  |  |
| --- | --- | --- |
| **Логическое имя атрибута** | **Тип данных** | **Ключ** |
| ID ошибки | Числовой | Первичный (PK) |
| Начало сбора данных | Дата и время |  |
| Количество ошибок | Числовой |  |
| ID файла | Числовой | Внешний (FK) |

*Таблица 8. Сущность «Данные скорости передачи»*

|  |  |  |
| --- | --- | --- |
| **Логическое имя атрибута** | **Тип данных** | **Ключ** |
| ID данных скорости передачи | Числовой | Первичный (PK) |
| Начало сбора данных | Дата и время |  |
| Максимальная скорость скачивания | Числовой |  |
| Максимальная скорость загрузки | Числовой |  |
| Время нахождения в первой группе | Числовой |  |
| Время нахождения во второй группе | Числовой |  |
| Время нахождения в третьей группе | Числовой |  |
| Время нахождения в четвертой группе | Числовой |  |
| Время нахождения в пятой группе | Числовой |  |
| ID файла | Числовой | Внешний (FK) |

*Таблица 9. Сущность «Данные сетевого уровня»*

|  |  |  |
| --- | --- | --- |
| **Логическое имя атрибута** | **Тип данных** | **Ключ** |
| ID данных сетевого уровня | Числовой | Первичный (PK) |
| Название хоста | Текстовый |  |
| MAC адрес | Текстовый |  |
| Тип интерфейса | Текстовый |  |
| Описание устройства | Текстовый |  |
| IPv6 | Текстовый |  |
| IPv4 | Текстовый |  |
| ID файла | Числовой | Внешний (FK) |

*Таблица 10. Сущность «ICMP тест»*

|  |  |  |
| --- | --- | --- |
| **Логическое имя атрибута** | **Тип данных** | **Ключ** |
| ID ICMP теста | Числовой | Первичный (PK) |
| URL теста | Текстовый |  |
| Результат | Текстовый |  |
| Передаваемые байты | Числовой |  |
| Время передачи | Числовой |  |
| Максимально допустимое время | Числовой |  |
| Количество прыжков | Числовой |  |
| ID файла | Числовой | Внешний (FK) |

*Таблица 11. Сущность «HTTP тест»*

|  |  |  |
| --- | --- | --- |
| **Логическое имя атрибута** | **Тип данных** | **Ключ** |
| ID HTTP теста | Числовой | Первичный (PK) |
| URL теста | Текстовый |  |
| Полученные байты | Числовой |  |
| Прошедшее время | Числовой |  |
| Результат | Текстовый |  |
| ID файла | Числовой | Внешний (FK) |

*Таблица 12. Сущность «Параметр»*

|  |  |  |
| --- | --- | --- |
| **Логическое имя атрибута** | **Тип данных** | **Ключ** |
| ID параметра | Числовой | Первичный (PK) |
| Уровень использования | Числовой |  |
| Скорость | Числовой |  |
| Задержка | Числовой |  |
| Ошибки сети | Числовой |  |
| Временное окно | Числовой |  |
| ID файла | Числовой | Внешний (FK) |

**Связи между сущностями**

1. «Файл принадлежит» связь между сущностями «Файл» и «Пользователь» типа «один-ко-многим»;
2. «Параметр характеризует» связь между сущностями «Параметр» и «Файл»;
3. «Файл описывает 1» связь между сущностями «Файл» и «Устройство» типа «один-к-одному»;
4. «Файл описывает 2» связь между сущностями «Файл» и «Ошибка на уровне ОС» типа «один-к-одному»;
5. Файл описывает 3» связь между сущностями «Файл» и «ICMP тест» типа «один-ко-многим»;
6. «Файл описывает 4» связь между сущностями «Файл» и «Сетевая ошибка» типа «один-к-одному»;
7. «Файл описывает 5» связь между сущностями «Файл» и «Данные скорости передачи» типа «один-ко-многим»;
8. «Файл описывает 6» связь между сущностями «Файл» и «HTTP тест» типа «один-ко-многим»;
9. «Файл описывает 7» связь между сущностями «Файл» и «Данные сетевого уровня» типа «один-к-одному»;

**Схема инфологической модели**

Описанные выше сущности и связи между ними можно представить в наглядном виде с помощью следующей схемы (рис. 3).

Изображение выглядит как текст, карта

Описание создано с очень высокой степенью достоверности

*Рис 3. Инфологическая модель*

* + - 1. Выбор СУБД

Для выбора средств реализации системы был произведен сравнительный анализ различных СУБД. Результаты представлены в таблице 2.

*Таблица 2. Сравнительный анализ СУБД*

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Microsoft SQL server** | **PostgreSQL** | **MySQL** |
| Цена | 290 тыс. руб. | freeware | freeware |
| Наличие документации | да | да | да |
| Поддержка технологии клиент-сервер | да | да | да |
| Наличие средств администрирования | да | да | да |
| Широко распространенная поддержка хостинг-провайдерами | да | нет | да |

В качестве системы управления базами данных (СУБД) была выбрана **MySQL (версия 5.5)**.

Данный выбор обусловлен тем, что MySQL относится к продуктам Open Source (открытые исходные тексты) и ее можно получить бесплатно. Кроме того, MySQL включает API для большого количества языков программирования (Delphi, C, C++, Java, Perl, PHP, Python, Ruby, и пр.), а также является кроссплатформенной системой (UNIX системы и среда Microsoft Windows).

**Описание MySQL 5.5**

MySQL – [свободная](http://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%BE%D0%B1%D0%BE%D0%B4%D0%BD%D0%BE%D0%B5_%D0%9F%D0%9E) [реляционная система управления базами данных](http://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%A1%D0%A3%D0%91%D0%94). MySQL является решением для малых и средних приложений. MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

MySQL – это система управления реляционными базами данных. SQL как часть системы MySQL можно охарактеризовать как язык структурированных запросов плюс наиболее распространенный стандартный язык, используемый для доступа к базам данных.

MySQL является системой клиент-сервер, которая содержит многопоточный SQL-сервер, обеспечивающий поддержку различных вычислительных машин баз данных, а также несколько различных клиентских программ и библиотек, средства администрирования и широкий спектр программных интерфейсов (API).

MySQL содержит следующие возможности:

* [хранимые процедуры](http://ru.wikipedia.org/wiki/%D0%A5%D1%80%D0%B0%D0%BD%D0%B8%D0%BC%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%B0) и функции;
* обработчики ошибок;
* [курсоры](http://ru.wikipedia.org/wiki/%D0%9A%D1%83%D1%80%D1%81%D0%BE%D1%80_%28%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85%29);
* [триггеры](http://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B8%D0%B3%D0%B3%D0%B5%D1%80_%28%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85%29);
* [представления](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%B4%D1%81%D1%82%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%28%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85%29);
* информационная схема (системный словарь, содержащий метаданные).

Система безопасности основана на привилегиях и паролях с возможностью верификации с удаленного компьютера, за счет чего обеспечивается гибкость и безопасность. Пароли при передаче по сети при соединении с сервером шифруются. Клиенты могут соединяться с MySQL, используя сокеты TCP/IP, сокеты Unix или именованные каналы (named pipes, под NT)

Начиная с MySQL версии 3.23 максимальный размер таблицы доведен до 8 миллионов терабайт (263 bytes). Однако следует заметить, что операционные системы имеют свои собственные ограничения по размерам файлов. Ниже приведено несколько примеров:

32-разрядная Linux-Intel – размер таблицы 4 Гб;

WindowsXP – размер таблицы 4 Гб.

Как можно видеть, размер таблицы в базе данных MySQL обычно лимитируется операционной системой. По умолчанию MySQL-таблицы имеют максимальный размер около 4 Гб.

* + - 1. Даталогическая модель

Построение даталогической модели основано на разработанной ранее инфологической модели.

Для обеспечения целостности данных в соответствующие сущности инфологической модели добавляются ключевые атрибуты, которые функционально определены и зависят только от первичного ключа, и не имеют связи между собой. При отображении инфологической модели в даталогическую, сущности соответствуют таблицами, атрибуты — полям таблиц. В даталогическую модель вводятся связи, аналогичные связям инфологической модели.

*Таблица 13. Collection File*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Логическое имя атрибута** | **Поле** | **Тип данных** | **Длина поля** | **Null/Not Null** |
| ID файла | collection\_id | int | 4 байта | Not null |
| Имя файла | file\_name | char(50) | 50 байт | Not null |
| Начало сбора данных | collection\_start | datetime | 8 байт | Not null |
| Конец сбора данных | collection\_end | datetime | 8 байт | Not null |
| Оценка | quality\_mark | int | 4 байта | Null |
| ID пользователя | user\_id | int | 4 байта | Not null |

*Таблица 14. Device Information*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Логическое имя атрибута** | **Поле** | **Тип данных** | **Длина поля** | **Null/Not Null** |
| ID устройства | device\_id | int | 4 байта | Not null |
| Производитель | manufacturer | char(50) | 50 байт | Not null |
| Модель | model | char(50) | 50 байт | Not null |
| Версия ОС | os\_version | char(50) | 50 байт | Not null |
| ID файла | collection\_id | int | 4 байта | Not null |

*Таблица 15. Users*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Логическое имя атрибута** | **Поле** | **Тип данных** | **Длина поля** | **Null/Not Null** |
| ID пользователя | user\_id | int | 4 байта | Not null |
| Идентификатор пользователя | user\_uid | char(20) | 20 байт | Not null |
| Фамилия | lastname | char(20) | 20 байт | Not null |
| Имя | firstname | char(20) | 20 байт | Not null |
| Отчество | thirdname | char(20) | 20 байт | Null |
| Пароль | user\_password | char(20) | 20 байт | Not null |
| Группа | user\_group | int | 4 байта | Not null |

*Таблица 16. Traffic Error*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Логическое имя атрибута** | **Поле** | **Тип данных** | **Длина поля** | **Null/Not Null** |
| ID сетевой ошибки | traffic\_error\_id | int | 4 байта | Not null |
| Начало сбора данных | collection\_datetime | datetime | 8 байт | Not null |
| Ошибки пакетов на вход | in\_pack\_discard | int | 4 байта | Not null |
| Процент ошибок на вход | in\_pack\_errors | int | 4 байта | Not null |
| Ошибки пакетов на выход | out\_pack\_discard | int | 4 байта | Not null |
| Процент ошибок на выход | out\_pack\_errors | int | 50 байт | Not null |
| ID файла | collection\_id | int | 4 байта | Not null |

*Таблица 17. Event Error*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Логическое имя атрибута** | **Поле** | **Тип данных** | **Длина поля** | **Null/Not Null** |
| ID ошибки | traffic\_error\_id | int | 4 байта | Not null |
| Начало сбора данных | collection\_datetime | int | 4 байта | Not null |
| Количество ошибок | error\_number | int | 4 байта | Not null |
| ID файла | collection\_id | int | 4 байта | Not null |

*Таблица 18. Throughput\_raw\_data*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Логическое имя атрибута** | **Поле** | **Тип данных** | **Длина поля** | **Null/Not Null** |
| ID данных скорости передачи | throughput\_ id | int | 4 байта | Not null |
| Начало сбора данных | collection\_datetime | datetime | 8 байт | Not null |
| Максимальная скорость скачивания | max\_download | int | 4 байта | Not null |
| Максимальная скорость загрузки | max\_upload | int | 4 байта | Not null |
| Время нахождения в первой группе | first\_scale | int | 4 байта | Not null |
| Время нахождения во второй группе | second\_scale | int | 4 байта | Not null |
| Время нахождения в третьей группе | third\_scale | int | 4 байта | Not null |
| Время нахождения в четвертой группе | fourth\_scale | int | 4 байта | Not null |
| Время нахождения в пятой группе | fifth\_scale | int | 4 байта | Not null |
| ID файла | collection\_id | int | 4 байта | Not null |

*Таблица 19. Network Information*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Логическое имя атрибута** | **Поле** | **Тип данных** | **Длина поля** | **Null/Not Null** |
| ID данных сетевого уровня | network\_id | int | 4 байта | Not null |
| Название хоста | hostname | char(50) | 50 байт | Not null |
| MAC адрес | mac | char(50) | 50 байт | Not null |
| Тип интерфейса | interface\_type | char(50) | 50 байт | Not null |
| Описание устройства | device\_description | char(50) | 50 байт | Not null |
| IPv6 | ipv6 | char(50) | 50 байт |  |
| IPv4 | ipv4 | char(50) | 50 байт | Not null |
| ID файла | collection\_id | int | 4 байта | Not null |

*Таблица 20. ICMP test*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Логическое имя атрибута** | **Поле** | **Тип данных** | **Длина поля** | **Null/Not Null** |
| ID ICMP теста | icmp\_test\_id | int | 4 байта | Not null |
| URL теста | url\_test | datetime | 8 байт | Not null |
| Результат | test\_result | char(50) | 50 байт | Not null |
| Передаваемые байты | icmp\_length | int | 4 байта | Not null |
| Время передачи | rtt | int | 4 байта | Not null |
| Максимально допустимое время | ttl\_default | int | 4 байта | Not null |
| Количество прыжков | ttl | int | 4 байта | Not null |
| ID файла | collection\_id | int | 4 байта | Not null |

*Таблица 21. HTTP test*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Логическое имя атрибута** | **Поле** | **Тип данных** | **Длина поля** | **Null/Not Null** |
| ID HTTP теста | http\_test\_id | int | 4 байта | Not null |
| URL теста | url\_test | char(50) | 50 байт | Not null |
| Полученные байты | bytes\_reecived | int | 4 байта | Not null |
| Прошедшее время | time\_elapsed | int | 4 байта | Not null |
| Результат | status\_code | char(50) | 50 байт | Not null |
| ID файла | collection\_id | int | 4 байта | Not null |

*Таблица 22. Parameter*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Логическое имя атрибута** | **Поле** | **Тип данных** | **Длина поля** | **Null/Not Null** |
| ID параметра | parameter\_id | int | 4 байта | Not null |
| Уровень использования | kRg | int | 4 байта | Not null |
| Скорость | kTh | int | 4 байта | Not null |
| Задержка | kDy | int | 4 байта | Not null |
| Ошибки сети | kEr | int | 4 байта | Not null |
| Временное окно | kWd | int | 4 байта | Not null |
| ID файла | collection\_id | int | 4 байта | Not null |

**Схема даталогической модели**

С помощью системы администрирования базами данных MySQL была получена следующая схема (рис. 4).



*Рис 4. Даталогическая модель*

* + - 1. Расчет объема базы данных

База данных состоит из таблиц, данные в которых и определяют объем, занимаемый всей БД. В данном расчете используются следующие типы данных:

* **числовое поле int(n)**: представляет собой целое 32 битное число, способное хранить n - значное число,
* **Text**: тип данных в СУБД, позволяющий хранить большие объемы текстовой информации. Плюс данного типа в том, что он не выделяет память сразу, а занимает лишь необходимое ему количество,
* **Datetime**: предназначен для хранения и даты и времени суток. Значение вводится и хранится в формате - MM-DD-YYYY hh:mm:ss. В качестве разделителей могут выступать любые символы отличные от цифры. Занимает 8 байт.

Объем каждой таблицы БД определяется для максимального количества пользователей системы – 2000:

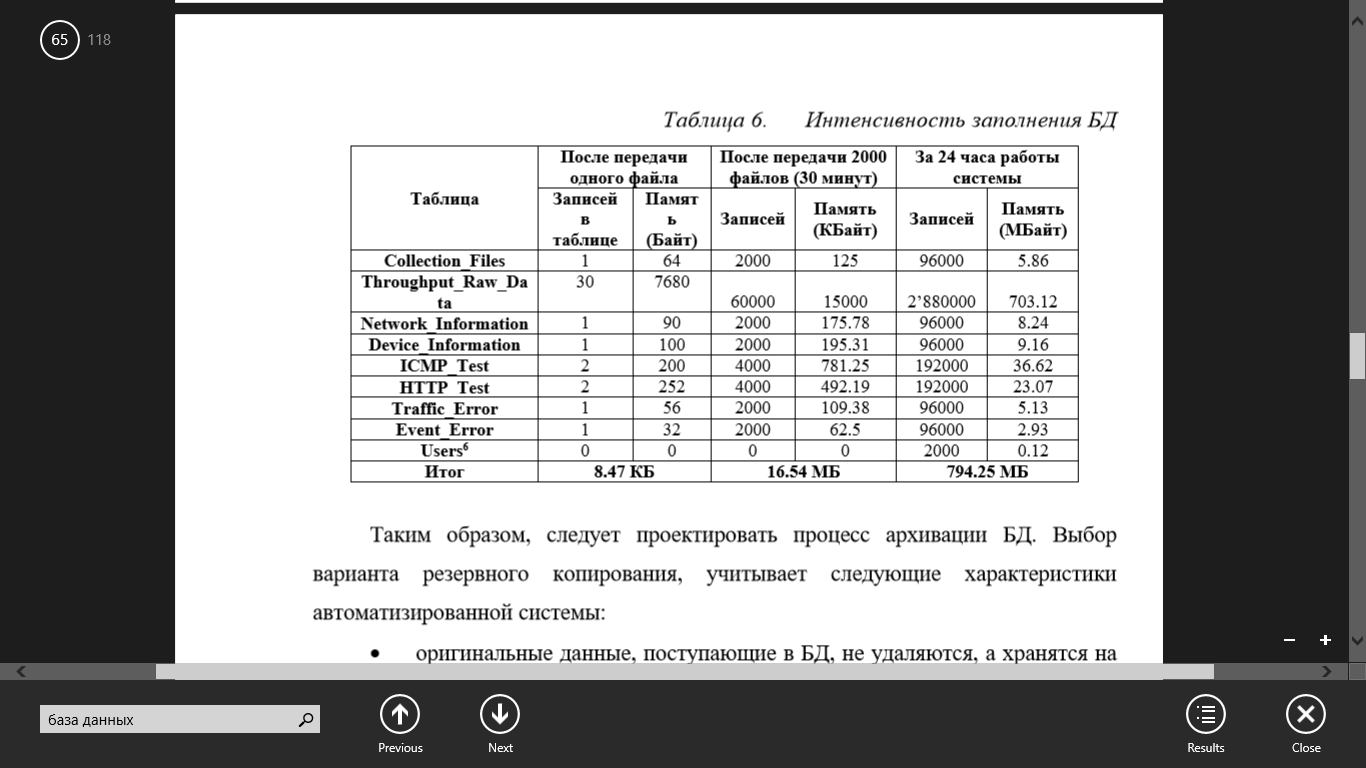
* Таблица Collection\_Files: Одна запись: 64 байт. Предполагаемый объем таблицы: до 5’760 000 строк. Вся таблица: не более 352 МБ.
* Таблица Throughput\_Raw\_Data: Одна запись: 256 байт. Предполагаемый объем таблицы: до 172’800 000 строк. Вся таблица: не более 41.19 ГБ.
* Таблица Network\_Information: Одна запись: 90 байт. Предполагаемый объем таблицы: до 72 000 строк. Вся таблица: не более 6.2 МБ.
* Таблица Device\_Information: Одна запись: 100 байт. Предполагаемый объем таблицы: до 4 000 строк. Вся таблица: не более 0.38 МБ.
* Таблица ICMP\_Test: Одна запись: 200 байт. Предполагаемый объем таблицы: до 11’ 520 000 строк. Вся таблица: не более 2.14 ГБ.
* Таблица HTTP\_Test: Одна запись: 126 байт. Предполагаемый объем таблицы: до 11’ 520 000 строк. Вся таблица: не более 1.35 ГБ.
* Таблица Traffic\_Error: Одна запись: 56 байт. Предполагаемый объем таблицы: до 360 000 строк. Вся таблица: не более 19.2 МБ.
* Таблица Event\_Error: Одна запись: 32 байт. Предполагаемый объем таблицы: до 48 000 строк. Вся таблица: не более 1.46 МБ.
* Таблица Users: Одна запись: 64 байт. Предполагаемый объем таблицы: до 4 000 строк. Вся таблица: не более 0.24 МБ.

Объем всей базы данных, рассчитывается как сумма объемов таблиц:

Объем БД = Ʃ Объема таблиц = 45.06 ГБ

Интенсивность заполнения БД представляется в таблице 23.

*Таблица 23. Интенсивность заполнения БД*



Таким образом, следует проектировать процесс архивации БД. Выбор варианта резервного копирования, учитывает следующие характеристики автоматизированной системы:

* оригинальные данные, поступающие в БД, не удаляются, а хранятся на внешнем носителе сервера. Поэтому, даже при полной потере БД, информация сможет быть восстановлена,
* БД хранит данные всех пользователей в течение 60 дней (2-х месяцев). После этого, БД копируется и создается новая, где обрабатываются новые данные,
* по практическим вычислениям, показанным в предыдущей таблице, база данных пополняется максимально на 795 МБ каждый день.

Система выполняет следующую структуру архивации данных:

* еженедельное полное копирование БД (в 00:00 каждого воскресенья), приблизительный объем данных: +5.4 ГБ,
* ежедневное разностное инкрементное копирование БД (в 02:00 каждый день), приблизительный объем данных: 795 МБ. Происходит копирование только тех файлов, которые были изменены с тех пор, как в последний раз выполнялось полное резервное копирование. Последующее инкрементное резервное копирование добавляет только файлы, которые были изменены с момента предыдущего. В среднем, инкрементное резервное копирование занимает меньше времени, так как копируется меньшее количество файлов. Однако процесс восстановления данных занимает больше времени, так как должны быть восстановлены данные последнего полного резервного копирования, плюс данные всех последующих инкрементных резервных копирований. При этом, файлы не замещают старые, а добавляются на носитель независимо.
* периодическое копирование журнала транзакций (2 раза в день в 10:00 и 18:00).

Далее, представлена диаграмма архивации БД.



*Рис. 5. Диаграмма архивации БД*

* + 1. Выбор программных средств

Для работы приложения на сервере требуется:

ОС семейства Windows;

Язык программирования C#;

База данных MySql версии 5.1.56 или выше.

* + 1. Выбор аппаратных средств

Требования к аппаратному обеспечению системы:

\* Процессор Intel Pentium 4 и выше;

\* Оперативная память 1024 МБ и выше;

\* Видеоадаптер и монитор, способные обеспечить графический режим 1920\*1080 точек с 16 битной цветопередачей и частотой регенерации на этом разрешении не менее 80 Гц;

\* наличие манипулятора «мышь»;

\* наличие клавиатуры.

* + 1. Структура программного изделия
    2. Диаграмма классов

**Элементы диаграммы**

Диаграмма классов является ключевым элементом в объектно-ориентированном моделировании. На диаграмме классы представлены в рамках, содержащих три компонента:

* В верхней части написано имя класса. Имя класса выравнивается по центру и пишется полужирным шрифтом. Имена классов начинаются с заглавной буквы. Если класс абстрактный — то его имя пишется полужирным курсивом.
* Посередине располагаются поля (атрибуты) класса. Они выровнены по левому краю и начинаются с маленькой буквы.
* Нижняя часть содержит методы класса. Они также выровнены по левому краю и пишутся с маленькой буквы.

**Взаимосвязи объектов классов**

*Зависимость*

[Зависимость](https://ru.wikipedia.org/w/index.php?title=%D0%97%D0%B0%D0%B2%D0%B8%D1%81%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D1%8C_(UML)&action=edit&redlink=1) обозначает такое отношение между классами, что изменение спецификации класса-поставщика может повлиять на работу зависимого класса, но не наоборот.

*Ассоциация*

[Ассоциация](https://ru.wikipedia.org/wiki/%D0%90%D1%81%D1%81%D0%BE%D1%86%D0%B8%D0%B0%D1%86%D0%B8%D1%8F_(%D0%BE%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) показывает, что объекты одной сущности (класса) связаны с объектами другой сущности таким образом, что можно перемещаться от объектов одного класса к другому. Является общим случаем композиции и агрегации.

Например, класс Человек и класс Школа имеют ассоциацию, так как человек может учиться в школе. Ассоциации можно присвоить имя «учится в».

Двойные ассоциации представляются линией без стрелочек на концах, соединяющей два классовых блока. Ассоциации более высокой степени имеют более двух концов и представляются линиями, один конец которых идёт к классовому блоку, а другой к общему ромбику. В представлении однонаправленной ассоциации добавляется стрелка, указывающая на направление ассоциации.

Ассоциация может быть именованной, и тогда на концах представляющей её линии будут подписаны роли, принадлежности, индикаторы, мультипликаторы, видимости или другие свойства.

*Агрегация*

[Агрегация](https://ru.wikipedia.org/wiki/%D0%90%D0%B3%D1%80%D0%B5%D0%B3%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) — это разновидность ассоциации при отношении между целым и его частями. Как тип ассоциации агрегация может быть именованной. Одно отношение агрегации не может включать более двух классов (контейнер и содержимое).

Агрегация встречается, когда один класс является коллекцией или контейнером других. Причём по умолчанию, агрегацией называют *агрегацию по ссылке*, то есть когда время существования содержащихся классов не зависит от времени существования содержащего их класса. Если контейнер будет уничтожен, то его содержимое — нет.

Графически агрегация представляется пустым ромбом на блоке класса, и линией, идущей от этого ромба к содержащемуся классу.

*Композиция*

[Композиция](https://ru.wikipedia.org/w/index.php?title=%D0%9A%D0%BE%D0%BC%D0%BF%D0%BE%D0%B7%D0%B8%D1%86%D0%B8%D1%8F_(%D0%BE%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)&action=edit&redlink=1) — более строгий вариант агрегации. Известна также как агрегация по значению.

Композиция имеет жёсткую зависимость времени существования экземпляров класса контейнера и экземпляров содержащихся классов. Если контейнер будет уничтожен, то всё его содержимое будет также уничтожено.

Графически представляется как и агрегация, но с закрашенным ромбиком.

*Различия между композицией и агрегацией*

Приведём наглядный пример. Комната является частью квартиры, следовательно, здесь подходит композиция, потому что комната без квартиры существовать не может. А, например, мебель не является неотъемлемой частью квартиры, но в то же время, квартира содержит мебель, поэтому следует использовать агрегацию.

**Взаимосвязи классов**

*Обобщение (наследование)*

Обобщение (Generalization) показывает, что один из двух связанных классов (*подтип*) является частной формой другого (*надтипа*), который называется обобщением первого. На практике это означает, что любой экземпляр подтипа является также экземпляром надтипа. Например: животные — супертип млекопитающих, которые, в свою очередь, — супертип приматов, и так далее. Эта взаимосвязь легче всего описывается фразой «А — это Б» (приматы — это млекопитающие, млекопитающие — это животные).

Графически обобщение представляется линией с пустым треугольником у супертипа.

Обобщение также известно как наследование или «[is a](https://ru.wikipedia.org/w/index.php?title=Is-a&action=edit&redlink=1)» взаимосвязь (или отношение «является»).

*Реализация*

Реализация — отношение между двумя элементами модели, в котором один элемент (*клиент*) реализует поведение, заданное другим (*поставщиком*). Реализация — отношение целое-часть. Графически реализация представляется так же, как и наследование, но с пунктирной линией.

Поставщик, как правило, является абстрактным классом или классом-интерфейсом.

**Общая взаимосвязь**

*Зависимость*

Зависимость (dependency) — это слабая форма отношения использования, при котором изменение в спецификации одного влечёт за собой изменение другого, причём обратное не обязательно. Возникает, когда объект выступает, например, в форме параметра или локальной переменной.

Графически представляется штриховой стрелкой, идущей от зависимого элемента к тому, от которого он зависит.

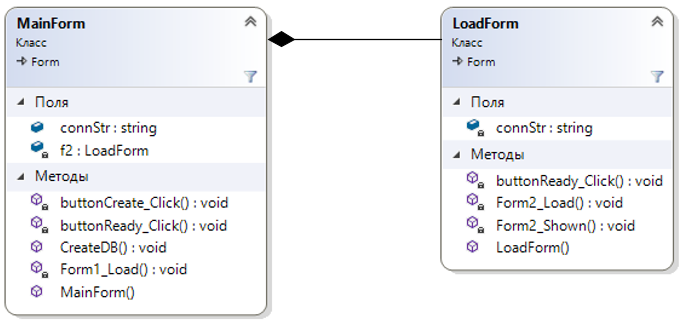
Существует несколько именованных вариантов.

Зависимость может быть между экземплярами, классами или экземпляром и классом.

*Уточнения отношений*

Уточнение имеет отношение к уровню детализации. Один пакет уточняет другой, если в нём содержатся те же самые элементы, но в более подробном представлении. Например, при написании книги вы наверняка начнете с формулировки предложения, в котором кратко будет представлено содержание каждой главы. Предположим, что резюме к каждой главе в качестве отдельного элемента входит в пакет «Предложение». Допустим также, что «Завершённая книга» — это пакет, элементами которого являются законченные главы. В этом контексте пакет «Завершённая книга» является уточнением пакета «Предложение».

**Мощность отношения** (мультипликатор) означает число связей между каждым экземпляром класса (объектом) в начале линии с экземпляром класса в её конце. Различают следующие типичные случаи:



*Рис. 6. Диаграмма классов программы QyalityCheck\_install*

Изображение выглядит как снимок экрана

Описание создано с высокой степенью достоверности

*Рис. 7. Диаграмма классов программы QyalityCheck*

Изображение выглядит как снимок экрана

Описание создано с высокой степенью достоверности

*Рис. 8. Диаграмма классов программы QyalityCheck*

* + - 1. Описание классов

*Таблица 24. Класс MainForm*

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

*Таблица 25. Класс LoadForm*

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

*Таблица 26. Класс MainForm*

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

*Таблица 27. Класс FileInfo*

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

* + 1. Программа и методика испытаний

**Объект испытаний**

Объектом испытаний является программный модуль обработки файла с данными об интернет соединении, создания на основе них базы данных и расчета нормированных показателей качества интернет соединения.

**Цель испытаний**

Цель испытания – проверка функционирования всех указанных в техническом задании функций программы.

**Состав предъявляемой документации**

На испытания программного продукта предъявляются следующие документы:

1) Техническое задание.

2) Программа и методика испытаний.

**Требования к составу аппаратного обеспечения**

* Процессор с частотой 2 ГГц;
* 2 ГБ оперативной памяти;
* Видеоадаптер и монитор, способные обеспечить графический режим 1024\*768 точек с 32 битной цветопередачей;
* Жесткий диск объемом 64 ГБ;
* Манипулятор «мышь» или другое указывающее устройство;
* Клавиатура;

**Требование к составу программного обеспечения**

* ОС семейства Windows;
* Язык программирования C#;
* База данных MySQL версии 5.1.56 или выше.

*Таблица 33. Последовательность испытаний*

|  |  |  |  |
| --- | --- | --- | --- |
| **N** | **N пункта ТЗ** | **Выполняемые действия** | **Результат** |
| 1 | 5.2.1 Создание базы данных | Запустить исполняемый файл «QualityCheck\_install» | На экране появится форма с данными для подключения к MySQL. |
| Ввести все данные для подключения к MySQL: логин - Admin, пароль - 123456, хост - localhost, порт - 3306; и нажать на кнопку «Создать базу данных» | На экране появится полоса загрузки, показывающая на каком этапе находиться создание базы данных. |
| Нажать на кнопку «Готово» дважды. | Закроются обе формы программы. |
| 2 | 5.2.2 Подключение к MySQL | Запустить исполняемый файл «QualityCheck» | На экране появится форма с данными для подключения к MySQL. |
| Ввести все данные для подключения к MySQL: логин, пароль, хост, порт; и нажать на кнопку «Подключиться к MySQL» | Кнопка «Подключиться к MySQL» станет неактивной, а кнопка «Начать работу» станет активной. |
| 3 | 5.2.3 Отслеживание появления нового файла в выбранном администратором каталоге | Во вкладке «Общие» выбрать каталог для отслеживания файлов (D:\VKR\_files) и ввеси данные для вычисления параметров (макс. скорость загрузки - 30000, макс. скорость скачивания и связи - 200000, параметр временного окна оценки качества – 4). Нажать на кнопку «Начать работу». | Кнопка «Начать работу», а также поля ввода данных станут неактивными, а кнопка «Остановить» станет активной. |
| 4 | 5.2.4 Извлечение необходимых данных из обнаруженного файла и занесение их в созданную базу данных | Перенести файл с данными в выбранный в п.3 каталог. | На экране появится сообщение об успешной обработке файла. |
| 5 | 5.2.5 Вычисление нормированных показателей качества интернет соединения на основании данных полученных из файла | Этот пункт ТЗ выполняется в рамках п.4. |  |
| 6 | 5.2.6 Изменение пороговых значений для расчета нормированных показателей качества интернет соединения | Нажать на кнопку «Остановить». | Кнопка «Начать работу», а также поля ввода данных станут активными, а кнопка «Остановить» станет неактивной. |
| Ввести в поле максимальная скорость загрузки значение 20000 и нажать на кнопку «Начать работу» | Кнопка «Начать работу», а также поля ввода данных станут неактивными, а кнопка «Остановить» станет активной. |
| 7 | 5.2.8 Изменение констант для расчета нормированных показателей качества интернет соединения | Во вкладке «Общие» нажать на кнопку «Изменить константы формул вычисления параметров» | Откроется файл с данными константами. |
| В файле изменить значение параметр а1, написав «а1=6». Закрыть файл с сохранением. | На экране останется только главная форма программы. |
| 8 | 5.2.7 Создание администратором соответствующей записи в базе данных при появлении нового пользователя | Во вкладке «Настройки БД» ввести данные нового пользователя: имя – Наталья, фамилия – Матвейчук, отчество – Алексеевна, пароль – 123456, UID – 987654, группа – 2.  Нажать на кнопку «Добавить». | Появится сообщение об успешном добавлении пользователя. |

* 1. Разработка интерфейса взаимодействия с пользователем

Исходя из поставленных выше задач, на основе разработанной модели данных были разработаны следующие экранные формы взаимодействия пользователя с системой.

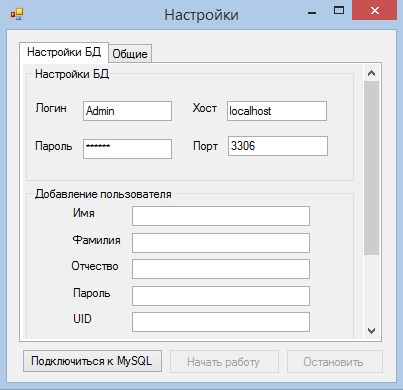
* + 1. Граф диалога

Изображение выглядит как текст, снимок экрана

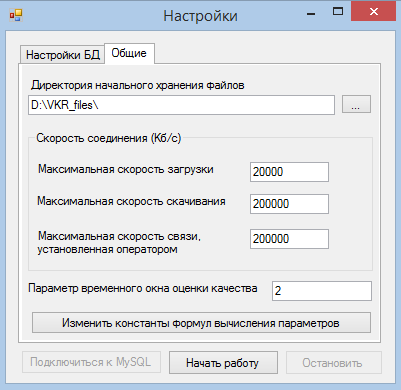
Описание создано с высокой степенью достоверности

*Рис.9. Граф диалога*

* + 1. Экранные формы

**

*Рис. 10. Главная форма. Вкладка настроек БД*

**

*Рис. 11. Главная форма. Вкладка общих настроек*

1. Научно-исследовательская часть
   1. Моделирование
      1. Параметризация системы

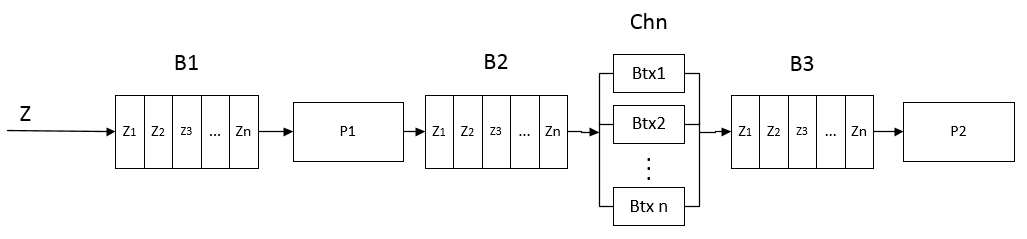
Подсистема рассматривается со следующими характеристиками:

* один канал связи (Chn): Интернет - канал между серверами,
* будет использован протокол для передачи данных (FTP, HTTP, и т.п.)
* заявки (Z) представляют собой файлы, приходящие на сервер.
* поток заявок образуют все пользователи или клиенты (Сlt) вошедшие в проект (имеют установленное ПО),
* действует накопитель – буфер на этапе обработки (Btx), и образуется очередь на обработку (Qtx).
* интервал времени генерации одного файла на каждом клиенте – 30 минут. Таким образом, заявки (Z) поступают, каждую секунду в количестве Сlt/1800. Если рассчитываем секунду как единица времени, то интенсивность λ = Сlt/1800. Интенсивность будет зависеть от количества клиентов, но в нормальном состоянии считается неизменной – поток является стационарным.
* считается, что в каждый момент времени может появиться только одна заявка, которая обслуживается очередным сокетом, деля для этого пропускную способность. Поток является простейшим.
* на этапе передачи в бд действует накопитель - буфер для файлов достигших сервера, но еще не обработанных.
* генерируется очередь на передачу. Длину очереди представляет количество не обработанных файлов. Обработать файл, значит проверить и обновить его данные в общую базу на сервере,
* емкость накопителя в данной модели предполагается неограниченной – переполнение накопителя очень маловероятно, учитывая средний размер файлов.
* заявки обрабатываются без приоритета, в порядке поступления (FIFO – Fist In First Out). Это касается дисциплин буферизации и обслуживания.

Обзор методов оптимизации вычислительных ресурсов для достижения наилучшей производительности системы в разных СУБД и операционных системах, позволяет в качестве критерия выбрать длительность обслуживания заявок на этапе обработки (Рrc) в секундах.

* + 1. Имитационная модель

Для оценки нагрузок на модули, обрабатывающие такие запросы, была использована специализированная система имитационного моделирования



*Рис.11. Имитационнная модель в терминах СМО*

Текст программы на языке GPSS (General Purpose Simulaton System):

Clt EQU 1000 ;пользователи вошедшие в проект

Z EQU 1800/Clt ;время генерации заявки каждого

;пользователя (в секундах)

Btx\_n STORAGE 10 ;количество допущенных одновременных

;соединений

Qtx\_n EQU 5 ;очередь на этапе передачи = Btx\_n

Rch EQU 512000 ;пропускная способность канала (б/сек)

Fsz EQU 2 ;размер генерирующегося файла

B1 EQU Uniform(3,2,3) ;длительность обслуживания заявок на

;этапе обработки (в секундах)

B2 EQU Uniform(3,0.5,2) ;длительность доступа к бд

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

GENERATE (Exponential(4,0,Z)) ;генерация заявок

QUEUE Total\_time ;контроль пребывания заявок в системе

QUEUE Qb1 ;счет очереди на этапе обработки файла

SEIZE Qb1 ;вход в блок обработки файла

ADVANCE B1 ;вычисление времени обработки файла

RELEASE Qb1 ;выход из блока обработки файла

DEPART Qb1 ;выход из очереди на этапе обработки файла

QUEUE Qtx ;счет очереди канала передачи

ENTER Btx\_n ;вход в очередной канал передачи

ADVANCE (TxTime()) ;вызов процедуры вычисления времени передачи

LEAVE Btx\_n ;выход из канала передачи

DEPART Qtx ;выход из очереди канала передачи

QUEUE Qb2 ;счет очереди на этапе доступа к бд

SEIZE Qb2 ;вход в блок доступа к бд

ADVANCE B2 ;вычисление времени доступа к бд

RELEASE Qb2 ;выход из блока доступа к бд

DEPART Qb2 ;выход из очереди на этапе доступа к бд

TERMINATE ;завершение модели в желательной

;последовательности

DEPART Total\_time ;контроль пребывания заявок в системе

GENERATE 604800

TERMINATE 1

START 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PROCEDURE TxTime() BEGIN

TEMPORARY Arg1, Time;

IF (Q$Qtx = 0) THEN BEGIN

Time = (Fsz # 8 # 1024) / Rch;

END;

ELSE BEGIN

Time = (Fsz # 8 # 1024 # Q$Qtx) / Rch;

END;

RETURN (Time);

END;

**Пример отчета:**



*Рис 12. Пример отчета*

* + 1. Результаты моделирования

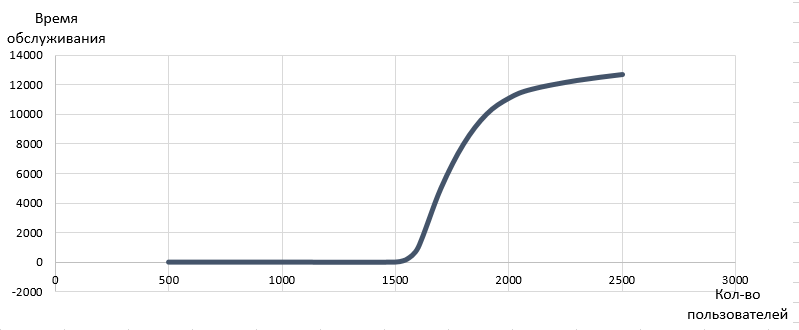
Система с временем обработки файла 2-3 сек стабильна примерно до 1500 пользователей, после чего время обслуживания файла значительно увеличивается до неприемлемых величин.

Результаты измерений представлены в таблице 33.

Зависимость представлена на рисунке 12.

*Таблица 33. Результаты моделирования*

|  |  |
| --- | --- |
| Количество пользователей | Время обслуживания |
| 3000 | 12985,34 |
| 2500 | 12683,56 |
| 2300 | 12279,03 |
| 2100 | 11710,34 |
| 2000 | 11106,5 |
| 1900 | 10153,1 |
| 1800 | 7982,16 |
| 1700 | 4098,59 |
| 1600 | 1012,04 |
| 1550 | 198,96 |
| 1500 | 11,43 |
| 1000 | 4,98 |
| 500 | 2,56 |



*Рис.13. Зависимость времени обслуживания файла от количества пользователей в системе*

1. Экономическая часть
   1. Смета затрат на создание программного изделия

Процесс разработки сложной программной продукции сопровождается необходимостью решения многих экономических проблем. Одна из важных экономических проблем – определение стоимости информационно-программного изделия (ИПИ) (сметной стоимости затрат).

Затраты на разработку программного изделия могут быть представлены в виде сметы затрат, включающей в себя следующие статьи:

*Таблица 24. Затраты на разработку программного изделия*

|  |  |  |
| --- | --- | --- |
| **№** | **Статья затрат** | **Сумма статьи (руб.)** |
| 1 | Расходные материалы | 1 420 |
| 2 | Затраты на программное обеспечение | 0 |
| 3 | Затраты на оборудование | 2 137 |
| 4 | Услуги сторонних организаций | 1 000 |
| 5 | Заработная плата | 67 800 |
| 6 | Отчисления на социальные нужды | 9695,40 |
| 7 | Налоги | 8814 |
| 8 | Накладные расходы | 9 086,5 |
| 9 | Себестоимость разработки | 99 953,5 |
| 10 | Прибыль | 17 991 |
| 11 | Цена | 117 944,5 |
| 12 | Договорная цена | 139174,51 |

Расчет затрат на разработку данного программного изделия проводился для уровня цен и окладов на 20.05.2018 г.

* 1. Обоснование сметы затрат
     1. Расчет затрат на расходные материалы

К данной статье затрат отнесены следующие расходы, представленные в таблице 35.

*Таблица 35. Затраты на расходные материалы*

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Количество** | **Цена (руб.)** |
| Бумага формата А1 (высокого качества: плотность 80 г/м2). | 6 листов | 600 |
| Бумага формата А4 | 3 пачки  (по 500 листов) | 320 |
| Flash-накопитель 32 Гб | 1 шт. | 500 |
| Итого | | 1 420 |

Итого: СМ = 1 420 рублей.

* + 1. Расчёт затрат на программное обеспечение

Были использованы Microsoft Office 365 Education и Visual Studio Community, которые не требуют платы за пользование.

Итого: СПО = 0 рублей.

* + 1. Расчёт затрат на оборудование

При разработке ОСС затраты по данной статье сводятся к затратам, связанным с использованием вычислительной техники с учетом ее ремонта. При разработке использовалось следующее оборудование:

*Таблица 46. Затраты на оборудование*

|  |  |
| --- | --- |
| **Наименование** | **Цена (руб.)** |
| Компьютер MacBook Pro 13-дюймовый: 2,4 ГГц с дисплеем Retina | 52 990 |
| [HP LaserJet Pro P1102w](http://market.yandex.ru/model.xml?modelid=6126370&hid=138608)http://market.yandex.ru/_/Z3jABDSRezQ_Z675TetexLKEBD4.pnghttp://market.yandex.ru/_/Z3jABDSRezQ_Z675TetexLKEBD4.pnghttp://market.yandex.ru/_/Z3jABDSRezQ_Z675TetexLKEBD4.pnghttp://market.yandex.ru/_/Z3jABDSRezQ_Z675TetexLKEBD4.pnghttp://market.yandex.ru/_/Z3jABDSRezQ_Z675TetexLKEBD4.png | 4 000 |
| Итого | 56 990 |

Итого: СОБ = 56 990 рублей.

Длительность использования оборудования составляет 3 мес.

При использовании ускоренных сроков амортизации затраты на вычислительную технику вычисляются по формуле: , где:

*k* – коэффициент амортизации на год (k=0,15 для ускоренной амортизации);

- стоимость компонентов оборудования;

 - период использования оборудования (в месяцах).

Таким образом, затраты на вычислительную технику при ускоренных сроках амортизации равны 2 137 руб.

Поскольку устройство новое и обслуживается по гарантии 1 год, то затраты на ремонт = 0 руб.

Таким образом, общие затраты на обслуживание оборудования составляют 2 137 руб.

* + 1. Расчёт затрат на услуги сторонних организаций

В статье учитываются затраты на выполнение сторонними организациями работ, непосредственно связанных с разработкой ИПО.

При разработке данного ИПО потребовались услуги сторонних организаций, представленные в таблице 37.

*Таблица 57. Затраты на услуги сторонних организаций*

|  |  |
| --- | --- |
| **Наименование** | **Цена (руб.)** |
| Переплет программной документации | 1 000 |

Итого: СИЗГ = 1 000 рублей.

* + 1. Расчёт заработной платы

В данную статью включается заработная плата исполнителей, непосредственно связанных с разработкой ИПО, с учетом их должностного оклада и времени участия в разработке. Расчет заработной платы (Сзп) производится по следующей формуле:

Затраты на выплату исполнителям заработной платы определяется следующим соотношением:

СЗП= СЗ.ОСН + СЗ.ДОП

где Сз.осн - основная заработанная плата, Сз.доп - дополнительная заработная плата.

СЗП = 11 500 руб. , следовательно СЗП = 23000 руб.

где n – количество разработчиков ИПИ, Сmin ЗП – минимальная заработанная плата, Кi – коэффициент, соответствующий разряду разработчика (Кi = 1) , – время разработки ( = 2 месяца). Но возможен расчет исходя из трудового договора, где прописан размер заработной платы, который в общем случае может отличаться от нормативной. Возьмем

В соответствии с налоговым законодательством Российской Федерации доходы физических лиц за вычетом некоторых льгот подлежат обязательному налогообложению (налог на доходы физических лиц). Для компенсации выплат размер месячного оклада увеличивается, что отражено в формуле:

СЗ.ОСН = СЗП (1+ НДФЛ)

Где СЗП = 50000 руб - сумма к выплате, которая была оговорена с работником. Ндфл - налог на доходы с физических лиц (13%). Таким образом,

СЗ.ОСН =50000\*1,13 = 56500 руб.

* + 1. Расходы на дополнительную заработанную плату

Расходы на дополнительную заработанную плату учитывают все выплаты непосредственно исполнителям за время, не проработанное на производстве, но предусмотренное законодательством, в том числе: оплата очередных отпусков, компенсация за недоиспользованный отпуск, и др. Величина этих выплат составляет 20% от размера основной заработной платы:

СЗ.ДОП = СЗ.ОСН \* 0,2

Т. е. в нашем случае

СЗ.ДОП = 56500 \* 0,2 = 11300 руб.

В результате получаем

СЗАРП= СЗ.ОСН + СЗ.ДОП = 67800 руб.

* + 1. Расчет отчислений на социальные нужды

В данной статье учитываются отчисления на социальные нужды, производимые в фонды социального страхования, обязательного медицинского страхования и пенсионный фонд. Расчет производится в соответствии с главой 24 Налогового Кодекса РФ «Единый социальный налог (взнос)».

Ставки единого социального налога определяются в зависимости от величины налогооблагаемой базы. Например, для налоговой базы, рассчитанной для каждого отдельного работника нарастающим итогом с начала года до 100 000 руб., величина единого социального налога рассчитывается по формуле:

СЕСН = КЕСН×СЗАРП

где КСН – коэффициент отчисления на соц. Нужды, СЗП – заработная плата (руб).

Коэффициент отчислений на социальные нужды (без учета льгот) включает в себя следующие отчисления:

Ндфл составляет 13% от заработной платы;

отчисление в фонд социального страхования составляет 2.9% от заработанной платы;

отчисление в пенсионный фонд составляет 22%

отчисление в фонд обязательного медицинского страхования составляет 5,1%

Отсюда: КСН = 0,143

СЕСН = КСН×СЗАРП= 67800\*0,143= 9695,40 руб.

* + 1. Расчет налоговых выплат

В качестве налоговых выплат выступает Ндфл (Ндфл - налог на доходы с физических лиц (13%)).

Вычисление 13% от заработной платы работника (67800 рублей) составляет 8814 рублей.

* + 1. Расчет накладных расходов

В данной статье учитываются затраты на общехозяйственные расходы, внепроизводственные расходы и расходы на управление.

Накладные расходы составляют 10% от суммы остальных расходов.

Получаем, что:

СНР = (СМ + СПО + СОБ + СИЗГ +СЗАРП + СЕСН) \*0,1 = 9 086,6 руб.

* + 1. Расчет прочих расходов

Данная статья представлена налогом на имущество и налогом на транспортные средства.

Налог на имущество в данном случае не платится, поскольку все имеющееся в наличии имущество, включаемое в налогооблагаемую базу в соответствии с инструкцией «О порядке исчисления и уплаты в бюджет налога на имущество предприятий», используется на нужды образования, и, следовательно, налогом на имущество не облагается.

Налог на владельцев транспортных средств не платится, в связи с отсутствием транспортных средств.

* + 1. Расчет себестоимости

Себестоимость рассчитывается как сумма по всем вышеперечисленным статьям затрат и составляет:

**S** = СНР + СМ + СПО + СОБ + СИЗГ +СЗАРП + СЕСН + СПРОЧ = 99 953,5руб.

* + 1. Расчет прибыли

Расчет прибыли ведется с учетом отчислений в местный бюджет и налога на прибыль.

Отчисления в местный бюджет составляют 4,5% от себестоимости ИПО и равны 4497,3 руб.

Расчет прибыли ведется с учетом налога на прибыль. Чистая прибыль составляет 10% от себестоимости и равна 9995,3 руб.

Налог на прибыль составляет 35% от прибыли и равен 3498,4 руб.

Итого: прибыль = 4497,3 + 9995,3 + 3498,4 = 17 991 руб.

* + 1. Цена

Цена программного продукта определяется путем суммирования прибыли и себестоимости и составляет: =117 944,5 руб.

* + 1. Договорная цена

В качестве конечной цены выступает договорная цена, включающая в себя налоги.

В виду того, что реализация товаров и услуг на территории РФ осуществляется с налогом на добавленную стоимость (18% от цены), в цену изделия включается НДС. Тогда окончательная цена ОСС с учетом НДС составляет:

= Ц\*1,18 = 139174,51 руб.

Таким образом, конечная цена продукта составляет 139174,51 рублей.

Заключение

При разработке подсистемы оценки качества были получены следующие результаты:

1. изучена предметная область, связанная с оценкой качества услуги «Доступ в интернет»;
2. сформулированы требования к разрабатываемой программе;
3. реализована база данных;
4. реализован алгоритм обработки файлов с параметрами услуги «Доступ в интернет»;
5. реализован интерфейс программы;
6. проведена оценка экономической себестоимости разработки;
7. разработан и написан комплект технической документации;
8. разработана графическая часть конструкторской документации.

Полученный информационно-программный продукт имеет возможность дальнейшего использования его в качестве модуля программы и возможность усовершенствования: имеется возможность добавления новых модулей и функциональных возможностей без серьезного вмешательства в уже разработанные модули.

Список использованных источников

1. ГОСТ 19.201-78. ЕСПД. Техническое задание. Требования к содержанию и оформлению.
2. ГОСТ 19.301-79. ЕСПД. Программа и методика испытаний. Требования к содержанию и оформлению.
3. ГОСТ 19.404-79. ЕСПД. Пояснительная записка. Требования к содержанию и оформлению.
4. ГОСТ 7.32—2001 Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления.
5. Федеральный закон от 24.07.2009 N 212-ФЗ (ред. от 25.12.2012) «О страховых взносах в Пенсионный фонд Российской Федерации, Фонд социального страхования Российской Федерации, Федеральный фонд обязательного медицинского страхования» // «Собрание законодательства РФ», 27.07.2009, N 30, ст. 3738
6. В. М. Постников. Основы эксплуатации АСОИиУ, Часть 1. – M.: Издательство МГТУ им. Н.Э. Баумана, 2015. - 191 с.
7. SQLite vs MySQL vs PostgreSQL: A Comparison Of Relation Database Management Systems. [Электронный ресурс]. Режим доступа: https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems (дата обращения 09.04.2018).
8. ГОСТ Р 55387-2012. Качество услуги «Доступ в Интернет». Показатели качества. Введ. 2014-01-01. М.: Изд-во стандартов, 2014. 10 с.
9. Гонсалес Х.К., Галкин В. А. Постановка задачи о разработке автоматизированной системы обработки технических данных конечного интернет-пользователя республики Эквадор. Инженерный вестник (МГТУ им. Н.Э.Баумана). Электронный журнал. 2013 - № 10. <http://engbul.bmstu.ru/doc/640684.html>.
10. Гонсалес Х.К., Галкин В. А. Анализ и синтез программного обеспечения сбора технических данных конечного Интернет – пользователя. // Инженерный вестник. МГТУ им. Н.Э.Баумана. Электрон. журн. 2014. № 11. Режим доступа: http://engbul.bmstu.ru/file/735786.html (дата обращения 21.03.2018).
11. Корпорация Майкрософт. Добавление в базу данных одной или нескольких записей. Режим доступа: http://office.microsoft.com/ruru/access-help/HA010043515.aspx (дата обращения 20.04.2018).
12. Справочник MySql. URL: <http://www.mysql.com/>

Приложение 1. Техническое задание

**Московский Государственный Технический Университет имени. Н. Э. Баумана**

Кафедра «Системы обработки информации и управления»

Утверждаю: Согласовано:

" " 2018 г. " " 2018 г.

**Подсистема сбора и обработки данных технических параметров конечных интернет пользователей**

техническое задание (вид документа)

листы А4 (вид носителя)

8

(количество листов)

Исполнитель:

Студент группы ИУ5-84

Матвейчук И.А.

" " 2018 г.

Москва, 2018

**Оглавление**

[1.Наименование 82](#_Toc514333208)

[2.Основание для разработки 82](#_Toc514333209)

[3.Исполнитель 82](#_Toc514333210)

[4.Назначение и цель разработки 82](#_Toc514333211)

[5.Содержание работы 83](#_Toc514333212)

[5.1 Задачи 83](#_Toc514333213)

[5.2 Требования к функциональным характеристикам 83](#_Toc514333214)

[5.3 Требования к архитектуре программного изделия 84](#_Toc514333215)

[5.4. Требования к входным и выходным данным 84](#_Toc514333216)

[5.4.1. Требования к входным данным 84](#_Toc514333217)

[5.4.2 Требования к выходным данным 85](#_Toc514333218)

[5.5 Требования к надежности 85](#_Toc514333219)

[5.6 Лингвистические требования 85](#_Toc514333220)

[5.7 Требования к программным компонентам 85](#_Toc514333221)

[5.7 Требования к составу программных средств 85](#_Toc514333222)

[5.8 Требования к составу технических средств 86](#_Toc514333223)

[6. Этапы разработки 86](#_Toc514333224)

[7. Техническая документация 87](#_Toc514333225)

[8.Порядок приема работы 87](#_Toc514333226)

[9.Дополнительные условия 87](#_Toc514333227)

1. **Наименование**

Подсистема сбора и обработки данных технических параметров конечных интернет-пользователей.

1. **Основание для разработки**

Основанием для разработки является задание на выпускную работу, подписанное руководителем выпускной работы и утвержденное заведующим кафедрой. Задание утверждено кафедрой «Системы обработки информации и управления» (ИУ5) МГТУ им. Н.Э. Баумана.

1. **Исполнитель**

Студентка четвертого курса группы ИУ5-84 Матвейчук И.А.

1. **Назначение и цель разработки**

Данная разработка представляет собой часть системы по оценке качества интернет соединения и предназначена для сбора и обработки данных технических параметров конечных интернет пользователей.

Целью разработки является проектирование и разработка приложения, позволяющего собирать и обрабатывать данные технических параметров конечных интернет пользователей. Система заносит полученные и вычисленные данные в базу данных для удобства их дальнейшего использования.

Система призвана уменьшить временные затраты на произведение анализа файла с данными об интернет соединении, на создание базы данных для их хранения и расчета нормированных показателей качества интернет соединения

.

1. **Содержание работы**
   1. **Задачи**

В процессе выполнения работы следующие задачи подлежат решению:

1. Исследование предметной области, определение функциональных задач;
2. Разработка инфологической и даталогической моделей;
3. Разработка архитектуры программного обеспечения;
4. Разработка алгоритмов основных функций программы;
5. Разработка интерфейса приложения
6. Реализация приложения;
7. Тестирование приложения;
8. Отладка приложения;
9. Оформление технической документации.
   1. **Требования к функциональным характеристикам**

Разрабатываемая система должна выполнять следующие функции:

* + 1. Создание базы данных;
    2. Подключение к MySQL;
    3. Отслеживание появления нового файла в выбранном администратором каталоге;
    4. Извлечение необходимых данных из обнаруженного файла и занесение их в созданную базу данных;
    5. Вычисление нормированных показателей качества интернет соединения на основании данных полученных из файла;
    6. Изменение пороговых значений для расчета нормированных показателей качества интернет соединения;
    7. Создание администратором соответствующей записи в базе данных при появлении нового пользователя;
    8. Изменение констант для расчета нормированных показателей качества интернет соединения.

**5.3 Требования к архитектуре программного изделия**

Приложение находится на сервере. База данных также находится на сервере. Данные для заполнения базы данных находятся в файлах, которые приходят в выбранный администратором каталог. Администратор может взаимодействовать с приложением посредствам интерфейса.

* 1. **Требования к входным и выходным данным**
     1. **Требования к входным данным**

Входные данные представляют собой файл, в котором хранятся такие характеристики интернет соединения как:

1. Уровень использования услуги;
2. Скорость передач данных;
3. Качество задержки;
4. Качество задержки сети;
5. Качество временного окна;
6. Группа пользователя на основе его субъективных требований к услуге
7. **DTsc**: время начала сбора данных,
8. **Thrp**: данные скорости передачи информации (вверх/вниз) и поминутная гистограмма эффективного распределения скорости (измеряется ежесекундно)
9. **HstN:** название хоста,
10. **MfMd:** производитель и модель ПК,
11. **OpSy:** данные ОС,
12. **AdpI:** данные сетевого устройства,
13. **IPvX:** приватные IP адреса,
14. **InEr:** ошибки сетевого уровня (передача пакетов),
15. **PuIP:** публичный IP адрес,
16. **ICMP:** показатели и результаты «пинг» теста,
17. **HTTP:** показатели и результаты HTTP теста,
18. **DTec:** время конца сбора данных.
19. **UID:** идентификатор пользователя
    * 1. **Требования к выходным данным**

Выходные данные представляют собой базу данных, в которой хранятся данные об интернет соединения, полученные из файла, и следующие просчитанные нормированные показатели качества интернет соединения:

1. Нормализованный показатель качества уровня использования услуги;
2. Нормализованный показатель качества скорости передач данных;
3. Нормализованный показатель качества задержки;
4. Нормализованный показатель качества задержки сети;
5. Нормализованный показатель качества временного окна;
6. Группа пользователя на основе его субъективных требований к услуге
   1. **Требования к надежности**

Программа не должна выдавать ошибок, не предусмотренных работой программы.

* 1. **Лингвистические требования**

Интерфейс программного изделия должен быть русифицирован.

* 1. **Требования к программным компонентам**

Основным требованием к программным компонентам является корректное исполнение функций, заявленных в пункте 5.2

**5.7 Требования к составу программных средств**

Для работы приложения на сервере требуется:

1. ОС семейства Windows;
2. Язык программирования C#;
3. База данных MySQL версии 5.1.56 или выше.
   1. **Требования к составу технических средств**
4. Процессор с частотой 2 ГГц;
5. 4 ГБ оперативной памяти;
6. Видеоадаптер и монитор, способные обеспечить графический режим 1024\*768 точек с 32 битной цветопередачей;
7. Жесткий диск объемом 64 ГБ;
8. Манипулятор «мышь» или другое указывающее устройство;
9. Клавиатура;

**6. Этапы разработки**

График выполнения отдельных этапов работ приведен в соответствии с приказом об организации учебного процесса в 2017/2018 учебном году. Этапы разработки программного изделия представлены в Таблице 1.

Таблица 1: Этапы разработки

|  |  |  |
| --- | --- | --- |
| № п/п | Наименование этапа и содержание работ | Сроки исполнения |
| 1 | Разработка и утверждение ТЗ | Декабрь 2017 г. |
| 2 | Исследование предметной области | Январь 2018 г. |
| 3 | Разработка архитектуры программного  обеспечения | Январь 2018 г. |
| 4 | Создание программного обеспечения | Февраль — апрель 2018 г. |
| 5 | Тестирование и отладка ПО | Май 2018 г. |
| 6 | Оформление документации | Май — июнь 2018 г. |
| 7 | Защита работы | Июнь 2018 г. |

**7. Техническая документация**

По окончании работы предъявляется следующая техническая документация:

1. Техническое задание;
2. Расчетно-пояснительная записка;
3. Программа и методика испытаний;
4. Руководство пользователя;
5. Текст программы;

5. Графический материал по проекту в формате листов А1.

1. **Порядок приема работы**

Прием и контроль программного изделия осуществляется в соответствие с методикой испытаний (см. документ «Программа и методика испытаний»).

1. **Дополнительные условия**

Данное техническое задание может уточняться в установленном порядке.

Приложение 2. Графическая часть

Изображение выглядит как снимок экрана

Описание создано с очень высокой степенью достоверностиИзображение выглядит как снимок экрана

Описание создано с высокой степенью достоверностиИзображение выглядит как снимок экрана

Описание создано с высокой степенью достоверностиИзображение выглядит как текст, карта

Описание создано с очень высокой степенью достоверностиИзображение выглядит как снимок экрана, текст

Описание создано с высокой степенью достоверностиИзображение выглядит как снимок экрана, текст

Описание создано с высокой степенью достоверностиИзображение выглядит как снимок экрана

Описание создано с очень высокой степенью достоверности

Изображение выглядит как текст, карта

Описание создано с очень высокой степенью достоверностиИзображение выглядит как снимок экрана

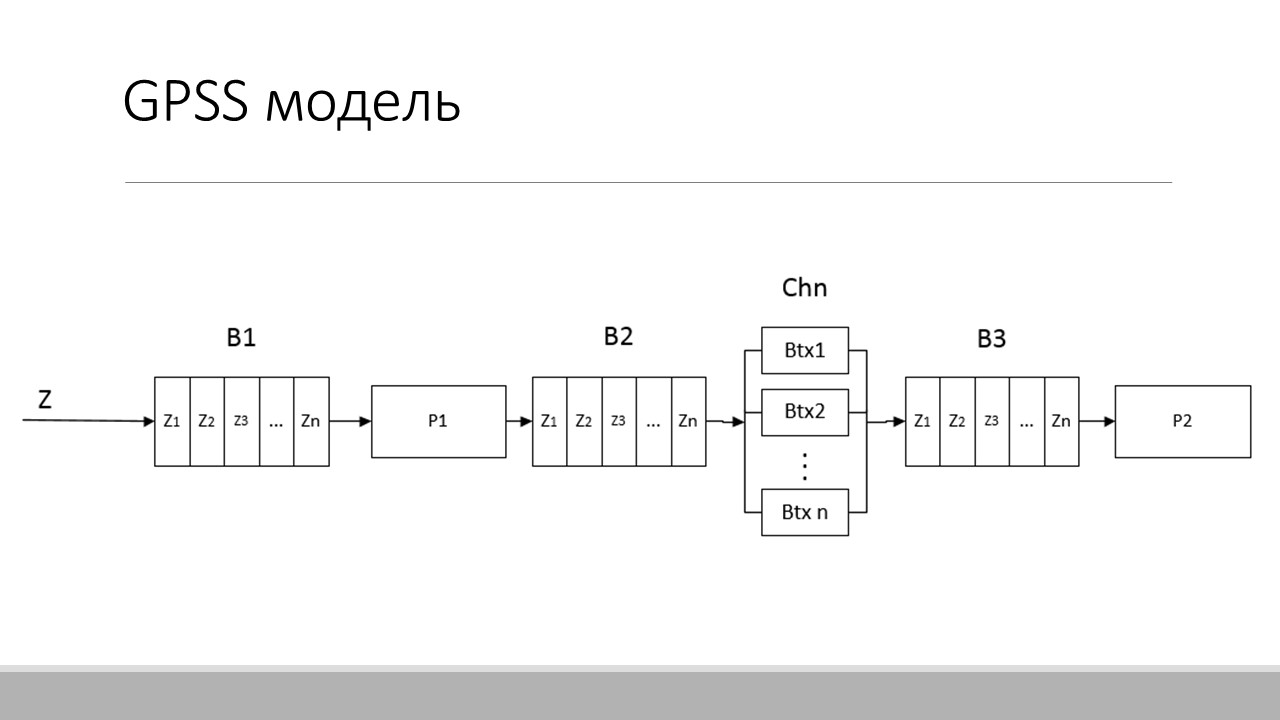
Описание создано с очень высокой степенью достоверностиИзображение выглядит как текст, карта

Описание создано с очень высокой степенью достоверностиИзображение выглядит как снимок экрана

Описание создано с очень высокой степенью достоверностиИзображение выглядит как снимок экрана

Описание создано с очень высокой степенью достоверностиИзображение выглядит как снимок экрана

Описание создано с очень высокой степенью достоверностиИзображение выглядит как снимок экрана

Описание создано с очень высокой степенью достоверностиИзображение выглядит как текст

Описание создано с высокой степенью достоверностиИзображение выглядит как снимок экрана

Описание создано с очень высокой степенью достоверностиИзображение выглядит как снимок экрана

Описание создано с высокой степенью достоверностиИзображение выглядит как животное

Описание создано с высокой степенью достоверности

Приложение 3. Текст программы

**Московский государственный технический университет им. Н.Э.Баумана**

Кафедра «Системы обработки информации и управления»

|  |  |  |  |
| --- | --- | --- | --- |
| Утверждаю: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Согласовано: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | "\_\_\_"\_\_\_\_\_\_\_\_\_\_2018 г. |  | "\_\_\_"\_\_\_\_\_\_\_\_\_\_\_2018 г. |

**Подсистема сбора и обработки данных технических параметров конечных интернет пользователей**

Текст программы

(вид документа)

Листы А4

(вид носителя)

25

(количество листов)

Исполнитель:

Студентка группы ИУ5-84

Матвейчук И.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_\_"\_\_\_\_\_\_\_\_\_\_2018 г.

Москва, 2018

**Аннотация**

В данном документе содержится текст программы «Подсистема сбора и обработки данных технических параметров конечных интернет пользователей». Разделы документа соответствуют модулям программы.

**Содержание**

[Аннотация 101](#_Toc515517106)

[Содержание 102](#_Toc515517107)

[1. Программа QualityCheck 103](#_Toc515517108)

[1.1. Модуль MainForm 103](#_Toc515517109)

[1.2. Модуль Database 106](#_Toc515517110)

[1.3. Модуль FileInfo 109](#_Toc515517111)

[2. Программа QualityCheck\_install 119](#_Toc515517112)

[2.1. Модуль MainForm 119](#_Toc515517113)

[2.2. Модуль LoadForm 120](#_Toc515517114)

[2.3. Скрипт create\_db.sql 122](#_Toc515517115)

**Программа QualityCheck**

**Модуль MainForm**

using System;

using System.Windows.Forms;

using System.IO;

using System.Diagnostics;

using MySql.Data.MySqlClient;

namespace QualityCheck

{

public partial class MainForm : Form

{

string directory, newDirectory;

FileInformation fi;

FileSystemWatcher watcher;

string path;

string connStr;

string text;

Database db;

public MainForm()

{

InitializeComponent();

textBoxThDMax.KeyPress += textBoxThDMax\_KeyPress;

textBoxThUMax.KeyPress += textBoxThUMax\_KeyPress;

textBoxThrpMax.KeyPress += textBoxThrpMax\_KeyPress;

}

public void Form1\_Load(object sender, EventArgs e)

{

PathArcTextBox.Text = "D:\\VKR\_files\\";

buttonStart.Enabled = false;

buttonStop.Enabled = false;

}

public void GetData()

{

fi.ThrpMax = Convert.ToDouble(textBoxThrpMax.Text);

fi.ThDMax = Convert.ToDouble(textBoxThDMax.Text);

fi.ThUMax = Convert.ToDouble(textBoxThUMax.Text);

fi.pl = Convert.ToInt32(textBoxWd.Text);

} //данные с формы

public void ProcessFile(string path)

{

Stopwatch sw = new Stopwatch();

sw.Start();

System.Threading.Thread.CurrentThread.CurrentCulture = new System.Globalization.CultureInfo("en-US");

fi.Clear();

fi.ReadData(directory, newDirectory);

GetData();

fi.CalculateParams();

db.FillInDB(fi, sw);

}

public string DBhost()

{

return textBoxHost.Text;

}

public string DBport()

{

return textBoxPort.Text;

}

public string DBlogin()

{

return textBoxLogin.Text;

}

public string DBpassword()

{

return textBoxPassword.Text;

}

private void button2\_Click(object sender, EventArgs e)

{

int gr = DefineGroup();

try

{

db.CreateUser(textBox\_firstname.Text, textBox\_secondname.Text, textBox\_thirdname.Text, textBox\_password.Text, textBox\_uid.Text, gr);

}

catch

{

MessageBox.Show("Проверьте правильность ввода даннных!");

}

textBox\_firstname.Text = "";

textBox\_secondname.Text = "";

textBox\_thirdname.Text = "";

textBox\_password.Text = "";

textBox\_uid.Text = "";

if (radioButton1.Checked == true) radioButton1.Checked = false;

else if (radioButton2.Checked == true) radioButton2.Checked = false;

else if (radioButton3.Checked == true) radioButton3.Checked = false;

MessageBox.Show("Пользователь добавлен");

} //добавление пользователя

public int DefineGroup()

{

if (radioButton1.Checked == true) return 1;

else if (radioButton2.Checked == true) return 2;

else if (radioButton3.Checked == true) return 3;

else return 0;

}

public void OnChanged(object source, FileSystemEventArgs e)

{

ProcessFile(directory);

}

private void buttonStart\_Click(object sender, EventArgs e)

{

bool fl = false;

db = new Database(connStr);

using (var conn = new MySqlConnection(connStr))

using (var cmd = conn.CreateCommand())

{

conn.Open();

cmd.CommandText = "use QualityInfo;";

cmd.ExecuteNonQuery();

cmd.CommandText = "select user\_id from users limit 1;";

MySqlDataReader reader = cmd.ExecuteReader();

if (reader.Read() == false) MessageBox.Show("Необходимо добавить пользователя");

else fl = true;

reader.Close(); // закрываем reader

conn.Close();

}

if (fl == true)

{

directory = PathArcTextBox.Text;

newDirectory = PathArcTextBox.Text + "\\OldFiles\\";

fi = new FileInformation(connStr);

fi.st.ReadSettings();

int nfiles = Directory.GetFiles(directory).Length;

if (nfiles != 0)

for (int i = 0; i < nfiles; i++)

ProcessFile(directory);

watcher = new FileSystemWatcher();

watcher.Path = PathArcTextBox.Text;

watcher.Filter = "\*.txt";

watcher.Created += new FileSystemEventHandler(OnChanged);

watcher.EnableRaisingEvents = true;

buttonStart.Enabled = false;

buttonStop.Enabled = true;

textBoxThDMax.Enabled = false;

textBoxThUMax.Enabled = false;

textBoxThrpMax.Enabled = false;

textBoxWd.Enabled = false;

}

}

private void buttonConnect\_Click(object sender, EventArgs e)

{

try

{

connStr = "server=" + DBhost() + ";user=" + DBlogin() + ";port=" + DBport() + ";password=" + DBpassword() + ";";

using (var conn = new MySqlConnection(connStr))

{

conn.Open();

conn.Close();

}

buttonStart.Enabled = true;

buttonConnect.Enabled = false;

}

catch

{

MessageBox.Show("Проверьте правильность ввода параметров");

}

}

private void buttonStop\_Click(object sender, EventArgs e)

{

buttonStart.Enabled = true;

buttonStop.Enabled = false;

textBoxThDMax.Enabled = true;

textBoxThUMax.Enabled = true;

textBoxThrpMax.Enabled = true;

textBoxWd.Enabled = true;

watcher.EnableRaisingEvents = false;

}

private void textBoxThDMax\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if (!Char.IsDigit(number) && number != 8 && number != 44)

e.Handled = true;

}

private void textBoxThUMax\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if (!Char.IsDigit(number) && number != 8 && number != 44)

e.Handled = true;

}

private void textBoxThrpMax\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if (!Char.IsDigit(number) && number != 8 && number != 44)

e.Handled = true;

}

private void buttonChangeConst\_Click(object sender, EventArgs e)

{

Process.Start(@"settings.txt");

}

private void buttonDir1\_Click(object sender, EventArgs e)

{

FolderBrowserDialog DirDialog = new FolderBrowserDialog();

DirDialog.Description = "Выбор директории";

DirDialog.SelectedPath = @"D:\";

if (DirDialog.ShowDialog() == DialogResult.OK)

{

PathArcTextBox.Text = DirDialog.SelectedPath;

directory = PathArcTextBox.Text;

newDirectory = PathArcTextBox.Text + "\\OldFiles\\";

}

}

}

}

**Модуль Database**

using System;

using System.IO;

using MySql.Data.MySqlClient;

using System.Windows.Forms;

using System.Diagnostics;

namespace QualityCheck

{

class Database

{

public string connStr;

public Database(string conn)

{

this.connStr = conn;

}

public void CreateDB()

{

string script = File.ReadAllText("create\_db.sql");

using (var conn = new MySqlConnection(connStr))

{

conn.Open();

var comd = new MySqlCommand(script, conn);

comd.ExecuteNonQuery();

conn.Close();

}

}

public void CreateUser(string firstname, string secondname, string thirdname, string password, string uid, int group)

{

//string connStr = "server=" + db.host + ";user=" + db.login + ";port=" + db.port + ";password=" + db.password + ";";

using (var conn = new MySqlConnection(connStr))

using (var cmd = conn.CreateCommand())

{

conn.Open();

cmd.CommandText = "use QualityInfo;";

cmd.ExecuteNonQuery();

string comm = "insert into users (user\_uid, lastname, firstname, thirdname, user\_password, user\_group) values " +

"(" + uid + ", '" + secondname + "', '" + firstname + "', '" + thirdname + "', '" + password + "', "+group+"); ";

cmd.CommandText = comm;

cmd.ExecuteNonQuery();

conn.Close();

}

}

public void FillInDB(FileInformation fi, Stopwatch sw)

{

string comm;

string dr = "";

int userid, collection\_id;

using (var conn = new MySqlConnection(connStr))

using (var cmd = conn.CreateCommand())

{

conn.Open();

cmd.CommandText = "use QualityInfo;";

cmd.ExecuteNonQuery();

cmd.CommandText = "SELECT user\_id FROM users WHERE user\_uid = " + fi.user\_uid + ";";

MySqlDataReader reader = cmd.ExecuteReader();

int k = 0;

while (reader.Read())

{

dr += reader[0].ToString();

k++;

}

reader.Close();

userid = Convert.ToInt32(dr);

dr = "";

fi.ChangeDateFormat(ref fi.DTsc);

fi.ChangeDateFormat(ref fi.DTec);

comm = "INSERT INTO collection\_file (file\_name, collection\_start, collection\_end, user\_id) VALUES " +

"('" + fi.filename + "', '" + fi.DTsc + "', '" + fi.DTec + "', " + userid + ");";

cmd.CommandText = comm;

cmd.ExecuteNonQuery();

cmd.CommandText = "SELECT collection\_id FROM collection\_file WHERE file\_name = '" + fi.filename + "';";

reader = cmd.ExecuteReader();

k = 0;

while (reader.Read())

{

dr += reader[0].ToString();

k++;

}

reader.Close();

collection\_id = Convert.ToInt32(dr);

dr = "";

comm = "insert into device\_information (manufacturer, model, os\_version, collection\_id) values " +

"('" + fi.Manufacturer + "', '" + fi.Model + "', '" + fi.OSversion + "', " + collection\_id + ");";

cmd.CommandText = comm;

cmd.ExecuteNonQuery();

for (int i = 0; i < fi.ThCount; i++)

{

fi.ChangeDateFormat(ref fi.Thrp[i].datetime);

comm = "insert into throughput\_raw\_data (collection\_datetime, max\_download, max\_upload, first\_scale, second\_scale, third\_scale, fourth\_scale, fifth\_scale, collection\_id) values " +

"('" + fi.Thrp[i].datetime + "', " + fi.Thrp[i].downspeed + ", " + fi.Thrp[i].upspeed + ", " + fi.Thrp[i].gr[0] + ", " + fi.Thrp[i].gr[1]

+ ", " + fi.Thrp[i].gr[2] + ", " + fi.Thrp[i].gr[3] + ", " + fi.Thrp[i].gr[4] + ", " + collection\_id + ");";

cmd.CommandText = comm;

cmd.ExecuteNonQuery();

}

for (int i = 0; i < 2; i++)

{

comm = "insert into icmp\_test (url\_test, test\_result, icmp\_length, rtt, ttl\_default, ttl, collection\_id) values " +

"('" + fi.ICMP[i].IP + "', '" + fi.ICMP[i].status + "', " + fi.ICMP[i].jumps + ", " + fi.ICMP[i].bytes +

", " + fi.ICMP[i].maxTime + ", " + fi.ICMP[i].time + ", " + collection\_id + ")";

cmd.CommandText = comm;

cmd.ExecuteNonQuery();

}

for (int i = 0; i < 2; i++)

{

comm = "insert into http\_test (url\_test, bytes\_received, time\_elapsed, status\_code, collection\_id) values " +

"('" + fi.HTTP[i].URL + "', " + fi.HTTP[i].bytesReceived + ", " + fi.HTTP[i].timeElapsed + ", '" + fi.HTTP[i].status + "', " + collection\_id + ");";

cmd.CommandText = comm;

cmd.ExecuteNonQuery();

}

fi.ChangeDateFormat(ref fi.InErdt);

comm = "insert into traffic\_error (collection\_datetime, in\_pack\_discard, in\_pack\_errors, out\_pack\_discard, out\_pack\_errors, collection\_id) values " +

"('" + fi.InErdt + "', " + fi.InEr[0] + ", " + fi.InEr[1] + ", " + fi.InEr[2] + ", " + fi.InEr[3] + ", " + collection\_id + ");";

cmd.CommandText = comm;

cmd.ExecuteNonQuery();

comm = "insert into parameter (kRg, kTh, kDy, kEr, kWd, collection\_id) values " +

"(" + fi.kRg + ", " + fi.kTh + ", " + fi.kDy + ", " + fi.kEr + ", " + fi.kWd + ", " + collection\_id + ");";

cmd.CommandText = comm;

cmd.ExecuteNonQuery();

conn.Close();

sw.Stop();

MessageBox.Show((sw.ElapsedMilliseconds / 100.0).ToString());

MessageBox.Show("Данные из последнего файла от пользователя с UID = " + fi.user\_uid + " обработаны и загружены в БД. Ожидается выставление оценки.");

}

}

}

}

**Модуль FileInfo**

using System;

using System.IO;

using System.Linq;

using System.Globalization;

using MySql.Data.MySqlClient;

namespace QualityCheck

{

class FileInformation

{

public Settings st;

string connStr;

public int pl=1; //ко-во последних измерений для оценки

public double HTTPSpeed;

public string DTsc, DTec; //начало сбора данных

public ICMPClass[] ICMP = new ICMPClass[2];

public HTTPClass[] HTTP = new HTTPClass[2];

public ThrpClass[] Thrp = new ThrpClass[25];

public string HstN; //название хоста

public string AdpI;

public string[] AdpI1 = new string[3];

public string MfMd; //проиводитель и модель ПК

public string Manufacturer;

public string Model;

public string OSversion; //данные ОС

public string IPvX;

public string[] IPv64 = new string[2];

public int[] InEr = new int[4];

public string InErdt;

public string EvErdt;

public int EvErnum;

public string filename;

public int ThCount = 25;

public double ThrpMax, ThDMax, ThUMax; //макс. скорость связи

public string user\_uid;

public int user\_group;

public double[] Qrg;

public double kRg;

public double[] Qth;

public double kTh;

public double Qdy;

public double kDy;

public double Qer;

public double kEr;

public double Qwd;

public double kWd;

public FileInformation(string conn)

{

st = new Settings();

this.connStr = conn;

for (int i = 0; i < 2; i++)

HTTP[i] = new HTTPClass();

for (int i = 0; i < 2; i++)

ICMP[i] = new ICMPClass();

for (int i = 0; i < ThCount; i++)

Thrp[i] = new ThrpClass();

}

public void CalculateParams()

{

Qrg = QrgCount();

kRg = kRgCount(Qrg);

Qth = QthCount();

kTh = kThCount(Qth);

Qdy = QdyCount();

kDy = kDyCount(Qdy);

Qer = QerCount();

kEr = kErCount(Qer);

Qwd = QwdCount(pl, connStr);

kWd = kWdCount(Qwd);

}

public double QerCount()

{

double Qer = st.d[0] \* Math.Pow(Math.E, -InEr[0]) + st.d[1] \* Math.Pow(Math.E, -InEr[2]);

return Qer;

}

public double kErCount(double Qer)

{

double Qermin = st.d[0] \* (1 / Math.E) + st.d[1] \* (1 / Math.E);

double Qermax = st.d[0] + st.d[1];

double kEr = (Qer - Qermin) / (Qermax - Qermin);

return kEr;

}

public double[] QrgCount()

{

double[] Qrg = new double[ThCount];

for (int i = 0; i < ThCount; i++)

{

for (int j = 0; j < st.qgr; j++)

{

st.p[j] = (100 \* Thrp[i].gr[j]) / 60;

st.p[j] = st.p[j] / 100;

}

Qrg[i] = 0;

for (int j = 0; j < st.qgr; j++)

{

Qrg[i] += st.a[j] \* Math.Pow(Math.E, st.p[j]);

}

}

return Qrg;

}

public double kRgCount(double[] Qrg)

{

double Qrgmin = 0, Qrgmax = 0, sum = 0, kRg;

double[] kRg1 = new double[ThCount];

for (int i = 0; i < st.qgr; i++)

{

if (i == 0) Qrgmax += st.a[i] \* Math.E;

else Qrgmax += st.a[i];

}

for (int i = 0; i < st.qgr; i++)

{

if (i == st.qgr - 1) Qrgmin += st.a[i] \* Math.E;

else Qrgmin += st.a[i];

}

for (int i = 0; i < ThCount; i++)

{

kRg1[i] = (Qrg[i] - Qrgmin) / (Qrgmax - Qrgmin);

sum += kRg1[i];

}

kRg = sum / kRg1.Length;

return kRg;

}

public double QdyCount()

{

double Qdy = 0;

for (int i = 0; i < st.qdy; i++)

{

double pow = -((double)ICMP[i].time / (double)ICMP[i].maxTime);

Qdy += st.c[i] \* Math.Pow(Math.E, pow);

}

return Qdy;

}

public double kDyCount(double Qdy)

{

double Qdymin = 0, Qdymax = 0;

double kDy;

for (int i = 0; i < st.qdy; i++)

{

Qdymax += st.c[i] \* (1 / Math.E);

Qdymin += st.c[0];

}

kDy = (Qdy - Qdymin) / (Qdymax - Qdymin);

return kDy;

}

public double[] QthCount()

{

double[] QTh = new double[ThCount];

double Tdw = (HTTP[0].bytesReceived + HTTP[1].bytesReceived) / 2;

for (int i = 0; i < ThCount; i++)

{

QTh[i] = st.b[0] \* Math.Pow(Math.E, Thrp[i].downspeed / ThrpMax) + st.b[1] \* Math.Pow(Math.E, Thrp[i].upspeed / ThrpMax)

+ st.b[2] \* Math.Pow(Math.E, Tdw / ThrpMax);

}

return QTh;

}

public double kThCount(double[] QTh)

{

double sum = 0, kTh;

double[] kTh1 = new double[ThCount];

double QThMax = st.b[0] \* Math.Pow(Math.E, ThDMax / ThrpMax) + st.b[1] \* Math.Pow(Math.E, ThUMax / ThrpMax)

+ st.b[2] \* Math.Pow(Math.E, HTTPSpeed / ThrpMax);

double QThMin = st.b[0] + st.b[1] + st.b[2];

for (int i = 0; i < ThCount; i++)

{

kTh1[i] = (QTh[i] - QThMin) / (QThMax - QThMin);

if (kTh1[i] > 1) kTh1[i] = 1;

sum += kTh1[i];

}

kTh = sum / kTh1.Length;

return kTh;

}

public double QwdCount(int n, string connStr)

{

double Qwd = 0;

int count;

using (var conn = new MySqlConnection(connStr))

using (var cmd = conn.CreateCommand())

{

conn.Open();

cmd.CommandText = "use QualityInfo;";

cmd.ExecuteNonQuery();

cmd.CommandText = "SELECT COUNT(\*) FROM collection\_file WHERE quality\_mark IS NOT NULL;";

count = Convert.ToInt32(cmd.ExecuteScalar());

if (n < count) count = n;

int[] marks = new int[count];

cmd.CommandText = "SELECT quality\_mark FROM collection\_file WHERE quality\_mark IS NOT NULL ORDER BY collection\_id DESC LIMIT " + count + ";";

MySqlDataReader reader = cmd.ExecuteReader();

int k = 0;

while (reader.Read())

{

marks[k] = Convert.ToInt32(reader[0]);

k++;

}

reader.Close();

conn.Close();

for (int i = 0; i < k; i++)

Qwd += marks[i] \* Math.Pow(Math.E, -(i + 1) / k);

pl = k;

}

return Qwd;

}

public double kWdCount(double Qwd)

{

double kWd;

double Qwdmin = 0, Qwdmax = 0;

for (int i = 0; i < pl; i++)

{

Qwdmin += 1 \* Math.Pow(Math.E, -(i + 1) / pl);

Qwdmax += 5 \* Math.Pow(Math.E, -(i + 1) / pl);

}

kWd = (Qwd - Qwdmin) / (Qwdmax - Qwdmin);

if (Qwd == 0) kWd = 1;

return kWd;

}

public void ReadData(string directory, string newDirectory)

{

string text;

DirectoryInfo inputDirectory = new DirectoryInfo(directory);

var myFile = inputDirectory.GetFiles().OrderByDescending(f => f.LastWriteTime).First();

filename = myFile.ToString();

string path = directory+"\\" + filename + "";

using (StreamReader sr = new StreamReader(path, System.Text.Encoding.Default))

text = sr.ReadLine();

int indDTsc = text.IndexOf("DTsc");

for (int i = indDTsc + 4; i < indDTsc + 23; i++)

DTsc += text[i];

int indDTec = text.IndexOf("DTec");

for (int i = indDTec + 4; i < indDTec + 23; i++)

DTec += text[i];

string data = "";

int j = 1, m = 0;

for (int i = 29; i < text.Length; i++)

{

data += text[i].ToString();

if (text[i].ToString() == "," && j == 1)

{

data = data.Remove(data.Length - 1);

Thrp[m].datetime = data;

//MessageBox.Show(fi.Thrp[m].datetime.ToString());

data = "";

j++;

continue;

}

else if (text[i].ToString() == "," && j == 2)

{

data = data.Remove(data.Length - 1);

Thrp[m].downspeed = Convert.ToDouble(data, CultureInfo.GetCultureInfo("en-US"));

data = "";

j++;

continue;

}

else if (text[i].ToString() == "," && j == 3)

{

data = data.Remove(data.Length - 1);

Thrp[m].upspeed = Convert.ToDouble(data, CultureInfo.GetCultureInfo("en-US"));

data = "";

j++;

continue;

}

else if (text[i].ToString() == "," && j == 4)

{

data = data.Remove(data.Length - 1);

Thrp[m].gr[0] = Convert.ToInt16(data);

data = "";

j++;

continue;

}

else if (text[i].ToString() == "," && j == 5)

{

data = data.Remove(data.Length - 1);

Thrp[m].gr[1] = Convert.ToInt16(data);

data = "";

j++;

continue;

}

else if (text[i].ToString() == "," && j == 6)

{

data = data.Remove(data.Length - 1);

Thrp[m].gr[2] = Convert.ToInt16(data);

data = "";

j++;

continue;

}

else if (text[i].ToString() == "," && j == 7)

{

data = data.Remove(data.Length - 1);

Thrp[m].gr[3] = Convert.ToInt16(data);

data = "";

j++;

continue;

}

else if ((text[i].ToString() == "," || text[i].ToString() == "|") && j == 8)

{

data = data.Remove(data.Length - 1);

if (text[i].ToString() == "|") i += 4;

Thrp[m].gr[4] = Convert.ToInt16(data);

data = "";

j = 1;

m++;

if (m == 25) break;

continue;

}

}

int indHstN = text.IndexOf("HstN");

for (int i = indHstN + 4; ; i++)

{

if (text[i] == '|') break;

HstN += text[i];

}

int indMfMd = text.IndexOf("MfMd");

for (int i = indMfMd + 4; ; i++)

{

if (text[i] == '|') break;

MfMd += text[i];

}

int indOpSy = text.IndexOf("OpSy");

for (int i = indOpSy + 4; ; i++)

{

if (text[i] == '|') break;

OSversion += text[i];

}

int indAdpI = text.IndexOf("AdpI");

for (int i = indAdpI + 4; ; i++)

{

if (text[i] == '|') break;

AdpI += text[i];

}

AdpI1 = AdpI.Split(',');

int indIPvX = text.IndexOf("IPvX");

for (int i = indIPvX + 4; ; i++)

{

if (text[i] == '|') break;

IPvX += text[i];

}

IPv64 = IPvX.Split(',');

int indHTTP1 = text.IndexOf("HTTP");

int indHTTP2 = text.IndexOf("HTTP", indHTTP1 + 1);

for (int i = 0; i < 2; i++)

{

data = "";

if (i == 0) j = indHTTP1 + 4;

else j = indHTTP2 + 4;

while (text[j] != ',')

{

HTTP[i].URL += text[j];

j++;

}

j++;

while (text[j] != ',')

{

data += text[j];

j++;

}

j++;

HTTP[i].bytesReceived = Convert.ToInt32(data);

data = "";

while (text[j] != ',')

{

data += text[j];

j++;

}

j++;

HTTP[i].timeElapsed = Convert.ToInt32(data);

while (text[j] != '|')

{

HTTP[i].status += text[j];

j++;

}

}

int indICMP1 = text.IndexOf("ICMP");

int indICMP2 = text.IndexOf("ICMP", indICMP1 + 1);

for (int i = 0; i < 2; i++)

{

data = "";

if (i == 0) j = indICMP1 + 4;

else j = indICMP2 + 4;

while (text[j] != ',')

{

ICMP[i].IP += text[j];

j++;

}

j++;

while (text[j] != ',')

{

ICMP[i].status += text[j];

j++;

}

j++;

while (text[j] != ',')

{

data += text[j];

j++;

}

j++;

ICMP[i].bytes = Convert.ToInt16(data);

data = "";

while (text[j] != ',')

{

data += text[j];

j++;

}

j++;

ICMP[i].time = Convert.ToInt16(data);

data = "";

while (text[j] != ',')

{

data += text[j];

j++;

}

j++;

ICMP[i].maxTime = Convert.ToInt16(data);

data = "";

while (text[j] != '|')

{

data += text[j];

j++;

}

j++;

ICMP[i].jumps = Convert.ToInt16(data);

data = "";

}

int indInEr = text.IndexOf("InEr");

for (int i = indInEr + 4; i < indInEr + 23; i++)

InErdt += text[i];

int indUID = text.IndexOf("UID");

for (int i = indUID + 3; ; i++)

{

if (text[i] == '|') break;

user\_uid += text[i];

}

j = indInEr + 24; data = "";

for (int i = 0; i < 4; i++)

{

while (text[j] != ',' && text[j] != '|')

{

data += text[j];

j++;

}

j++;

InEr[i] = Convert.ToInt16(data);

data = "";

}

int indEvEr = text.IndexOf("EvEr");

for (int i = indEvEr + 4; i < indEvEr + 23; i++)

EvErdt += text[i];

j = indEvEr + 24; data = "";

while (text[j] != '|')

{

data += text[j];

j++;

}

EvErnum = Convert.ToInt16(data);

Directory.CreateDirectory(newDirectory);

File.Move(directory + "\\" + filename, newDirectory + filename);

}

public void ChangeDateFormat(ref string date)

{

string day = date[0].ToString() + date[1].ToString();

string month = date[3].ToString() + date[4].ToString();

string year = date.Substring(6, 4);

date = year + "-" + month + "-" + day + " " + date.Substring(11);

}

public void Clear()

{

DTsc = "";

DTec = ""; //начало сбора данных

HstN = ""; //название хоста

AdpI = "";

MfMd = ""; //проиводитель и модель ПК

Manufacturer = "";

Model = "";

OSversion = ""; //данные ОС

IPvX = "";

InErdt = "";

EvErdt = "";

filename = "";

user\_uid = "";

for (int i=0;i<2;i++)

{

ICMP[i].IP = "";

HTTP[i].URL = "";

HTTP[i].status = "";

Thrp[i].datetime = "";

}

for (int i = 0; i < ThCount; i++)

{

Thrp[i].datetime = "";

}

filename = "";

}

}

public class Settings

{

string settingspath = "settings.txt";

public int qgr = 5; //кол-во скоростных групп

public int qdy = 2; //кол-во серверов для измерения задержки

public int[] a;

public double[] p;

public int[] b = new int[3];

public int[] c;

public int[] d = new int[2];

public Settings()

{

a = new int[qgr];

p = new double[qgr];

c = new int[qdy];

d = new int[2];

}

public void ReadSettings()

{

string[] strings = new string[17];

strings = File.ReadAllLines(settingspath);

for (int i = 0; i < 5; i++)

{

string data = "";

int j = 3;

while (strings[i + 2][j] != ';')

{

data += strings[i + 2][j];

j++;

}

a[i] = Convert.ToInt16(data);

}

for (int i = 0; i < 3; i++)

{

string data = "";

int j = 3;

while (strings[i + 8][j] != ';')

{

data += strings[i + 8][j];

j++;

}

b[i] = Convert.ToInt16(data);

}

for (int i = 0; i < 2; i++)

{

string data = "";

int j = 3;

while (strings[i + 12][j] != ';')

{

data += strings[i + 12][j];

j++;

}

c[i] = Convert.ToInt16(data);

}

for (int i = 0; i < 2; i++)

{

string data = "";

int j = 3;

while (strings[i + 15][j] != ';' && strings[i + 15][j] != '.')

{

data += strings[i + 15][j];

j++;

}

d[i] = Convert.ToInt16(data);

}

} //данные из файла настроек

}

public class ThrpClass

{

public string datetime;

public double downspeed, upspeed;

public int[] gr = new int[5];

}

public class ICMPClass

{

public string IP;

public double status;

public int bytes;

public int time;

public int maxTime;

public int jumps;

}

public class HTTPClass

{

public string URL;

public int bytesReceived;

public int timeElapsed;

public string status;

}

}

**Программа QualityCheck\_install**

**Модуль MainForm**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

namespace QualityCheck\_install

{

public partial class MainForm : Form

{

public string connStr;

LoadForm f2;

public MainForm()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

//connStr = "server=" + textBoxHost.Text + ";user=" + textBoxLogin.Text + ";port=" + textBoxPort.Text + ";password=" + textBoxPassword.Text + ";";

buttonReady.Enabled = false;

}

private void buttonCreate\_Click(object sender, EventArgs e)

{

try

{

CreateDB();

}

catch

{

MessageBox.Show("Проверьте правильность ввода параметров");

}

}

private void buttonReady\_Click(object sender, EventArgs e)

{

this.Close();

}

public void CreateDB()

{

connStr = "server=" + textBoxHost.Text + ";user=" + textBoxLogin.Text + ";port=" + textBoxPort.Text + ";password=" + textBoxPassword.Text + ";";

using (var conn = new MySqlConnection(connStr))

{

conn.Open();

conn.Close();

}

f2 = new LoadForm(connStr);

buttonCreate.Enabled = false;

f2.ShowDialog();

buttonReady.Enabled = true;

}

}

}

**Модуль LoadForm**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

namespace QualityCheck\_install

{

public partial class LoadForm : Form

{

string connStr;

public LoadForm(string text)

{

InitializeComponent();

Shown += new EventHandler(Form2\_Shown);

connStr = text;

}

private void Form2\_Load(object sender, EventArgs e)

{

progressBar1.Maximum = 100;

progressBar1.Value = 0;

buttonReady.Enabled = false;

}

private void Form2\_Shown(object sender, EventArgs e)

{

using (var conn = new MySqlConnection(connStr))

{

conn.Open();

var comd = new MySqlCommand("CREATE DATABASE IF NOT EXISTS `QualityInfo`;", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 5;

comd = new MySqlCommand("USE qualityinfo;", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 10;

label1.Text = "Создание таблиц...";

comd = new MySqlCommand("CREATE TABLE IF NOT EXISTS users (" +

"user\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT," +

"user\_uid char(20) NOT NULL unique," +

"lastname char(20)," +

"firstname char(20)," +

"thirdname char(20)," +

"user\_password char(20)," +

"user\_group int); ", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 20;

comd = new MySqlCommand("CREATE TABLE IF NOT EXISTS collection\_file (" +

"collection\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT, " +

"file\_name CHAR(50) unique, collection\_start DATETIME, " +

"collection\_end DATETIME, " +

"quality\_mark INT,user\_id INT, " +

"FOREIGN KEY (user\_id) REFERENCES users(user\_id));", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 30;

comd = new MySqlCommand("CREATE TABLE IF NOT EXISTS device\_information (" +

"device\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT, " +

"manufacturer CHAR(50), model CHAR(50), os\_version char(50), " +

"collection\_id INT, " +

"FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 30;

comd = new MySqlCommand("CREATE TABLE IF NOT EXISTS network\_information (" +

"network\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT, hostname CHAR(50), " +

"mac CHAR(50), interface\_type char(50), device\_description CHAR(50), " +

"ipv6 CHAR(50), ipv4 char(50), collection\_id INT, " +

"FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 40;

comd = new MySqlCommand("CREATE TABLE IF NOT EXISTS throughput\_raw\_data (" +

"throughput\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT, " +

"collection\_datetime DATETIME," +

"max\_download float," +

"max\_upload float," +

"first\_scale INT," +

"second\_scale INT," +

"third\_scale INT," +

"fourth\_scale INT," +

"fifth\_scale INT, " +

"collection\_id INT, " +

"FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 50;

comd = new MySqlCommand("CREATE TABLE IF NOT EXISTS icmp\_test (" +

"icmp\_test\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT, " +

"url\_test CHAR(50), " +

"test\_result char(10), " +

"icmp\_length INT, rtt INT, ttl\_default INT, " +

"ttl INT," +

"collection\_id INT," +

"FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 60;

comd = new MySqlCommand("CREATE TABLE IF NOT EXISTS http\_test (" +

"http\_test\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT," +

"url\_test CHAR(50), " +

"bytes\_received INT, " +

"time\_elapsed INT, " +

"status\_code CHAR(10)," +

"collection\_id INT," +

"FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 70;

comd = new MySqlCommand("CREATE TABLE IF NOT EXISTS traffic\_error (" +

"traffic\_error\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT, " +

"collection\_datetime DATETIME, " +

"in\_pack\_discard INT, " +

"in\_pack\_errors INT, " +

"out\_pack\_discard INT, " +

"out\_pack\_errors INT," +

"collection\_id INT," +

"FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 80;

comd = new MySqlCommand("CREATE TABLE IF NOT EXISTS event\_error (" +

"traffic\_error\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT, " +

"collection\_datetime DATETIME, " +

"error\_number INT, " +

"collection\_id INT," +

"FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 90;

comd = new MySqlCommand("CREATE TABLE IF NOT EXISTS parameter (" +

"parameter\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT, " +

"kRg FLOAT, " +

"kTh FLOAT, " +

"kDy FLOAT, " +

"kEr FLOAT, " +

"kWd FLOAT, " +

"collection\_id INT," +

"FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));", conn);

comd.ExecuteNonQuery();

progressBar1.Value = 100;

conn.Close();

}

buttonReady.Enabled = true;

}

private void buttonReady\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

**Скрипт create\_db.sql**

CREATE DATABASE IF NOT EXISTS `QualityInfo`;

USE qualityinfo;

CREATE TABLE IF NOT EXISTS users (

user\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT,

user\_uid INT NOT NULL unique,

lastname char(20),

firstname char(20),

thirdname char(20),

user\_password char(20));

CREATE TABLE IF NOT EXISTS collection\_file (

collection\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT,

file\_name CHAR(50),

collection\_start DATETIME,

collection\_end DATETIME,

quality\_mark INT,

user\_id INT,

FOREIGN KEY (user\_id) REFERENCES users(user\_id));

CREATE TABLE IF NOT EXISTS device\_information (

device\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT,

manufacturer CHAR(50),

model CHAR(50),

os\_version char(50),

collection\_id INT,

FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));

CREATE TABLE IF NOT EXISTS network\_information (

network\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT,

hostname CHAR(50),

mac CHAR(50),

interface\_type char(50),

device\_decription CHAR(50),

ipv6 CHAR(50),

ipv4 char(50),

collection\_id INT,

FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));

CREATE TABLE IF NOT EXISTS throughput\_raw\_data (

throughput\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT,

collection\_datetime DATETIME,

max\_download float,

max\_upload float,

first\_scale INT,

second\_scale INT,

third\_scale INT,

fourth\_scale INT,

fifth\_scale INT,

collection\_id INT,

FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));

CREATE TABLE IF NOT EXISTS icmp\_test (

icmp\_test\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT,

test\_datetime DATETIME,

url\_test CHAR(50),

test\_result char(10),

icmp\_length INT,

rtt INT,

ttl\_default INT,

ttl INT,

collection\_id INT,

FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));

CREATE TABLE IF NOT EXISTS http\_test (

http\_test\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT,

test\_datetime DATETIME,

url\_test CHAR(50),

bytes\_received INT,

time\_elapsed INT,

status\_code CHAR(10),

collection\_id INT,

FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));

CREATE TABLE IF NOT EXISTS traffic\_error (

traffic\_error\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT,

collection\_datetime DATETIME,

in\_pack\_discard INT,

in\_pack\_errors INT,

out\_pack\_discard INT,

out\_pack\_errors INT,

collection\_id INT,

FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));

CREATE TABLE IF NOT EXISTS event\_error (

traffic\_error\_id INT NOT NULL PRIMARY KEY AUTO\_iNCREMENT,

collection\_datetime DATETIME,

error\_number INT,

collection\_id INT,

FOREIGN KEY (collection\_id) REFERENCES collection\_file(collection\_id));